Benutzerhandbuch

AWS Kryptografie für Zahlungen

AWS Kryptografie für Zahlungen: Benutzerhandbuch

Table of Contents

Was istAWSZahlungskryptographie?

AWSPayment Cryptography ist eine verwalteteAWSService, der den Zugriff auf kryptografische Funktionen und die Schlüsselverwaltung ermöglicht, die bei der Zahlungsabwicklung gemäß den Standards der Payment Card Industry (PCI) verwendet werden, ohne dass Sie spezielle HSM-Instances für Zahlungen beschaffen müssen.AWS Payment Cryptography bietet Kunden, die Zahlungsfunktionen ausführen, wie Acquirer, Zahlungsvermittler, Netzwerke, Switches, Prozessoren und Banken, die Möglichkeit, ihre kryptografischen Zahlungsvorgänge näher an Anwendungen in der Cloud zu verlagern und die Abhängigkeit von zusätzlichen Rechenzentren oder Colocation-Einrichtungen mit speziellen Zahlungs-HSM zu minimieren.

Der Service ist so konzipiert, dass er die geltenden Branchenregeln wie PCI-PIN, PCI P2PE und PCI DSS erfüllt, und der Service nutzt Hardware, die er istPCI PTS HSM V3 und FIPS 140-2 Level 3 zertifiziert. Es wurde entwickelt, um eine niedrige Latenz zu unterstützen undhohe Verfügbarkeit und Belastbarkeit.AWS Die Zahlungskryptografie ist vollständig elastisch und macht viele der betrieblichen Anforderungen von lokalen HSMs überflüssig, z. B. die Notwendigkeit, Hardware bereitzustellen, Schlüsselmaterial sicher zu verwalten und Notfall-Backups in sicheren Einrichtungen zu führen.AWS Zahlungskryptografie bietet Ihnen auch die Möglichkeit, Schlüssel elektronisch mit Ihren Partnern zu teilen, wodurch die Notwendigkeit entfällt, Klartext-Komponenten auf Papier auszutauschen.

Sie können das verwendenAWSAPI zur Steuerung der Zahlungskryptographieum Schlüssel zu erstellen und zu verwalten.

Sie können das verwendenAWSDatenplane-API für Zahlungskryptographiezur Verwendung von Verschlüsselungsschlüsseln für die zahlungsbezogene Transaktionsverarbeitung und die damit verbundenen kryptografischen Operationen.

AWSPayment Cryptography bietet wichtige Funktionen, mit denen Sie Ihre Schlüssel verwalten können:

- Symmetrisch und asymmetrisch erstellen und verwaltenAWSVerschlüsselungsschlüssel für Zahlungen, einschließlich TDES-, AES- und RSA-Schlüssel, und spezifizieren deren Verwendungszweck, z. B. für die CVV-Generierung oder die DUKPT-Schlüsselableitung.
- Speichern Sie automatisch IhreAWSVerschlüsselungsschlüssel für Zahlungen sind sicher, geschützt durch Hardware-Sicherheitsmodule (HSMs), während gleichzeitig die Schlüsseltrennung zwischen den Anwendungsfällen durchgesetzt wird.

- Aliase erstellen, löschen, auflisten und aktualisieren. Dabei handelt es sich um „benutzerfreundliche Namen“, die verwendet werden können, um auf IhreAWSVerschlüsselungsschlüssel für Zahlungen.
- Markiere deinAWSVerschlüsselungsschlüssel für Zahlungen zur Identifizierung, Gruppierung, Automatisierung, Zugriffskontrolle und Kostenverfolgung.
- Importiere und exportiere symmetrische Schlüssel zwischenAWSZahlungskryptografie und Ihr HSM (oder Drittanbieter) verwenden Key Encryption Keys (KEK) gemäß TR-31 (Interoperable Secure Key Exchange Key Block Specification).
- Importieren und exportieren Sie symmetrische Schlüsselverschlüsselungsschlüssel (KEK) zwischenAWSZahlungskryptographie und andere Systeme, die asymmetrische Schlüsselpaare verwenden, gefolgt von elektronischen Mitteln wie TR-34 (Method For Distribution Of Symmetric Keys Using Asymmetric Techniques).

Sie können Ihre verwendenAWSVerschlüsselungsschlüssel für Zahlungen bei kryptografischen Vorgängen, wie z. B.:

- Daten mit symmetrischen oder asymmetrischen Daten verschlüsseln, entschlüsseln und erneut verschlüsselnAWSVerschlüsselungsschlüssel für Zahlungen.
- Übersetzen Sie vertrauliche Daten (wie die PINS von Karteninhabern) sicher zwischen Verschlüsselungsschlüsseln, ohne den Klartext gemäß den PCI-PIN-Regeln preiszugeben.
- Generieren oder validieren Sie Karteninhaberdaten wie CVV, CVV2 oder ARQC.
- Generieren und validieren Sie Pins für Karteninhaber.
- Generieren oder validieren Sie MAC-Signaturen.

Konzepte

Lernen Sie die grundlegenden Begriffe und Konzepte der AWS Zahlungskryptografie kennen und erfahren Sie, wie Sie sie zum Schutz Ihrer Daten verwenden können.

Alias

> Ein benutzerfreundlicher Name, der mit einem AWS Zahlungskryptografie-Schlüssel verknüpft ist. Der Alias kann in vielen AWS Payment Cryptography API-Vorgängen synonym mit dem Schlüssel-ARN verwendet werden. Mithilfe von Aliasen können Schlüssel rotiert oder auf andere Weise geändert werden, ohne dass sich dies auf Ihren Anwendungscode auswirkt. Der

Aliasname besteht aus einer Zeichenfolge mit bis zu 256 Zeichen. Es identifiziert eindeutig einen zugehörigen AWS Zahlungskryptografie-Schlüssel innerhalb eines Kontos und einer Region. In der AWS Zahlungskryptografie beginnen Aliasnamen immer mit. `alias/`

Das Format eines Aliasnamens lautet wie folgt:

```
alias/<alias-name>
```

Beispielsweise:

```
alias/sampleAlias2
```

Schüssel-ARN

Der Schlüssel-ARN ist der Amazon-Ressourcenname (ARN) eines Schlüsseleintrags in AWS Payment Cryptography. Es handelt sich um eine eindeutige, vollqualifizierte Kennung für den AWS Payment Cryptography Key. Ein Schlüssel-ARN umfasst eine AWS-Konto Region und eine zufällig generierte ID. Der ARN steht nicht im Zusammenhang mit dem Schlüsselmaterial oder leitet sich davon ab. Da sie bei Erstellungs- oder Importvorgängen automatisch zugewiesen werden, sind diese Werte nicht idempotent. Das mehrfache Importieren desselben Schlüssels führt zu mehreren Schlüssel-ARNs mit eigenem Lebenszyklus.

Das Format eines Schlüssel-ARN lautet wie folgt:

```
arn:<partition>:payment-cryptography:<region>:<account-id>:alias/<alias-name>
```

Im Folgenden finden Sie ein Beispiel für einen Schlüssel-ARN:

```
arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h
```

Schlüssel-ID

Eine Schlüssel-ID ist ein Verweis auf einen Schlüssel, und einer (oder mehrere) von ihnen sind typische Eingaben für AWS kryptografische Zahlungsvorgänge. Gültige Schlüsselkennungen können entweder ein Schlüssel oder ein Schlüsselalias sein.

AWS Kryptografie-Schlüssel für Zahlungen

AWS Kryptografie-Schlüssel (Schlüssel) für Zahlungen werden für alle kryptografischen Funktionen verwendet. Schlüssel werden entweder direkt von Ihnen mit dem Befehl create

key generiert oder dem System hinzugefügt, indem Sie den Schlüsselimport aufrufen. Der Ursprung eines Schlüssels kann anhand des Attributs bestimmt werden KeyOrigin. AWS Die Zahlungskryptografie unterstützt auch abgeleitete Schlüssel oder Zwischenschlüssel, die bei kryptografischen Vorgängen verwendet werden, wie sie beispielsweise von DUKPT verwendet werden.

Diese Schlüssel haben sowohl unveränderliche als auch veränderbare Attribute, die bei der Erstellung definiert wurden. Attribute wie Algorithmus, Länge und Verwendung werden bei der Erstellung definiert und können nicht geändert werden. Andere, wie z. B. das Gültigkeitsdatum oder das Ablaufdatum, können geändert werden. Eine vollständige Liste der Schlüsselattribute für AWS Zahlungskryptografie finden Sie in der API-Referenz für AWS Zahlungskryptografie.

AWS Für Schlüssel zur Zahlungskryptografie gibt es Schlüsseltypen, die hauptsächlich durch ANSI X9 TR 31 definiert sind. Sie beschränken ihre Verwendung auf den in PCI PIN v3.1 Requirement 19 festgelegten Verwendungszweck.

Attribute werden mithilfe von Schlüsselblöcken an Schlüssel gebunden, wenn sie gespeichert, mit anderen Konten geteilt oder exportiert werden, wie in PCI-PIN v3.1 Anforderung 18-3 spezifiziert.

Schlüssel werden auf der AWS Payment Cryptography Platform anhand eines eindeutigen Werts identifiziert, der als Amazon Resource Name (ARN) als Schlüssel bezeichnet wird.

> **Note**
>
> Der Schlüssel ARN wird generiert, wenn ein Schlüssel zum ersten Mal erstellt oder in den AWS Payment Cryptography Service importiert wird. Wenn also dasselbe Schlüsselmaterial mithilfe der Schlüsselimportfunktion mehrmals hinzugefügt wird, befindet sich dasselbe Schlüsselmaterial unter mehreren Schlüsseln, die jedoch jeweils einen anderen Schlüssellebenszyklus haben.

Branchenterminologie

Themen

Allgemeine Schlüsseltypen

AWK

Ein Acquirer Working Key (AWK) ist ein Schlüssel, der typischerweise für den Datenaustausch zwischen einem Acquirer/Acquirer-Prozessor und einem Netzwerk (wie Visa oder Mastercard) verwendet wird. In der Vergangenheit nutzte AWK 3DES zur Verschlüsselung und wurde als TR31_P0_PIN_ENCRYPTION_KEY dargestellt.

BDK

Ein Basisableitungsschlüssel (Base Derivation Key, BDK) ist ein funktionierender Schlüssel, der häufig als Teil des PCI-PIN- und PCI P2PE-DUKPT-Prozesses verwendet wird. Er wird als TR31_B0_BASE_DERIVATION_KEY bezeichnet.

CMK

Ein Kartenhauptschlüssel (CMK) ist ein oder mehrere kartenspezifische Schlüssel, die in der Regel von einem Issuer Master Key, PAN und PSN abgeleitet werden und sind in der Regel 3DES-Schlüssel. Diese Schlüssel werden während der Personalisierung auf dem EMV-Chip gespeichert. Beispiele für CMKs sind AC-, SMI- und SMC-Schlüssel.

CMK-AC

Ein Anwendungskryptogrammschlüssel (AC) wird als Teil von EMV-Transaktionen verwendet, um das Transaktionskryptogramm zu generieren. Dabei handelt es sich um eine Art Kartenhauptschlüssel.

CMK-SMI

Ein SMI-Schlüssel (Secure Messaging Integrity) wird als Teil von EMV verwendet, um die Integrität von Payloads zu überprüfen, die über MAC an die Karte gesendet werden, z. B. von Pin-Aktualisierungsskripten. Es handelt sich um eine Art Hauptschlüssel für Karten.

CMK-SMC

Ein SMC-Schlüssel (Secure Messaging Confidentiality) wird als Teil von EMV verwendet, um an die Karte gesendete Daten zu verschlüsseln, z. B. PIN-Updates. Es handelt sich um eine Art Hauptschlüssel für Karten.

CVK

Ein Kartenverifikationsschlüssel (CVK) ist ein Schlüssel, der zur Generierung von CVV-, CVV2- und ähnlichen Werten unter Verwendung eines definierten Algorithmus sowie zur Validierung einer Eingabe verwendet wird. Er wird als TR31_C0_CARD_VERIFICATION_KEY bezeichnet.

iCVV

iCVV ist ein CVV2-ähnlicher Wert, der jedoch in die Track2-äquivalenten Daten auf einer EMV-Karte (Chip) eingebettet ist. Dieser Wert wird anhand des Servicecodes 999 berechnet und unterscheidet sich vom CVV1/CVV2, um zu verhindern, dass gestohlene Informationen verwendet werden, um neue Zahlungsinformationen eines anderen Typs zu erstellen. Wenn beispielsweise Chip-Transaktionsdaten abgerufen wurden, ist es nicht möglich, diese Daten zur Generierung eines Magnetstreifens (CVV1) oder für Online-Einkäufe (CVV2) zu verwenden.

Es verwendet einen Schlüssel ???

IMK

Ein Issuer Master Key (IMK) ist ein Masterschlüssel, der im Rahmen der Personalisierung von EMV-Chipkarten verwendet wird. In der Regel gibt es 3 IMKs — jeweils einen für AC- (Kryptogramm), SMI- (Skript-Masterschlüssel für Integrität/Signatur) und SMC-Schlüssel (Skript-Masterschlüssel für Vertraulichkeit/Verschlüsselung).

IK

Ein Initialschlüssel (IK) ist der erste Schlüssel, der im DUKPT-Prozess verwendet wird. Er wird vom Base Derivation Key (BDK) abgeleitet. Mit diesem Schlüssel werden keine Transaktionen verarbeitet, aber er wird verwendet, um future Schlüssel abzuleiten, die für Transaktionen verwendet werden. Die Ableitungsmethode für die Erstellung eines IK wurde in X9. 24-1:2017 definiert. Wenn ein TDES-BDK verwendet wird, ist X9. 24-1:2009 der geltende Standard und IK wird durch den Initial Pin Encryption Key (IPEK) ersetzt.

IPEK

Ein initialer PIN-Verschlüsselungsschlüssel (IPEK) ist der erste Schlüssel, der im DUKPT-Prozess verwendet wird. Er wird vom Base Derivation Key (BDK) abgeleitet. Mit diesem Schlüssel werden keine Transaktionen verarbeitet, aber er wird verwendet, um future Schlüssel abzuleiten, die für Transaktionen verwendet werden. IPEK ist eine Fehlbezeichnung, da dieser Schlüssel auch zur Ableitung von Datenverschlüsselung und Mac-Schlüsseln verwendet werden kann. Die Ableitungsmethode zur Erstellung eines IPEK wurde in X9. 24-1:2009 definiert. Wenn ein AES-BDK verwendet wird, ist X9. 24-1:2017 der geltende Standard und IPEK wird durch Initial Key (IK) ersetzt.

IWK

Ein Issuer Working Key (IWK) ist ein Schlüssel, der typischerweise für den Datenaustausch zwischen einem Emittenten/Emittenten und einem Netzwerk (wie Visa oder Mastercard)

verwendet wird. In der Vergangenheit nutzte IWK 3DES zur Verschlüsselung und wurde als TR31_P0_PIN_ENCRYPTION_KEY dargestellt.

KEK

Ein Key Encryption Key (KEK) ist ein Schlüssel, mit dem andere Schlüssel entweder für die Übertragung oder Speicherung verschlüsselt werden. Schlüssel, die zum Schutz anderer Schlüssel bestimmt sind, haben laut Standard in der Regel den Wert TR31_K0_KEY_ENCRYPTION_KEY. KeyUsage TR-31

SPITZE

Ein PIN-Verschlüsselungsschlüssel (PEK) ist eine Art Arbeitsschlüssel, der zur Verschlüsselung von PINs entweder für die Speicherung oder Übertragung zwischen zwei Parteien verwendet wird. IWK und AWK sind zwei Beispiele für spezifische Anwendungen von PIN-Verschlüsselungsschlüsseln. Diese Schlüssel werden als TR31_P0_PIN_ENCRYPTION_KEY dargestellt.

PVK

Ein PIN-Bestätigungsschlüssel (PVK) ist eine Art Arbeitsschlüssel, der zur Generierung von PIN-Bestätigungswerten wie PVV verwendet wird. Die beiden gängigsten Typen sind TR31_V1_IBM3624_PIN_VERIFICATION_KEY, der für die Generierung von IBM3624-Offsetwerten verwendet wird, und TR31_V2_VISA_PIN_VERIFICATION_KEY, der für Visa/ABA-Bestätigungswerte verwendet wird.

Andere Begriffe

ARQC

Das Authorization Request Cryptogram (ARQC) ist ein Kryptogramm, das während der Transaktion mit einer EMV-Standard-Chipkarte (oder einer gleichwertigen kontaktlosen Implementierung) generiert wird. In der Regel wird ein ARQC durch eine Chipkarte generiert und zur Überprüfung bei der Transaktion an einen Emittenten oder dessen Beauftragten weitergeleitet.

DUPT

Derived Unique Key Per Transaction (DUKPT) ist ein Schlüsselverwaltungsstandard, der in der Regel verwendet wird, um die Verwendung von Verschlüsselungsschlüsseln zur einmaligen Verwendung an physischen POS/POI zu definieren. In der Vergangenheit nutzte DUKPT 3DES zur Verschlüsselung. Der Industriestandard für DUKPT ist in ANSI X9.24-3-2017 definiert.

EMV

EMV (ursprünglich Europay, Mastercard, Visa) ist ein technisches Gremium, das mit Interessenträgern im Zahlungsverkehr zusammenarbeitet, um interoperable Zahlungsstandards und -technologien zu entwickeln. Ein Beispiel für Standards sind Chip-/kontaktlose Karten und die Zahlungsterminals, mit denen sie interagieren, einschließlich der verwendeten Kryptografie. Die EMV-Schlüsselableitung bezieht sich auf Verfahren zur Generierung eindeutiger Schlüssel für jede Zahlungskarte auf der Grundlage eines anfänglichen Schlüsselsatzes, wie z. B. IMK

HSM

Ein Hardware-Sicherheitsmodul (HSM) ist ein physisches Gerät, das kryptografische Operationen (z. B. Verschlüsselung, Entschlüsselung und digitale Signaturen) sowie die zugrunde liegenden Schlüssel, die für diese Operationen verwendet werden, schützt.

KCV

Key Check Value (KCV) bezieht sich auf eine Vielzahl von Prüfsummenmethoden, die hauptsächlich dazu verwendet werden, Schlüssel miteinander zu vergleichen, ohne Zugriff auf das eigentliche Schlüsselmaterial zu haben. KCV wurde auch für die Integritätsprüfung verwendet (insbesondere beim Austausch von Schlüsseln), obwohl diese Rolle jetzt Teil von Schlüsselblockformaten wie z. TR-31 Bei TDES-Schlüsseln wird der KCV berechnet, indem 8 Byte, jedes mit dem Wert Null, verschlüsselt werden, wobei der zu prüfende Schlüssel und die 3 höchstwertigen Byte des verschlüsselten Ergebnisses beibehalten werden. Bei AES-Schlüsseln wird der KCV mithilfe eines CMAC-Algorithmus berechnet, bei dem die Eingabedaten aus 16 Byte Null bestehen und die 3 Byte der höchsten Ordnung des verschlüsselten Ergebnisses beibehalten werden.

KDH

Ein Key Distribution Host (KDH) ist ein Gerät oder System, das Schlüssel in einem Schlüsselaustauschprozess wie TR-34 sendet. Beim Senden von Schlüsseln aus AWS Payment Cryptography wird dies als KDH betrachtet.

KIF

Eine Key Injection Facility (KIF) ist eine sichere Einrichtung zur Initialisierung von Zahlungsterminals, einschließlich des Ladens dieser mit Verschlüsselungsschlüsseln.

KRD

Ein Schlüsselempfangsgerät (KRD) ist ein Gerät, das Schlüssel im Rahmen eines Schlüsselaustauschprozesses wie TR-34 empfängt. Beim Senden von Schlüsseln an AWS Payment Cryptography wird es als KRD betrachtet.

KSN

Eine Schlüsselseriennummer (KSN) ist ein Wert, der als Eingabe für die DUKPT-Verschlüsselung/Entschlüsselung verwendet wird, um eindeutige Verschlüsselungsschlüssel pro Transaktion zu erstellen. Die KSN besteht in der Regel aus einer BDK-Kennung, einer halbeindeutigen Terminal-ID sowie einem Transaktionszähler, der bei jedem auf einem bestimmten Zahlungsterminal verarbeiteten Übergang inkrementiert wird.

PFANNE

Eine primäre Kontonummer (PAN) ist eine eindeutige Kennung für ein Konto, z. B. eine Kredit- oder Debitkarte. In der Regel 13 bis 19 Ziffern lang. Die ersten 6-8 Ziffern identifizieren das Netzwerk und die ausstellende Bank.

PIN-Block

Ein Datenblock, der während der Verarbeitung oder Übertragung eine PIN sowie andere Datenelemente enthält. PIN-Blockformate standardisieren den Inhalt des PIN-Blocks und die Art und Weise, wie er verarbeitet werden kann, um die PIN abzurufen. Die meisten PIN-Blocks bestehen aus der PIN und der PIN-Länge und enthalten häufig einen Teil oder die gesamte PAN. AWS Die Zahlungskryptografie unterstützt die ISO 9564-1-Formate 0, 1, 3 und 4. Format 4 ist für AES-Schlüssel erforderlich. Bei der Überprüfung oder Übersetzung von PINs muss der PIN-Block der eingehenden oder ausgehenden Daten angegeben werden.

POI

Point of Interaction (POI), auch häufig synonym mit Point of Sale (POS) verwendet, ist das Hardwaregerät, mit dem der Karteninhaber interagiert, um seine Zahlungsdaten vorzuzeigen. Ein Beispiel für einen POI ist das physische Terminal an einem Händlerstandort. Eine Liste der zertifizierten PCI-PTS-POI-Terminals finden Sie auf der PCI-Website.

PSN

Die PAN-Sequenznummer (PSN) ist ein numerischer Wert, der zur Unterscheidung mehrerer Karten verwendet wird, die mit derselben PAN ausgegeben wurden.

Öffentlicher Schlüssel

Bei der Verwendung von asymmetrischen Chiffren (RSA) ist der öffentliche Schlüssel die öffentliche Komponente eines öffentlich-privaten key pair. 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 von asymmetrischen Chiffren (RSA) ist der private Schlüssel die private Komponente eines öffentlich-privaten key pair. Mit dem privaten Schlüssel werden dann Daten entschlüsselt oder digitale Signaturen erstellt. Ähnlich wie bei symmetrischen Schlüsseln für die AWS Zahlungskryptografie werden private Schlüssel auf sichere Weise von HSMs erstellt. Sie werden nur in den flüchtigen Speicher des HSM entschlüsselt und nur für die Zeit, die für die Bearbeitung Ihrer kryptografischen Anfrage benötigt wird.

PVV

Der Pin Verification Value (PVV) ist ein Wert, der algorithmisch aus einer Reihe von Eingaben wie Kartennummer und PIN abgeleitet wird und einen Wert generiert, der für die nachfolgende Validierung verwendet werden kann. Ein solches Schema ist als Visa PVV (auch bekannt als ABA-Methode) bekannt, obwohl es für PINs in jedem Netzwerk verwendet wird.

RSA Wrap/Unwrap

RSA Wrap verwendet einen asymmetrischen Schlüssel, um einen symmetrischen Schlüssel (z. B. einen TDES-Schlüssel) für die Übertragung an ein anderes System zu umschließen. Nur das System mit dem passenden privaten Schlüssel kann die Nutzdaten entschlüsseln und den symmetrischen Schlüssel laden. Umgekehrt entschlüsselt RSA Unwrap einen mit RSA verschlüsselten Schlüssel sicher und lädt den Schlüssel dann in die Zahlungskryptografie. AWS RSA Wrap ist eine Low-Level-Methode für den Austausch von Schlüsseln, bei der Schlüssel nicht im Schlüsselblockformat übertragen werden und auch keine Payload-Signierung durch die sendende Partei verwendet wird. Alternative Kontrollen sollten in Betracht gezogen werden, um sicherzustellen, dass Providence und Schlüsselattribute nicht verändert sind.

TR-34 verwendet RSA auch intern, ist jedoch ein separates Format und nicht interoperabel.

TR-31

TR-31 (formal definiert als ANSI X9 TR 31) ist ein Schlüsselblockformat, das vom American National Standards Institute (ANSI) definiert wurde, um die Definition von Schlüsselattributen

in derselben Datenstruktur wie die Schlüsseldaten selbst zu unterstützen. Das TR-31-Schlüsselblockformat definiert eine Reihe von Schlüsselattributen, die an den Schlüssel gebunden sind, sodass sie zusammengehalten werden. AWS Die Zahlungskryptografie verwendet, wann immer möglich, standardisierte TR-31-Begriffe, um eine korrekte Trennung der Schlüssel und den Hauptzweck zu gewährleisten. TR-31 wurde durch ANSI X9.143-2022 ersetzt.

TR-34

TR-34 ist eine Implementierung von ANSI X9.24-2, in der ein Protokoll zur sicheren Verteilung symmetrischer Schlüssel (wie 3DES und AES) mithilfe asymmetrischer Techniken (wie RSA) beschrieben wurde. AWS Die Zahlungskryptografie verwendet TR-34-Methoden, um einen sicheren Import und Export von Schlüsseln zu ermöglichen.

Zugehörige Services

AWS Key Management Service

AWSSchlüsselverwaltungsdienst (AWSKMS) ist ein verwalteter Dienst, mit dem Sie die kryptografischen Schlüssel, die zum Schutz Ihrer Daten verwendet werden, auf einfache Weise erstellen und kontrollieren können.AWS KMS verwendet Hardware-Sicherheitsmodule (HSMs) zum Schutz und zur Validierung IhresAWSKMS-Schlüssel.

AWS CloudHSM

AWS CloudHSMbietet Kunden dedizierte Allzweck-HSM-Instances in derAWSWolke.AWS CloudHSMkann eine Vielzahl kryptografischer Funktionen bereitstellen, z. B. das Erstellen von Schlüsseln, das Signieren von Daten oder das Verschlüsseln und Entschlüsseln von Daten.

Weitere Informationen

- Um mehr über die Begriffe und Konzepte zu erfahren, die inAWSZahlungskryptografie, sieheAWSKonzepte der Zahlungskryptographie.
- Für Informationen über dieAWSAPI zur Steuerung der Zahlungskryptographie, sieheAWSAPI-Referenz zur Payment Cryptography Control Plane.
- Für Informationen über dieAWSAPI für Zahlungskryptographie auf Datenebene, sieheAWSAPI-Referenz zur Zahlungskryptografie auf Datenebene.

- Für detaillierte technische Informationen darüber, wieAWSZahlungskryptographie verwendet Kryptographie und schütztAWSVerschlüsselungsschlüssel für Zahlungen, siehe[Kryptografische Details](#).

Endpunkte für AWS Payment Cryptography

Um programmgesteuert eine Verbindung zu herzustellen AWS Payment Cryptography, verwenden Sie einen Endpunkt, die URL des Einstiegspunkts für den Service. Die - AWS SDKs und die Befehlszeilen-Tools verwenden automatisch den Standardendpunkt für den Service in einem AWS-Region , basierend auf dem Regionskontext einer Anforderung, sodass diese Werte in der Regel nicht explizit festgelegt werden müssen. Bei Bedarf können Sie einen anderen Endpunkt für Ihre API-Anforderungen angeben.

Endpunkte der Steuerebene

Name der Region	Region	Endpunkt	Protokoll
USA Ost (Nord-Virginia)	`us-east-1`	controlplane.payment-cryptography.us-east-1.amazonaws.com	HTTPS
USA Ost (Ohio)	`us-east-2`	controlplane.payment-cryptography.us-east-2.amazonaws.com	HTTPS
USA West (Oregon)	`us-west-2`	controlplane.payment-cryptography.us-west-2.amazonaws.com	HTTPS

Endpunkte der Datenebene

Name der Region	Region	Endpunkt	Protokoll
USA Ost (Nord-Virginia)	`us-east-1`	dataplane.payment-cryptography.us-east-1.amazonaws.com	HTTPS
USA Ost (Ohio)	`us-east-2`	dataplane.payment-cryptography.us-east-2.amazonaws.com	HTTPS

Name der Region	Region	Endpunkt	Protokoll
USA West (Oregon)	`us-west-2`	dataplane.payment-cryptography.us-west-2.amazonaws.com	HTTPS

Erste Schritte mit AWS Payment Cryptography

Um mit der AWS Zahlungskryptografie zu beginnen, sollten Sie zunächst Schlüssel erstellen und diese dann für verschiedene kryptografische Operationen verwenden. Das folgende Tutorial bietet einen einfachen Anwendungsfall für die Generierung eines Schlüssels, der zur Generierung/Überprüfung von CVV2-Werten verwendet werden soll. Um andere Beispiele auszuprobieren und Bereitstellungsmuster innerhalb von AWS zu untersuchen, probieren Sie bitte den folgenden AWSPayment Cryptography Workshop aus oder schauen Sie sich unser Beispielprojekt auf Github an

Dieses Tutorial führt Sie durch die Erstellung eines einzelnen Schlüssels und die Durchführung kryptografischer Operationen mit diesem Schlüssel. Anschließend löschen Sie den Schlüssel, wenn Sie ihn nicht mehr benötigen, wodurch der Schlüssellebenszyklus abgeschlossen ist.

Themen

Voraussetzungen

Bevor Sie beginnen, stellen Sie sicher, dass:

- Sie haben die Erlaubnis, auf den Dienst zuzugreifen. Weitere Informationen finden Sie unter IAM-Richtlinien.
- Sie haben das AWS CLIinstalliert. Sie können auch AWSSDKs oder AWSAPIs verwenden, um auf AWS Payment Cryptography zuzugreifen, aber die Anweisungen in diesem Tutorial verwenden die. AWS CLI

Schritt 1: Erstellen Sie einen Schlüssel

Der erste Schritt besteht darin, einen Schlüssel zu erstellen. In diesem Tutorial erstellen Sie einen CVK-3DES-Schlüssel mit doppelter Länge (2KEY TDES) zum Generieren und Überprüfen von CVV/CVV2-Werten.

```
$ aws payment-cryptography create-key \
    --exportable
    --key-attributes KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,\
        KeyClass=SYMMETRIC_KEY,\
        KeyModesOfUse='{Generate=true,Verify=true}'
```

Die Antwort gibt die Anforderungsparameter zurück, einschließlich eines ARN für nachfolgende Aufrufe sowie eines Key Check Value (KCV).

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_2KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
```

```
        "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
    }
}
```

Notieren Sie sich den Wert, der den Schlüssel darstelltKeyArn, zum Beispiel arn:aws:payment-cryptography:us-east - 2:111122223333:key/tqv5yij6wtxx64pi. Das benötigen Sie im nächsten Schritt.

Schritt 2: Generieren Sie mit dem Schlüssel einen CVV2-Wert

In diesem Schritt generieren Sie mithilfe des Schlüssels aus Schritt 1 einen CVV2-Wert für ein bestimmtes PAN Ablaufdatum.

```
$  aws payment-cryptography-data generate-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
    --generation-attributes CardVerificationValue2={CardExpiryDate=0123}
```

```
{
    "CardDataGenerationKeyCheckValue": "CADDA1",
    "CardDataGenerationKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi",
    "CardDataType": "CARD_VERIFICATION_VALUE_2",
    "CardDataValue": "144"
}
```

Notieren Sie sich diecardDataValue, in diesem Fall die 3-stellige Zahl 144. Das brauchst du im nächsten Schritt.

Schritt 3: Überprüfen Sie den in Schritt 2 generierten Wert

In diesem Beispiel validieren Sie den CVV2 aus Schritt 2 mit dem Schlüssel, den Sie in Schritt 1 erstellt haben.

Führen Sie den folgenden Befehl aus, um den CVV2 zu validieren.

```
$  aws payment-cryptography-data  verify-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
```

```
    --primary-account-number=171234567890123 \
    --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
    --validation-data 144
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1"
}
```

Der Dienst gibt eine HTTP-Antwort von 200 zurück, um anzuzeigen, dass er den CVV2 validiert hat.

Schritt 4: Führen Sie einen negativen Test durch

In diesem Schritt erstellen Sie einen negativen Test, bei dem der CVV2 nicht korrekt ist und nicht validiert wird. Sie versuchen, einen falschen CVV2 mit dem Schlüssel zu validieren, den Sie in Schritt 1 erstellt haben. Dies ist ein erwarteter Vorgang, z. B. wenn ein Karteninhaber beim Checkout den falschen CVV2 eingegeben hat.

```
$  aws payment-cryptography-data  verify-card-validation-data \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --primary-account-number=171234567890123 \
    --verification-attributes CardVerificationValue2={CardExpiryDate=0123} \
    --validation-data 999
```

```
Card validation data verification failed.
```

Der Dienst gibt eine HTTP-Antwort von 400 mit der Meldung „Überprüfung der Kartenvalidierungsdaten fehlgeschlagen“ und dem Grund INVALID_VALIDATION_DATA zurück.

Schritt 5: (Optional) Aufräumen

Jetzt können Sie den Schlüssel löschen, den Sie in Schritt 1 erstellt haben. Um die Anzahl der nicht wiederherstellbaren Änderungen zu minimieren, beträgt die Standardlöschfrist für Schlüssel sieben Tage.

```
$ aws payment-cryptography delete-key \
```

```
    --key-identifier=arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi
```


```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T08:27:51.795000-07:00",
        "DeletePendingTimestamp": "2022-11-03T13:37:12.114000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY"
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "DELETE_PENDING",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```


Notieren Sie sich zwei Felder in der Ausgabe. Die `deletePendingTimestamp` ist standardmäßig auf sieben Tage in der future festgelegt. Der KeyState ist auf eingestellt. `DELETE_PENDING` Sie können diesen Löschvorgang jederzeit vor dem geplanten Löschtermin stornieren, indem Sie anrufen. `restore-key`

Schlüssel verwalten

Um mit der AWS Zahlungskryptografie zu beginnen, sollten Sie einen AWS Zahlungskryptografie-Schlüssel erstellen.

In den Themen dieses Abschnitts wird erklärt, wie Sie eine Vielzahl von Schlüsseltypen für die AWS Zahlungskryptografie erstellen und verwalten, von der Erstellung bis zur Löschung. Es umfasst Themen zum Erstellen, Bearbeiten und Anzeigen von Schlüsseln, zum Markieren von Schlüsseln, zum Erstellen von Schlüsselaliasnamen sowie zum Aktivieren und Deaktivieren von Schlüsseln.

Themen

Generieren von Schlüsseln

Mithilfe der API-Operation können Sie Schlüssel für die AWS Zahlungskryptografie erstellen. CreateKey Während dieses Vorgangs geben Sie verschiedene Attribute des Schlüssels oder der resultierenden Ausgabe an, z. B. den Schlüsselalgorithmus (z. B. TDES_3KEY), den (zum Beispiel TR31_P0_PIN_ENCRYPTION_KEY), die zulässigen Operationen KeyUsage (z. B. Verschlüsseln, Signieren) und ob der Schlüssel exportierbar ist. Sie AWS können diese Eigenschaften nicht mehr ändern, nachdem der Payment Cryptography Key erstellt wurde.

Beispiele

- Generieren eines PVV-Schlüssels (PIN Verification Value)

Generieren eines 2KEY-TDES-Schlüssels

Example

Dieser Befehl generiert einen 2KEY-TDES-Schlüssel zum Generieren und Überprüfen von CVV/CVV2-Werten. Die Antwort gibt die Anforderungsparameter zurück, einschließlich eines ARN für nachfolgende Aufrufe sowie eines KCV (Key Check Value).

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,\
KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY, \
KeyModesOfUse='{Generate=true,Verify=true}'
```


```
                        {
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
hjprdg5o4jtgs5tw",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "B72F",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
```

```
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-26T16:04:11.559000-07:00"
    }
}
```

Generieren eines PIN-Verschlüsselungsschlüssels

Example Generieren eines PIN-Verschlüsselungsschlüssels (PEK)

Dieser Befehl generiert einen 3KEY-TDES-Schlüssel zum Verschlüsseln von PIN-Werten (bekannt als PIN-Verschlüsselungsschlüssel). Dieser Schlüssel kann zur Sicherung der Speicherung von PINs oder zum Entschlüsseln von PINs verwendet werden, die bei einem Überprüfungsversuch, beispielsweise während einer Transaktion, bereitgestellt wurden. Die Antwort gibt die Anforderungsparameter zurück, einschließlich eines ARN für nachfolgende Aufrufe sowie eines KCV (Key Check Value).

```
$ aws payment-cryptography create-key --exportable --key-attributes \
                       KeyAlgorithm=TDES_3KEY,KeyUsage=TR31_P0_PIN_ENCRYPTION_KEY, \
                       KeyClass=SYMMETRIC_KEY,/

 KeyModesOfUse='{Encrypt=true,Decrypt=true,Wrap=true,Unwrap=true}'
```

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T08:27:51.795000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
```

```
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY"
        },
        "KeyCheckValue": "9CA6",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```

Erstellen Sie einen asymmetrischen Schlüssel (RSA)

Example

In diesem Beispiel werden wir ein neues asymmetrisches RSA 2048-Bit-Schlüsselpaar generieren. Es werden ein neuer privater Schlüssel sowie der passende öffentliche Schlüssel generiert. Der öffentliche Schlüssel kann mit der PublicCertificate Get-API abgerufen werden.

```
$ aws payment-cryptography create-key --exportable \
--key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION, \
KeyClass=ASYMMETRIC_KEY_PAIR,KeyModesOfUse='{Encrypt=true,
 Decrypt=True,Wrap=True,Unwrap=True}'
```

```
{
    "Key": {
        "CreateTimestamp": "2022-11-15T11:15:42.358000-08:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
nsq2i3mbg6sn775f",
        "KeyAttributes": {
            "KeyAlgorithm": "RSA_2048",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyModesOfUse": {
```

```
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION"
        },
        "KeyCheckValue": "40AD487F",
        "KeyCheckValueAlgorithm": "CMAC",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-11-15T11:15:42.182000-08:00"
    }
}
```

Generieren eines PVV-Schlüssels (PIN Verification Value)

Example

Dieser Befehl generiert einen 3KEY-TDES-Schlüssel zum Generieren von PVV-Werten (bekannt als PIN-Bestätigungswert). Sie können diesen Schlüssel verwenden, um einen PVV-Wert zu generieren, der mit einem anschließend berechneten PVV-Wert verglichen werden kann. Die Antwort gibt die Anforderungsparameter zurück, einschließlich eines ARN für nachfolgende Aufrufe sowie eines KCV (Key Check Value).

```
$ aws payment-cryptography create-key --exportable/
--key-attributes KeyAlgorithm=TDES_3KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,/
KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Generate=true,Verify=true}'
```

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T10:22:59.668000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
j4u4cmnzkelhc6yb",
```

```
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": true,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY"
        },
        "KeyCheckValue": "5132",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T10:22:59.614000-07:00"
    }
}
```

Schlüssel auflisten

List Keys zeigt eine Liste von Schlüsseln an, auf die der Anrufer in diesem Konto und in dieser Region zugreifen kann.

Example

```
$ aws payment-cryptography list-keys
```

```
 {"Keys": [
    {
        "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
        "Enabled": false,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
```

```
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStopTimestamp": "2022-10-27T14:19:42.488000-07:00"
    }
    ]
}
```

Aktivieren und Deaktivieren von Schlüsseln

Sie können Payment Cryptography Keys deaktivieren und wieder aktivieren AWS . Wenn Sie einen Schlüssel erstellen, ist er standardmäßig aktiviert. Wenn Sie einen Schlüssel deaktivieren, kann er erst dann für kryptografische Operationen verwendet werden, wenn Sie ihn erneut aktivieren. Befehle zum Starten/Beenden der Verwendung werden sofort wirksam. Es wird daher empfohlen, die Verwendung zu überprüfen, bevor Sie eine solche Änderung vornehmen. Mit dem optionalen `timestamp` Parameter können Sie auch festlegen, dass eine Änderung (Nutzung starten oder beenden) in der future wirksam wird.

Da die Deaktivierung eines AWS Zahlungskryptografie-Schlüssels temporär ist und leicht rückgängig gemacht werden kann, ist sie eine sicherere Alternative zum Löschen eines AWS Zahlungskryptografie-Schlüssels, eine Aktion, die destruktiv und irreversibel ist. Wenn Sie erwägen, einen AWS Payment Cryptography Key zu löschen, deaktivieren Sie ihn zunächst und stellen Sie sicher, dass Sie den Schlüssel in future nicht mehr zum Verschlüsseln oder Entschlüsseln von Daten verwenden müssen.

Themen

Starten Sie die Schlüsselnutzung

Die Schlüsselverwendung muss aktiviert sein, um einen Schlüssel für kryptografische Operationen verwenden zu können. Wenn ein Schlüssel nicht aktiviert ist, können Sie diesen Vorgang verwenden, um ihn nutzbar zu machen. Das Feld `UsageStartTimestamp` gibt an, wann der Schlüssel aktiv wurde/wird. Dies gilt für ein aktiviertes Token in der Vergangenheit und in der future, wenn die Aktivierung noch aussteht.

Example

In diesem Beispiel wird die Aktivierung eines Schlüssels für die Schlüsselverwendung angefordert. Die Antwort enthält die wichtigsten Informationen und das Aktivierungs-Flag wurde auf „true“ umgestellt. Dies wird sich auch im Antwortobjekt list-keys widerspiegeln.

```
$ aws payment-cryptography start-key-usage --key-identifier "arn:aws:payment-
cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh"
```

```
{
      "Key": {
      "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
      "Enabled": true,
      "Exportable": true,
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
      "KeyAttributes": {
          "KeyAlgorithm": "TDES_3KEY",
          "KeyClass": "SYMMETRIC_KEY",
          "KeyModesOfUse": {
              "Decrypt": true,
              "DeriveKey": false,
              "Encrypt": true,
              "Generate": false,
              "NoRestrictions": false,
              "Sign": false,
              "Unwrap": true,
```

```
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T14:09:59.468000-07:00"
    }
}
```

Beenden Sie die Verwendung der Schlüssel

Wenn Sie nicht mehr vorhaben, einen Schlüssel zu verwenden, können Sie die Schlüsselverwendung beenden, um weitere kryptografische Operationen zu verhindern. Dieser Vorgang ist nicht permanent, sodass Sie ihn rückgängig machen können, indem Sie die Verwendung des Startschlüssels verwenden. Sie können auch festlegen, dass ein Schlüssel in future deaktiviert wird. Das Feld `UsageStopTimestamp` gibt an, wann der Schlüssel deaktiviert wurde/wird.

Example

In diesem Beispiel wird es aufgefordert, die Verwendung von Schlüsseln in future einzustellen. Nach der Ausführung kann dieser Schlüssel nicht für kryptografische Operationen verwendet werden, es sei denn, er wird über die Verwendung des Startschlüssels erneut aktiviert. Die Antwort enthält die Schlüsselinformationen und das Aktivierungskennzeichen wurde auf „Falsch“ gesetzt. Dies wird sich auch im Antwortobjekt list-Keys widerspiegeln.

```
$ aws payment-cryptography stop-key-usage --key-identifier "arn:aws:payment-
cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh"
```

```
{
    "Key": {
        "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
        "Enabled": false,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
```

```
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_P1_PIN_GENERATION_KEY"
        },
        "KeyCheckValue": "369D",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStopTimestamp": "2022-10-27T14:09:59.468000-07:00"
    }
}
```

Löschen von Schlüsseln

Durch das Löschen eines AWS Zahlungskryptografie-Schlüssels werden das Schlüsselmaterial und alle mit dem Schlüssel verknüpften Metadaten gelöscht. Dies ist irreversibel, sofern keine Kopie des Schlüssels außerhalb von AWS Payment Cryptography verfügbar ist. Nach dem Löschen eines Schlüssels können Sie die mit diesem Schlüssel verschlüsselten Daten nicht mehr entschlüsseln, was bedeutet, dass Daten möglicherweise nicht mehr wiederhergestellt werden können. Sie sollten einen Schlüssel nur löschen, wenn Sie sicher sind, dass Sie ihn nicht mehr benötigen und keine anderen Parteien diesen Schlüssel verwenden. Wenn Sie sich nicht sicher sind, sollten Sie erwägen, den Schlüssel zu deaktivieren, anstatt ihn zu löschen. Sie können einen deaktivierten Schlüssel wieder aktivieren, wenn Sie ihn später erneut verwenden müssen, aber Sie können einen gelöschten AWS Payment Cryptography Key nur wiederherstellen, wenn Sie ihn aus einer anderen Quelle erneut importieren können.

Bevor Sie einen Schlüssel löschen, sollten Sie sicherstellen, dass Sie den Schlüssel nicht mehr benötigen. AWS Die Zahlungskryptografie speichert nicht die Ergebnisse kryptografischer Operationen wie CVV2 und kann nicht feststellen, ob ein Schlüssel für persistentes kryptografisches Material benötigt wird.

AWS Payment Cryptography löscht niemals Schlüssel, die zu aktiven AWS Konten gehören, es sei denn, Sie planen ausdrücklich deren Löschung und die vorgeschriebene Wartezeit läuft ab.

Sie können sich jedoch aus einem oder mehreren der folgenden Gründe dafür entscheiden, einen AWS Zahlungskryptografie-Schlüssel zu löschen:

- Um den Schlüssellebenszyklus für einen Schlüssel abzuschließen, den Sie nicht mehr benötigen
- Um den Verwaltungsaufwand zu vermeiden, der mit der Aufbewahrung ungenutzter AWS Payment Cryptography Keys verbunden ist

> **Note**
>
> Wenn Sie Ihren schließen oder löschen AWS-Konto, kann nicht mehr auf Ihren AWS Payment Cryptography Key zugegriffen werden. Sie müssen die Löschung Ihres AWS Payment Cryptography Keys nicht getrennt von der Schließung des Kontos planen.

AWS Die Zahlungskryptografie zeichnet einen Eintrag in Ihrem AWS CloudTrailProtokoll auf, wenn Sie das Löschen des AWS Zahlungskryptografie-Schlüssels planen und wenn der AWS Zahlungskryptografie-Schlüssel tatsächlich gelöscht wird.

Über die Wartezeit

Da das Löschen eines Schlüssels irreversibel ist, müssen Sie für die AWS Zahlungskryptografie eine Wartezeit zwischen 3 und 180 Tagen festlegen. Die standardmäßige Wartezeit beträgt sieben Tage.

Die tatsächliche Wartezeit kann jedoch bis zu 24 Stunden länger sein als die, die Sie geplant haben. Um das tatsächliche Datum und die Uhrzeit zu ermitteln, zu der der AWS Payment Cryptography Key gelöscht wird, verwenden Sie die folgenden GetKey Operationen. Achten Sie darauf, die Zeitzone zu notieren.

Während der Wartezeit lautet der Status und der Schlüsselstatus des AWS Zahlungskryptografie-Schlüssels „Ausstehende Löschung“.

> **Note**
>
> Ein AWS Zahlungskryptografie-Schlüssel, dessen Löschung noch aussteht, kann für keine kryptografischen Operationen verwendet werden.

Nach Ablauf der Wartezeit löscht AWS Payment Cryptography den Zahlungskryptografie-Schlüssel, seine Aliase und alle zugehörigen AWS Zahlungskryptografie-Metadaten. AWS

Nutzen Sie die Wartezeit, um sicherzustellen, dass Sie den AWS Payment Cryptography Key weder jetzt noch in future benötigen. Wenn Sie feststellen, dass Sie den Schlüssel während der Wartezeit benötigen, können Sie die Löschung des Schlüssels vor Ablauf der Wartezeit abbrechen. Nach Ablauf der Wartezeit können Sie das Löschen des Schlüssels nicht mehr abbrechen und der Dienst löscht den Schlüssel.

Example

In diesem Beispiel wird die Löschung eines Schlüssels angefordert. Neben den grundlegenden Schlüsselinformationen geben zwei relevante Felder an, dass der Schlüsselstatus in DELETE_PENDING geändert wurde, und geben an deletePendingTimestamp, wann der Schlüssel aktuell gelöscht werden soll.

```
$ aws payment-cryptography delete-key \
                    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

```
  {
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_3KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
```

```
        },
        "KeyCheckValue": "",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": false,
        "Exportable": true,
        "KeyState": "DELETE_PENDING",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-05T12:01:29.969000-07:00",
        "UsageStopTimestamp": "2023-06-05T14:31:13.399000-07:00",
        "DeletePendingTimestamp": "2023-06-12T14:58:32.865000-07:00"
    }
}
```

Example

In diesem Beispiel wird ein ausstehender Löschvorgang storniert. Nach erfolgreichem Abschluss wird ein Schlüssel nicht mehr gemäß dem vorherigen Zeitplan gelöscht. Die Antwort enthält die grundlegenden Schlüsselinformationen. Darüber hinaus wurden zwei relevante Felder geändert - `KeyState` und`deletePendingTimestamp`. `KeyState`wird auf den Wert CREATE_COMPLETE zurückgegeben und gleichzeitig `DeletePendingTimestamp` entfernt.

```
$ aws payment-cryptography restore-key --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_V2_VISA_PIN_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_3KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
```

```
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": false,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-08T12:01:29.969000-07:00",
        "UsageStopTimestamp": "2023-06-08T14:31:13.399000-07:00"
    }
}
```

Schlüssel importieren und exportieren

AWS Kryptografie-Schlüssel für Zahlungen können aus anderen Lösungen importiert oder in andere Lösungen (z. B. andere HSMs) exportiert werden. Es ist ein häufiger Anwendungsfall, Schlüssel mithilfe von Import- und Exportfunktionen mit Dienstanbietern auszutauschen. Als Cloud-Dienst verfolgt AWS Payment Cryptography einen modernen, elektronischen Ansatz für die Schlüsselverwaltung und hilft Ihnen gleichzeitig dabei, die geltenden Vorschriften und Kontrollen aufrechtzuerhalten. Langfristiges Ziel ist die Abkehr von papiergestützten Schlüsselkomponenten hin zu standardbasierten, elektronischen Mitteln für den Schlüsselaustausch.

Austausch von Schlüsselverschlüsselungsschlüsseln (KEK)

AWS Die Zahlungskryptografie fördert die Verwendung von Public-Key-Kryptografie (RSA) für den ersten Schlüsselaustausch unter Verwendung der etablierten ANSI X9.24 TR-34-Norm. Zu den gebräuchlichen Bezeichnungen für diesen ersten Schlüsseltyp gehören Key Encryption Key (KEK), Zone Master Key (ZMK) und Zone Control Master Key (ZCMK). Wenn Ihre Systeme oder Partner TR-34 noch nicht unterstützen können, können Sie auch die Verwendung von RSA Wrap/Unwrap in Betracht ziehen.

Wenn Sie die Verarbeitung von Schlüsselkomponenten auf paper fortsetzen müssen, bis alle Partner den elektronischen Schlüsselaustausch unterstützen, können Sie erwägen, zu diesem Zweck ein Offline-HSM beizubehalten.

> **Note**
>
> Wenn Sie Ihre eigenen Testschlüssel importieren möchten, schauen Sie sich bitte das Beispielprojekt auf Github an. Anweisungen zum Importieren/Exportieren von Schlüsseln von anderen Plattformen finden Sie im Benutzerhandbuch für diese Plattformen.

Working Key (WK) Exchange

AWS Die Zahlungskryptografie verwendet die entsprechende Industrienorm (ANSI X9.24 TR 31-2018) für den Austausch funktionierender Schlüssel. TR-31 geht davon aus, dass zuvor ein KEK ausgetauscht wurde. Dies entspricht der Anforderung der PCI-PIN, Schlüsselmaterial jederzeit kryptografisch an seinen Schlüsseltyp und seine Verwendung zu binden. Arbeitsschlüssel haben verschiedene Namen, darunter Acquirer-Arbeitsschlüssel, Issuer-Arbeitsschlüssel, BDK, IPEK usw.

Themen

Schlüssel importieren

> **Important**
>
> Für Beispiele ist möglicherweise die neueste Version von AWS CLI V2 erforderlich. Bevor Sie beginnen, stellen Sie bitte sicher, dass Sie auf die neueste Version aktualisiert haben.

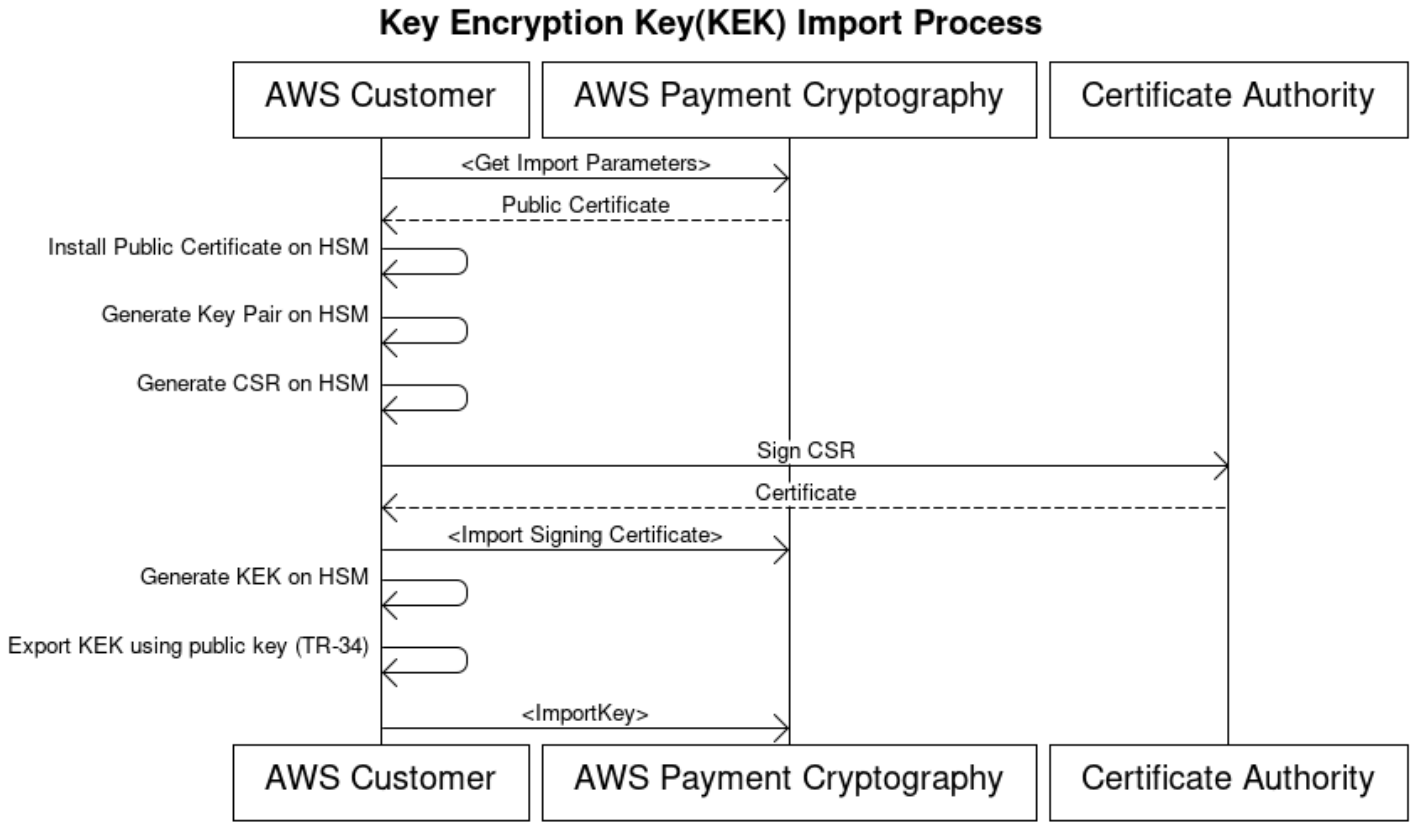
Themen

Symmetrische Schlüssel importieren

Themen

- Importieren Sie Schlüssel mithilfe asymmetrischer Techniken (RSA Unwrap)
- Importieren Sie symmetrische Schlüssel mithilfe eines vorab festgelegten Schlüsselaustauschschlüssels (TR-31)

Schlüssel mithilfe asymmetrischer Techniken importieren (TR-34)



Überblick: TR-34 verwendet asymmetrische RSA-Kryptografie, um symmetrische Schlüssel für den Austausch zu verschlüsseln und die Herkunft der Daten sicherzustellen (Signierung). Dadurch werden sowohl die Vertraulichkeit (Verschlüsselung) als auch die Integrität (Signatur) des verpackten Schlüssels gewährleistet.

Wenn Sie Ihre eigenen Schlüssel importieren möchten, schauen Sie sich bitte das Beispielprojekt auf Github an. Anweisungen zum Importieren/Exportieren von Schlüsseln von anderen Plattformen finden Sie im Benutzerhandbuch für diese Plattformen.

1. Rufen Sie den Befehl initialize import auf

   Rufen Sie `get-parameters-for-import` auf, um den Importvorgang zu initialisieren. Diese API generiert ein Schlüsselpaar für Schlüsselimporte, signiert den Schlüssel und gibt das Zertifikat und den Zertifikatsstamm zurück. Letztlich sollte der zu exportierende Schlüssel mit

diesem Schlüssel verschlüsselt werden. In der TR-34-Terminologie wird dies als KRD-Zertifikat bezeichnet. Beachten Sie, dass diese Zertifikate nur von kurzer Dauer sind und nur für diesen Zweck bestimmt sind.

2. Installieren Sie das öffentliche Zertifikat auf dem Schlüsselquellsystem

   Bei vielen HSMs müssen Sie möglicherweise das in Schritt 1 generierte öffentliche Zertifikat installieren/laden/als vertrauenswürdig einstufen, um damit Schlüssel exportieren zu können.

3. Generieren Sie einen öffentlichen Schlüssel und stellen Sie den Zertifikatsstamm für Payment Cryptography bereit AWS

   Um die Integrität der übertragenen Nutzdaten zu gewährleisten, wird sie von der sendenden Partei signiert (bekannt als Key Distribution Host oder KDH). Die sendende Partei möchte zu diesem Zweck einen öffentlichen Schlüssel generieren und anschließend ein Public-Key-Zertifikat (X509) erstellen, das an Payment Cryptography zurückgegeben werden kann. AWS AWS Private CA ist eine Option zum Generieren von Zertifikaten, es gibt jedoch keine Einschränkungen hinsichtlich der verwendeten Zertifizierungsstelle.

   Sobald Sie das Zertifikat haben, sollten Sie das Stammzertifikat mithilfe der `importKey` Befehle and KeyMaterialType of `ROOT_PUBLIC_KEY_CERTIFICATE` und KeyUsageType of `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` in AWS Payment Cryptography laden.

4. Schlüssel aus dem Quellsystem exportieren

   Viele HSMs und verwandte Systeme unterstützen die Möglichkeit, Schlüssel gemäß der TR-34-Norm zu exportieren. Sie sollten den öffentlichen Schlüssel aus Schritt 1 als KRD-Zertifikat (Verschlüsselung) und den Schlüssel aus Schritt 3 als KDH-Zertifikat (Signierzertifikat) angeben. Für den Import in AWS Payment Cryptography sollten Sie das Format TR-34.2012 angeben, das kein CMS-Format mit zwei Durchgängen ist. Dieses Format kann auch als TR-34 Diebold-Format bezeichnet werden.

5. Rufen Sie den Importschlüssel auf

   Als letzten Schritt rufen Sie die ImportKey-API mit einem KeyMaterialType von `TR34_KEY_BLOCK` auf. Das `certificate-authority-public-key-identifier` ist der KeyARN der in Schritt 3 importierten Root-CA, das Schlüsselmaterial aus Schritt 4 `key-material` wird verpackt und `signing-key-certificate` ist das Leaf-Zertifikat aus Schritt 3. Sie müssen auch das Import-Token aus Schritt 1 angeben.

6. Verwenden Sie den importierten Schlüssel für kryptografische Operationen oder nachfolgenden Import

   Wenn der importierte Schlüssel TR31_K0_KEY_ENCRYPTION_KEY KeyUsage war, kann dieser Schlüssel für nachfolgende Schlüsselimporte mit TR-31 verwendet werden. Wenn der Schlüsseltyp ein anderer Typ war (z. B. TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), kann der Schlüssel direkt für kryptografische Operationen verwendet werden.

Importieren Sie Schlüssel mithilfe asymmetrischer Techniken (RSA Unwrap)

Überblick: AWS Payment Cryptography unterstützt RSA Wrap/Unwrap für den Schlüsselaustausch, wenn TR-34 nicht möglich ist. Ähnlich wie bei TR-34 verwendet diese Technik asymmetrische RSA-Kryptografie, um symmetrische Schlüssel für den Austausch zu verschlüsseln. Im Gegensatz zu TR-34 ist bei dieser Methode die Nutzdaten jedoch nicht von der sendenden Partei signiert. Außerdem gewährleistet diese RSA-Wrap-Technik nicht die Integrität der Schlüsselmetadaten während der Übertragung, da keine Schlüsselblöcke enthalten sind.

> **Note**
>
> RSA Wrap kann verwendet werden, um TDES- und AES-128-Schlüssel zu importieren oder zu exportieren.

1. Rufen Sie den Befehl initialize import auf

   Rufen Sie `get-parameters-for-import` auf, um den Importvorgang mit dem Schlüsselmaterialtyp KEY_CRYPTOGRAM zu initialisieren. WrappingKeyAlgorithm kann beim Austausch von TDES-Schlüsseln RSA_2048 sein. RSA_3072 oder RSA_4096 können beim Austausch von TDES- oder AES-128-Schlüsseln verwendet werden. Diese API generiert ein Schlüsselpaar für Schlüsselimporte, signiert den Schlüssel mit einem Zertifikatsstamm und gibt sowohl das Zertifikat als auch den Zertifikatsstamm zurück. Letztlich sollte der zu exportierende Schlüssel mit diesem Schlüssel verschlüsselt werden. Beachten Sie, dass diese Zertifikate nur von kurzer Dauer sind und nur für diesen Zweck bestimmt sind.

```
$ aws payment-cryptography get-parameters-for-import --key-material-type
 KEY_CRYPTOGRAM --wrapping-key-algorithm RSA_4096
```

```
{
    "ImportToken": "import-token-bwxli6ocftypneu5",
    "ParametersValidUntilTimestamp": 1698245002.065,
    "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0....",
    "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0....",
    "WrappingKeyAlgorithm": "RSA_4096"
}
```

2. Installieren Sie das öffentliche Zertifikat auf dem Schlüsselquellsystem

   Bei vielen HSMs müssen Sie möglicherweise das in Schritt 1 generierte öffentliche Zertifikat (und/oder sein Stammzertifikat) installieren/laden/als vertrauenswürdig einstufen, um damit Schlüssel exportieren zu können.

3. Schlüssel aus dem Quellsystem exportieren

   Viele HSMs und verwandte Systeme unterstützen die Möglichkeit, Schlüssel mithilfe von RSA Wrap zu exportieren. Sie sollten den öffentlichen Schlüssel aus Schritt 1 als (Verschlüsselungs-) Zertifikat (WrappingKeyZertifikat) angeben. Wenn Sie die Vertrauenskette benötigen, ist diese im Antwortfeld WrappingKeyCertificateChain in Schritt #1 enthalten. Wenn Sie den Schlüssel aus Ihrem HSM exportieren, sollten Sie als Format RSA, Padding Mode = PKCS #1 v2.2 OAEP (mit SHA 256 oder SHA 512) angeben.

4. Rufen Sie den Importschlüssel auf

   Als letzten Schritt rufen Sie die ImportKey-API mit einem KeyMaterialType von `KeyMaterial` auf. Sie benötigen das Import-Token aus Schritt 1 und das `key-material` (verpackte Schlüsselmaterial) aus Schritt 3. Sie müssen die wichtigsten Parameter (z. B. die Schlüsselverwendung) angeben, da RSA Wrap keine Schlüsselblöcke verwendet.

```
$ cat import-key-cryptogram.json
              {
                  "KeyMaterial": {
                      "KeyCryptogram": {
                          "Exportable": true,
                          "ImportToken": "import-token-bwxli6ocftypneu5",
                          "KeyAttributes": {
                              "KeyAlgorithm": "AES_128",
                              "KeyClass": "SYMMETRIC_KEY",
                              "KeyModesOfUse": {
                                  "Decrypt": true,
                                  "DeriveKey": false,
```

```
                                "Encrypt": true,
                                "Generate": false,
                                "NoRestrictions": false,
                                "Sign": false,
                                "Unwrap": true,
                                "Verify": false,
                                "Wrap": true
                            },
                            "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY"
                        },
                        "WrappedKeyCryptogram": "18874746731....",
                        "WrappingSpec": "RSA_OAEP_SHA_256"
                    }
                }
            }
```

```
$ aws payment-cryptography import-key --cli-input-json file://import-key-
cryptogram.json
```

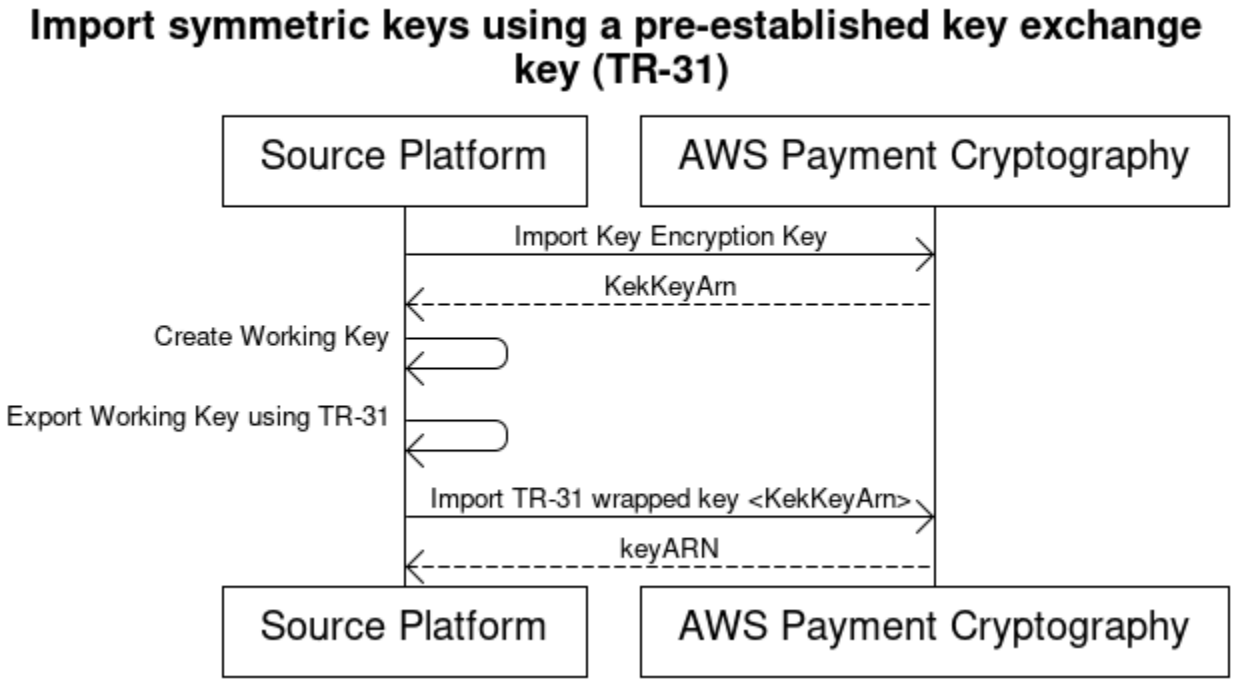
```
{
      "Key": {
          "KeyOrigin": "EXTERNAL",
          "Exportable": true,
          "KeyCheckValue": "DA1ACF",
          "UsageStartTimestamp": 1697643478.92,
          "Enabled": true,
          "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
          "CreateTimestamp": 1697643478.92,
          "KeyState": "CREATE_COMPLETE",
          "KeyAttributes": {
              "KeyAlgorithm": "AES_128",
              "KeyModesOfUse": {
                  "Encrypt": true,
                  "Unwrap": true,
                  "Verify": false,
                  "DeriveKey": false,
                  "Decrypt": true,
                  "NoRestrictions": false,
                  "Sign": false,
                  "Wrap": true,
                  "Generate": false
```

```
            },
            "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY",
            "KeyClass": "SYMMETRIC_KEY"
        },
        "KeyCheckValueAlgorithm": "CMAC"
    }
}
```

5. Verwenden Sie den importierten Schlüssel für kryptografische Operationen oder nachfolgenden Import

> Wenn der importierte Schlüssel TR31_K0_KEY_ENCRYPTION_KEY KeyUsage war, kann dieser Schlüssel für nachfolgende Schlüsselimporte mit TR-31 verwendet werden. Wenn der Schlüsseltyp ein anderer Typ war (z. B. TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), kann der Schlüssel direkt für kryptografische Operationen verwendet werden.

Importieren Sie symmetrische Schlüssel mithilfe eines vorab festgelegten Schlüsselaustauschschlüssels (TR-31)



Wenn Partner mehrere Schlüssel austauschen (oder um die Schlüsselrotation zu unterstützen), ist es üblich, zunächst einen Initial Key Encryption Key (KEK) unter Verwendung von Techniken wie Papierschlüsselkomponenten oder im Fall von AWS Zahlungskryptografie mithilfe von TR-34 auszutauschen.

Sobald ein KEK eingerichtet ist, können Sie diesen Schlüssel verwenden, um nachfolgende Schlüssel (einschließlich anderer KEK) zu transportieren. AWS Die Zahlungskryptografie unterstützt diese Art des Schlüsselaustauschs mithilfe von ANSI TR-31, das weit verbreitet ist und von HSM-Anbietern weitgehend unterstützt wird.

1. Schlüsselverschlüsselungsschlüssel (KEK) importieren

   Es wird davon ausgegangen, dass Sie Ihren KEK bereits importiert haben und dass Ihnen der KeyARN (oder KeyAlias) zur Verfügung steht.
2. Schlüssel auf der Quellplattform erstellen

   Wenn der Schlüssel noch nicht existiert, erstellen Sie den Schlüssel auf der Quellplattform. Umgekehrt können Sie den Schlüssel in AWS Payment Cryptography erstellen und stattdessen den `export` Befehl verwenden.
3. Exportieren Sie den Schlüssel von der Quellplattform

   Achten Sie beim Exportieren darauf, dass Sie als Exportformat TR-31 angeben. Die Quellplattform fragt Sie außerdem nach dem Schlüssel, der exportiert werden soll, und nach dem zu verwendenden Schlüssel.
4. In AWS Payment Cryptography importieren

   Wenn Sie den ImportKey-Befehl aufrufen, WrappingKeyIdentifier sollte dies der KeyARN (oder Alias) Ihres Schlüsselverschlüsselungsschlüssels sein und WrappedKeyBlock die Ausgabe von der Quellplattform sein.

   Example

```
$ aws payment-cryptography import-key \
            --key-material="Tr31KeyBlock={WrappingKeyIdentifier="arn:aws:payment-
cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza",\

 WrappedKeyBlock="D0112B0AX00E00002E0A3D58252CB67564853373D1EBCC1E23B2ADE7B15E967CC27B85D599
```

```
     {
       "Key": {
           "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
           "KeyAttributes": {
```

```
                "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyAlgorithm": "AES_128",
                "KeyModesOfUse": {
                    "Encrypt": true,
                    "Decrypt": true,
                    "Wrap": true,
                    "Unwrap": true,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": false,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "0A3674",
            "KeyCheckValueAlgorithm": "CMAC",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "EXTERNAL",
            "CreateTimestamp": "2023-06-02T07:38:14.913000-07:00",
            "UsageStartTimestamp": "2023-06-02T07:38:14.857000-07:00"
        }
    }
```

Importiert asymmetrische (RSA) Schlüssel

Öffentliche RSA-Schlüssel importieren

AWS Payment Cryptography unterstützt den Import von öffentlichen RSA-Schlüsseln in Form von X.509-Zertifikaten. Um ein Zertifikat zu importieren, müssen Sie zuerst das Stammzertifikat importieren. Alle Zertifikate sollten zum Zeitpunkt des Imports noch nicht abgelaufen sein. Das Zertifikat sollte im PEM-Format vorliegen und Base64-codiert sein.

1. In das Stammzertifikat in Payment Cryptography importieren AWS

   Example

```
$ aws payment-cryptography import-key \
  --key-material='{"RootCertificatePublicKey":{"KeyAttributes":
{"KeyAlgorithm":"RSA_2048", \
```

```
  "KeyClass":"PUBLIC_KEY", "KeyModesOfUse":{"Verify":
 true},"KeyUsage":"TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"}, \

 "PublicKeyCertificate":"LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURKVENDQWcyZ0F3SUJBZ0lCWkR
```

```
  {
    "Key": {
        "CreateTimestamp": "2023-08-08T18:52:01.023000+00:00",
        "Enabled": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
zabouwe3574jysdl",
        "KeyAttributes": {
            "KeyAlgorithm": "RSA_2048",
            "KeyClass": "PUBLIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": false,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
        },
        "KeyOrigin": "EXTERNAL",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2023-08-08T18:52:01.023000+00:00"
    }
}
```

2. Importieren Sie das Public-Key-Zertifikat in die AWS Zahlungskryptografie

   Sie können jetzt einen öffentlichen Schlüssel importieren. Es gibt zwei Möglichkeiten, öffentliche Schlüssel zu importieren. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`kann verwendet werden, wenn der Schlüssel dazu dient, Signaturen zu verifizieren (z. B. beim Import mit TR-34). `TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`kann beim Verschlüsseln von Daten verwendet werden, die für die Verwendung mit einem anderen System bestimmt sind.

Example

```
$ aws payment-cryptography import-key \
    --key-material='{"TrustedCertificatePublicKey":
{"CertificateAuthorityPublicKeyIdentifier":"arn:aws:payment-cryptography:us-
east-2:111122223333:key/zabouwe3574jysdl", \
    "KeyAttributes":
{"KeyAlgorithm":"RSA_2048","KeyClass":"PUBLIC_KEY","KeyModesOfUse":
{"Verify":true},"KeyUsage":"TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"},\
    "PublicKeyCertificate":"LS0tLS1CRUdJTiB..."}}'
```


```
{
  "Key": {
      "CreateTimestamp": "2023-08-08T18:55:46.815000+00:00",
      "Enabled": true,
      "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/4kd6xud22e64wcbk",
      "KeyAttributes": {
          "KeyAlgorithm": "RSA_4096",
          "KeyClass": "PUBLIC_KEY",
          "KeyModesOfUse": {
              "Decrypt": false,
              "DeriveKey": false,
              "Encrypt": false,
              "Generate": false,
              "NoRestrictions": false,
              "Sign": false,
              "Unwrap": false,
              "Verify": true,
              "Wrap": false
          },
          "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
      },
      "KeyOrigin": "EXTERNAL",
      "KeyState": "CREATE_COMPLETE",
      "UsageStartTimestamp": "2023-08-08T18:55:46.815000+00:00"
  }
}
```

Schlüssel exportieren

Themen

Exportieren symmetrischer Schlüssel

> **Important**
>
> Für Beispiele ist möglicherweise die neueste Version von AWS CLI V2 erforderlich. Bevor Sie beginnen, stellen Sie bitte sicher, dass Sie auf die neueste Version aktualisiert haben.

Themen

Exportieren Sie Schlüssel mithilfe asymmetrischer Techniken (TR-34)

Überblick: TR-34 verwendet asymmetrische RSA-Kryptografie, um symmetrische Schlüssel für den Austausch zu verschlüsseln und die Herkunft der Daten sicherzustellen (Signierung). Dadurch werden sowohl die Vertraulichkeit (Verschlüsselung) als auch die Integrität (Signatur) des verpackten Schlüssels gewährleistet. Beim Export wird AWS Payment Cryptography zum Key Distribution Host (KDH) und das Zielsystem zum Key Receiving Device (KRD).

1. Rufen Sie den Befehl „Export initialisieren“ auf

   Rufen Sie `get-parameters-for-export` auf, um den Exportvorgang zu initialisieren. Diese API generiert ein Schlüsselpaar für Schlüsselexporte, signiert den Schlüssel und gibt das Zertifikat und den Zertifikatsstamm zurück. Letztlich wurde der durch diesen Befehl generierte private Schlüssel zum Signieren der Exportnutzdaten verwendet. In der TR-34-Terminologie wird dies als KDH-Signaturzertifikat bezeichnet. Beachten Sie, dass diese Zertifikate nur von kurzer Dauer sind

und nur für diesen Zweck bestimmt sind. Der Parameter ParametersValidUntilTimestamp gibt ihre Dauer an.

HINWEIS: Alle Zertifikate werden in einem Base64-codierten Format zurückgegeben

Example

```
$ aws payment-cryptography get-parameters-for-export \
                    --signing-key-algorithm RSA_2048 --key-material-type
 TR34_KEY_BLOCK
```

```
{
              "SigningKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJBZ0lRZFAzSzNHNEFKT0I4WTNpTmUvYl
              "SigningKeyCertificateChain":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUY0VENDQThtZ0F3SUJBZ0lSQUtlN2piaHFKZjJPd3FGUWI5c3
              "SigningKeyAlgorithm": "RSA_2048",
              "ExportToken": "export-token-au7pvkbsq4mbup6i",
              "ParametersValidUntilTimestamp": "2023-06-13T15:40:24.036000-07:00"
}
```

2. Importieren Sie das AWS Payment Cryptography Zertifikat in das Empfangssystem

Importieren Sie die in Schritt 1 angegebene Zertifikatskette nach Bedarf in Ihr Empfangssystem.

3. Generieren Sie ein key pair, erstellen Sie ein öffentliches Zertifikat und stellen Sie den Zertifikatsstamm für AWS Payment Cryptography bereit

Um die Vertraulichkeit der übertragenen Nutzdaten zu gewährleisten, wird sie von der sendenden Partei verschlüsselt (bekannt als Key Distribution Host oder KDH). Die empfangende Partei (in der Regel Ihr HSM oder das HSM Ihrer Partner) möchte zu diesem Zweck einen öffentlichen Schlüssel generieren und anschließend ein Public-Key-Zertifikat (x.509) erstellen, das an Payment Cryptography zurückgegeben werden kann. AWS AWS Private CA ist eine Option zum Generieren von Zertifikaten, aber es gibt keine Einschränkungen hinsichtlich der verwendeten Zertifizierungsstelle.

Sobald Sie das Zertifikat haben, sollten Sie das Stammzertifikat mithilfe der `ImportKey` Befehle and KeyMaterialType of `ROOT_PUBLIC_KEY_CERTIFICATE` und KeyUsageType of `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` in AWS Payment Cryptography laden.

Das KeyUsageType Zertifikat ist TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE, weil es der Stammschlüssel ist und zum Signieren des Leaf-Zertifikats verwendet wird. Leaf-Zertifikate für den Import/Export werden nicht in Payment Cryptography importiert, sondern inline übergeben. AWS

> **Note**
>
> Wenn das Stammzertifikat zuvor importiert wurde, kann dieser Schritt übersprungen werden.

4. Rufen Sie den Exportschlüssel auf

Als letzten Schritt rufen Sie die ExportKey API mit einem KeyMaterialType von aufTR34_KEY_BLOCK. `certificate-authority-public-key-identifier`Dies `export-key-identifier` ist der KeyARN des Root-CA-Imports in Schritt 3, `WrappingKeyCertificate` das Leaf-Zertifikat aus Schritt 3 und der KeyARN (oder Alias), der exportiert werden soll. Sie müssen auch das Export-Token aus Schritt 1 angeben.

Exportieren Sie Schlüssel mithilfe asymmetrischer Techniken (RSA Wrap)

Überblick: AWS Payment Cryptography unterstützt RSA Wrap/Unwrap für den Schlüsselaustausch, wenn TR-34 von der Gegenpartei nicht verfügbar ist. Ähnlich wie bei TR-34 verwendet diese Technik asymmetrische RSA-Kryptografie, um symmetrische Schlüssel für den Austausch zu verschlüsseln. Im Gegensatz zu TR-34 ist bei dieser Methode die Nutzdaten jedoch nicht von der sendenden Partei signiert. Außerdem beinhaltet diese RSA-Wrap-Technik keine Schlüsselblöcke, die verwendet werden, um die Integrität der Schlüsselmetadaten während des Transports aufrechtzuerhalten.

> **Note**
>
> RSA Wrap kann zum Exportieren von TDES- und AES-128-Schlüsseln verwendet werden.

1. Generieren Sie einen RSA-Schlüssel und ein Zertifikat auf dem Empfangssystem

Erstellen (oder identifizieren) Sie einen RSA-Schlüssel, der für den Empfang des verpackten Schlüssels verwendet wird. AWS Payment Cryptography erwartet Schlüssel im X.509-Zertifikatsformat. Das Zertifikat sollte mit einem Stammzertifikat signiert werden, das in AWS Payment Cryptography importiert wurde (oder importiert werden kann).

2. Installieren Sie das öffentliche Stammzertifikat für AWS Payment Cryptography

```
$ aws payment-cryptography  import-key --key-material='{"RootCertificatePublicKey":
{"KeyAttributes":{"KeyAlgorithm":"RSA_4096","KeyClass":"PUBLIC_KEY","KeyModesOfUse":
{"Verify":
 true},"KeyUsage":"TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"},"PublicKeyCertificate":"LS
```

```
    {
        "Key": {
        "CreateTimestamp": "2023-09-14T10:50:32.365000-07:00",
        "Enabled": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
nsq2i3mbg6sn775f",
        "KeyAttributes": {
        "KeyAlgorithm": "RSA_4096",
        "KeyClass": "PUBLIC_KEY",
        "KeyModesOfUse": {
        "Decrypt": false,
        "DeriveKey": false,
        "Encrypt": false,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": false,
        "Verify": true,
        "Wrap": false
        },
        "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"
        },
        "KeyOrigin": "EXTERNAL",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2023-09-14T10:50:32.365000-07:00“
        }
       }
```

3. Rufen Sie den Exportschlüssel auf

Als Nächstes möchten Sie AWS Payment Cryptography anweisen, Ihren Schlüssel mithilfe Ihres Leaf-Zertifikats zu exportieren. Sie geben den ARN für das zuvor importierte Stammzertifikat, das für den Export zu verwendende Blattzertifikat und den symmetrischen Schlüssel für den

Export an. Die Ausgabe wird eine hexadezimale, binär umhüllte (verschlüsselte) Version Ihres symmetrischen Schlüssels sein.

```
$ cat export-key.json
```

```
  {
      "ExportKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi",
      "KeyMaterial": {
          "KeyCryptogram": {
              "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-2:111122223333:key/zabouwe3574jysdl",
              "WrappingKeyCertificate": "LS0tLS1CRUdJTiBD...",
              "WrappingSpec": "RSA_OAEP_SHA_256"
          }
      }
  }
```

```
$ aws payment-cryptography export-key --cli-input-json file://export-key.json
```

```
{
      "WrappedKey": {
          "KeyMaterial":
 "18874746731E9E1C4562E4116D1C2477063FCB08454D757D81854AEAEE0A52B1F9D303FA29C02DC82AE7785353
          "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM"
      }
  }
```

4. Schlüssel in das Empfangssystem importieren

   Viele HSMs und verwandte Systeme unterstützen die Möglichkeit, Schlüssel mithilfe von RSA Unwrap (einschließlich AWS Payment Cryptography) zu importieren. Geben Sie dazu den öffentlichen Schlüssel aus Schritt 1 als (Verschlüsselungs-) Zertifikat an. Das Format sollte als RSA, Padding Mode = PKCS #1 v2.2 OAEP (mit SHA 256) angegeben werden. Die genaue Terminologie kann je nach HSM variieren.

> **Note**
>
> AWS Payment Cryptography gibt den verpackten Schlüssel in HexBinary aus. Möglicherweise müssen Sie das Format vor dem Import konvertieren, wenn Ihr System eine andere binäre Darstellung wie Base64 benötigt.

Exportieren Sie symmetrische Schlüssel mithilfe eines vorab festgelegten Schlüsselaustauschschlüssels (TR-31)

Wenn Partner mehrere Schlüssel austauschen (oder um die Schlüsselrotation zu unterstützen), ist es üblich, zunächst einen Initial Key Encryption Key (KEK) unter Verwendung von Techniken wie Papierschlüsselkomponenten oder im Fall von AWS Zahlungskryptografie mithilfe von TR-34 auszutauschen. Sobald ein KEK eingerichtet ist, können Sie diesen Schlüssel verwenden, um nachfolgende Schlüssel (einschließlich anderer KEK) zu transportieren. AWS Die Zahlungskryptografie unterstützt diese Art des Schlüsselaustauschs mithilfe von ANSI TR-31, das weit verbreitet ist und von HSM-Anbietern weitgehend unterstützt wird.

1. Exchange Key Encryption Key (KEK)

   Es wird davon ausgegangen, dass Sie Ihren KEK bereits ausgetauscht haben und dass Ihnen der KeyARN (oder KeyAlias) zur Verfügung steht.

2. Schlüssel zur Zahlungskryptografie AWS erstellen

   Wenn der Schlüssel noch nicht existiert, erstellen Sie ihn. Umgekehrt können Sie den Schlüssel auf dem anderen System erstellen und stattdessen den Importbefehl verwenden.

3. Exportieren Sie den Schlüssel aus AWS Payment Cryptography

   Beim Exportieren wird das Format TR-31 verwendet. Beim Aufrufen der API geben Sie den zu exportierenden Schlüssel und den zu verwendenden Wrapping-Schlüssel an.

```
$ aws payment-cryptography export-key --key-material='{"Tr31KeyBlock":
 {"WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza"}}' --export-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp
```

```
{
            "WrappedKey": {
```

```
                "KeyCheckValue": "73C263",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "KeyMaterial":
"D0144K0AB00E0000A24D3ACF3005F30A6E31D533E07F2E1B17A2A003B338B1E79E5B3AD4FBF7850FACF9A37844
                "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK"
            }
    }
```

4. In Ihr System importieren

   Sie oder Ihr Partner verwenden die Implementierung des Importschlüssels auf Ihrem System, um den Schlüssel zu importieren.

Exportieren Sie die DUKPT-Anfangsschlüssel (IPEK/IK)

Bei Verwendung von DUKPT kann ein einziger Base Derivation Key (BDK) für eine Flotte von Terminals generiert werden. Terminals haben jedoch nie Zugriff auf diesen ursprünglichen BDK, sondern erhalten jeweils einen eindeutigen, anfänglichen Terminalschlüssel, der als IPEK oder Initial Key (IK) bezeichnet wird. Jeder IPEK ist ein vom BDK abgeleiteter Schlüssel, der für jedes Terminal eindeutig sein soll, aber vom ursprünglichen BDK abgeleitet ist. Die Ableitungsdaten für diese Berechnung werden als Key Serial Number (KSN) bezeichnet. Gemäß X9.24 besteht das 10-Byte-KSN für TDES typischerweise aus 24 Bit für die Schlüsselsatz-ID, 19 Bit für die Terminal-ID und 21 Bit für den Transaktionszähler. Bei AES besteht das 12-Byte-KSN typischerweise aus 32 Bit für die BDK-ID, 32 Bit für den Derivation Identifier (ID) und 32 Bit für den Transaktionszähler.

AWS Die Zahlungskryptografie bietet einen Mechanismus zum Generieren und Exportieren dieser Anfangsschlüssel. Nach der Generierung können diese Schlüssel mit den Methoden TR-31, TR-34 und RSA Wrap exportiert werden. IPEK-Schlüssel werden nicht dauerhaft gespeichert und können nicht für nachfolgende Operationen zur Zahlungskryptografie verwendet werden AWS

AWS Die Zahlungskryptografie erzwingt die Aufteilung zwischen den ersten beiden Teilen des KSN nicht. Wenn Sie den Ableitungsbezeichner zusammen mit dem BDK speichern möchten, können Sie zu diesem Zweck die AWS Tags-Funktion verwenden.

> **Note**
>
> Der Zählerteil des KSN (32 Bit für AES DUKPT) wird nicht für die IPEK/IK-Ableitung verwendet. Daher gibt eine Eingabe von 12345678901234560001 und 12345678901234569999 dasselbe IPEK aus.

```
$ aws payment-cryptography export-key --key-material='{"Tr31KeyBlock":
 {"WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza"}}' --export-key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi --export-attributes
 'ExportDukptInitialKey={KeySerialNumber=12345678901234560001}'
```

```
{
    "WrappedKey": {
        "KeyCheckValue": "73C263",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyMaterial":
 "B0096B1TX00S000038A8A06588B9011F0D5EEF1CCAECFA6962647A89195B7A98BDA65DDE7C57FEA507559AF2A5D60
        "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK"
    }
}
```

Exportieren von asymmetrischen (RSA) Schlüsseln

Rufen Sie `get-public-key-certificate` auf, um einen öffentlichen Schlüssel in Zertifikatsform zu exportieren. Diese API exportiert das Zertifikat sowie sein Stammzertifikat, das im Base64-Format codiert ist.

HINWEIS: Diese API ist nicht idempotent — nachfolgende Aufrufe können zu unterschiedlichen Zertifikaten führen, obwohl der zugrunde liegende Schlüssel derselbe ist.

Example

```
$ aws payment-cryptography get-public-key-certificate \
         –key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5dza7xqd6soanjtb
```

```
{
        "KeyCertificate": "LS0tLS1CRUdJTi...",
        "KeyCertificateChain": "LS0tLS1CRUdJT...“
}
```

Verwenden von Aliassen

Ein Alias ist ein benutzerfreundlicher Name für einen AWS Payment Cryptography Key. Mit einem Alias können Sie beispielsweise auf einen Schlüssel als `alias/test-key` statt auf verweisen. `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`

Sie können einen Alias verwenden, um einen Schlüssel bei den meisten Schlüsselverwaltungsvorgängen (Steuerungsebene) und bei kryptografischen Vorgängen (Datenebene) zu identifizieren.

Sie können auch den Zugriff auf AWS Payment Cryptography Keys anhand ihrer Aliase zulassen oder verweigern, ohne Richtlinien bearbeiten oder Zuschüsse verwalten zu müssen. Diese Funktion ist Teil der Unterstützung des Dienstes für die attributebasierte Zugriffskontrolle (ABAC).

Ein Großteil der Leistungsfähigkeit von Aliasen beruht auf Ihrer Fähigkeit, den mit einem Alias verknüpften Schlüssel jederzeit zu ändern. Aliasse machen Ihren Code einfacher zu schreiben und zu verwalten. Nehmen wir zum Beispiel an, Sie verwenden einen Alias, um auf einen bestimmten AWS Zahlungskryptografie-Schlüssel zu verweisen, und Sie möchten den AWS Zahlungskryptografie-Schlüssel ändern. In diesem Fall ordnen Sie den Alias einfach einem anderen Schlüssel zu. Sie müssen Ihren Code oder Ihre Anwendungskonfiguration nicht ändern.

Darüber hinaus erleichtern Aliasse die Wiederverwendung des gleichen Codes in verschiedenen AWS-Regionen. Erstellen Sie Aliase mit demselben Namen in mehreren Regionen und verknüpfen Sie jeden Alias mit einem AWS Payment Cryptography Key in seiner Region. Wenn der Code in jeder Region ausgeführt wird, bezieht sich der Alias auf den zugehörigen AWS Payment Cryptography Key in dieser Region.

Mithilfe der API können Sie einen Alias für einen AWS Payment Cryptography Key erstellen. `CreateAlias`

Die AWS Payment Cryptography API bietet die vollständige Kontrolle über Aliase in jedem Konto und jeder Region. Die API umfasst Operationen zum Erstellen eines Alias (CreateAlias), zum Anzeigen von Aliasnamen und dem verknüpften KeyARN (Listen-Aliase), zum Ändern des mit einem Alias verknüpften AWS Zahlungskryptografie-Schlüssels (Update-Alias) und zum Löschen eines Alias (Delete-Alias).

Themen

- Verbundene APIs

Über Aliasse

Erfahren AWS Sie, wie Aliase in der Zahlungskryptografie funktionieren.

Ein Alias ist eine unabhängige Ressource AWS

Ein Alias ist kein Eigentum eines AWS Zahlungskryptografie-Schlüssels. Die Aktionen, die Sie mit dem Alias ausführen, wirken sich nicht auf den zugehörigen Schlüssel aus. Sie können einen Alias für einen AWS Payment Cryptography Key erstellen und dann den Alias so aktualisieren, dass er einem anderen AWS Payment Cryptography Key zugeordnet ist. Sie können den Alias sogar löschen, ohne dass dies Auswirkungen auf den zugehörigen AWS Payment Cryptography Key hat. Wenn Sie einen AWS Payment Cryptography Key löschen, wird die Zuweisung aller mit diesem Schlüssel verknüpften Aliase aufgehoben.

Wenn Sie in einer IAM-Richtlinie einen Alias als Ressource angeben, bezieht sich die Richtlinie auf den Alias, nicht auf den zugehörigen AWS Payment Cryptography Key.

Jeder Alias hat einen benutzerfreundlichen Namen

Wenn Sie einen Alias erstellen, geben Sie den Aliasnamen mit dem Präfix an`alias/`. Zum Beispiel `alias/test_1234`

Jeder Alias ist jeweils einem AWS Payment Cryptography Key zugeordnet

Der Alias und sein AWS Payment Cryptography Key müssen sich im selben Konto und in derselben Region befinden.

Ein AWS Payment Cryptography Key kann mehreren Alias gleichzeitig zugeordnet werden, aber jeder Alias kann nur einem einzigen Schlüssel zugeordnet werden

Diese `list-aliases` Ausgabe zeigt beispielsweise, dass der `alias/sampleAlias1` Alias genau einem Zielschlüssel für AWS Payment Cryptography zugeordnet ist, der durch die Eigenschaft repräsentiert wird. `KeyArn`

```
$ aws payment-cryptography list-aliases
```

```
{
```

```
    "Aliases": [
        {
            "AliasName": "alias/sampleAlias1",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
        }
    ]
}
```

Diesem AWS Payment Cryptography Schlüssel können mehrere Aliase zugeordnet werden

Sie können beispielsweise die `alias/sampleAlias2` Aliase `alias/sampleAlias1;` und demselben Schlüssel zuordnen.

```
$ aws payment-cryptography list-aliases
```

```
{
        "Aliases": [
            {
                "AliasName": "alias/sampleAlias1",
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
            },
            {
                "AliasName": "alias/sampleAlias2",
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
            }
        ]
    }
```

Ein Alias muss für ein bestimmtes Konto und eine bestimmte Region eindeutig sein

Zum Beispiel können Sie nur einen `alias/sampleAlias1`-Alias in jedem Konto und in jeder Region haben. Bei Aliasen wird Groß- und Kleinschreibung beachtet. Wir empfehlen jedoch, Aliase nicht zu verwenden, die sich nur in der Groß-/Kleinschreibung unterscheiden, da sie fehleranfällig sein können. Sie können keinen Aliasnamen ändern. Sie können den Alias jedoch löschen und einen neuen Alias mit dem gewünschten Namen erstellen.

Sie können jedoch einen Alias mit demselben Namen in verschiedenen Regionen erstellen.

Sie können beispielsweise einen Alias `alias/sampleAlias2` in USA Ost (Nord-Virginia) und einen Alias `alias/sampleAlias2` in USA West (Oregon) haben. Jeder Alias wäre mit einem AWS Payment Cryptography Key in seiner Region verknüpft. Wenn Ihr Code auf einen Aliasnamen wie `alias/finance-key` verweist, können Sie ihn in mehreren Regionen ausführen. In jeder Region wird ein anderer Alias/SampleAlias2 verwendet. Details hierzu finden Sie unter Verwenden von Aliassen in Ihren Anwendungen.

Sie können den AWS Payment Cryptography Key ändern, der einem Alias zugeordnet ist

Sie können diesen `UpdateAlias` Vorgang verwenden, um einen Alias mit einem anderen AWS Payment Cryptography Key zu verknüpfen. Wenn der `alias/sampleAlias2` Alias beispielsweise mit dem `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` AWS Payment Cryptography Key verknüpft ist, können Sie ihn so aktualisieren, dass er dem `arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi` Schlüssel zugeordnet ist.

> **Warning**
>
> AWS Mit der Zahlungskryptografie wird nicht überprüft, ob der alte und der neue Schlüssel dieselben Attribute haben, wie z. B. die Schlüsselverwendung. Die Aktualisierung mit einem anderen Schlüsseltyp kann zu Problemen in Ihrer Anwendung führen.

Einige Schlüssel haben keine Aliase

Ein Alias ist eine optionale Funktion, und nicht alle Schlüssel haben Aliase, es sei denn, Sie entscheiden sich dafür, Ihre Umgebung auf diese Weise zu betreiben. Schlüssel können mithilfe des Befehls mit Aliasnamen verknüpft werden. `create-alias` Sie können auch den Vorgang „Alias aktualisieren“ verwenden, um den mit einem Alias verknüpften AWS Payment Cryptography-Schlüssel zu ändern, und den Vorgang „Alias löschen“, um einen Alias zu löschen. Daher können einige Schlüssel zur AWS Zahlungskryptografie mehrere Aliase haben, andere wiederum keine.

Zuordnen eines Schlüssels zu einem Alias

Mit dem `create-alias` Befehl können Sie einen Schlüssel (dargestellt durch einen ARN) einem oder mehreren Aliasen zuordnen. Dieser Befehl ist nicht idempotent. Um einen Alias zu aktualisieren, verwenden Sie den Befehl update-alias.

```
$ aws payment-cryptography create-alias --alias-name alias/sampleAlias1 \
             --key-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h
```

```
{
    "Alias": {
        "AliasName": "alias/alias/sampleAlias1",
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
    }
}
```

Verwenden von Aliassen in Ihren Anwendungen

Sie können einen Alias verwenden, um einen AWS Payment Cryptography Key in Ihrem Anwendungscode darzustellen. Der `key-identifier` Parameter bei AWS Payment Cryptography-Datenoperationen sowie bei anderen Operationen wie List Keys akzeptiert einen Aliasnamen oder Alias-ARN.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier alias/
BIN_123456_CVK --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2={CardExpiryDate=0123}
```

Denken Sie bei der Verwendung eines Alias-ARN daran, dass die Aliaszuweisung zu einem AWS Payment Cryptography Key in dem Konto definiert ist, dem der AWS Payment Cryptography Key gehört, und dass sich die Alias-Zuordnung je nach Region unterscheiden kann.

Eine der mächtigsten Verwendungen von Aliasen ist in Anwendungen, die in mehreren AWS-Regionen ausgeführt werden.

Sie können in jeder Region eine andere Version Ihrer Anwendung erstellen oder ein Wörterbuch, eine Konfiguration oder eine Switch-Anweisung verwenden, um den richtigen AWS Payment Cryptography Key für jede Region auszuwählen. Es könnte jedoch einfacher sein, in jeder Region einen Alias mit

demselben Aliasnamen zu erstellen. Bei dem Aliasnamen wird zwischen Groß-/Kleinschreibung unterschieden.

Verbundene APIs

Tags

Tags sind Schlüssel- und Wertepaare, die als Metadaten für die Organisation Ihrer AWS Payment Cryptography Keys dienen. Sie können verwendet werden, um Schlüssel flexibel zu identifizieren oder einen oder mehrere Schlüssel zu gruppieren.

Holen Sie sich Schlüssel

Ein AWS Payment Cryptography Key stellt eine einzelne Einheit von kryptografischem Material dar und kann nur für kryptografische Operationen für diesen Dienst verwendet werden. Die GetKeys API verwendet a KeyIdentifier als Eingabe und gibt die unveränderlichen und veränderbaren Attribute des Schlüssels zurück, enthält jedoch kein kryptografisches Material.

Example

```
$ aws payment-cryptography get-key --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

```
{
  "Key": {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
      "KeyAttributes": {
          "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY",
          "KeyClass": "SYMMETRIC_KEY",
          "KeyAlgorithm": "AES_128",
          "KeyModesOfUse": {
              "Encrypt": true,
              "Decrypt": true,
              "Wrap": true,
              "Unwrap": true,
              "Generate": false,
```

```
            "Sign": false,
            "Verify": false,
            "DeriveKey": false,
            "NoRestrictions": false
        }
    },
    "KeyCheckValue": "0A3674",
    "KeyCheckValueAlgorithm": "CMAC",
    "Enabled": true,
    "Exportable": true,
    "KeyState": "CREATE_COMPLETE",
    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
    "CreateTimestamp": "2023-06-02T07:38:14.913000-07:00",
    "UsageStartTimestamp": "2023-06-02T07:38:14.857000-07:00"
  }
}
```

Rufen Sie den öffentlichen Schlüssel/das Zertifikat ab, das einem key pair zugeordnet ist

Get Public Key/Certificate gibt den öffentlichen Schlüssel zurück, der durch den gekennzeichnet ist. `KeyArn` Dies kann der öffentliche Schlüsselteil eines key pair sein, das mit AWS Payment Cryptography generiert wurde, oder ein zuvor importierter öffentlicher Schlüssel. Der häufigste Anwendungsfall ist die Bereitstellung des öffentlichen Schlüssels an einen externen Dienst, der Daten verschlüsselt. Diese Daten können dann an eine Anwendung weitergegeben werden, die AWS Zahlungskryptografie nutzt, und die Daten können mit dem privaten Schlüssel, der in der Zahlungskryptografie gesichert ist, entschlüsselt werden. AWS

Der Dienst gibt öffentliche Schlüssel als öffentliches Zertifikat zurück. Das API-Ergebnis enthält die CA und das Public-Key-Zertifikat. Beide Datenelemente sind Base64-codiert.

> **Note**
>
> Das zurückgegebene öffentliche Zertifikat soll kurzlebig und nicht idempotent sein. Möglicherweise erhalten Sie bei jedem API-Aufruf ein anderes Zertifikat, auch wenn der öffentliche Schlüssel selbst unverändert bleibt.

Example

```
$ aws payment-cryptography get-public-key-certificate  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/nsq2i3mbg6sn775f
```


```
{
  "KeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2VENDQXFXZ0F3SUJBZ0lSQUo1OWd2VkpDd3dlYldMNldYZEpYY
  "KeyCertificateChain":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUY0VENDQThtZ0F3SUJBZ0lSQUtlN2piaHFKZjJPd3FGUWI5c3VuC
}
```


Tagging von Schlüsseln

In AWS Zahlungskryptografie können Sie einem AWS Zahlungskryptografie-Schlüssel Tags hinzufügen, wenn Sie einen Schlüssel erstellen, und vorhandene Schlüssel kennzeichnen oder deren Markierung aufheben, sofern sie nicht gelöscht werden müssen. Tags sind optional, aber sie können sehr nützlich sein.

Allgemeine Informationen zu Stichwörtern, einschließlich bewährter Methoden, Tagging-Strategien sowie Format und Syntax von Stichwörtern, finden Sie unter Ressourcen AWS taggen in der. Allgemeine Amazon Web Services-Referenz

Themen

- Informationen zu Tags in der Zahlungskryptografie AWS
- Schlüssel-Tags in der Konsole anzeigen
- Verwaltung von Schlüssel-Tags mit API-Operationen
- Zugriffssteuerung mit Tags
- Verwendung von Tags zur Steuerung des Zugriffs auf Schlüssel

Informationen zu Tags in der Zahlungskryptografie AWS

Ein Tag ist ein optionales Metadaten-Label, das Sie einer AWS Ressource zuweisen (oder zuweisen AWS können). Jedes Tag besteht aus einem Tag-Schlüssel und einem Tag-Wert, wobei beide Zeichenfolgen zwischen Groß-/Kleinschreibung unterscheiden. Der Wert kann auch eine leere (Null) Zeichenfolge sein. Jedes Tag auf einer Ressource muss einen anderen Tag-Schlüssel haben, aber Sie können dasselbe Tag mehreren AWS Ressourcen hinzufügen. Eine Ressource kann bis zu 50 Tags besitzen, die von Benutzern erstellt wurden.

Geben Sie keine vertraulichen oder sensiblen Informationen in den Tag-Schlüssel oder Tag-Wert ein. Tags sind für viele zugänglich AWS-Services, auch für die Abrechnung.

In der AWS Zahlungskryptografie können Sie einem Schlüssel bei der Erstellung Tags hinzufügen und vorhandene Schlüssel kennzeichnen oder deren Markierung aufheben, sofern sie nicht gelöscht werden müssen. Sie können Aliase nicht taggen. Tags sind optional, aber sie können sehr nützlich sein.

Sie können beispielsweise allen AWS Payment Cryptography Keys und Amazon S3 S3-Buckets, die Sie für das Alpha-Projekt verwenden, ein `"Project"="Alpha"` Tag hinzufügen. Ein anderes Beispiel ist das Hinzufügen eines `"BIN"="20130622"` Tags zu allen Schlüsseln, die einer bestimmten Bank-Identifikationsnummer (BIN) zugeordnet sind.

```
    [
    {
      "Key": "Project",
      "Value": "Alpha"
    },
    {
      "Key": "BIN",
      "Value": "20130622"
    }
  ]
```

Allgemeine Informationen zu Tags, einschließlich Format und Syntax, finden Sie unter AWS Ressourcen taggen in der Allgemeine Amazon Web Services-Referenz.

Tags sind für folgende Aktivitäten nützlich:

- Identifizieren und organisieren Sie Ihre AWS Ressourcen. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Sie können beispielsweise einem AWS Payment Cryptography Keys und einem Amazon Elastic Block Store (Amazon EBS) -Volume oder AWS Secrets Manager Secret dasselbe Tag zuweisen. Sie können Tags auch verwenden, um Schlüssel für die Automatisierung zu identifizieren.
- Verfolgen Sie Ihre AWS Kosten. Wenn Sie Ihren AWS Ressourcen Tags hinzufügen, AWS wird ein Kostenverteilungsbericht generiert, in dem die Nutzung und die Kosten nach Stichwörtern zusammengefasst sind. Sie können diese Funktion verwenden, um die Kosten der AWS Zahlungskryptografie für ein Projekt, eine Anwendung oder eine Kostenstelle nachzuverfolgen.

  Weitere Informationen zur Verwendung von Tags für die Kostenzuordnung finden Sie unter Verwenden von Kostenzuordnungs-Tags im AWS Billing -Benutzerhandbuch. Weitere Informationen über die Regeln, die für die Tag-Schlüssel und Tag-Werte gelten, finden Sie unter Beschränkungen benutzerdefinierter Tags im AWS Billing -Benutzerhandbuch.
- Kontrollieren Sie den Zugriff auf Ihre AWS Ressourcen. Das Zulassen und Verweigern des Zugriffs auf Schlüssel auf der Grundlage ihrer Tags ist Teil der AWS Payment Cryptography-Unterstützung für die attributebasierte Zugriffskontrolle (ABAC). Informationen zur Steuerung des Zugriffs auf AWS Payment Cryptography anhand ihrer Tags finden Sie unter. Autorisierung auf der Grundlage von Payment Cryptography Tags AWS Weitere allgemeine Informationen zur Verwendung von Tags zur Steuerung des Zugriffs auf AWS Ressourcen finden Sie unter Steuern des Zugriffs auf AWS Ressourcen mithilfe von Ressourcen-Tags im IAM-Benutzerhandbuch.

AWS Payment Cryptography schreibt einen Eintrag in Ihr AWS CloudTrail Protokoll, wenn Sie die Operationen TagResource UntagResource, oder ListTagsForResource verwenden.

Schlüssel-Tags in der Konsole anzeigen

Um Tags in der Konsole anzeigen zu können, benötigen Sie die Tagging-Berechtigung für den Schlüssel aus einer IAM-Richtlinie, die den Schlüssel enthält. Sie benötigen diese Berechtigungen zusätzlich zu den Berechtigungen zum Anzeigen von Schlüsseln in der Konsole.

Verwaltung von Schlüssel-Tags mit API-Operationen

Sie können die AWS Payment Cryptography API verwenden, um Tags für die von Ihnen verwalteten Schlüssel hinzuzufügen, zu löschen und aufzulisten. Für diese Beispiele wird die AWS Command

Line Interface (AWS CLI) verwendet. Sie können aber jede unterstützte Programmiersprache nutzen. Sie können nicht taggen Von AWS verwaltete Schlüssel.

Um Tags für einen Schlüssel hinzuzufügen, zu bearbeiten, anzuzeigen und zu löschen, benötigen Sie die erforderlichen Berechtigungen. Details hierzu finden Sie unter Zugriffssteuerung mit Tags.

Themen

- CreateKey: Fügen Sie einem neuen Schlüssel Tags hinzu
- TagResource: Tags für einen Schlüssel hinzufügen oder ändern
- ListResourceTags: Ruft die Tags für einen Schlüssel ab
- UntagResource: Löscht Tags aus einem Schlüssel

CreateKey: Fügen Sie einem neuen Schlüssel Tags hinzu

Sie können Tags hinzufügen, wenn Sie einen Schlüssel erstellen. Verwenden Sie den `Tags` Parameter der CreateKeyOperation, um die Tags anzugeben.

Um beim Erstellen eines Schlüssels Tags hinzuzufügen, muss der Aufrufer über die entsprechenden `payment-cryptography:TagResource` Rechte in einer IAM-Richtlinie verfügen. Die Erlaubnis muss mindestens alle Schlüssel im Konto und in der Region abdecken. Details hierzu finden Sie unter Zugriffssteuerung mit Tags.

Der Wert des `Tags`-Parameters von `CreateKey` ist eine Sammlung von Tag-Schlüssel- und Tag-Wert-Paaren, unter Beachtung der Groß-/Kleinschreibung. Jedes Tag auf einem Schlüssel muss einen anderen Tagnamen haben. Der Tag-Wert kann auch eine leere (Null) Zeichenfolge sein.

Mit dem folgenden AWS CLI Befehl wird beispielsweise ein symmetrischer Verschlüsselungsschlüssel mit einem `Project:Alpha` Tag erstellt. Wenn Sie mehr als ein Schlüssel-Wert-Paar angeben, verwenden Sie ein Leerzeichen, um die einzelnen Paare zu trennen.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY, \
       KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY, \
       KeyModesOfUse='{Generate=true,Verify=true}' \
       --tags '[{"Key":"Project","Value":"Alpha"},{"Key":"BIN","Value":"123456"}]'
```

Wenn dieser Befehl erfolgreich ist, gibt er ein `Key` Objekt mit Informationen über den neuen Schlüssel zurück. Die `Key` enthalten jedoch keine Tags. Verwenden Sie die Operation Tags, um die ListResourceTags abzurufen.

TagResource: Tags für einen Schlüssel hinzufügen oder ändern

Die TagResourceOperation fügt einem Schlüssel ein oder mehrere Tags hinzu. Sie können diese Operation nicht verwenden, um Tags in einem anderen AWS-Konto hinzuzufügen oder zu bearbeiten.

Geben Sie zum Hinzufügen eines Tags einen neuen Tag-Schlüssel und einen Tag-Wert an. Um ein Tag zu bearbeiten, geben Sie einen vorhandenen Tag-Schlüssel und einen neuen Tag-Wert an. Jedes Tag auf einem Schlüssel muss einen anderen Tag-Schlüssel haben. Der Tag-Wert kann auch eine leere (Null) Zeichenfolge sein.

Der folgende Befehl fügt beispielsweise einem Beispielschlüssel **`BIN`** Tags hinzu **`UseCase`** und fügt ihm Tags hinzu.

```
$ aws payment-cryptography tag-resource --resource-arn arn:aws:payment-
cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h --tags
 '[{"Key":"UseCase","Value":"Acquiring"},{"Key":"BIN","Value":"123456"}]'
```

Wenn der Befehl erfolgreich war, erfolgt keine Ausgabe. Um die Tags eines Schlüssels anzuzeigen, verwenden Sie die Operation ListResourceTags.

Sie können auch TagResource verwenden, um den Tag-Wert für ein vorhandenes Tag zu ändern. Um einen Tag-Wert zu ersetzen, geben Sie den gleichen Tag-Schlüssel mit einem unterschiedlichen Wert an. Tags, die nicht in einem Änderungsbefehl aufgeführt sind, werden nicht geändert oder entfernt.

Beispielsweise ändert dieser Befehl den Wert des `Project`-Tag von `Alpha` auf `Noe`.

Der Befehl gibt http/200 ohne Inhalt zurück. Um Ihre Änderungen zu sehen, verwenden Sie `ListTagsForResource`

```
$ aws payment-cryptography tag-resource --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
        --tags '[{"Key":"Project","Value":"Noe"}]'
```

ListResourceTags: Ruft die Tags für einen Schlüssel ab

Die ListResourceTags-Operation ruft die Tags für einen Schlüssel ab. Der Parameter `ResourceArn` (KeyArn oder KeyAlias) ist erforderlich. Sie können diesen Vorgang nicht verwenden, um die Tags auf Schlüsseln in einem anderen Format anzuzeigen. AWS-Konto

Mit dem folgenden Befehl werden beispielsweise die Tags für einen Beispielschlüssel abgerufen.

```
$ aws payment-cryptography list-tags-for-resource --resource-arn arn:aws:payment-
cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h

  {
    "Tags": [
        {
            "Key": "BIN",
            "Value": "20151120"
        },
        {
            "Key": "Project",
            "Value": "Production"
        }
    ]
}
```

UntagResource: Löscht Tags aus einem Schlüssel

Die UntagResourceOperation löscht Tags aus einem Schlüssel. Um die zu löschenden Tags zu identifizieren, geben Sie die Tag-Schlüssel an. Sie können diesen Vorgang nicht verwenden, um Tags aus einem anderen AWS-Konto Schlüssel zu löschen.

Wenn sie erfolgreich ist, gibt die `UntagResource`-Operation keine Ausgabe zurück. Wenn der angegebene Tag-Schlüssel auf dem Schlüssel nicht gefunden wird, wird auch keine Ausnahme ausgelöst und es wird keine Antwort zurückgegeben. Verwenden Sie den Vorgang ListResourceTags, um zu überprüfen, ob der Vorgang funktioniert hat.

Mit diesem Befehl werden beispielsweise das **`Purpose`** Tag und sein Wert aus dem angegebenen Schlüssel gelöscht.

```
$ aws payment-cryptography untag-resource \
       --resource-arn arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h --tag-keys Project
```

Zugriffssteuerung mit Tags

Um Tags mithilfe der API hinzuzufügen, anzuzeigen und zu löschen, benötigen Principals Tagging-Berechtigungen in den IAM-Richtlinien.

Sie können diese Berechtigungen auch einschränken, indem Sie AWS globale Bedingungsschlüssel für Tags verwenden. In der AWS Zahlungskryptografie können diese Bedingungen den Zugriff auf Tagging-Operationen wie TagResourceund steuern. UntagResource

Beispielrichtlinien und weitere Informationen finden Sie unter Zugriffssteuerung anhand von Tag-Schlüsseln im IAM-Benutzerhandbuch.

Berechtigungen zum Erstellen und Verwalten von Tags funktionieren wie folgt.

Zahlungskryptografie: TagResource

Erlaubt es Prinzipalen, Tags hinzuzufügen oder zu bearbeiten. Um beim Erstellen eines Schlüssels Tags hinzufügen zu können, muss der Principal über die entsprechenden Rechte in einer IAM-Richtlinie verfügen, die nicht auf bestimmte Schlüssel beschränkt ist.

Zahlungskryptografie: ListTags ForResource

Ermöglicht Prinzipalen das Anzeigen von Tags auf Schlüsseln.

Zahlungskryptografie: UntagResource

Ermöglicht Prinzipalen, Tags aus Schlüsseln zu löschen.

Tag-Berechtigungen in Richtlinien

Sie können Markierungs-Berechtigungen in einer Schlüsselrichtlinie oder einer IAM-Richtlinie bereitstellen. Das folgende Beispiel für eine Schlüsselrichtlinie gibt ausgewählten Benutzern beispielsweise die Möglichkeit, den Schlüssel mit Tags zu versehen. Sie erteilt allen Benutzern, die die Beispiel-Administrator- oder Entwicklerrollen übernehmen können, die Berechtigung zum Anzeigen von Tags.

```
{
  "Version": "2012-10-17",
  "Id": "example-key-policy",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "payment-cryptography:*",
      "Resource": "*"
    },
```

```
    {
      "Sid": "Allow all tagging permissions",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:user/LeadAdmin",
        "arn:aws:iam::111122223333:user/SupportLead"
      ]},
      "Action": [
          "payment-cryptography:TagResource",
          "payment-cryptography:ListTagsForResource",
          "payment-cryptography:UntagResource"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow roles to view tags",
      "Effect": "Allow",
      "Principal": {"AWS": [
        "arn:aws:iam::111122223333:role/Administrator",
        "arn:aws:iam::111122223333:role/Developer"
      ]},
      "Action": "payment-cryptography:ListResourceTags",
      "Resource": "*"
    }
  ]
}
```

Um Prinzipalen die Möglichkeit zu geben, mehrere Schlüssel zu kennzeichnen, können Sie eine IAM-Richtlinie verwenden. Damit diese Richtlinie wirksam ist, muss die Schlüsselrichtlinie für jeden Schlüssel es dem Konto ermöglichen, den Zugriff auf den Schlüssel mithilfe von IAM-Richtlinien zu steuern.

Die folgende IAM-Richtlinie ermöglicht es den Prinzipalen beispielsweise, Schlüssel zu erstellen. Sie ermöglicht ihnen auch, Tags für alle Schlüssel im angegebenen Konto zu erstellen und zu verwalten. Diese Kombination ermöglicht es den Prinzipalen, den Tags-Parameter des CreateKeyVorgangs zu verwenden, um einem Schlüssel während der Erstellung Tags hinzuzufügen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKeys",
```

```
      "Effect": "Allow",
      "Action": "payment-cryptography:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyTags",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:TagResource",
        "payment-cryptography:UntagResource",
        "payment-cryptography:ListTagsForResource"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*"
    }
  ]
}
```

Beschränken von Tag-Berechtigungen

Sie können Markierungs-Berechtigungen einschränken, indem Sie Richtlinienbedingungen verwenden. Die folgenden Richtlinienbedingungen können auf die `payment-cryptography:TagResource`- und `payment-cryptography:UntagResource`-Berechtigungen angewendet werden. Beispielsweise können Sie mit der `aws:RequestTag/`*`tag-key`*-Bedingung einem Prinzipal erlauben, nur bestimmte Tags hinzuzufügen, oder verhindern, dass ein Prinzipal Tags mit bestimmten Tag-Schlüsseln hinzufügen kann.

- aws: RequestTag
- aws:ResourceTag/tag-key (nur IAM-Richtlinien)
- aws: TagKeys

Wenn Sie Tags verwenden, um den Zugriff auf Schlüssel zu kontrollieren, empfiehlt es sich, den `aws:TagKeys` Bedingungsschlüssel `aws:RequestTag/`*`tag-key`* oder zu verwenden, um zu bestimmen, welche Tags (oder Tag-Schlüssel) zulässig sind.

Die folgende IAM-Richtlinie ähnelt beispielsweise der vorherigen. Diese Richtlinie erlaubt den Prinzipalen jedoch das Erstellen von Tags (`TagResource`) und das Löschen von Tags (`UntagResource`) nur für Tags mit einem `Project`-Tag-Schlüssel.

Da `TagResource` `UntagResource` Anfragen mehrere Tags enthalten können, müssen Sie einen Operator `ForAllValues` or `ForAnyValue` set mit der TagKeys Bedingung aws: angeben. Der

`ForAnyValue`-Operator erfordert, dass mindestens einer der Tag-Schlüssel in der Anforderung mit einem der Tag-Schlüssel in der Richtlinie übereinstimmen muss. Der `ForAllValues`-Operator erfordert, dass alle der Tag-Schlüssel in der Anforderung mit einem der Tag-Schlüssel in der Richtlinie übereinstimmen müssen. Der `ForAllValues` Operator gibt auch zurück, `true` wenn die Anfrage keine Tags enthält, schlägt jedoch TagResource UntagResource fehl, wenn keine Tags angegeben sind. Ausführliche Informationen zu den Satz-Operatoren finden Sie unter Verwenden mehrerer Schlüssel und Werte im IAM-Benutzerhandbuch.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyCreateKey",
      "Effect": "Allow",
      "Action": "payment-cryptography:CreateKey",
      "Resource": "*"
    },
    {
      "Sid": "IAMPolicyViewAllTags",
      "Effect": "Allow",
      "Action": "payment-cryptography:ListResourceTags",
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*"
    },
    {
      "Sid": "IAMPolicyManageTags",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:TagResource",
        "payment-cryptography:UntagResource"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
          "ForAllValues:StringEquals": {"aws:TagKeys": "Project"}
      }
    }
  ]
}
```

Verwendung von Tags zur Steuerung des Zugriffs auf Schlüssel

Sie können den Zugriff auf AWS Payment Cryptography anhand der Tags auf dem Schlüssel steuern. Sie können beispielsweise eine IAM-Richtlinie schreiben, die es Prinzipalen ermöglicht, nur die

Schlüssel zu aktivieren und zu deaktivieren, die über ein bestimmtes Tag verfügen. Oder Sie können eine IAM-Richtlinie verwenden, um zu verhindern, dass Prinzipale Schlüssel für kryptografische Operationen verwenden, es sei denn, der Schlüssel hat ein bestimmtes Tag.

Diese Funktion ist Teil der AWS Payment Cryptography-Unterstützung für die attributebasierte Zugriffskontrolle (ABAC). Informationen zur Verwendung von Tags zur Steuerung des Zugriffs auf AWS Ressourcen finden Sie unter Wozu dient ABAC? AWS und Steuern des Zugriffs auf AWS Ressourcen mithilfe von Ressourcen-Tags im IAM-Benutzerhandbuch.

> **Note**

AWS Die Zahlungskryptografie unterstützt den globalen Bedingungskontextschlüssel aws:ResourceTag/tag-key, mit dem Sie den Zugriff auf Schlüssel anhand der Tags auf dem Schlüssel steuern können. Da mehrere Schlüssel dasselbe Tag haben können, können Sie mit dieser Funktion die Berechtigung auf einen ausgewählten Schlüsselsatz anwenden. Sie können die Schlüssel im Satz auch einfach ändern, indem Sie ihre Tags ändern.

In der AWS Zahlungskryptografie wird der `aws:ResourceTag/`*`tag-key`* Bedingungsschlüssel nur in IAM-Richtlinien unterstützt. Er wird in wichtigen Richtlinien, die nur für einen Schlüssel gelten, oder bei Vorgängen, die keinen bestimmten Schlüssel verwenden, wie z. B. die ListKeysOder-Operationen, nicht unterstützt. ListAliases

Die Steuerung des Zugriffs mit Tags bietet eine einfache, skalierbare und flexible Möglichkeit, Berechtigungen zu verwalten. Wenn es jedoch nicht richtig konzipiert und verwaltet wird, kann es versehentlich den Zugriff auf Ihre Schlüssel gewähren oder verweigern. Wenn Sie Tags verwenden, um den Zugriff zu steuern, sollten Sie die folgenden Methoden berücksichtigen.

- Verwenden Sie Tags, um beim Zugriff die bewährte Methode der geringsten Berechtigung zu befolgen. Erteilen Sie IAM-Prinzipalen nur die Berechtigungen, die sie benötigen, und zwar nur für die Schlüssel, die sie verwenden oder verwalten müssen. Verwenden Sie beispielsweise Tags, um die für ein Projekt verwendeten Schlüssel zu kennzeichnen. Erteilen Sie dann dem Projektteam die Erlaubnis, nur Schlüssel mit dem Projekt-Tag zu verwenden.
- Seien Sie vorsichtig, wenn Sie Prinzipalen die `payment-cryptography:TagResource`- und `payment-cryptography:UntagResource`-Berechtigung erteilen, mit denen sie Tags hinzufügen, bearbeiten und löschen können. Wenn Sie Tags verwenden, um den Zugriff auf Schlüssel zu steuern, kann das Ändern eines Tags den Prinzipalen die Erlaubnis geben, Schlüssel

zu verwenden, zu deren Verwendung sie sonst nicht berechtigt waren. Es kann auch den Zugriff auf Schlüssel verweigern, die andere Prinzipale für ihre Arbeit benötigen. Schlüsseladministratoren, die nicht berechtigt sind, wichtige Richtlinien zu ändern oder Zuweisungen zu vergeben, können den Zugriff auf Schlüssel kontrollieren, sofern sie über die Berechtigung zur Verwaltung von Stichwörtern verfügen.

Verwenden Sie nach Möglichkeit eine Richtlinienbedingung, z. B. `aws:RequestTag/`*`tag-key`* `aws:TagKeys` um die Tag-Berechtigungen eines Prinzipals auf bestimmte Tags oder Tag-Muster für bestimmte Schlüssel zu beschränken.

- Prüfen Sie die Prinzipale in Ihrem System AWS-Konto , die derzeit über Berechtigungen zum Markieren und Entmarkieren verfügen, und passen Sie diese gegebenenfalls an. In IAM-Richtlinien können für alle Schlüssel Berechtigungen zum Markieren und Aufheben von Tags zulässig sein. Die vom Administrator verwaltete Richtlinie ermöglicht es Prinzipalen beispielsweise, Tags für alle Schlüssel zu kennzeichnen, die Markierung aufzuheben und sie aufzulisten.
- Bevor Sie eine Richtlinie festlegen, die von einem Tag abhängt, überprüfen Sie die Tags auf den Schlüsseln in Ihrem. AWS-Konto Stellen Sie sicher, dass Ihre Richtlinie nur für die Tags gilt, die Sie einschließen möchten. Verwenden Sie CloudTrail Protokolle und CloudWatch Alarme, um Sie auf Änderungen am Tag aufmerksam zu machen, die sich auf den Zugriff auf Ihre Schlüssel auswirken könnten.
- Die tag-basierten Richtlinienbedingungen verwenden Musterabgleich; sie sind nicht an eine bestimmte Instance eines Tags gebunden. Eine Richtlinie, die Tag-basierte Bedingungsschlüssel verwendet, wirkt sich auf alle neuen und vorhandenen Tags aus, die dem Muster entsprechen. Wenn Sie ein Tag löschen und neu erstellen, das einer Richtlinienbedingung entspricht, gilt die Bedingung für das neue Tag, genau wie für das alte Tag.

Betrachten Sie beispielsweise die folgende IAM-Richtlinie: Dadurch können die Principals die Entschlüsselungsvorgänge nur für Schlüssel in Ihrem Konto aufrufen, die sich in der Region USA Ost (Nord-Virginia) befinden und über ein `"Project"="Alpha"` Schlagwort verfügen. Sie können diese Richtlinie an Rollen im Beispiel Alpha-Projekt anfügen.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
```

```
        "payment-cryptography:DecryptData"
      ],
      "Resource": "arn:aws::us-east-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

Das folgende Beispiel für eine IAM-Richtlinie ermöglicht es den Prinzipalen, jeden beliebigen Schlüssel im Konto für bestimmte kryptografische Operationen zu verwenden. Sie verbietet den Prinzipalen jedoch, diese kryptografischen Operationen für Schlüssel mit oder ohne Tag zu verwenden. `"Type"="Reserved"` `"Type"`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAllowCryptographicOperations",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:EncryptData",
        "payment-cryptography:DecryptData",
        "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*"
    },
    {
      "Sid": "IAMDenyOnTag",
      "Effect": "Deny",
      "Action": [
        "payment-cryptography:EncryptData",
        "payment-cryptography:DecryptData",
        "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Type": "Reserved"
        }
```

```
      }
    },
    {
      "Sid": "IAMDenyNoTag",
      "Effect": "Deny",
      "Action": [
        "payment-cryptography:EncryptData",
        "payment-cryptography:DecryptData",
        "payment-cryptography:ReEncrypt*"
      ],
      "Resource": "arn:aws:kms:*:111122223333:key/*",
      "Condition": {
        "Null": {
          "aws:ResourceTag/Type": "true"
        }
      }
    }
  ]
}
```

Grundlegendes zu den Schlüsselattributen des Payment Cryptography AWS Keys

Ein Grundsatz der ordnungsgemäßen Schlüsselverwaltung besteht darin, dass Schlüssel einen angemessenen Gültigkeitsbereich haben und nur für zulässige Operationen verwendet werden können. Daher können bestimmte Schlüssel nur mit bestimmten Verwendungsarten erstellt werden. Wann immer möglich, entspricht dies den verfügbaren Verwendungsmodi, wie sie in TR-31 definiert sind.

AWS Die Zahlungskryptografie verhindert zwar, dass Sie ungültige Schlüssel erstellen, aber der Einfachheit halber finden Sie hier gültige Kombinationen.

Symmetrische Schlüssel

- TR31_B0_BASE_DERIVATION_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_C0_CARD_VERIFICATION KEY

  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_E1_EMV_MKEY_CONFIDENTIALITY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_E2_EMV_MKEY_INTEGRITY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_E5_EMV_MKEY_CARD_PERSONALIZATION
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions
- TR31_E6_EMV_MKEY_OTHER
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {= true}, {= true} DeriveKey NoRestrictions

- TR31_K0_KEY_ENCRYPTION_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_K1_KEY_BLOCK_PROTECTION_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_M1_ISO_9797_1_MAC_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M3_ISO_9797_3_MAC_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M6_ISO_9797_5_CMAC_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M7_HMAC_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_P0_PIN_ENCRYPTION_KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_V1_IBM3624_PIN_VERIFICATION_KEY

  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_V2_VISA_PIN_VERIFICATION KEY
  - Zulässige Schlüsselalgorithmen: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions

Asymmetrische Schlüssel

- TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION
  - Zulässige Schlüsselalgorithmen: RSA_2048, RSA_3072, RSA_4096
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}
  - HINWEIS: {Encrypt = true, Wrap = true} ist die einzig gültige Option beim Import eines öffentlichen Schlüssels, der zum Verschlüsseln von Daten oder zum Umschließen eines Schlüssels vorgesehen ist
- TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE
  - Zulässige Schlüsselalgorithmen: RSA_2048, RSA_3072, RSA_4096
  - Zulässige Kombination der wichtigsten Verwendungsmodi: {Sign = true}, {Verify = true}
  - HINWEIS: {Verify = true} ist die einzig gültige Option beim Import eines Schlüssels, der zum Signieren bestimmt ist, z. B. ein Stammzertifikat, ein Zwischenzertifikat oder Signaturzertifikate für TR-34.

Datenoperationen

Nachdem Sie einen Schlüssel für die AWS Zahlungskryptografie eingerichtet haben, kann dieser zur Durchführung kryptografischer Operationen verwendet werden. Verschiedene Operationen führen unterschiedliche Arten von Aktivitäten aus, die von Verschlüsselung und Hashing bis hin zu domänenspezifischen Algorithmen wie der CVV2-Generierung reichen.

Verschlüsselte Daten können ohne den entsprechenden Entschlüsselungsschlüssel (je nach Verschlüsselungstyp der symmetrische Schlüssel oder der private Schlüssel) nicht entschlüsselt werden. Hashing- und domänenspezifische Algorithmen können ohne den symmetrischen Schlüssel oder den öffentlichen Schlüssel ebenfalls nicht verifiziert werden.

Informationen zu gültigen Schlüsseltypen für bestimmte Operationen finden Sie unter Gültige Schlüssel für kryptografische Operationen

> **Note**
>
> Wir empfehlen, Testdaten in einer Umgebung außerhalb der Produktionsumgebung zu verwenden. Die Verwendung von Produktionsschlüsseln und -daten (PAN, BDK-ID usw.) in einer Umgebung außerhalb der Produktion kann sich auf Ihren Compliance-Umfang auswirken, z. B. für PCI DSS und PCI P2PE.

Themen

Daten verschlüsseln, entschlüsseln und erneut verschlüsseln

Verschlüsselungs- und Entschlüsselungsmethoden können verwendet werden, um Daten mithilfe einer Vielzahl von symmetrischen und asymmetrischen Techniken wie TDES, AES und RSA zu

verschlüsseln oder zu entschlüsseln. Diese Methoden unterstützen auch Schlüssel, die mit DUKPT- und EMV-Techniken abgeleitet wurden. Für Anwendungsfälle, in denen Sie Daten unter einem neuen Schlüssel sichern möchten, ohne die zugrunde liegenden Daten offenzulegen, kann der ReEncrypt Befehl auch verwendet werden.

> **Note**
>
> Bei der Verwendung der Verschlüsselungs-/Entschlüsselungsfunktionen wird davon ausgegangen, dass alle Eingaben in HexBinary erfolgen. Beispielsweise wird ein Wert von 1 als 31 (hex) eingegeben und ein kleingeschriebenes t wird als 74 (hex) dargestellt. Alle Ausgaben sind ebenfalls in HexBinary.

Einzelheiten zu allen verfügbaren Optionen finden Sie im API-Leitfaden für Encrypt, Decrypt and Re-Encrypt.

Themen

- Daten verschlüsseln
- Daten entschlüsseln

Daten verschlüsseln

Die `Encrypt Data` API wird verwendet, um Daten mit symmetrischen und asymmetrischen Datenverschlüsselungsschlüsseln sowie von DUKPT und EMV abgeleiteten Schlüsseln zu verschlüsseln. Verschiedene Algorithmen und Varianten werden unterstützt, darunter, und. `TDES` `RSA` `AES`

Die wichtigsten Eingaben sind der Verschlüsselungsschlüssel, der zum Verschlüsseln der Daten verwendet wird, die Klartextdaten im HexBinary-Format, die verschlüsselt werden sollen, und Verschlüsselungsattribute wie Initialisierungsvektor und Modus für Blockchiffren wie TDES. Die Klartextdaten müssen ein Vielfaches von 8 Byte für`TDES`, 16 Byte für `AES` und der Länge des Schlüssels im Fall von sein. `RSA` Symmetrische Tasteneingaben (TDES, AES, DUKPT, EMV) sollten aufgefüllt werden, falls die Eingabedaten diese Anforderungen nicht erfüllen. Die folgende Tabelle zeigt die maximale Klartext-Länge für jeden Schlüsseltyp und den Fülltyp, den Sie für RSA-Schlüssel definieren. `EncryptionAttributes`

Füllungstyp	RSA_2048	RSA_3072	RSA_4096
OAEP SHA1	428	684	940
OAEP SHA256	380	636	892
OAEP SHA512	252	508	764
PKCS1	488	744	1000
None	488	744	1000

Die primären Ausgaben enthalten die verschlüsselten Daten als Chiffretext im HexBinary-Format und den Prüfsummenwert für den Verschlüsselungsschlüssel. Einzelheiten zu allen verfügbaren Optionen finden Sie im API-Leitfaden für Encrypt.

Beispiele

- Verschlüsseln Sie Daten mit einem symmetrischen AES-Schlüssel
- Verschlüsseln Sie Daten mit dem DUKPT-Schlüssel
- Verschlüsseln Sie Daten mit einem von EMV abgeleiteten symmetrischen Schlüssel
- Verschlüsseln Sie Daten mit einem RSA-Schlüssel

Verschlüsseln Sie Daten mit einem symmetrischen AES-Schlüssel

> **Note**
>
> Alle Beispiele gehen davon aus, dass der entsprechende Schlüssel bereits existiert. Schlüssel können mithilfe der CreateKeyOperation erstellt oder mithilfe der ImportKeyOperation importiert werden.

Example

In diesem Beispiel verschlüsseln wir Klartextdaten mit einem symmetrischen Schlüssel, der mithilfe der Operation erstellt oder mithilfe der CreateKeyOperation importiert wurde. ImportKey Für diesen Vorgang muss der Schlüssel auf `Encrypt` und KeyUsage auf KeyModesOfUse gesetzt sein.

TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen.

```
$ aws payment-cryptography-data encrypt-data --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi --plain-text
 313233343132333431323334313233343 --encryption-attributes 'Symmetric={Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Verschlüsseln Sie Daten mit dem DUKPT-Schlüssel

Example

In diesem Beispiel verschlüsseln wir Klartextdaten mit einem DUKPT-Schlüssel. AWS Zahlungskryptografie und DUKPT-Schlüssel werden unterstütztTDES. AES Für diesen Vorgang muss der Schlüssel auf DeriveKey und KeyUsage auf KeyModesOfUse gesetzt sein. TR31_B0_BASE_DERIVATION_KEY Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen.

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --plain-text 313233343132333431323334313233343 --encryption-attributes
 'Dukpt={KeySerialNumber=FFFF9876543210E00001}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
```

```
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Verschlüsseln Sie Daten mit einem von EMV abgeleiteten symmetrischen Schlüssel

Example

In diesem Beispiel verschlüsseln wir Klartextdaten mit einem von EMV abgeleiteten symmetrischen Schlüssel, der bereits erstellt wurde. Sie könnten einen Befehl wie diesen verwenden, um Daten an eine EMV-Karte zu senden. Für diesen Vorgang muss KeyModesOfUse der Schlüssel auf `Derive` und auf `TR31_E1_EMV_MKEY_CONFIDENTIALITY` oder KeyUsage `TR31_E6_EMV_MKEY_OTHER` gesetzt sein. Weitere Informationen finden Sie unter Schlüssel für kryptografische Operationen.

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --plain-text 33612AB9D6929C3A828EB6030082B2BD --encryption-attributes
 'Emv={MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber=27,PrimaryAccountNumber=1000000000
 InitializationVector=1500000000000999,Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "CipherText": "33612AB9D6929C3A828EB6030082B2BD"
}
```

Verschlüsseln Sie Daten mit einem RSA-Schlüssel

Example

In diesem Beispiel verschlüsseln wir Klartextdaten mit einem öffentlichen RSA-Schlüssel, der mit der Operation importiert wurde. ImportKey Für diesen Vorgang muss der Schlüssel auf `Encrypt` und KeyUsage auf KeyModesOfUse gesetzt sein. `TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION` Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen.

Für PKCS #7 oder andere Padding-Schemata, die derzeit nicht unterstützt werden, bewerben Sie sich bitte, bevor Sie den Service aufrufen, und wählen Sie kein Padding aus, indem Sie den Padding-Indikator 'Asymmetric= {} 'weglassen

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/thfezpmsalcfwmsg
 --plain-text 313233343132333431323334313233343 --encryption-attributes
 'Asymmetric={PaddingType=OAEP_SHA256}'
```

```
{
 "CipherText":
 "12DF6A2F64CC566D124900D68E8AFEAA794CA819876E258564D525001D00AC93047A83FB13 \
 E73F06329A100704FA484A15A49F06A7A2E55A241D276491AA91F6D2D8590C60CDE57A642BC64A897F4832A3930
 \
 0FAEC7981102CA0F7370BFBF757F271EF0BB2516007AB111060A9633D1736A9158042D30C5AE11F8C5473EC70F067
 \
 72590DEA1638E2B41FAE6FB1662258596072B13F8E2F62F5D9FAF92C12BB70F42F2ECDCF56AADF0E311D4118FE3591
 \
 FB672998CCE9D00FFFE05D2CD154E3120C5443C8CF9131C7A6A6C05F5723B8F5C07A4003A5A6173E1B425E2B5E42AD
 \
 7A2966734309387C9938B029AFB20828ACFC6D00CD1539234A4A8D9B94CDD4F23A",
 "KeyArn": "arn:aws:payment-cryptography:us-east-1:529027455495:key/5dza7xqd6soanjtb",
 "KeyCheckValue": "FF9DE9CE"
}
```

Daten entschlüsseln

Die `Decrypt Data` API wird verwendet, um Daten mit symmetrischen und asymmetrischen Datenverschlüsselungsschlüsseln sowie von DUKPT und EMV abgeleiteten Schlüsseln zu entschlüsseln. Verschiedene Algorithmen und Varianten werden unterstützt, darunter, und. `TDES` `RSA` `AES`

Die wichtigsten Eingaben sind der Entschlüsselungsschlüssel, der zum Entschlüsseln der Daten verwendet wird, die Chiffretextdaten im HexBinary-Format, die entschlüsselt werden sollen, und Entschlüsselungsattribute wie Initialisierungsvektor, Modus als Blockchiffren usw. Die primären Ausgaben umfassen die entschlüsselten Daten als Klartext im HexBinary-Format und den Prüfsummenwert für den Entschlüsselungsschlüssel. Einzelheiten zu allen verfügbaren Optionen finden Sie im API-Leitfaden für Decrypt.

Beispiele

- Entschlüsseln Sie Daten mit dem symmetrischen AES-Schlüssel
- Entschlüsseln Sie Daten mit dem DUKPT-Schlüssel
- Entschlüsseln Sie Daten mithilfe eines von EMV abgeleiteten symmetrischen Schlüssels
- Entschlüsseln Sie Daten mit einem RSA-Schlüssel

Entschlüsseln Sie Daten mit dem symmetrischen AES-Schlüssel

Example

In diesem Beispiel werden wir Chiffretextdaten mit einem symmetrischen Schlüssel entschlüsseln. Dieses Beispiel zeigt einen `AES` Schlüssel, aber `TDES_2KEY` er wird auch unterstützt. `TDES_3KEY` Für diesen Vorgang muss der Schlüssel auf KeyModesOfUse gesetzt `Decrypt` und auf KeyUsage gesetzt sein`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen.

```
$ aws payment-cryptography-data decrypt-data --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi --cipher-text
 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes 'Symmetric={Mode=CBC}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "PlainText": "31323334313233343132333431323334"
}
```

Entschlüsseln Sie Daten mit dem DUKPT-Schlüssel

> **Note**
>
> Die Verwendung von Entschlüsselungsdaten mit DUKPT für P2PE-Transaktionen kann dazu führen, dass Kreditkarten-PAN und andere Karteninhaberdaten an Ihre Anwendung

zurückgegeben werden, die bei der Bestimmung des PCI-DSS-Umfangs berücksichtigt werden müssen.

Example

In diesem Beispiel werden wir Chiffretextdaten mithilfe eines DUKPT-Schlüssels entschlüsseln, der mit der Operation erstellt oder mit der Operation importiert wurde. CreateKeyImportKey Für diesen Vorgang muss der Schlüssel auf gesetzt und auf KeyModesOfUse gesetzt seinDeriveKey. KeyUsage TR31_B0_BASE_DERIVATION_KEY Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen. Wenn Sie für den TDES Algorithmus verwendenDUKPT, muss die Länge der Chiffretext-Daten ein Vielfaches von 16 Byte sein. Für AES Algorithmen muss die Länge der Chiffretext-Daten ein Vielfaches von 32 Byte sein.

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --cipher-text 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes
 'Dukpt={KeySerialNumber=FFFF9876543210E00001}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "71D7AE",
    "PlainText": "31323334313233343132333431323334"
}
```

Entschlüsseln Sie Daten mithilfe eines von EMV abgeleiteten symmetrischen Schlüssels

Example

In diesem Beispiel werden wir Chiffretextdaten mithilfe eines von EMV abgeleiteten symmetrischen Schlüssels entschlüsseln, der mit der Operation erstellt oder mit der Operation importiert wurde. CreateKeyImportKey Für diesen Vorgang muss der Schlüssel auf und auf oder

KeyModesOfUse gesetzt sein. `Derive` KeyUsage `TR31_E1_EMV_MKEY_CONFIDENTIALITY` `TR31_E6_EMV_MKEY_OTHER` Weitere Informationen finden Sie unter Schlüssel für kryptografische Operationen.

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --cipher-text 33612AB9D6929C3A828EB6030082B2BD --decryption-attributes
 'Emv={MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber=27,PrimaryAccountNumber=1000000000
 InitializationVector=1500000000000999,Mode=CBC}'
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi",
"KeyCheckValue": "71D7AE",
"PlainText": "31323334313233343132333431323334"
}
```

Entschlüsseln Sie Daten mit einem RSA-Schlüssel

Example

In diesem Beispiel werden wir Chiffretextdaten mit einem RSA-Schlüsselpaar entschlüsseln, das mit der Operation erstellt wurde. CreateKey Für diesen Vorgang muss der Schlüssel auf aktiviert `Decrypt` und auf KeyModesOfUse gesetzt sein. KeyUsage `TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION` Weitere Optionen finden Sie unter Schlüssel für kryptografische Operationen.

Für PKCS #7 oder andere Padding-Schemata, die derzeit nicht unterstützt werden, wählen Sie bitte kein Padding aus, indem Sie den Padding-Indikator 'Asymmetric= {} 'weglassen und das Padding nach dem Aufrufen des Dienstes entfernen.

```
$ aws payment-cryptography-data decrypt-data \
       --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5dza7xqd6soanjtb --cipher-text
 8F4C1CAFE7A5DEF9A40BEDE7F2A264635C... \
       --decryption-attributes 'Asymmetric={PaddingType=OAEP_SHA256}'
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-
east-1:529027455495:key/5dza7xqd6soanjtb",
    "KeyCheckValue": "FF9DE9CE",
    "PlainText": "31323334313233343132333431323334"
}
```

Kartendaten generieren und verifizieren

Kartendaten generieren und verifizieren beinhalten Daten, die aus Kartendaten abgeleitet wurden, zum Beispiel CVV, CVV2, CVC und DCVV.

Themen

- Kartendaten generieren
- Überprüfen Sie die Kartendaten

Kartendaten generieren

Die `Generate Card Data` API wird verwendet, um Kartendaten mithilfe von Algorithmen wie CVV, CVV2 oder Dynamic CVV2 zu generieren. Welche Schlüssel für diesen Befehl verwendet werden können, finden Sie im Abschnitt Gültige Schlüssel für kryptografische Operationen.

Viele kryptografische Werte wie CVV, CVV2, iCVV, CAVV V8 verwenden denselben kryptografischen Algorithmus, variieren jedoch die Eingabewerte. CardVerificationValue1 hat beispielsweise Eingaben für Kartennummer und Ablaufdatum. ServiceCode CardVerificationValue2 hat zwar nur zwei dieser Eingaben, das liegt daran, dass für CVV2/CVC2 der Wert auf 000 festgelegt ist. ServiceCode In ähnlicher Weise ist der Wert für iCVV auf 999 festgelegt. ServiceCode Einige Algorithmen können die vorhandenen Felder wiederverwenden, z. B. CAVV V8. In diesem Fall müssen Sie die korrekten Eingabewerte im Handbuch Ihres Anbieters nachlesen.

> **Note**
>
> Das Ablaufdatum muss für die Generierung und Validierung im gleichen Format eingegeben werden (z. B. MMYY oder YYMM), damit korrekte Ergebnisse erzielt werden.

Generieren Sie CVV2

Example

In diesem Beispiel werden wir einen CVV2 für eine bestimmte PAN mit Eingaben von PAN und dem Ablaufdatum der Karte generieren. Dies setzt voraus, dass Sie einen Kartenbestätigungsschlüssel generiert haben.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2={CardExpiryDate=0123}
```

```
  {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
      "KeyCheckValue": "CADDA1",
      "ValidationData": "801"
  }
```

Generieren Sie iCVV

Example

In diesem Beispiel generieren wir ein iCVV für eine bestimmte PAN mit Eingaben vonPAN, einem Servicecode von 999 und dem Ablaufdatum der Karte. Dies setzt voraus, dass Sie einen Kartenbestätigungsschlüssel generiert haben.

Alle verfügbaren Parameter finden Sie unter CardVerificationValue1 im API-Referenzhandbuch.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999}'
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1",
    "ValidationData": "801"
}
```

Überprüfen Sie die Kartendaten

`Verify Card Data`wird verwendet, um Daten zu verifizieren, die mithilfe von Zahlungsalgorithmen erstellt wurden, die auf Verschlüsselungsprinzipien basieren, wie `DISCOVER_DYNAMIC_CARD_VERIFICATION_CODE` z.

Die Eingabewerte werden in der Regel im Rahmen einer eingehenden Transaktion an einen Emittenten oder einen unterstützenden Plattformpartner weitergegeben. Informationen zur Überprüfung eines ARQC-Kryptogramms (das für EMV-Chipkarten verwendet wird) finden Sie unter ARQC verifizieren.

Weitere Informationen finden Sie im API-Leitfaden. VerifyCardValidationData

Wenn der Wert verifiziert ist, gibt die API http/200 zurück. Wenn der Wert nicht verifiziert ist, wird http/400 zurückgegeben.

Überprüfen Sie CVV2

Example

In diesem Beispiel validieren wir ein CVV/CVV2 für eine bestimmte PAN. Der CVV2 wird in der Regel vom Karteninhaber oder Benutzer während der Transaktion zur Validierung bereitgestellt. Um ihre Eingaben zu validieren, werden zur Laufzeit die folgenden Werte bereitgestellt: Key to Use for Validation (CVK)PAN, Kartenablaufdatum und CVV2 werden eingegeben. Das Kartenablaufformat muss mit dem Format übereinstimmen, das bei der ursprünglichen Wertgenerierung verwendet wurde.

Alle verfügbaren Parameter finden Sie unter CardVerificationValue2 im API-Referenzhandbuch.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue2={CardExpiryDate=0123} --validation-data 801
```

```
  {
      "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
      "KeyCheckValue": "CADDA1"
  }
```

Überprüfen Sie iCVV

Example

In diesem Beispiel verifizieren wir ein iCVV für eine bestimmte PAN mit Eingaben von Key to Use for Validation (CVK), einem Servicecode von 999PAN, dem Ablaufdatum der Karte und dem iCVV, das von der zu validierenden Transaktion bereitgestellt wurde.

iCVV ist kein vom Benutzer eingegebener Wert (wie CVV2), sondern auf einer EMV-Karte eingebettet. Es sollte geprüft werden, ob es immer validiert werden sollte, wenn es bereitgestellt wird.

Alle verfügbaren Parameter finden Sie unter CardVerificationValue1 im API-Referenzhandbuch.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "CADDA1",
    "ValidationData": "801"
}
```

Generieren, übersetzen und verifizieren Sie PIN-Daten

Mit den Funktionen für PIN-Daten können Sie zufällige Pins und PIN-Bestätigungswerte (PVV) generieren und eingehende verschlüsselte Pins anhand von PVV- oder PIN-Offsets validieren.

Mit der Pinübersetzung können Sie eine PIN von einem funktionierenden Schlüssel in einen anderen übersetzen, ohne dass die PIN im Klartext angezeigt wird, wie in PCI-PIN-Anforderung 1 festgelegt.

> **Note**
>
> Da es sich bei der PIN-Generierung und -Validierung in der Regel um Funktionen des Ausstellers handelt und die PIN-Übersetzung eine typische Acquirer-Funktion ist, empfehlen wir Ihnen, den Zugriff mit den geringsten Rechten in Betracht zu ziehen und die Richtlinien entsprechend Ihrem Systemanwendungsfall festzulegen.

Themen

PIN-Daten Translate

Funktionen zum Translate von PIN-Daten werden verwendet, um verschlüsselte PIN-Daten von einem Schlüsselsatz in einen anderen zu übersetzen, ohne dass die verschlüsselten Daten das HSM verlassen. Es wird für die P2PE-Verschlüsselung verwendet, bei der sich die funktionierenden Schlüssel ändern sollten, das Verarbeitungssystem die Daten jedoch entweder nicht entschlüsseln muss oder darf. Die primären Eingaben sind die verschlüsselten Daten, der Verschlüsselungsschlüssel, der zur Verschlüsselung der Daten verwendet wird, die Parameter, die zur Generierung der Eingabewerte verwendet werden. Bei den anderen Eingaben handelt es sich um die angeforderten Ausgabeparameter, z. B. den Schlüssel, der zur Verschlüsselung der Ausgabe verwendet werden soll, und die Parameter, mit denen diese Ausgabe erstellt wurde. Die primären Ausgaben sind ein neu verschlüsselter Datensatz sowie die Parameter, die zu seiner Generierung verwendet wurden.

> **Note**
>
> AES-Schlüsseltypen unterstützen nur 4-polige Blöcke im ISO-Format.

Themen

- PIN von PEK zu DUKPT
- PIN von DUKPT zu AWK

PIN von PEK zu DUKPT

Example

In diesem Beispiel übersetzen wir eine PIN aus der PEK-TDES-Verschlüsselung mithilfe des ISO-0-PIN-Blocks in einen AES-ISO-4-PIN-Block mithilfe des DUKPT-Algorithmus. In der Regel kann dies in umgekehrter Reihenfolge geschehen, wobei ein Zahlungsterminal eine PIN in ISO 4 verschlüsselt und sie dann zur Weiterverarbeitung wieder in TDES übersetzt werden kann.

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "AC17DC148BDA645E" --incoming-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}' --incoming-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt  --outgoing-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/4pmyquwjs3yj4vwe --outgoing-translation-attributes
 IsoFormat4="{PrimaryAccountNumber=171234567890123}" --outgoing-dukpt-attributes
 KeySerialNumber="FFFF9876543210E00008"
```

```
    {
            "PinBlock": "1F4209C670E49F83E75CC72E81B787D9",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
            "KeyCheckValue": "7CC9E2"
        }
```

PIN von DUKPT zu AWK

Example

In diesem Beispiel übersetzen wir eine PIN von einer mit AES DUKPT verschlüsselten PIN in eine PIN, die unter einer AWK verschlüsselt ist. Es ist funktionell das Gegenteil des vorherigen Beispiels.

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-
block "1F4209C670E49F83E75CC72E81B787D9" --outgoing-translation-
```

```
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}' --outgoing-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt   --incoming-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/4pmyquwjs3yj4vwe --incoming-translation-attributes
 IsoFormat4="{PrimaryAccountNumber=171234567890123}" --incoming-dukpt-attributes
 KeySerialNumber="FFFF9876543210E00008"
```

```
        {
            "PinBlock": "AC17DC148BDA645E",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
            "KeyCheckValue": "FE23D3"
        }
```

Generieren Sie PIN-Daten

Die Funktionen zum Generieren von PIN-Daten werden zur Generierung von PIN-bezogenen Werten wie PVV und Pin-Block-Offsets verwendet, die zur Überprüfung der PIN-Eingabe durch Benutzer während der Transaktions- oder Autorisierungszeit verwendet werden. Diese API kann auch mithilfe verschiedener Algorithmen eine neue zufällige PIN generieren.

Generieren Sie Visa PVV für eine PIN

Example

In diesem Beispiel werden wir eine neue (zufällige) PIN generieren, bei der die Ausgänge verschlüsselt sind `PIN block` (PinData. PinBlock) und a PVV (pinData.Offset). Die wichtigsten Eingaben sindPAN, der`Pin Verification Key`, der und der`Pin Encryption Key`. `PIN block format`

Dieser Befehl setzt voraus, dass der Schlüssel vom Typ ist`TR31_V2_VISA_PIN_VERIFICATION_KEY`.

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
```

```
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-
attributes VisaPin={PinVerificationKeyIndex=1}
```

```
{
            "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "GenerationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",
            "EncryptedPinBlock": "AC17DC148BDA645E",
            "PinData": {
                "VerificationValue": "5507"
            }
        }
```

Generieren Sie den IBM3624-Pin-Offset für einen Anschluss

IBM 3624 PIN Offset wird manchmal auch als IBM-Methode bezeichnet. Diese Methode generiert anhand der Validierungsdaten (normalerweise die PAN) und eines PIN-Schlüssels (PVK) eine natürliche/zwischengeschaltete PIN. Natürliche Pins sind quasi ein abgeleiteter Wert, und da sie deterministisch sind, sind sie für einen Emittenten sehr effizient zu handhaben, da keine PIN-Daten auf Karteninhaberebene gespeichert werden müssen. Der offensichtlichste Nachteil ist, dass dieses Schema vom Karteninhaber auswählbare oder zufällige Pins nicht berücksichtigt. Um diese Arten von Pins zu berücksichtigen, wurde dem Schema ein Offset-Algorithmus hinzugefügt. Der Offset stellt den Unterschied zwischen dem vom Benutzer ausgewählten (oder zufälligen) Pin und dem natürlichen Schlüssel dar. Der Offsetwert wird vom Kartenaussteller oder Kartenverarbeiter gespeichert. Zum Zeitpunkt der Transaktion berechnet der AWS Payment Cryptography Service intern die natürliche PIN neu und wendet den Offset an, um die PIN zu ermitteln. Dieser Wert wird dann mit dem Wert verglichen, der in der Transaktionsautorisierung angegeben wurde.

Für IBM 3624 gibt es mehrere Optionen:

- `Ibm3624NaturalPin`gibt die natürliche PIN und einen verschlüsselten Pin-Block aus
- `Ibm3624PinFromOffset`generiert bei gegebenem Offset einen verschlüsselten Pin-Block
- `Ibm3624RandomPin`generiert eine zufällige PIN und dann den passenden Offset und den verschlüsselten Pinblock.
- `Ibm3624PinOffset`generiert den Pin-Offset anhand einer vom Benutzer ausgewählten PIN.

Bei der AWS Zahlungskryptografie werden die folgenden Schritte ausgeführt:

- Füllen Sie das bereitgestellte Feld mit 16 Zeichen aus. Wenn <16 angegeben sind, füllen Sie den Text mit dem angegebenen Füllzeichen auf der rechten Seite auf.
- Verschlüsselt die Validierungsdaten mithilfe des PIN-Generierungsschlüssels.
- Dezimalisieren Sie die verschlüsselten Daten mithilfe der Dezimalisierungstabelle. Dadurch werden Hexadezimalziffern Dezimalziffern zugeordnet, zum Beispiel kann 'A' 9 und 1 1 1 zugeordnet werden.
- Ermittelt die ersten 4 Ziffern aus einer Hexadezimaldarstellung der Ausgabe. Das ist der natürliche Stift.
- Wenn ein Benutzer eine PIN ausgewählt hat oder eine zufällige PIN generiert wurde, subtrahiert Modulo die natürliche PIN mit der Kunden-PIN. Das Ergebnis ist der Pin-Offset.

Beispiele

- Generieren Sie den IBM3624-Pin-Offset für einen Pin

Generieren Sie den IBM3624-Pin-Offset für einen Pin

In diesem Beispiel werden wir einen neuen (zufälligen) Pin generieren, bei dem die Ausgänge verschlüsselt sind (. `PIN block` PinData PinBlock) und einen `IBM3624` Offset-Wert (pinData.Offset). Die Eingaben sind `PAN` Validierungsdaten (normalerweise der Pan), das Füllzeichen, das`Pin Verification Key`, das und das. `Pin Encryption Key` `PIN block format`

Für diesen Befehl müssen der Pin-Generierungsschlüssel vom Typ `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` und der Verschlüsselungsschlüssel vom Typ `TR31_P0_PIN_ENCRYPTION_KEY`

Example

Das folgende Beispiel zeigt die Generierung einer zufälligen PIN und die anschließende Ausgabe des verschlüsselten Pinblocks und des IBM3624-Offsetwerts mithilfe von Ibm3624 RandomPin

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt --primary-account-number
 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-attributes
 Ibm3624RandomPin="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
              "GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
              "GenerationKeyCheckValue": "7F2363",
              "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
              "EncryptionKeyCheckValue": "7CC9E2",
              "EncryptedPinBlock": "AC17DC148BDA645E",
              "PinData": {
                  "PinOffset": "5507"
              }
          }
```

Überprüfen Sie die PIN-Daten

Mit den Funktionen zur Überprüfung der PIN-Daten wird überprüft, ob eine PIN korrekt ist. Dabei wird in der Regel der zuvor gespeicherte PIN-Wert mit dem verglichen, den der Karteninhaber an einem POI eingegeben hat. Diese Funktionen vergleichen zwei Werte, ohne den zugrunde liegenden Wert einer der beiden Quellen offenzulegen.

Überprüfen Sie die verschlüsselte PIN mit der PVV-Methode

Example

In diesem Beispiel validieren wir eine PIN für eine bestimmte PAN. Die PIN wird in der Regel vom Karteninhaber oder Benutzer während der Transaktion zur Validierung bereitgestellt und mit dem in der Datei hinterlegten Wert verglichen (die Eingabe des Karteninhabers wird als verschlüsselter Wert vom Terminal oder einem anderen Upstream-Anbieter bereitgestellt). Um diese Eingabe zu validieren, werden zur Laufzeit auch die folgenden Werte bereitgestellt: Der Schlüssel, mit dem die eingegebene PIN verschlüsselt wurde (dies ist häufig eine`IWK`), PAN und der Wert, anhand dessen verifiziert werden soll (entweder ein `PVV` oder). `PIN offset`

Wenn AWS Payment Cryptography die PIN validieren kann, wird ein http/200 zurückgegeben. Wenn die PIN nicht validiert wird, wird ein http/400 zurückgegeben.

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2 --encryption-
key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt
 --primary-account-number 171234567890123 --pin-block-format ISO_FORMAT_0  --
verification-attributes VisaPin="{PinVerificationKeyIndex=1,VerificationValue=5507}" --
encrypted-pin-block AC17DC148BDA645E
```

```
        {
            "VerificationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "VerificationKeyCheckValue": "7F2363",
            "EncryptionKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt",
            "EncryptionKeyCheckValue": "7CC9E2",

        }
```

Validieren Sie eine PIN anhand des zuvor gespeicherten IBM3624-Pin-Offsets.

In diesem Beispiel werden wir die von einem Karteninhaber eingegebene PIN mit dem PIN-Offset vergleichen, der in der Datei beim Kartenaussteller/-verarbeiter gespeichert ist. Die Eingaben sind ähnlich wie ??? bei der zusätzlichen verschlüsselten PIN, die vom Zahlungsterminal (oder einem anderen Upstream-Anbieter wie dem Kartennetzwerk) bereitgestellt wird. Wenn die PIN übereinstimmt, gibt die API http 200 zurück, wobei die Ausgaben verschlüsselt sind `PIN block` (PinData. PinBlock) und einen `IBM3624` Offsetwert (pinData.Offset).

Dieser Befehl erfordert, dass der Schlüssel zur PIN-Generierung vom Typ `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` und der Verschlüsselungsschlüssel vom Typ `TR31_P0_PIN_ENCRYPTION_KEY`

Example

```
$ aws payment-cryptography-data generate-pin-data --generation-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt --primary-account-number
 171234567890123 --pin-block-format ISO_FORMAT_0 --generation-attributes
 Ibm3624RandomPin="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
"GenerationKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
"GenerationKeyCheckValue": "7F2363",
```

```
"EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
"EncryptionKeyCheckValue": "7CC9E2",
"EncryptedPinBlock": "AC17DC148BDA645E",
"PinData": {
  "PinOffset": "5507"
}
}
```

Kryptogramm für Authentifizierungsanfragen (ARQC) verifizieren

Die Kryptogramm-API für Verify Auth Requests wird zur Überprüfung von ARQC verwendet. Die Generierung des ARQC fällt nicht in den Anwendungsbereich der AWS Zahlungskryptografie und wird in der Regel während der Transaktionsautorisierung auf einer EMV-Chipkarte (oder einem digitalen Äquivalent wie einer mobilen Geldbörse) durchgeführt. Ein ARQC ist für jede Transaktion einzigartig und dient dazu, sowohl die Gültigkeit der Karte kryptografisch nachzuweisen als auch sicherzustellen, dass die Transaktionsdaten exakt mit der aktuellen (erwarteten) Transaktion übereinstimmen.

AWS Die Zahlungskryptografie bietet eine Vielzahl von Optionen zur Validierung von ARQC und zur Generierung optionaler ARPC-Werte, einschließlich der in EMV 4.4 Buch 2 definierten und anderen von Visa und Mastercard verwendeten Schemata. Eine vollständige Liste aller verfügbaren Optionen finden Sie im Abschnitt im VerifyCardValidationData API-Leitfaden.

ARQC-Kryptogramme erfordern normalerweise die folgenden Eingaben (obwohl dies je nach Implementierung variieren kann):

- PAN — Im Feld angegeben PrimaryAccountNumber
- PAN-Sequenznummer (PSN) — im PanSequenceNumber Feld angegeben
- Methode zur Schlüsselableitung wie Common Session Key (CSK) — Spezifiziert in SessionKeyDerivationAttributes
- Hauptschlüsselableitungsmodus (z. B. EMV-Option A) — Spezifiziert in MajorKeyDerivationMode
- Transaktionsdaten — eine Zeichenfolge verschiedener Transaktions-, Terminal- und Kartendaten wie Betrag und Datum —, die im Feld angegeben sind TransactionData
- Hauptschlüssel des Ausstellers — der Hauptschlüssel, der zur Ableitung des Kryptogrammschlüssels (AC) verwendet wird, der zum Schutz einzelner Transaktionen verwendet und im Feld angegeben ist KeyIdentifier

Themen

Transaktionsdaten erstellen

Der genaue Inhalt (und die Reihenfolge) des Transaktionsdatenfeldes variiert je nach Implementierung und Netzwerkschema, aber die empfohlenen Mindestfelder (und die Verkettungsreihenfolge) sind in EMV 4.4, Buch 2, Abschnitt 8.1.1 — Datenauswahl, definiert. Wenn die ersten drei Felder Betrag (17,00), sonstiger Betrag (0,00) und Land des Kaufs lauten, würde dies dazu führen, dass die Transaktionsdaten wie folgt beginnen würden:

- 000000001700 — Betrag — 12 Stellen impliziert eine zweistellige Dezimalzahl
- 000000000000 — anderer Betrag — 12 Stellen impliziert eine zweistellige Dezimalzahl
- 0124 — vierstelliger Ländercode
- Transaktionsdaten (teilweise) ausgeben - 00000000170000000000000000124

Auffüllen von Transaktionsdaten

Transaktionsdaten sollten vor dem Senden an den Dienst aufgefüllt werden. Die meisten Schemas verwenden das Auffüllen nach ISO 9797 Methode 2. Dabei wird an eine Hexadezimalzahl 80 gefolgt von 00 angehängt, bis das Feld ein Vielfaches der Größe des Verschlüsselungsblocks ist: 8 Byte oder 16 Zeichen für TDES und 16 Byte oder 32 Zeichen für AES. Die Alternative (Methode 1) ist nicht so üblich, verwendet aber nur 00 als Füllzeichen.

ISO 9797 Methode 1: Innenabstand

Ohne Füllung: 00000000170000000000000084000800080000840160517000000000093800000B03011203 (74 Zeichen oder 37 Byte)

Gepolstert: 00000000170000000000084000800080000840160517000000000093800000B03011203 000000 (80 Zeichen oder 40 Byte)

Polsterung nach ISO 9797 Methode 2

Ohne Füllung:
000000001700000000000000084000800080000840160517000000000938000 00B1F220103000000
(80 Zeichen oder 40 Byte)

Gepolstert:
000000001700000000000000084000800080000840160517000000000938000 00B1F220103000000
8000000000000000 (88 Zeichen oder 44 Byte)

Beispiele

Visum CVN10

Example

In diesem Beispiel validieren wir einen mit Visa CVN10 generierten ARQC.

Wenn AWS Payment Cryptography den ARQC validieren kann, wird ein http/200 zurückgegeben. Wenn der ARQC nicht validiert wird, gibt er eine http/400-Antwort zurück.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram D791093C8A921769 \
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A \
--transaction-data
 000000001700000000000000084000800080000840160517000000000938000 00B03011203000000 \
--session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
,"PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
  "KeyCheckValue": "08D7B4"
}
```

Visa CVN-18 und Visa CVN-22.

Example

In diesem Beispiel validieren wir einen ARQC, der mit Visa CVN18 oder CVN22 generiert wurde. Die kryptografischen Operationen zwischen CVN18 und CVN22 sind dieselben, aber die in den

Transaktionsdaten enthaltenen Daten variieren. Im Vergleich zu CVN10 wird selbst bei denselben Eingaben ein völlig anderes Kryptogramm generiert.

Wenn AWS Payment Cryptography den ARQC validieren kann, wird ein http/200 zurückgegeben. Wenn der ARQC nicht validiert ist, wird ein http/400 zurückgegeben.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram \
--auth-request-cryptogram 61EDCC708B4C97B4
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A
--transaction-data
 00000000170000000000000084000800080000840160517000000000093800000B1F22010300000000000
 \
00000000000000000000000000000000000000000800000000000000
--session-key-derivation-attributes='{"EmvCommon":
{"ApplicationTransactionCounter":"000B", \
"PanSequenceNumber":"01","PrimaryAccountNumber":"9137631040001422"}}'
```

```
{
  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
  "KeyCheckValue": "08D7B4"
}
```

MAC generieren und verifizieren

Message Authentication Codes (MAC) werden normalerweise verwendet, um die Integrität einer Nachricht zu authentifizieren (unabhängig davon, ob sie geändert wurde). Kryptografische Hashes wie HMAC (Hash-Based Message Authentication Code), CBC-MAC und CMAC (Cipher-based Message Authentication Code) bieten durch den Einsatz von Kryptografie zusätzliche Sicherheit beim Absender des MAC. HMAC basiert auf Hash-Funktionen, während CMAC auf Blockchiffren basiert.

Alle MAC-Algorithmen dieses Dienstes kombinieren eine kryptografische Hash-Funktion und einen gemeinsamen geheimen Schlüssel. Sie nehmen eine Nachricht und einen geheimen Schlüssel, z. B. das Schlüsselmaterial in einem Schlüssel, und geben ein eindeutiges Tag oder einen eindeutigen Mac zurück. Wenn sich auch nur ein Zeichen der Nachricht ändert oder wenn sich der geheime Schlüssel ändert, ist das resultierende Tag völlig anders. Durch die Anforderung eines geheimen Schlüssels bietet kryptografische MACs auch Authentizität. Ohne den geheimen Schlüssel ist es unmöglich, einen identischen Mac zu generieren. Kryptografische MACs werden manchmal als

symmetrische Signaturen bezeichnet, weil sie wie digitale Signaturen funktionieren, aber einen einzigen Schlüssel sowohl für das Signieren als auch für die Überprüfung verwenden.

AWSDie Zahlungskryptografie unterstützt verschiedene Arten von MACs:

ISO9797, ALGORITHMUS 1

Bezeichnet mit oder ISO9797_ALGORITHM1 `KeyUsage`

ISO9797-ALGORITHMUS 3 (MAC für den Einzelhandel)

Bezeichnet mit oder ISO9797_ALGORITHM3 `KeyUsage`

ISO9797-ALGORITHMUS 5 (CMAC)

Wird mit TR31_M6_ISO_9797_5_CMAC_KEY bezeichnet `KeyUsage`

HMAC

Wird mit `KeyUsage` TR31_M7_HMAC_KEY bezeichnet, einschließlich HMAC_SHA224, HMAC_SHA256, HMAC_SHA384 und HMAC_SHA512

Themen

MAC generieren

Die Generate MAC API wird verwendet, um kartenbezogene Daten zu authentifizieren, z. B. Trackdaten von einem Kartenmagnetstreifen, indem bekannte Datenwerte verwendet werden, um einen MAC (Message Authentication Code) für die Datenvalidierung zwischen sendenden und empfangenden Parteien zu generieren. Die zur Generierung von MAC verwendeten Daten umfassen Nachrichtendaten, einen geheimen MAC-Verschlüsselungsschlüssel und einen MAC-Algorithmus zur Generierung eines eindeutigen MAC-Werts für die Übertragung. Die empfangende Partei des MAC verwendet dieselben MAC-Nachrichtendaten, denselben MAC-Verschlüsselungsschlüssel und denselben Algorithmus, um einen anderen MAC-Wert für den Vergleich und die Datenauthentifizierung zu reproduzieren. Selbst wenn sich ein Zeichen der Nachricht ändert oder der zur Überprüfung verwendete MAC-Schlüssel nicht identisch ist, ist der resultierende MAC-Wert unterschiedlich. Die API unterstützt DUPKT MAC-, HMAC- und EMV-MAC-Verschlüsselungsschlüssel für diesen Vorgang.

Der Eingabewert für `message-data` muss HexBinary-Daten sein.

In diesem Beispiel werden wir einen HMAC (Hash-Based Message Authentication Code) für die Kartendatenauthentifizierung mithilfe des HMAC-Algorithmus `HMAC_SHA256` und des HMAC-Verschlüsselungsschlüssels generieren. Der Schlüssel muss auf und auf KeyUsage eingestellt sein. `TR31_M7_HMAC_KEY` KeyModesOfUse `Generate` Der MAC-Schlüssel kann entweder mit AWS Payment Cryptography per Anruf erstellt CreateKeyoder per Anruf importiert werden. ImportKey

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes Algorithm=HMAC_SHA256
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6,
    "KeyCheckValue": "2976E7",
    "Mac": "ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C"
}
```

Überprüfen Sie den MAC

Überprüfen Sie, ob die MAC-API verwendet wird, um den MAC (Message Authentication Code) für die kartenbezogene Datenauthentifizierung zu verifizieren. Es muss derselbe Verschlüsselungsschlüssel verwendet werden, der bei der MAC-Generierung verwendet wurde, um den MAC-Wert für die Authentifizierung zu reproduzieren. Der MAC-Verschlüsselungsschlüssel kann entweder mit AWS Payment Cryptography per Anruf erstellt CreateKeyoder per Anruf importiert werden. ImportKey Die API unterstützt DUPKT MAC-, HMAC- und EMV-MAC-Verschlüsselungsschlüssel für diesen Vorgang.

Wenn der Wert verifiziert ist, kehrt der Antwortparameter zurück`Http/200`, andernfalls `MacDataVerificationSuccessful` wird eine `Http/400` entsprechende Meldung angezeigt. `Mac verification failed`

In diesem Beispiel verifizieren wir einen HMAC (Hash-Based Message Authentication Code) für die Kartendatenauthentifizierung mithilfe des HMAC-Algorithmus `HMAC_SHA256` und des HMAC-Verschlüsselungsschlüssels. Der Schlüssel muss auf und auf KeyUsage eingestellt sein. `TR31_M7_HMAC_KEY` KeyModesOfUse `Verify`

Example

```
$ aws payment-cryptography-data verify-mac \
     --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
     --message-data
 "3b343038383439303031303733393431353d323430383232363030303730303033f33" \
     --verification-attributes='Algorithm=HMAC_SHA256' \
     --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6,
    "KeyCheckValue": "2976E7",
}
```

Gültige Schlüssel für kryptografische Operationen

Bestimmte Schlüssel können nur für bestimmte Operationen verwendet werden. Darüber hinaus können einige Operationen die wichtigsten Verwendungsmodi für Schlüssel einschränken. Die zulässigen Kombinationen finden Sie in der folgenden Tabelle.

> **Note**
>
> Bestimmte Kombinationen sind zwar zulässig, können aber zu unbrauchbaren Situationen führen, z. B. beim Generieren von CVV-Codes, die dann `(generate)` aber nicht verifiziert werden können. `(verify)`

Themen

- GenerateCardDaten
- VerifyCardDaten

- GeneratePinData (für VISA/ABA-Programme)
- GeneratePinData (fürIBM3624)
- VerifyPinData (für VISA/ABA-Programme)
- VerifyPinData (fürIBM3624)
- Daten entschlüsseln
- Encrypt Data
- Translate Pin Data
- MAC generieren/verifizieren
- VerifyAuthRequestCryptogram
- Schlüssel Import/Export
- Unbenutzte Schlüsseltypen

GenerateCardDaten

API-Endpunkt	Kryptografischer Vorgang oder Algorithmus	Zulässige Verwendung von Schlüsseln	Zulässige r Schlüssel algorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
GenerateCardDaten	• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SICHERHEITSCODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	{Generieren = wahr}, {Generieren = wahr, Überprüfen = wahr}
GenerateCardDaten	• CARD_VERIFICATION VALUE_1	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2-SCHLÜSSEL	{Generieren = wahr}, {Generieren = wahr, Überprüfen = wahr}

API-Endpunkt	Kryptografischer Vorgang oder Algorithmus	Zulässige Verwendung von Schlüsseln	Zulässige r Schlüssel algorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
	• WERT_2 FÜR DIE KARTENÜBERPRÜFUNG			
GenerateCardDaten	• WERT FÜR DIE AUTHENTIFIZIERUNG DES KARTENINHABERS	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{= wahr} DeriveKey
GenerateCardDaten	• DYNAMIC_CARD_VERIFICATION CODE	TR31_E4_EMV_MKEY_DYNAMISCHE_ZAHLEN	• TDES_2KEY	{= wahr} DeriveKey
GenerateCardDaten	• DYNAMIC_CARD_VERIFICATION VALUE	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{= wahr} DeriveKey

VerifyCardDaten

Kryptografischer Vorgang oder Algorithmus	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüssel algorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
• AMEX_CARD_SECURITY	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2-SCHLÜSSEL	{Generieren = wahr}, {Generieren = wahr, Überprüfen = wahr}

Kryptografischer Vorgang oder Algorithmus	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
_CODE_VERSION_1 • AMEX_CARD_SICHERHEITSCODE_VERSION_2		• TDES_3KEY	
• CARD_VERIFICATION_VALUE_1 • WERT_2 FÜR DIE KARTENÜBERPRÜFUNG	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2-SCHLÜSSEL	{Generieren = wahr}, {Generieren = wahr, Überprüfen = wahr}
• WERT DER AUTHENTIFIZIERUNG_DES KARTENINHABERS	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{= wahr} DeriveKey
• VERIFIZIERUNGSCODE FÜR DYNAMISCHE_KARTEN	TR31_E4_EMV_MKEY_DYNAMISCHE_ZAHLEN	• TDES_2KEY	{= wahr} DeriveKey
• WERT FÜR DIE ÜBERPRÜFUNG DER DYNAMISCHEN KARTE	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{= wahr} DeriveKey

GeneratePinData (für VISA/ABA-Programme)

`VISA_PIN or VISA_PIN_VERIFICATION_VALUE`

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
PIN-Verschlüsselungsschlüssel	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	• {Verschlüsseln = wahr, Wrap = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Umbrechen = wahr, Auspacken = wahr} • {= wahr} NoRestrictions
Schlüssel zur PIN-Generierung	TR31_V2_VISA_PIN_VERIFICATION KEY	• TDES_3-SCHLÜSSEL	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr}

GeneratePinData (für**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
PIN-Verschlüsselungsschlüssel	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	Für IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
			• {Verschlüsseln = wahr, Wrap = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Umbrechen = wahr, Auspacken = wahr} • {= wahr} NoRestrictions Für IBM3624_PIN_OFFSET • {Verschlüsseln = wahr, Auspacken = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Umbrechen = wahr, Auspacken = wahr} • {= wahr} NoRestrictions
Schlüssel zur PIN-Generierung	TR31_V1_IBM3624_PIN_VERIFICATION KEY	• TDES_3-SCHLÜSSEL	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr}

VerifyPinData (für VISA/ABA-Programme)

`VISA_PIN`

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
PIN-Verschlüsselungsschlüssel	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	• {Entschlüsseln = wahr, Auspacken = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Wrap = wahr, Auspacken = wahr} • {= wahr} NoRestrictions
Schlüssel zur PIN-Generierung	TR31_V2_VISA_PIN_VERIFICATION KEY	• TDES_3-SCHLÜSSEL	• {Verifizieren = wahr} • {Generieren = wahr, Überprüfen = wahr}

VerifyPinData (für**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
PIN-Verschlüsselungsschlüssel	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	Für IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
			• {Entschlüsseln = wahr, Auspacken = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Wrap = wahr, Auspacken = wahr} • {= wahr} NoRestrictions
PIN-Bestätigungsschlüssel	TR31_V1_IBM3624_PIN_VERIFICATION KEY	• TDES_3-SCHLÜSSEL	• {Verifizieren = wahr} • {Generieren = wahr, Überprüfen = wahr}

Daten entschlüsseln

Schlüsseltyp	Zulässige Schlüsselverwendung	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
DUMPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2-SCHLÜSSEL • AES_128 • AES_192 • AES_256	• { DeriveKey = wahr} • { NoRestrictions = wahr}
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITY	• TDES_2KEY	• {= wahr} DeriveKey

Schlüsseltyp	Zulässige Schlüsselverwendung	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
	TR31_E6_EMV_MKEY_SONSTIGES		
RSA	TR31_D1_ASYMMETRISCHER_SCHLÜSSEL_FÜR_DATENVERSCHLÜSSELUNG	• RSA_2048 • RSA_3072 • RSA_4096	• {Entschlüsseln = wahr, entpacken = wahr} • {Encrypt=Wahr, Wrap=Wahr, Entschlüsseln = wahr, Unwrap=Wahr}
Symmetrische Schlüssel	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Entschlüsseln = wahr, entpacken = wahr} • {Encrypt=Wahr, Wrap=Wahr, Entschlüsseln = wahr, Unwrap=Wahr} • NoRestrictions {= wahr}

Encrypt Data

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
DUMPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2-SCHLÜSSEL • AES_128 • AES_192 • AES_256	• { DeriveKey = wahr} • { NoRestrictions = wahr}
EMV	TR31_E1_EMV_MKEY_CONFIDENTIALITY TR31_E6_EMV_MKEY_SONSTIGES	• TDES_2KEY	• {= wahr} DeriveKey
RSA	TR31_D1_ASYMMETRISCHER_SCHLÜSSEL_FÜR_DATENVERSCHLÜSSELUNG	• RSA_2048 • RSA_3072 • RSA_4096	• {Verschlüsseln = wahr, Wrap=wahr} • {Encrypt=Wahr, Wrap=Wahr, Entschlüsseln = wahr, Unwrap=Wahr}
Symmetrische Schlüssel	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Verschlüsseln = wahr, Wrap=wahr} • {Encrypt=Wahr, Wrap=Wahr, Entschlüsseln = wahr, Unwrap=Wahr}

Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
			• NoRestrictions {= wahr}

Translate Pin Data

Richtung	Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
Eingehende Datenquelle	DEPUTIERT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2-SCHLÜSSEL • AES_128 • AES_192 • AES_256	• { DeriveKey = wahr} • { NoRestrictions = wahr}
Eingehende Datenquelle	Unverfälschte Daten (PEK, AWK, IWK usw.)	TR31_P0_PIN_VERSCHLÜSSELUNGSSCHLÜSSEL	• TDES_2-SCHLÜSSEL • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Entschlüsseln = wahr, Auspacken = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Wrap = wahr, Auspacken = wahr} • {= wahr} NoRestrictions

Richtung	Schlüsseltyp	Zulässige Verwendung von Schlüsseln	Zulässige r Schlüssel algorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
Ziel für ausgehende Daten	DUPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2-SCHLÜSSEL • AES_128 • AES_192 • AES_256	• { DeriveKey = wahr} • { NoRestrictions = wahr}
Ziel für ausgehende Daten	Unmanipuliert (PEK, IWK, AWK usw.)	TR31_P0_PIN_VERSCHLÜSSELUNGSSCHLÜSSEL	• TDES_2-SCHLÜSSEL • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Verschlüsseln = wahr, Wrap = wahr} • {Verschlüsseln = wahr, Entschlüsseln = wahr, Umbrechen = wahr, Auspacken = wahr} • {= wahr} NoRestrictions

MAC generieren/verifizieren

MAC-Schlüssel werden verwendet, um kryptografische Hashes für eine Nachricht/einen Datentext zu erstellen. Es wird nicht empfohlen, einen Schlüssel mit eingeschränkten Verwendungsmöglichkeiten zu erstellen, da Sie den Abgleichvorgang dann nicht durchführen können. Sie können jedoch einen Schlüssel mit nur einer Operation importieren/exportieren, wenn das andere System die andere Hälfte des Operationspaars ausführen soll.

Zulässige Verwendung von Schlüsseln	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
MAC-Schlüssel	TR31_M1_ISO_9797_1_MAC_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr} • {Verifizieren = wahr} • {Generieren = wahr}
MAC-Schlüssel (MAC für den Einzelhandel)	TR31_M1_ISO_9797_3_MAC_KEY	• TDES_2KEY • TDES_3KEY	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr} • {Verifizieren = wahr} • {Generieren = wahr}
MAC-Schlüssel (CMAC)	TR31_M6_ISO_9797_5_CMAC_KEY	• TDES_2-TASTE • TDES_3KEY • AES_128 • AES_192 • AES_256	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr} • {Verifizieren = wahr} • {Generieren = wahr}
MAC-Schlüssel (HMAC)	TR31_M7_HMAC_KEY	• TDES_2KEY • TDES_3KEY • AES_128 • AES_192	• {Generieren = wahr} • {Generieren = wahr, Überprüfen = wahr}

Zulässige Verwendung von Schlüsseln	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
		• AES_256	• {Verifizieren = wahr} • {Generieren = wahr}

VerifyAuthRequestCryptogram

Zulässige Verwendung von Schlüsseln	EMV-Option	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
• OPTION A • OPTION B	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• TDES_2KEY	• {= wahr} DeriveKey

Schlüssel Import/Export

Vorgangstyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
TR-31 Wickelschlüssel	TR31_K1_KEY_BLOCK_PROTECTION_KEY TR31_K0_KEY_ENCRYPTION_KEY	• TDES_2-SCHLÜSSEL • TDES_3KEY • AES_128	• {Encrypt = true, Wrap = true} (nur Export) • {Decrypt = true, Unwrap = true} (nur Import) • {Verschlüsseln = wahr, Entschlüsseln = wahr, Wrap =

Vorgangstyp	Zulässige Verwendung von Schlüsseln	Zulässiger Schlüsselalgorithmus	Zulässige Kombination der wichtigsten Nutzungsmodi
			wahr, Auspacken = wahr}
Import einer vertrauenswürdigen CA	TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE	• RSA_2048 • RSA_3072 • RSA_4096	• {Verifizieren = wahr}
Import eines Public-Key-Zertifikats für asymmetrische Verschlüsselung	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION	• RSA_2048 • RSA_3072 • RSA_4096	• {encrypt=Wahr, wrap=Wahr}

Unbenutzte Schlüsseltypen

Die folgenden Schlüsseltypen werden derzeit nicht von AWS Payment Cryptography verwendet

- TR31_P1_PIN_GENERATION_KEY
- TR31_K3_ASYMMETRISCHER SCHLÜSSEL ZUR SCHLÜSSELVEREINBARUNG

Sicherheit in der Zahlungskryptografie AWS

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der übergreifenden Verantwortlichkeit beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud —AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für AWS Zahlungskryptografie gelten, finden Sie unter AWS-Services im Umfang nach Compliance-Programm .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. 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.

In diesem Thema erfahren Sie, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von AWS Zahlungskryptografie anwenden können. Es zeigt Ihnen, wie Sie die AWS Zahlungskryptografie konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Ressourcen zur AWS Zahlungskryptografie überwachen und sichern können.

Themen

Datenschutz in der AWS Zahlungskryptografie

Das Modell der AWS gemeinsamen Verantwortung gilt für den Datenschutz in der AWS Zahlungskryptografie. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der AWS Cloud alle Systeme laufen. 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, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. 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, um mit Ressourcen zu kommunizieren. AWS Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen 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 AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-2-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2.

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 AWS Payment Cryptography oder anderen Methoden arbeiten und die Konsole, die AWS-Services API oder SDKs verwenden. AWS CLI AWS 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.

AWS Payment Cryptography speichert und schützt Ihre Verschlüsselungsschlüssel für Zahlungen, um sie hochverfügbar zu machen und Ihnen gleichzeitig eine starke und flexible Zugriffskontrolle zu bieten.

Themen

Schutz von Schlüsselmaterial

Standardmäßig schützt AWS Payment Cryptography das kryptografische Schlüsselmaterial für Zahlungsschlüssel, die vom Service verwaltet werden. Darüber hinaus bietet AWS Payment Cryptography Optionen für den Import von Schlüsselmaterial, das außerhalb des Services erstellt wurde. Technische Informationen zu Zahlungsschlüsseln und Schlüsselmaterial finden Sie unter AWS Payment Cryptography Cryptography Cryptographic Details.

Datenverschlüsselung

Die Daten in AWS Payment Cryptography bestehen aus AWS Payment Cryptography-Schlüsseln, dem darin enthaltenen Verschlüsselungsschlüsselmaterial und ihren Nutzungsattributen. Schlüsselmaterial ist nur im Klartext in den Hardware-Sicherheitsmodulen (HSM) von AWS Payment Cryptography und nur dann verfügbar, wenn es verwendet wird. Andernfalls werden das Schlüsselmaterial und die Schlüsselattribute verschlüsselt und in einem dauerhaften persistenten Speicher gespeichert.

Das Schlüsselmaterial, das AWS Payment Cryptography für Zahlungsschlüssel generiert oder lädt, verlässt niemals unverschlüsselt die Grenzen von AWS Payment Cryptography HSMs. Es kann verschlüsselt durch API-Operationen von AWS Payment Cryptography exportiert werden.

Verschlüsselung im Ruhezustand

AWS Payment Cryptography generiert Schlüsselmaterial für Zahlungsschlüssel in PCI PTS HSM-gelisteten HSMs. Bei Nichtgebrauch wird Schlüsselmaterial durch einen HSM-Schlüssel verschlüsselt und in dauerhaftem persistenten Speicher geschrieben. Das Schlüsselmaterial für Payment Cryptography Keys und die Verschlüsselungsschlüssel, die das Schlüsselmaterial schützen, verlassen die HSMs niemals im Klartext-Format.

Die Verschlüsselung und Verwaltung des Schlüsselmaterials für Schlüssel zur Zahlungskryptografie erfolgt vollständig durch den Dienst.

Weitere Informationen finden Sie unter Kryptografische Details des AWS Key Management Service.

Verschlüsselung während der Übertragung

Schlüsselmaterial, das AWS Payment Cryptography für Zahlungsschlüssel generiert oder lädt, wird bei API-Vorgängen von AWS Payment Cryptography niemals im Klartext exportiert oder übertragen. AWS Payment Cryptography verwendet Schlüsselkennungen, um die Schlüssel bei API-Vorgängen darzustellen.

Einige API-Operationen von AWS Payment Cryptography exportieren jedoch Schlüssel, die mit einem zuvor gemeinsam genutzten oder asymmetrischen Schlüsselaustauschschlüssel verschlüsselt wurden. Außerdem können Kunden API-Operationen verwenden, um verschlüsseltes Schlüsselmaterial für Zahlungsschlüssel zu importieren.

Alle API-Aufrufe von AWS Payment Cryptography müssen signiert und mit Transport Layer Security (TLS) übertragen werden. AWS Payment Cryptography erfordert TLS-Versionen und Cipher Suites, die von PCI als „starke Kryptografie“ definiert wurden. Alle Service-Endpunkte unterstützen TLS 1.0—1.3 und hybrides Post-Quantum-TLS.

Weitere Informationen finden Sie unter Kryptografische Details des AWS Key Management Service.

Richtlinie für den Datenverkehr zwischen Netzwerken

AWS Payment Cryptography unterstützt eine AWS-Managementkonsole und eine Reihe von API-Vorgängen, mit denen Sie Zahlungsschlüssel erstellen und verwalten und sie für kryptografische Operationen verwenden können.

AWS Pament Cryptography unterstützt zwei Netzwerkverbindungsoptionen von Ihrem privaten Netzwerk zu AWS.

- Eine IPSec-VPN-Verbindung über das Internet.
- AWS Direct Connect, das Ihr internes Netzwerk über ein Standard-Ethernet-Glasfaserkabel mit einem AWS Direct Connect-Standort verbindet.

Alle API-Aufrufe für Zahlungskryptografie müssen signiert und mithilfe von Transport Layer Security (TLS) übertragen werden. Die Anrufe erfordern auch eine moderne Verschlüsselungssammlung, die Perfect Forward Secrecy unterstützt. Der Datenverkehr zu den Hardware-Sicherheitsmodulen (HSMs), die Schlüsselmaterial für Zahlungsschlüssel speichern, ist nur von bekannten AWS Payment Cryptography API-Hosts über das interne AWS-Netzwerk zulässig.

Verwenden Sie VPC-Endpunkte, die von AWS bereitgestellt werden, um von Ihrer Virtual Private Cloud (VPC) aus eine direkte Verbindung zu AWS Payment Cryptography herzustellen, ohne Datenverkehr über das öffentliche Internet zu senden. PrivateLink Weitere Informationen finden Sie unter Herstellen einer Verbindung zu AWS Payment Cryptography über einen VPC-Endpunkt.

AWS Payment Cryptography unterstützt auch eine hybride Post-Quantum-Schlüsselaustauschoption für das Netzwerkverschlüsselungsprotokoll Transport Layer Security (TLS). Sie können diese Option mit TLS verwenden, wenn Sie eine Verbindung zu AWS Payment Cryptography API-Endpunkten herstellen.

Resilienz in AWS der Zahlungskryptografie

AWS Die globale Infrastruktur basiert auf AWS Regionen und Availability Zones. 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.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Regionale Isolierung

AWS Payment Cryptography ist ein regionaler Service, der in mehreren Regionen verfügbar ist.

Das regional isolierte Design von AWS Payment Cryptography stellt sicher, dass ein Verfügbarkeitsproblem in einer AWS-Region den Betrieb von AWS Payment Cryptography in

keiner anderen Region beeinträchtigen kann. AWS Payment Cryptography wurde entwickelt, um sicherzustellen, dass keine geplanten Ausfallzeiten auftreten, da alle Softwareupdates und Skalierungsvorgänge nahtlos und unmerklich ausgeführt werden.

Das Service Level Agreement (SLA) von AWS Payment Cryptography beinhaltet eine Serviceverpflichtung von 99,99% für alle Payment Cryptography APIs. Um dieser Verpflichtung nachzukommen, stellt AWS Payment Cryptography sicher, dass alle Daten und Autorisierungsinformationen, die zur Ausführung einer API-Anfrage erforderlich sind, auf allen regionalen Hosts verfügbar sind, die die Anfrage erhalten.

Die Infrastruktur von AWS Payment Cryptography wird in mindestens drei Availability Zones (AZs) in jeder Region repliziert. Um sicherzustellen, dass mehrere Hostausfälle die Leistung von AWS Payment Cryptography nicht beeinträchtigen, ist AWS Payment Cryptography so konzipiert, dass Kundenverkehr von allen AZs in einer Region bedient wird.

Änderungen, die Sie an den Eigenschaften oder Berechtigungen eines Zahlungsschlüssels vornehmen, werden auf alle Hosts in der Region repliziert, um sicherzustellen, dass nachfolgende Anfragen von jedem Host in der Region korrekt verarbeitet werden können. Anfragen für kryptografische Operationen unter Verwendung Ihres Zahlungsschlüssels werden an eine Flotte von Hardware-Sicherheitsmodulen (HSMs) von AWS Payment Cryptography weitergeleitet, von denen jedes den Vorgang mit dem Zahlungsschlüssel ausführen kann.

Design mit mehreren Mandanten

Das mandantenfähige Design von AWS Payment Cryptography ermöglicht es, die Verfügbarkeits-SLA zu erfüllen und hohe Anforderungsraten aufrechtzuerhalten und gleichzeitig die Vertraulichkeit Ihrer Schlüssel und Daten zu schützen.

Es werden mehrere Mechanismen zur Durchsetzung der Integrität eingesetzt, um sicherzustellen, dass der Zahlungsschlüssel, den Sie für den kryptografischen Vorgang angegeben haben, immer der verwendete ist.

Das Klartext-Schlüsselmaterial für Ihre kryptographischen Zahlungsschlüssel ist umfassend geschützt. Das Schlüsselmaterial wird im HSM verschlüsselt, sobald es erstellt wurde, und das verschlüsselte Schlüsselmaterial wird sofort in einen sicheren Speicher verschoben. Der verschlüsselte Schlüssel wird im HSM abgerufen und entschlüsselt, sobald er benutzt wird. Der Klartextschlüssel verbleibt nur so lange im HSM-Speicher, wie er für den Abschluss der kryptografischen Operation benötigt wird. Das Klartext-Schlüsselmaterial verlässt die HSMs nie; es wird nie in den permanenten Speicher geschrieben.

Weitere Informationen zu den Mechanismen, mit denen AWS Payment Cryptography Ihre Schlüssel schützt, finden Sie unter AWS Payment Cryptography Cryptography Cryptographic Details.

Sicherheit der Infrastruktur in AWS Payment Cryptography

Als verwalteter Service AWS Payment Cryptography ist er durch die AWS globalen Netzwerksicherheitsverfahren geschützt, die im Whitepaper Amazon Web Services: Sicherheitsprozesse im Überblick beschrieben sind.

Sie verwenden AWS veröffentlichte API-Aufrufe für den Zugriff AWS Payment Cryptography über das Netzwerk. Clients müssen Transport Layer Security (TLS) 1.2 oder höher unterstützen. Clients müssen außerdem Verschlüsselungssammlungen mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Isolierung von physischen Hosts

Die Sicherheit der physischen Infrastruktur, die AWS Payment Cryptography verwendet, unterliegt den Kontrollen, die im Abschnitt Physische Sicherheit und Umweltsicherheit von Amazon Web Services: Überblick über Sicherheitsprozesse beschrieben sind. Weitere Details finden Sie in Compliance-Berichten und Prüfungsergebnissen von Drittanbietern, die im vorherigen Abschnitt aufgeführt sind.

AWS Payment Cryptography wird von speziellen commercial-off-the-shelf PCI PTS HSM-gelisteten Hardware-Sicherheitsmodulen (HSMs) unterstützt. Das Schlüsselmaterial für AWS Payment Cryptography-Schlüssel wird nur im flüchtigen Speicher der HSMs gespeichert und nur solange der Payment Cryptography-Schlüssel verwendet wird. HSMs befinden sich in zugriffskontrollierten Racks in Amazon-Rechenzentren, die eine doppelte Kontrolle für jeden physischen Zugriff erzwingen. Detaillierte Informationen zum Betrieb von AWS Payment Cryptography HSMs finden Sie unter AWS Payment Cryptography Cryptography Cryptographic Details.

Über einen VPC-Endpunkt eine Verbindung zur AWS Zahlungskryptografie herstellen

Sie können über einen privaten Schnittstellenendpunkt in Ihrer Virtual Private Cloud (VPC) eine direkte Verbindung zu AWS Payment Cryptography herstellen. Wenn Sie einen VPC-Schnittstellen-Endpunkt verwenden, erfolgt die Kommunikation zwischen Ihrer VPC und AWS Payment Cryptography ausschließlich innerhalb des Netzwerks. AWS

AWS Payment Cryptography unterstützt Amazon Virtual Private Cloud (Amazon VPC) -Endpunkte, die von bereitgestellt werden. AWS PrivateLink Jeder VPC-Endpunkt wird durch eine oder mehrere Elastic Network-Schnittstellen (ENIs) mit privaten IP-Adressen in Ihren VPC-Subnetzen repräsentiert.

Der VPC-Endpunkt der Schnittstelle verbindet Ihre VPC direkt mit AWS Payment Cryptography, ohne dass ein Internet-Gateway, ein NAT-Gerät, eine VPN-Verbindung oder eine Verbindung erforderlich ist. AWS Direct Connect Die Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit AWS Payment Cryptography zu kommunizieren.

Regionen

AWS Payment Cryptography unterstützt VPC-Endpunkte und VPC-Endpunktrichtlinien in allen Bereichen, AWS-Regionen in denen AWS Zahlungskryptografie unterstützt wird.

Themen

Überlegungen zu AWS VPC-Endpunkten für Zahlungskryptografie

Bevor Sie einen VPC-Schnittstellen-Endpunkt für AWS Payment Cryptography einrichten, lesen Sie das Thema Eigenschaften und Einschränkungen von Schnittstellenendpunkten im AWS PrivateLink Handbuch.

AWS Die Unterstützung der Zahlungskryptografie für einen VPC-Endpunkt umfasst Folgendes.

- Sie können Ihren VPC-Endpunkt verwenden, um alle AWS Payment Cryptography Controlplane-Operationen und AWS Payment Cryptography Dataplane-Operationen von einer VPC aus aufzurufen.
- Sie können einen VPC-Schnittstellen-Endpunkt erstellen, der eine Verbindung zu einem Endpunkt der AWS Payment Cryptography Region herstellt.
- AWS Die Zahlungskryptografie besteht aus einer Steuerungsebene und einer Datenebene. Sie können wählen, ob Sie einen oder beide Unterdienste einrichten möchten, aber jeder wird separat konfiguriert.
- Sie können AWS CloudTrail Protokolle verwenden, um Ihre Verwendung von AWS Payment Cryptography Keys über den VPC-Endpunkt zu überprüfen. Details hierzu finden Sie unter Protokollieren des VPC-Endpunkts.

Einen VPC-Endpunkt für AWS Zahlungskryptografie erstellen

Sie können mithilfe der Amazon VPC-Konsole oder der Amazon VPC-API einen VPC-Endpunkt für AWS Payment Cryptography erstellen. Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts im AWS PrivateLink -Leitfaden.

- Verwenden Sie die folgenden Dienstnamen, um einen VPC-Endpunkt für AWS Payment Cryptography zu erstellen:

```
com.amazonaws.region.payment-cryptography.controlplane
```

```
com.amazonaws.region.payment-cryptography.dataplane
```

  In der Region USA West (Oregon) (`us-west-2`) wären die Dienstnamen beispielsweise:

```
com.amazonaws.us-west-2.payment-cryptography.controlplane
```

```
com.amazonaws.us-west-2.payment-cryptography.dataplane
```

Um die Verwendung des VPC-Endpunkts zu vereinfachen, können Sie einen privaten DNS-Namen für Ihren VPC-Endpunkt aktivieren. Wenn Sie die Option „DNS-Namen aktivieren“ auswählen, wird der standardmäßige DNS-Hostname für AWS Payment Cryptography zu Ihrem VPC-Endpunkt aufgelöst. `https://controlplane.payment-cryptography.us-`

`west-2.amazonaws.com` würde beispielsweise in einen VPC-Endpunkt aufgelöst, der mit dem Servicenamen `com.amazonaws.us-west-2.payment-cryptography.controlplane` verbunden ist.

Diese Option vereinfacht die Verwendung des VPC-Endpunkts. Die AWS SDKs AWS CLI verwenden standardmäßig den AWS Standard-Payment Cryptography DNS-Hostnamen, sodass Sie die VPC-Endpunkt-URL in Anwendungen und Befehlen nicht angeben müssen.

Weitere Informationen finden Sie unter Zugriff auf einen Service über einen Schnittstellenendpunkt im AWS PrivateLink -Leitfaden.

Verbindung zu einem VPC-Endpunkt für AWS Payment Cryptography herstellen

Sie können über den VPC-Endpunkt eine Verbindung zu AWS Payment Cryptography herstellen, indem Sie ein AWS SDK, das AWS CLI oder, verwenden. AWS Tools for PowerShell Um den VPC-Endpunkt anzugeben, verwenden Sie seinen DNS-Namen.

Dieser list-keys-Befehl verwendet zur Angabe des VPC-Endpunkts z. B. den Parameter `endpoint-url`. Wenn Sie einen solchen Befehl verwenden möchten, ersetzen Sie die Beispiels-ID des VPC-Endpunkts durch eine ID in Ihrem Konto.

```
$ aws payment-cryptography list-keys --endpoint-url
```

Wenn Sie beim Erstellen Ihres VPC-Endpunkts private Hostnamen aktiviert waren, müssen Sie die URL des Endpunkts in Ihren CLI-Befehlen oder in Ihrer Anwendungskonfiguration angeben. Der standardmäßige DNS-Hostname für AWS Payment Cryptography wird zu Ihrem VPC-Endpunkt aufgelöst. Die SDKs AWS CLI und verwenden standardmäßig diesen Hostnamen, sodass Sie den VPC-Endpunkt verwenden können, um eine Verbindung zu einem regionalen AWS Payment Cryptography-Endpunkt herzustellen, ohne etwas an Ihren Skripten und Anwendungen zu ändern.

Zur Verwendung privater Hostnamen müssen die Attribute `enableDnsHostnames` und `enableDnsSupport` Ihrer VPC auf `true` eingestellt sein. Verwenden Sie den Attributvorgang, um diese Attribute festzulegen. ModifyVpc Details dazu finden Sie unter Anzeigen und Aktualisieren von DNS-Attributen für Ihre VPC im Amazon-VPC-Benutzerhandbuch.

Steuern des Zugriffs auf einen VPC-Endpunkt

Um den Zugriff auf Ihren VPC-Endpunkt für AWS Payment Cryptography zu kontrollieren, fügen Sie VPC VPC-Endpunkt eine VPC-Endpunktrichtlinie hinzu. Die Endpunktrichtlinie legt fest, ob Principals den VPC-Endpunkt verwenden können, um AWS Zahlungskryptografieoperationen mit bestimmten AWS Zahlungskryptografie-Ressourcen aufzurufen.

Sie können beim Erstellen des Endpunkts eine VPC-Endpunktrichtlinie erstellen und die VPC-Endpunktrichtlinie jederzeit ändern. Verwenden Sie die VPC-Managementkonsole oder die CreateVpcEndpoint- oder ModifyVpcEndpoint-Operationen. Sie können eine VPC-Endpunktrichtlinie auch mithilfe einer AWS CloudFormation Vorlage erstellen und ändern. Hilfe zur Verwendung der VPC-Managementkonsole finden Sie unter Erstellen eines Schnittstellenendpunkts und Ändern eines Schnittstellenendpunkts im AWS PrivateLink -Leitfaden.

Themen

Weitere Informationen über VPC-Endpunktrichtlinien

Damit eine AWS Zahlungskryptografieanfrage, die einen VPC-Endpunkt verwendet, erfolgreich ist, benötigt der Principal Berechtigungen aus zwei Quellen:

- Eine identitätsbasierte Richtlinie muss dem Prinzipal die Erlaubnis erteilen, den Vorgang für die Ressource (AWS Payment Cryptography Keys oder Alias) aufzurufen.
- Eine VPC-Endpunktrichtlinie muss dem Prinzipal die Berechtigung erteilen, den Endpunkt für die Anforderung zu verwenden.

Beispielsweise kann eine Schlüsselrichtlinie einem Prinzipal die Erlaubnis erteilen, Decrypt für einen bestimmten AWS Schlüssel zur Zahlungskryptografie aufzurufen. Die VPC-Endpunktrichtlinie erlaubt es diesem Principal jedoch möglicherweise nicht, diese AWS Payment Cryptography Keys mithilfe des Endpunkts aufzurufen`Decrypt`.

Oder eine VPC-Endpunktrichtlinie könnte es einem Principal ermöglichen, den Endpunkt zu verwenden, um StopKeyUsage für bestimmte AWS Payment Cryptography Keys aufzurufen. Wenn der Principal jedoch nicht über die in einer IAM-Richtlinie festgelegten Berechtigungen verfügt, schlägt die Anfrage fehl.

Standard-VPC-Endpunktrichtlinie

Jeder VPC-Endpunkt verfügt über eine VPC-Endpunktrichtlinie. Sie müssen die Richtlinie jedoch nicht angeben. Wenn Sie keine Richtlinie angeben, erlaubt die standardmäßige Endpunktrichtlinie alle Operationen aller Prinzipale auf allen Ressourcen über den Endpunkt.

Für AWS Zahlungskryptografie-Ressourcen muss der Prinzipal jedoch auch die Erlaubnis haben, den Vorgang über eine IAM-Richtlinie aufzurufen. Daher besagt die Standardrichtlinie in der Praxis, dass, wenn ein Prinzipal über die Berechtigung zum Aufrufen einer Operation für eine Ressource verfügt, diese auch mithilfe des Endpunkts aufrufen kann.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

Damit Prinzipale den VPC-Endpunkt nur für eine Teilmenge ihrer zulässigen Operationen verwenden können, erstellen oder aktualisieren Sie die VPC-Endpunktrichtlinie.

Erstellen einer VPC-Endpunktrichtlinie

Eine VPC-Endpunktrichtlinie bestimmt, ob ein Prinzipal die Berechtigung hat, den VPC-Endpunkt zum Ausführen von Operationen auf einer Ressource zu verwenden. Für AWS Zahlungskryptografie-Ressourcen muss der Principal außerdem über die Erlaubnis verfügen, die Operationen anhand einer IAM-Richtlinie auszuführen.

Für jede VPC-Endpunktrichtlinie sind die folgenden Elemente erforderlich:

- Der Prinzipal, der die Aktionen ausführen kann
- Aktionen, die ausgeführt werden können

- Ressourcen, für die Aktionen ausgeführt werden können

Die Richtlinienanweisung gibt den VPC-Endpunkt nicht an. Stattdessen gilt sie für jeden VPC-Endpunkt, dem die Richtlinie angefügt ist. Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für eine VPC-Endpunktrichtlinie für AWS Payment Cryptography. Wenn diese Richtlinie an einen VPC-Endpunkt angehängt ist, ermöglicht `ExampleUser` sie die Verwendung des VPC-Endpunkts, um die angegebenen Operationen mit den angegebenen AWS Zahlungskryptografie-Schlüsseln aufzurufen. Bevor Sie eine Richtlinie wie diese verwenden, ersetzen Sie den Beispielprinzipal und die Schlüssel-ID durch gültige Werte aus Ihrem Konto.

```
{
   "Statement":[
      {
         "Sid": "AllowDecryptAndView",
         "Principal": {"AWS": "arn:aws:iam::111122223333:user/ExampleUser"},
         "Effect":"Allow",
         "Action": [
             "payment-cryptography:Decrypt",
             "payment-cryptography:GetKey",
             "payment-cryptography:ListAliases",
             "payment-cryptography:ListKeys",
             "payment-cryptography:GetAlias"
          ],
         "Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
      }
   ]
}
```

AWS CloudTrail protokolliert alle Operationen, die den VPC-Endpunkt verwenden. Ihre CloudTrail Protokolle enthalten jedoch keine Vorgänge, die von Prinzipalen in anderen Konten angefordert wurden, oder Vorgänge für AWS Zahlungskryptografie-Schlüssel in anderen Konten.

Daher möchten Sie möglicherweise eine VPC-Endpunktrichtlinie erstellen, die verhindert, dass Prinzipale in externen Konten den VPC-Endpunkt verwenden, um AWS Zahlungskryptografievorgänge für Schlüssel im lokalen Konto aufzurufen.

Im folgenden Beispiel wird der PrincipalAccount globale Bedingungsschlüssel aws: verwendet, um allen Prinzipalen den Zugriff auf alle Vorgänge mit allen AWS Payment Cryptography Keys zu verweigern, sofern sich der Principal nicht im lokalen Konto befindet. Bevor Sie eine Richtlinie wie diese verwenden, ersetzen Sie die Beispiel-Konto-ID durch eine gültige.

```
{
  "Statement": [
    {
      "Sid": "AccessForASpecificAccount",
      "Principal": {"AWS": "*"},
      "Action": "payment-cryptography:*",
      "Effect": "Deny",
      "Resource": "arn:aws:payment-cryptography:*:111122223333:key/*",
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": "111122223333"
        }
      }
    }
  ]
}
```

Anzeigen einer VPC-Endpunktrichtlinie

Um die VPC-Endpunktrichtlinie für einen Endpunkt anzuzeigen, verwenden Sie die VPC-Managementkonsole oder den DescribeVpcEndpoint-Vorgang.

Mit dem folgenden AWS CLI Befehl wird die Richtlinie für den Endpunkt mit der angegebenen VPC-Endpunkt-ID abgerufen.

Bevor Sie diesen Befehl ausführen, ersetzen Sie die Beispiel-Endpunkt-ID durch eine gültige aus Ihrem Konto.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==``].[PolicyDocument]'
--output text
```

Verwenden eines VPC-Endpunkts in einer Richtlinienanweisung

Sie können den Zugriff auf Ressourcen und Vorgänge im Bereich AWS Payment Cryptography kontrollieren, wenn die Anfrage von einer VPC kommt oder einen VPC-Endpunkt verwendet. Verwenden Sie dazu eine IAM-Richtlinie

- Verwenden Sie den `aws:sourceVpce`-Bedingungsschlüssel zum Erteilen oder Beschränken des Zugriffs anhand des VPC-Endpunkts.
- Verwenden Sie den `aws:sourceVpc`-Bedingungsschlüssel zum Erteilen oder Beschränken des Zugriffs anhand des VPC, auf der der private Endpunkt gehostet wird.

> **Note**
>
> Der `aws:sourceIP` Bedingungsschlüssel ist nicht wirksam, wenn die Anfrage von einem Amazon VPC-Endpunkt kommt. Um die Anforderungen an einen VPC-Endpunkt zu beschränken, verwenden Sie die Bedingungsschlüssel `aws:sourceVpce` oder `aws:sourceVpc`. Weitere Informationen finden Sie unter Identity and Access Management für VPC-Endpunkte und VPC-Endpunkt-Services im AWS PrivateLink -Leitfaden.

Sie können diese globalen Bedingungsschlüssel verwenden, um den Zugriff auf AWS Payment Cryptography Keys und Aliase zu kontrollieren. Solche CreateKeyOperationen hängen nicht von einer bestimmten Ressource ab.

Die folgende Beispielschlüsselrichtlinie ermöglicht es einem Benutzer beispielsweise, bestimmte kryptografische Operationen mit AWS Zahlungskryptografie-Schlüsseln nur dann durchzuführen, wenn die Anfrage den angegebenen VPC-Endpunkt verwendet, wodurch der Zugriff sowohl über das Internet als auch über Verbindungen (falls eingerichtet) blockiert wird. Wenn ein Benutzer eine Anfrage an AWS Payment Cryptography stellt, wird die VPC-Endpunkt-ID in der Anfrage mit dem `aws:sourceVpce` Bedingungsschlüsselwert in der Richtlinie verglichen. Wenn sie nicht übereinstimmen, wird die Anforderung abgelehnt.

Um eine Richtlinie wie diese zu verwenden, ersetzen Sie die AWS-Konto Platzhalter-ID und die VPC-Endpunkt-IDs durch gültige Werte für Ihr Konto.

```
{
    "Id": "example-key-1",
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "Enable IAM policies",
            "Effect": "Allow",
            "Principal": {"AWS":["111122223333"]},
            "Action": ["payment-cryptography:*"],
            "Resource": "*"
        },
        {
            "Sid": "Restrict usage to my VPC endpoint",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "payment-cryptography:Encrypt",
                "payment-cryptography:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:sourceVpce": ""
                }
            }
        }

    ]
}
```

Sie können den `aws:sourceVpc` Bedingungsschlüssel auch verwenden, um den Zugriff auf Ihre AWS Payment Cryptography Keys basierend auf der VPC, in der sich der VPC-Endpunkt befindet, einzuschränken.

Das folgende Beispiel für eine Schlüsselrichtlinie erlaubt Befehle, die die AWS Payment Cryptography Keys verwalten, nur dann, wenn sie stammen. `vpc-12345678` Darüber hinaus sind Befehle, die AWS Payment Cryptography Keys für kryptografische Operationen verwenden, nur zulässig, wenn sie von `vpc-2b2b2b2b` Sie verwenden eine solche Richtlinie wie diese möglicherweise, wenn eine Anwendung in einer VPC ausgeführt wird, aber eine zweite, isolierte VPC für die Verwaltungsfunktionen genutzt wird.

Um eine Richtlinie wie diese zu verwenden, ersetzen Sie die AWS-Konto Platzhalter-ID und die VPC-Endpunkt-IDs durch gültige Werte für Ihr Konto.

```
{
```

```
    "Id": "example-key-2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow administrative actions from vpc-12345678",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "payment-cryptography:Create*","payment-
cryptography:Encrypt*","payment-cryptography:ImportKey*","payment-
cryptography:GetParametersForImport*",
                "payment-cryptography:TagResource", "payment-
cryptography:UntagResource"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-12345678"
                }
            }
        },
        {
            "Sid": "Allow key usage from vpc-2b2b2b2b",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "payment-cryptography:Encrypt","payment-cryptography:Decrypt"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-2b2b2b2b"
                }
            }
        },
        {
            "Sid": "Allow list/read actions from everywhere",
            "Effect": "Allow",
            "Principal": {"AWS": "111122223333"},
            "Action": [
                "payment-cryptography:List*","payment-cryptography:Get*"
            ],
            "Resource": "*",
        }
```

```
    ]
}
```

Protokollieren des VPC-Endpunkts

AWS CloudTrail protokolliert alle Operationen, die den VPC-Endpunkt verwenden. Wenn eine Anfrage an AWS Payment Cryptography einen VPC-Endpunkt verwendet, erscheint die VPC-Endpunkt-ID im AWS CloudTrail Protokolleintrag, der die Anfrage aufzeichnet. Sie können die Endpunkt-ID verwenden, um die Nutzung Ihres AWS Payment Cryptography VPC-Endpunkts zu überprüfen.

Um Ihre VPC zu schützen, werden Anfragen, die durch eine VPC-Endpunktrichtlinie abgelehnt wurden, aber andernfalls zugelassen worden wären, nicht aufgezeichnet. AWS CloudTrail

In diesem Beispielprotokolleintrag wird beispielsweise eine GenerateMacAnfrage aufgezeichnet, die den VPC-Endpunkt verwendet hat. Das Feld `vpcEndpointId` erscheint am Ende des Protokolleintrags.

```
{
      "eventVersion": "1.08",
      "userIdentity": {
          "principalId": "TESTXECZ5U9M4LGF2N6Y5:",
          "arn": "arn:aws:sts::111122223333:assumed-role//",
          "accountId": "111122223333",
          "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
          "sessionContext": {
              "sessionIssuer": {
                  "type": "Role",
                  "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                  "arn": "arn:aws:iam::111122223333:role/",
                  "accountId": "111122223333",
                  "userName": ""
              },
              "webIdFederationData": {},
              "attributes": {
                  "creationDate": "2024-05-27T19:34:10Z",
                  "mfaAuthenticated": "false"
              },
              "ec2RoleDelivery": "2.0"
          }
      },
      "eventTime": "2024-05-27T19:49:54Z",
```

```
      "eventSource": "payment-cryptography.amazonaws.com",
      "eventName": "CreateKey",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "172.31.85.253",
      "userAgent": "aws-cli/2.14.5 Python/3.9.16 Linux/6.1.79-99.167.amzn2023.x86_64
 source/x86_64.amzn.2023 prompt/off command/payment-cryptography.create-key",
      "requestParameters": {
          "keyAttributes": {
              "keyUsage": "TR31_M1_ISO_9797_1_MAC_KEY",
              "keyClass": "SYMMETRIC_KEY",
              "keyAlgorithm": "TDES_2KEY",
              "keyModesOfUse": {
                  "encrypt": false,
                  "decrypt": false,
                  "wrap": false,
                  "unwrap": false,
                  "generate": true,
                  "sign": false,
                  "verify": true,
                  "deriveKey": false,
                  "noRestrictions": false
              }
          },
          "exportable": true
      },
      "responseElements": {
          "key": {
              "keyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
              "keyAttributes": {
                  "keyUsage": "TR31_M1_ISO_9797_1_MAC_KEY",
                  "keyClass": "SYMMETRIC_KEY",
                  "keyAlgorithm": "TDES_2KEY",
                  "keyModesOfUse": {
                      "encrypt": false,
                      "decrypt": false,
                      "wrap": false,
                      "unwrap": false,
                      "generate": true,
                      "sign": false,
                      "verify": true,
                      "deriveKey": false,
                      "noRestrictions": false
                  }
```

```
            },
            "keyCheckValue": "A486ED",
            "keyCheckValueAlgorithm": "ANSI_X9_24",
            "enabled": true,
            "exportable": true,
            "keyState": "CREATE_COMPLETE",
            "keyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "createTimestamp": "May 27, 2024, 7:49:54 PM",
            "usageStartTimestamp": "May 27, 2024, 7:49:54 PM"
        }
    },
    "requestID": "f3020b3c-4e86-47f5-808f-14c7a4a99161",
    "eventID": "b87c3d30-f3ab-4131-87e8-bc54cfef9d29",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "vpcEndpointId": "",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "-oo28vrvr.controlplane.payment-cryptography.us-
east-1.vpce.amazonaws.com"
    }
}
```


Bewährte Sicherheitsmethoden für AWS Zahlungskryptografie

AWS Die Zahlungskryptografie unterstützt viele Sicherheitsfunktionen, die entweder integriert sind oder die Sie optional implementieren können, um den Schutz Ihrer Verschlüsselungsschlüssel zu verbessern und sicherzustellen, dass sie für den vorgesehenen Zweck verwendet werden, darunter IAM-Richtlinien, eine umfangreiche Reihe von Richtlinienbedingungsschlüsseln zur Verfeinerung Ihrer Schlüsselrichtlinien und IAM-Richtlinien sowie die integrierte Durchsetzung von PCI-PIN-Regeln in Bezug auf Schlüsselblöcke.

> **Important**
>
> Die bereitgestellten allgemeinen Richtlinien stellen keine vollständige Sicherheitslösung dar. Da nicht alle bewährten Methoden für alle Situationen geeignet sind, sollten diese nicht vorschriftig sein.

- Verwendung und Verwendungsarten von Schlüsseln: Bei der AWS Zahlungskryptografie werden die in der ANSI X9 TR 31-2018 Interoperable Secure Key Exchange Key Block Specification beschriebenen Beschränkungen für die Verwendung und Nutzung von Schlüsseln sowie die Einhaltung der PCI-PIN-Sicherheitsanforderung 18-3 befolgt und durchgesetzt. Dadurch wird die Möglichkeit eingeschränkt, einen einzelnen Schlüssel für mehrere Zwecke zu verwenden, und die Schlüsselmetadaten (z. B. zulässige Operationen) werden kryptografisch an das Schlüsselmaterial selbst gebunden. AWS Die Zahlungskryptografie erzwingt diese Einschränkungen automatisch, z. B. dass ein Schlüsselverschlüsselungsschlüssel (TR31_K0_KEY_ENCRYPTION_KEY) nicht auch für die Datenentschlüsselung verwendet werden kann. Weitere Details finden Sie unter Grundlegendes zu den Schlüsselattributen des Payment Cryptography AWS Keys.
- Beschränken Sie die gemeinsame Nutzung von symmetrischem Schlüsselmaterial: Teilen Sie symmetrisches Schlüsselmaterial (wie PIN-Verschlüsselungsschlüssel oder Schlüsselverschlüsselungsschlüssel) nur mit höchstens einer anderen Entität. Wenn vertrauliches Material an mehrere Entitäten oder Partner übertragen werden muss, erstellen Sie zusätzliche Schlüssel. AWS Bei der Zahlungskryptografie werden symmetrisches Schlüsselmaterial oder asymmetrisches privates Schlüsselmaterial niemals unverschlüsselt offengelegt.
- Verwenden Sie Aliase oder Tags, um Schlüssel bestimmten Anwendungsfällen oder Partnern zuzuordnen: Aliase können verwendet werden, um auf einfache Weise den Anwendungsfall zu kennzeichnen, der mit einem Schlüssel verknüpft ist, z. B. Alias/BIN_12345_CVK, um einen Kartenverifizierungsschlüssel zu bezeichnen, der mit BIN 12345 verknüpft ist. Um mehr Flexibilität zu bieten, sollten Sie in Erwägung ziehen, Tags wie bin=12345, use_case=acquire, country=us, partner=foo zu erstellen. Aliase und Tags können auch verwendet werden, um den Zugriff einzuschränken, z. B. um Zugriffskontrollen zwischen ausstellenden und akquirierenden Anwendungsfällen durchzusetzen.
- Praktischer Zugriff mit den geringsten Rechten: IAM kann verwendet werden, um den Produktionszugriff auf Systeme und nicht auf einzelne Personen zu beschränken. So kann beispielsweise verhindert werden, dass einzelne Benutzer Schlüssel erstellen oder kryptografische Operationen ausführen. IAM kann auch verwendet werden, um den Zugriff auf Befehle und Schlüssel einzuschränken, was für Ihren Anwendungsfall möglicherweise nicht zutrifft, z. B. die

Möglichkeit, Pins für einen Acquirer zu generieren oder zu validieren. Eine weitere Möglichkeit, den Zugriff mit den geringsten Rechten zu nutzen, besteht darin, sensible Vorgänge (wie den Schlüsselimport) auf bestimmte Dienstkonten zu beschränken. Beispiele finden Sie unter AWS Beispiele für identitätsbasierte Richtlinien zur Zahlungskryptografie.

Informationen finden Sie auch unter:

- Identitäts- und Zugriffsmanagement für AWS Zahlungskryptografie
- Bewährte Methoden für die Sicherheit im IAM-Benutzerhandbuch.

Konformitätsvalidierung fürAWSZahlungskryptografie

Externe Auditoren bewerten die Sicherheit und Einhaltung vonAWSZahlungskryptographie als Teil mehrererAWSCompliance-Programme. Dazu gehören SOC, PCI und andere.

AWSDie Zahlungskryptographie wurde zusätzlich zu PCI DSS für mehrere PCI-Standards bewertet. Dazu gehören PCI-PIN-Sicherheit (PCI-PIN) und PCI-Punkt-zu-Punkt-Verschlüsselung (P2PE). Bitte sehen SieAWS Artifactfür verfügbare Bescheinigungen und Compliance-Leitfäden.

Eine Liste der AWS-Services im Bereich bestimmter Compliance-Programme finden Sie unter AWS-Services im Bereich 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 Berichte herunterladen in AWS Artifact.

Ihre Verantwortung für die Einhaltung gesetzlicher Vorschriften bei der VerwendungAWSDie Zahlungskryptografie hängt von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab.AWSstellt die folgenden Ressourcen bereit, um Sie bei der Einhaltung von Vorschriften zu unterstützen:

- Schnellstartanleitungen zu Sicherheit und Compliance—In diesen Bereitstellungshandbüchern werden Überlegungen zur Architektur erörtert und Schritte für die Bereitstellung von sicherheits- und Compliance-orientierten Baseline-Umgebungen beschriebenAWS.
- AWSRessourcen zur Einhaltung von Vorschriften—Diese Sammlung von Arbeitsmappen und Leitfäden könnte für Ihre Branche und Ihren Standort gelten.
- Ressourcen anhand von Regeln bewertenin derAWS ConfigLeitfaden für Entwickler—AWS Config; bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub—DasAWSDer Service bietet einen umfassenden Überblick über Ihren SicherheitsstatusAWSdamit Sie überprüfen können, ob Sie die Standards und Best Practices der Sicherheitsbranche einhalten.

Identitäts- und Zugriffsmanagement für AWS Zahlungskryptografie

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um AWS Zahlungskryptografie-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

Zielgruppe

Die Art und Weise, wie Sie AWS Identity and Access Management (IAM) verwenden, hängt davon ab, welche Arbeit Sie im Bereich AWS Zahlungskryptografie ausführen.

Dienstbenutzer — Wenn Sie den AWS Payment Cryptography Service für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr Funktionen der AWS Zahlungskryptografie verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Fuktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Wenn Sie auf eine Funktion in AWS Payment Cryptography nicht zugreifen können, finden Sie weitere Informationen unter. Fehlerbehebung bei AWS Zahlungen: Kryptografie, Identität und Zugriff

Dienstadministrator — Wenn Sie in Ihrem Unternehmen für die Ressourcen zur AWS Zahlungskryptografie verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf AWS Zahlungskryptografie. Es ist Ihre Aufgabe, zu bestimmen, auf welche Funktionen und Ressourcen für die AWS Zahlungskryptografie Ihre Servicebenutzer 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 darüber, wie Ihr Unternehmen IAM mit AWS Zahlungskryptografie verwenden kann, finden Sie unter. So funktioniert AWS Zahlungskryptografie mit IAM

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien zur Verwaltung des Zugriffs auf Payment Cryptography verfassen können. AWS Beispiele für identitätsbasierte AWS Zahlungskryptografie-Richtlinien, die Sie in IAM verwenden können, finden Sie unter. AWS Beispiele für identitätsbasierte Richtlinien zur Zahlungskryptografie

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS , übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter So melden Sie sich bei Ihrem an AWS-Konto im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, mit denen Sie Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch signieren können. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS Empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere

Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center - Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität innerhalb von Ihnen AWS-Konto , die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen wechseln. Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center -Benutzerhandbuch.
- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff — Einige AWS-Services verwenden Funktionen in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.

- Forward Access Sessions (FAS) — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
- Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, 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.
- Dienstbezogene Rolle — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

- Anwendungen, die auf Amazon EC2 ausgeführt werden — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder

Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

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.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

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. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

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.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.

- Service Control Policies (SCPs) — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter Bewertungslogik für Richtlinien.

So funktioniert AWS Zahlungskryptografie mit IAM

Bevor Sie IAM zur Verwaltung des Zugriffs auf AWS Zahlungskryptografie verwenden, sollten Sie wissen, welche IAM-Funktionen für die Verwendung mit Zahlungskryptografie verfügbar sind. AWS Einen allgemeinen Überblick darüber, wie AWS Zahlungskryptografie und andere AWS Dienste mit IAM funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren.

Themen

- AWS Zahlungskryptografie Identitätsbasierte Richtlinien
- Autorisierung auf der Grundlage von Payment Cryptography Tags AWS

AWS Zahlungskryptografie Identitätsbasierte 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. AWS Die Zahlungskryptografie unterstützt bestimmte Aktionen, Ressourcen und Zustandsschlüssel. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Aktionen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. 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 der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang 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.

Richtlinienaktionen in der AWS Zahlungskryptografie verwenden vor der Aktion das folgende Präfix:`payment-cryptography:`. Um beispielsweise jemandem die Erlaubnis zu erteilen, einen AWS Payment Cryptography `VerifyCardData` API-Vorgang auszuführen, nehmen Sie die `payment-cryptography:VerifyCardData` Aktion in seine Richtlinie auf. Richtlinienanweisungen müssen entweder ein – `Action`oder ein `NotAction`-Element enthalten. AWS Payment Cryptography definiert eigene Aktionen, die Aufgaben beschreiben, die Sie mit diesem Dienst ausführen können.

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie wie folgt durch Kommata:

```
"Action": [
      "payment-cryptography:action1",
      "payment-cryptography:action2"
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Um beispielsweise alle Aktionen anzugeben, die mit dem Wort beginnen `List` (wie `ListKeys` und`ListAliases`), schließen Sie die folgende Aktion ein:

```
"Action": "payment-cryptography:List*"
```

Eine Liste der AWS Zahlungskryptografie-Aktionen finden Sie im IAM-Benutzerhandbuch unter Durch AWS Zahlungskryptografie definierte Aktionen.

Ressourcen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. 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 Schlüsselressource für Zahlungskryptografie hat den folgenden ARN:

```
arn:${Partition}:payment-cryptography:${Region}:${Account}:key/${keyARN}
```

Weitere Informationen zum Format von ARNs finden Sie unter Amazon Resource Names (ARNs) und AWS Service Namespaces.

Wenn Sie beispielsweise die `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`-Instance in Ihrer Anweisung angeben möchten, verwenden Sie den folgenden ARN:

```
"Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h"
```

Um alle Schlüssel anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie den Platzhalter (*):

```
"Resource": "arn:aws:payment-cryptography:us-east-2:111122223333:key/*"
```

Einige Kryptografie-Aktionen für AWS Zahlungen, z. B. zum Erstellen von Schlüsseln, können nicht für eine bestimmte Ressource ausgeführt werden. In diesen Fällen müssen Sie den Platzhalter (*) verwenden.

```
"Resource": "*"
```

Um mehrere Ressourcen in einer einzigen Anweisung anzugeben, verwenden Sie ein Komma wie unten dargestellt:

```
"Resource": [
      "resource1",
      "resource2"
```

Beispiele

Beispiele für identitätsbasierte AWS Zahlungskryptografie-Richtlinien finden Sie unter. AWS Beispiele für identitätsbasierte Richtlinien zur Zahlungskryptografie

Autorisierung auf der Grundlage von Payment Cryptography Tags AWS

AWS Beispiele für identitätsbasierte Richtlinien zur Zahlungskryptografie

Standardmäßig sind IAM-Benutzer und -Rollen nicht berechtigt, Payment Cryptography-Ressourcen zu erstellen oder zu ändern AWS . Sie können auch keine Aufgaben mit der AWS Management Console AWS CLI, oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Themen

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand AWS Zahlungskryptografie-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:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS -verwaltete Richtlinien oder AWS -verwaltete Richtlinien für Auftrags-Funktionen 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 Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. 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.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, 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 Sie die Payment Cryptography Console AWS

Um auf die AWS Payment Cryptography Console zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, die AWS Zahlungskryptografie-Ressourcen in Ihrem AWS Konto aufzulisten und einzusehen. 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 diese Entitäten weiterhin die AWS Payment Cryptography Console verwenden können, fügen Sie den Entitäten außerdem die folgende AWS verwaltete Richtlinie bei. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die den API-Operation entsprechen, die Sie ausführen möchten.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Möglichkeit, auf alle Aspekte der AWS Zahlungskryptografie zuzugreifen

> **Warning**
>
> Dieses Beispiel bietet umfassende Berechtigungen und wird nicht empfohlen. Ziehen Sie stattdessen Modelle mit den geringsten Zugriffsrechten in Betracht.

In diesem Beispiel möchten Sie einem IAM-Benutzer in Ihrem AWS Konto Zugriff auf alle Ihre AWS Payment Cryptography Keys gewähren und die Möglichkeit geben, alle Payment Cryptography APIs aufzurufen, AWS einschließlich der beiden Operationen. ControlPlane DataPlane

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
            "Effect": "Allow",
            "Action": [
               "payment-cryptography:*"
            ],
            "Resource": [
               "*"
            ]
         }
   ]
}
```

Möglichkeit, APIs mit bestimmten Schlüsseln aufzurufen

In diesem Beispiel möchten Sie einem IAM-Benutzer in Ihrem AWS Konto Zugriff auf einen Ihrer AWS Payment Cryptography Keys gewähren `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` und diese Ressource dann in zwei APIs verwenden, `GenerateCardData` und. `VerifyCardData` Umgekehrt hat der IAM-Benutzer keinen Zugriff darauf, diesen Schlüssel für andere Operationen wie oder zu verwenden `DeleteKey` `ExportKey`

Bei Ressourcen kann es sich entweder um Schlüssel mit Präfix `key` oder um Aliase mit Präfix handeln. `alias`

```
   {
      "Version": "2012-10-17",
      "Statement": [
            {
               "Effect": "Allow",
               "Action": [
                  "payment-cryptography:VerifyCardData",
                  "payment-cryptography:GenerateCardData"
               ],
               "Resource": [
                  "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
               ]
            }
      ]
   }
```

Fähigkeit, eine Ressource gezielt abzulehnen

> **Warning**
>
> Überlegen Sie sich sorgfältig, welche Auswirkungen die Gewährung von Wildcard-Zugriff hat. Ziehen Sie stattdessen ein Modell mit den geringsten Rechten in Betracht.

In diesem Beispiel möchten Sie einem IAM-Benutzer in Ihrem AWS Konto Zugriff auf einen beliebigen Ihrer AWS Payment Cryptography-Schlüssel gewähren, aber Sie möchten die Berechtigungen für einen bestimmten Schlüssel verweigern. Der Benutzer hat Zugriff auf `VerifyCardData` und `GenerateCardData` mit allen Schlüsseln, mit Ausnahme des Schlüssels, der in der Ablehnungsanweisung angegeben ist.

```
   {
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
```

```
            "Action": [
                "payment-cryptography:VerifyCardData",
                "payment-cryptography:GenerateCardData"
            ],
            "Resource": [
                "arn:aws:payment-cryptography:us-east-2:111122223333:key/*"
            ]
        },
        {
            "Effect": "Deny",
            "Action": [
                "payment-cryptography:GenerateCardData"
            ],
            "Resource": [
                "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h"
            ]
        }
    ]
}
```

Fehlerbehebung bei AWS Zahlungen: Kryptografie, Identität und Zugriff

Weitere Themen werden diesem Abschnitt hinzugefügt, sobald Probleme im Zusammenhang mit IAM identifiziert werden, die sich speziell auf die AWS Zahlungskryptografie beziehen. Allgemeine Informationen zur Fehlerbehebung zu IAM-Themen finden Sie im Abschnitt zur Fehlerbehebung im IAM-Benutzerhandbuch.

Überwachung der AWS Zahlungskryptographie

Überwachung ist wichtig, um die Zuverlässigkeit, Verfügbarkeit und Leistung von AWS Payment Cryptography und Ihren anderen AWS-Lösungen aufrechtzuerhalten. AWS stellt die folgenden Überwachungstools zur Verfügung, um die Kryptografie von AWS Zahlungen zu überwachen, zu melden, wenn etwas nicht stimmt, und um gegebenenfalls automatische Aktionen durchzuführen:

- Amazon CloudWatch überwacht Ihre AWS -Ressourcen und -Anwendungen, die Sie auf ausführen, AWS in Echtzeit. Sie können Metriken erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Beispielsweise können Sie mit die CPU-Auslastung oder andere Metriken Ihrer Amazon EC2 EC2-Instances CloudWatch erfassen und bei Bedarf automatisch neue Instances starten. Weitere Informationen finden Sie im Amazon CloudWatch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch.
- Amazon CloudWatch Logs ermöglicht Ihnen die Überwachung, Speicherung und den Zugriff auf Ihre Protokolldateien von Amazon EC2 EC2-InstancesCloudTrail, und anderen Quellen. CloudWatchLogs kann Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch.
- Amazon EventBridge kann verwendet werden, um Ihre AWS -Services zu automatisieren und automatisch auf Systemereignisse zu reagieren, wie z. B. Probleme mit der Anwendungsverfügbarkeit oder Ressourcenänderungen. Ereignisse von AWS-Services werden für EventBridge nahezu in Echtzeit bereitgestellt. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen ausgeführt werden sollen, wenn ein Ereignis mit einer Regel übereinstimmt. Weitere Informationen finden Sie im Amazon EventBridge Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch Benutzerhandbuch.
- AWS CloudTrail erfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS-Kontos erfolgten, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon-S3-Bucket. Sie können die Benutzer und Konten, die AWS aufgerufen haben,

identifizieren, sowie die Quell-IP-Adresse, von der diese Aufrufe stammen, und den Zeitpunkt der Aufrufe ermitteln. Weitere Informationen finden Sie im AWS CloudTrail-Benutzerhandbuch.

> **Note**
>
> AWS CloudTrailProtokolle werden für Steuerungsebenenoperationen unterstütztCreateKey, z. B. für Datenebenenoperationen wie Kartendaten generieren

Protokollieren von API-Aufrufen zur AWS Zahlungskryptographie mit AWS CloudTrail

AWSPayment Cryptography ist in einen ServiceAWS CloudTrail, der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einem AWS -Service in der AWS Zahlungskryptographie durchgeführten Aktionen bietet. CloudTrailerfasst alle API-Aufrufe für AWS Payment Cryptography als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der AWS Payment Cryptography-Konsole und Code-Aufrufe der AWS Payment Cryptography-API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen für einen Amazon S3 S3-Bucket, einschließlich Ereignissen für AWS Payment Cryptography. Auch wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail-Konsole in Event history (Ereignisverlauf) anzeigen. Mit den von gestellten CloudTrail Informationen können Sie die an gestellte AnfrageAWS, die IP-Adresse, von der die Anfrage gestellt wurde, den Zeitpunkt der Anfrage und zusätzliche Details bestimmen.

Weitere Informationen zu CloudTrail finden Sie im AWS CloudTrail-Benutzerhandbuch.

> **Note**
>
> Die Cloudtrail-Integration wird derzeit nur für den Betrieb auf der Steuerungsebene unterstützt.

AWSInformationen zur Zahlungskryptographie in CloudTrail

CloudTrail wird beim Erstellen Ihres AWS-Kontos für Sie aktiviert. Die in der AWS Zahlungskryptografie auftretenden Aktivitäten werden als CloudTrail Ereignis zusammen mit anderen AWS -Serviceereignissen im Ereignisverlauf aufgezeichnet. Sie können die neusten Ereignisse in

Ihr AWS-Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail -API-Ereignisverlauf.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS -Konto, einschließlich Ereignissen für AWS Payment Cryptography, erstellen Sie einen Trail. Ein Trail ermöglicht esCloudTrail, Protokolldateien in einem Amazon S3 S3-Bucket bereitzustellen. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon S3 Bucket bereit. Darüber hinaus können Sie andere AWS-Services konfigurieren, um die in den CloudTrail-Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Trails
- In CloudTrail 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

CloudTrailprotokolliert kryptografische AWS Zahlungsvorgänge wie, CreateKey, ImportKey, DeleteKeyListKeysTagResource, und alle anderen Operationen auf der Steuerungsebene.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Anhand der Identitätsinformationen zur Benutzeridentität können Sie Folgendes bestimmen:

- Ob die Anfrage mit Stammbenutzer- oder AWS Identity and Access Management (IAM)-Anmeldeinformationen ausgeführt wurde.
- Ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer ausgeführt wurde.
- Gibt an, ob die Anforderung aus einem anderen AWS-Service gesendet wurde

Weitere Informationen finden Sie unter CloudTrail-Element userIdentity.

Grundlegendes zu Protokolldateieinträgen zur AWS Zahlungskryptographie

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen Amazon-S3-Bucket übermittelt werden. CloudTrail-Protokolldateien können einen oder

mehrere Einträge enthalten. Ein Ereignis stellt eine einzelne Anfrage aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anfrageparameter. CloudTrail-Protokolleinträge sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die `CreateKey` Aktion AWS Payment Cryptography demonstriert.

```
      {
    CloudTrailEvent: {
      tlsDetails= {
        TlsDetails: {
          cipherSuite=TLS_AES_128_GCM_SHA256,
          tlsVersion=TLSv1.3,
          clientProvidedHostHeader=pdx80.controlplane.paymentcryptography.us-
west-2.amazonaws.com
        }
      },
    requestParameters=CreateKeyInput (
      keyAttributes=KeyAttributes(
        KeyUsage=TR31_B0_BASE_DERIVATION_KEY,
        keyClass=SYMMETRIC_KEY,
        keyAlgorithm=AES_128,
        keyModesOfUse=KeyModesOfUse(
          encrypt=false,
          decrypt=false,
          wrap=false
          unwrap=false,
          generate=false,
          sign=false,
          verify=false,
          deriveKey=true,
          noRestrictions=false)
        ),
      keyCheckValueAlgorithm=null,
      exportable=true,
      enabled=true,
      tags=null),
    eventName=CreateKey,
    userAgent=Coral/Apache-HttpClient5,
    responseElements=CreateKeyOutput(
      key=Key(
```

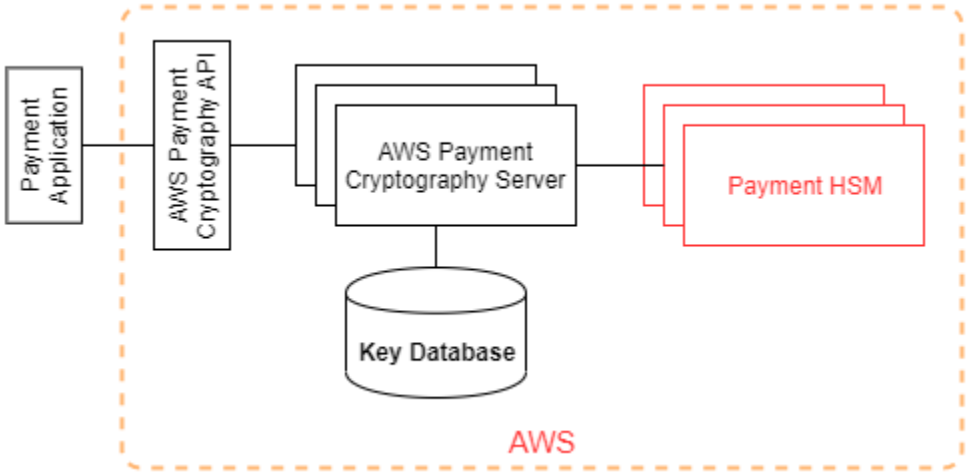
```
        keyArn=arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp,
          keyAttributes=KeyAttributes(
            KeyUsage=TR31_B0_BASE_DERIVATION_KEY,
            keyClass=SYMMETRIC_KEY,
            keyAlgorithm=AES_128,
            keyModesOfUse=KeyModesOfUse(
              encrypt=false,
              decrypt=false,
              wrap=false,
              unwrap=false,
              generate=false,
              sign=false,
              verify=false,
              deriveKey=true,
              noRestrictions=false)
            ),
          keyCheckValue=FE23D3,
          keyCheckValueAlgorithm=ANSI_X9_24,
          enabled=true,
          exportable=true,
          keyState=CREATE_COMPLETE,
          keyOrigin=AWS_PAYMENT_CRYPTOGRAPHY,
          createTimestamp=Sun May 21 18:58:32 UTC 2023,
          usageStartTimestamp=Sun May 21 18:58:32 UTC 2023,
          usageStopTimestamp=null,
          deletePendingTimestamp=null,
          deleteTimestamp=null)
        ),
      sourceIPAddress=192.158.1.38,
        userIdentity={
          UserIdentity: {
            arn=arn:aws:sts::111122223333:assumed-role/TestAssumeRole-us-west-2-PDX80/
ControlPlane-IntegTest-68211a2a-3e9d-42b7-86ac-c682520e0410,
            invokedBy=null,
            accessKeyId=,
            type=AssumedRole,
            sessionContext={
              SessionContext: {
                sessionIssuer={
                  SessionIssuer: {arn=arn:aws:iam::111122223333:role/TestAssumeRole-us-
west-2-PDX80,
                  type=Role,
                  accountId=111122223333,
```

```
                userName=TestAssumeRole-us-west-2-PDX80,
                principalId=}
              },
              attributes={
                SessionContextAttributes: {
                  creationDate=Sun May 21 18:58:31 UTC 2023,
                  mfaAuthenticated=false
                }
              },
              webIdFederationData=null
            }
          },
          username=null,
          principalId=:ControlPlane-User,
          accountId=111122223333,
          identityProvider=null
        }
      },
      eventTime=Sun May 21 18:58:32 UTC 2023,
      managementEvent=true,
      recipientAccountId=111122223333,
      awsRegion=us-west-2,
      requestID=151cdd67-4321-1234-9999-dce10d45c92e,
      eventVersion=1.08, eventType=AwsApiCall,
      readOnly=false,
      eventID=c69e3101-eac2-1b4d-b942-019919ad2faf,
      eventSource=payment-cryptography.amazonaws.com,
      eventCategory=Management,
      additionalEventData={
    }
  }
}
```

Kryptografische Details

AWSPayment Cryptography bietet eine Weboberfläche zur Generierung und Verwaltung kryptografischer Schlüssel für Zahlungsvorgänge. AWS Payment Cryptography bietet standardmäßige Schlüsselverwaltungsdienste und Kryptografie für Zahlungsvorgänge sowie Tools, die Sie für die zentrale Verwaltung und Prüfung verwenden können. Diese Dokumentation enthält eine detaillierte Beschreibung zu den kryptografischen Operationen, die Sie in der AWS Zahlungskryptografie verwenden können, um Sie bei der Bewertung der vom Service angebotenen Funktionen zu unterstützen.

AWSPayment Cryptography enthält mehrere Schnittstellen (einschließlich einer RESTful-API über die AWS-CLI, das AWS-SDK und dasAWS Management Console), um kryptografische Operationen einer verteilten Flotte von PCI PTS HSM-validierten Hardwaresicherheitsmodulen anzufordern.



AWSPayment Cryptography ist ein mehrstufiger Service, der aus AWS -Hosts und einer Reihe von HSMs besteht. Die Gruppierung dieser gestuften Hosts bildet die AWS -Stacks. Alle Anfragen an AWS Payment Cryptography müssen über das Transport-Layer-Security-Protokoll (TLS) erfolgen und auf einem AWS Payment Cryptography Hosts enden. Die Service-Hosts erlauben nur TLS mit einer Verschlüsselungssuite, die Perfect Forward Secrecy bietet. Der Dienst authentifiziert und autorisiert Ihre Anfragen mithilfe derselben Anmeldeinformationen und Richtlinienmechanismen von IAM, die für alle anderen API-Operationen verfügbar sind. AWS

AWSPayment Cryptography Server stellen über ein privates, nicht virtuelles Netzwerk eine Verbindung zum zugrunde liegenden HSM her. Verbindungen zwischen Servicekomponenten und HSM werden mit gegenseitigem TLS (mTLS) zur Authentifizierung und Verschlüsselung gesichert.

Designziele

AWSDie ZahlungsKryptographie wurde entwickelt, um die folgenden Anforderungen zu erfüllen:

- Vertrauenswürdig — Die Verwendung von Schlüsseln ist durch Zugriffssteuerungsrichtlinien geschützt, die Sie definieren und verwalten. Es gibt keinen Mechanismus, um AWS Klartext-ZahlungssteuerungsSchlüssel 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. Keine Amazon-Mitarbeiter haben Zugriff auf HSM-Haupt- (oder Master-) Schlüssel oder Backups. Hauptschlüssel können nicht mit HSMs synchronisiert werden, die nicht Teil einer Region für AWS Zahlungskryptographie sind. Alle anderen Schlüssel sind durch HSM-Hauptschlüssel geschützt. Daher können die AWS Zahlungskryptographie-Schlüssel des Kunden außerhalb des AWS Zahlungskryptographiedienstes, der innerhalb des Kundenkontos betrieben wird, nicht verwendet werden.
- Niedrige Latenz und hoher Durchsatz — Die AWS Zahlungskryptographie bietet kryptografische Operationen auf Latenz- und Durchsatzebene, die für die Verwaltung kryptografischer Zahlungsschlüssel und die Verarbeitung von Zahlungstransaktionen geeignet sind.
- Haltbarkeit — Die Haltbarkeit von kryptografischen Schlüsseln ist so ausgelegt, dass sie der der Services mit der höchsten Haltbarkeit in AWS entspricht. Ein einziger kryptografischer Schlüssel kann mit einem Zahlungsterminal, einer EMV-Chipkarte oder einem anderen sicheren kryptografischen Gerät (SCD) geteilt werden, das seit vielen Jahren verwendet wird.
- Unabhängige Regionen — AWS bietet unabhängige Regionen für Kunden, die den Datenzugriff in verschiedenen Regionen einschränken oder die Anforderungen an den Datenspeicher erfüllen müssen. Die Schlüsselverwendung kann innerhalb einer AWS-Region isoliert werden.
- Sichere Quelle für Zufallszahlen — Da starke Kryptographie von einer wirklich unvorhersehbaren Zufallszahlengenerierung abhängt, bietet eine qualitativ hochwertige und validierte Quelle für AWS Zufallszahlen. Die gesamte Schlüsselgenerierung für AWS Zahlungskryptografie verwendet HSM, das auf PCI PTS HSM gelistet ist und im PCI-Modus arbeitet.
- Prüfung — AWS Payment Cryptography zeichnet die Verwendung und Verwaltung kryptografischer Schlüssel in CloudTrail Protokollen und Serviceprotokollen auf, die über Amazon verfügbar sind. CloudWatch Sie können CloudTrail -Protokolle verwenden, um die Verwendung Ihrer kryptografischen Schlüssel zu überprüfen, einschließlich der Verwendung von Schlüsseln durch Konten, mit denen Sie Schlüssel geteilt haben. AWS Die Zahlungskryptografie wird von externen Gutachtern anhand der geltenden PCI-, Kartenmarken- und regionalen

Zahlungssicherheitsstandards geprüft. Bescheinigungen und Leitfäden zur gemeinsamen Verantwortung sind auf AWS Artifact verfügbar.

- Elastic — AWS Payment Cryptography skaliert je nach Bedarf. Anstatt HSM-Kapazität vorherzusagen und zu reservieren, bietet Payment Cryptography AWS Zahlungskryptografie auf Abruf an. AWS Payment Cryptography übernimmt die Verantwortung für die Aufrechterhaltung der Sicherheit und Konformität von HSM, um ausreichend Kapazität bereitzustellen, um die Spitzennachfrage der Kunden zu decken.

Grundlagen

In den Themen in diesem Kapitel werden die kryptografischen Primitiven von AWS Payment Cryptography und deren Verwendung beschrieben. Sie führen auch die grundlegenden Elemente des Services ein.

Themen

Kryptografische Primitive

AWS Payment Cryptography verwendet parameterfähige, standardmäßige kryptografische Algorithmen, damit Anwendungen die für ihren Anwendungsfall erforderlichen Algorithmen implementieren können. Der Satz kryptografischer Algorithmen wird durch PCI-, ANSI X9-, EMVco und ISO-Standards definiert. Die gesamte Kryptografie wird von standardmäßig aufgelisteten PCI-PHS-HSMs durchgeführt, die im PCI-Modus ausgeführt werden.

Entropie und Zufallszahlengenerierung

AWS Die Schlüsselgenerierung von Payment Cryptography wird auf den AWS Payment Cryptography HSMs durchgeführt. Die HSMs implementieren einen Zufallszahlengenerator, der die PCI PTS HSM-Anforderung für alle unterstützten Schlüsseltypen und Parameter erfüllt.

Symmetrische Schlüsseloperationen

Symmetrische Schlüsselalgorithmen und Schlüsselstärken, die in ANSI X9 TR 31, ANSI X9.24 und PCI PIN Anhang C definiert sind, werden unterstützt:

- Hash-Funktionen – Algorithmen aus der SHA2- und SHA3-Familie mit einer Ausgabegröße von mehr als 2551. Mit Ausnahme der Abwärtskompatibilität mit PTS-POI-v3-Terminals vor CI.
- Verschlüsselung und Entschlüsselung – AES mit einer Schlüsselgröße von mehr als oder gleich 128 Bit oder TDEA mit einer Schlüsselgröße von mehr als oder gleich 112 Bit (2 Schlüssel oder 3 Schlüssel).
- Message Authentication Codes (MACs)CMAC oder Bol mit AES sowie HMAC mit einer genehmigten Hash-Funktion und einer Schlüsselgröße größer oder gleich 128.

AWS Payment Cryptography verwendet AES 256 für HSM-Hauptschlüssel, Datenschutzschlüssel und TLS-Sitzungsschlüssel.

Asymmetrische Schlüsseloperationen

Asymmetrische Schlüsselalgorithmen und Schlüsselstärken, die in ANSI X9 TR 31, ANSI X9.24 und PCI PIN Anhang C definiert sind, werden unterstützt:

- Genehmigte Schlüsseleinrichtungsschemata – wie in NIST SP800-56A (ECC/FCC2-basierte Schlüsselvereinbarung), NIST SP800-56B (IFC-basierte Schlüsselvereinbarung) und NIST SP800-38F (AES-basierte Schlüsselverschlüsselung/-umbruch) beschrieben.

AWS Payment-Cryptography-Hosts erlauben nur Verbindungen zum Service mithilfe von TLS mit einer Verschlüsselungssuite, die Perfect Forward Secrecy bietet.

Schlüsselspeicher

AWS Zahlungskryptografieschlüssel werden durch HSM-AES-256-Hauptschlüssel geschützt und in ANSI-X9-TR-31-Schlüsselblöcken in einer verschlüsselten Datenbank gespeichert. Die Datenbank wird auf AWS Payment-Cryptography-Servern in die In-Memory-Datenbank repliziert.

Laut PCI PIN Security Normative Anhang C sind AES-256-Schlüssel genauso stark wie oder stärker als:

- TDEA mit 3 Schlüsseln
- RSA 15360 Bit
- ECC 512 Bit
- DSA, DH und MQV 15360/512

Schlüsselimport mit symmetrischen Schlüsseln

AWS Payment Cryptography unterstützt den Import von Kryptogrammen und Schlüsselblöcken mit symmetrischen oder öffentlichen Schlüsseln mit einem symmetrischen Schlüsselverschlüsselungsschlüssel (KEK), der so stark oder stärker ist als der geschützte Schlüssel für den Import.

Schlüsselimport mit asymmetrischen Schlüsseln

AWS Payment Cryptography unterstützt den Import von Kryptogrammen und Schlüsselblöcken mit symmetrischen oder öffentlichen Schlüsseln, die durch einen privaten Schlüsselverschlüsselungsschlüssel (KEK) geschützt sind, der so stark oder stärker ist als der geschützte Schlüssel für den Import. Der öffentliche Schlüssel, der für die Entschlüsselung bereitgestellt wird, muss seine Authentizität und Integrität durch ein Zertifikat einer vertrauenswürdigen Stelle des Kunden gewährleistet haben.

Öffentlicher KEK, der von AWS Payment Cryptography bereitgestellt wird, verfügt über den Authentifizierungs- und Integritätsschutz einer Zertifizierungsstelle (CA) mit bestätigter Einhaltung von PCI PIN Security und PCI P2PE Anhang A.

Schlüsselexport

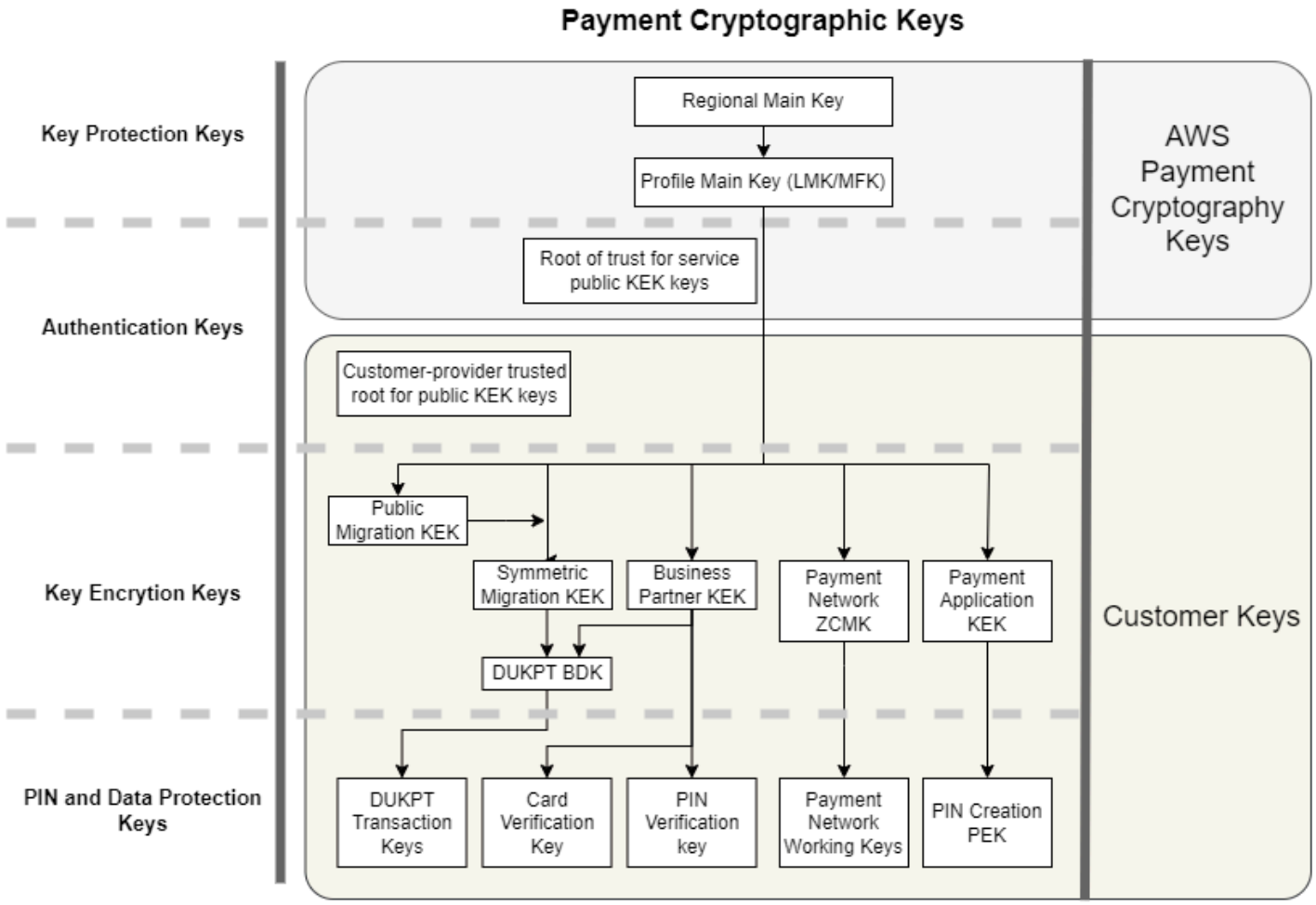
Schlüssel können exportiert und durch Schlüssel mit den entsprechenden geschützt werden KeyUsage , die so stark oder stärker sind als der zu exportierende Schlüssel.

Abgeleitetes Unique Key Per Transaction (DUKPT)-Protokoll

AWS Payment Cryptography unterstützt mit TDEA- und AES-Basisableitungsschlüsseln (BDK), wie in ANSI X9.24-3 beschrieben.

Schlüsselhierarchie

Die Schlüsselhierarchie von AWS Payment Cryptography stellt sicher, dass Schlüssel immer durch Schlüssel geschützt sind, die so stark oder stärker sind als die Schlüssel, die sie schützen.



AWS Zahlungskryptografieschlüssel werden für den Schlüsselschutz innerhalb des Services verwendet:

Schlüssel	Beschreibung
Regionaler Hauptschlüssel	Schutz virtueller HSM-Images oder Profile, die für die kryptografische Verarbeitung verwendet

Schlüssel	Beschreibung
	werden. Dieser Schlüssel existiert nur in HSM und sicheren Backups.
Profil-Hauptschlüssel	Kundenschlüsselschutzschlüssel der obersten Ebene, traditionell als lokaler Masterschlüssel (LMK) oder Masterdateischlüssel (MFK) für Kundenschlüssel bezeichnet. Dieser Schlüssel existiert nur in HSM und sicheren Backups. Profile definieren unterschiedliche HSM-Konfigurationen, die gemäß den Sicherheitsstandards für Zahlungsanwendungsfälle erforderlich sind.
Root of Trust für öffentliche Schlüsselverschlüsselungsschlüssel (KEK) von AWS Payment Cryptography	Der vertrauenswürdige öffentliche Root-Schlüssel und das Zertifikat für die Authentifizierung und Validierung öffentlicher Schlüssel, die von AWS Payment Cryptography für den Schlüsselimport und -export mit asymmetrischen Schlüsseln bereitgestellt werden.

Kundenschlüssel sind nach Schlüsseln gruppiert, die zum Schutz anderer Schlüssel und Schlüssel verwendet werden, die zahlungsbezogene Daten schützen. Dies sind Beispiele für Kundenschlüssel beider Typen:

Schlüssel	Beschreibung
Vom Kunden bereitgestelltes vertrauenswürdiges Stammverzeichnis für öffentliche KEK-Schlüssel	Öffentlicher Schlüssel und Zertifikat, die von Ihnen als Vertrauensstellung für die Authentifizierung und Validierung öffentlicher Schlüssel bereitgestellt werden, die Sie für den Schlüsselimport und -export mit asymmetrischen Schlüsseln bereitstellen.
Schlüsselverschlüsselungsschlüssel (KEK)	KEK werden ausschließlich zur Verschlüsselung anderer Schlüssel für den Austausch

Schlüssel	Beschreibung
	zwischen externen Schlüsselspeichern und AWS Payment Cryptography, Geschäfts partnern, Zahlungsnetzwerken oder verschied enen Anwendungen in Ihrer Organisation verwendet.
Abgeleiteter eindeutiger Schlüssel pro Transaktion (DUKPT)-Basisableitungsschlüssel (BDK)	BDKs werden verwendet, um eindeutige Schlüssel für jedes Zahlungsterminal zu erstellen und Transaktionen von mehreren Terminals in einen einzigen Arbeitsschlüssel der erwerbenden Bank oder des Erwerbers zu übersetzen. Die bewährte Methode, die für PCI Point-to-Point Encryption (P2PE) erforderl ich ist, besteht darin, dass verschiedeneBDKs für verschiedene Terminalmodelle, Schlüssel injektions- oder Initialisierungsservices oder andere Segmentierungen verwendet werden, um die Auswirkungen einer Gefährdung einesK zu begrenzen.
Hauptschlüssel (ZCMK) für die Kontrolle der Zahlungsnetzwerkzone	ZCMK, auch als Zonenschlüssel oder Zonenmasterschlüssel bezeichnet, werden von Zahlungsnetzwerken bereitgestellt, um anfängliche Arbeitsschlüssel einzurichten.
DUKPT-Transaktionsschlüssel	Zahlungsterminals, die für DUKPT konfiguri ert sind, leiten einen eindeutigen Schlüssel für das Terminal und die Transaktion ab. Das HSM, das die Transaktion empfängt, kann den Schlüssel anhand der Terminal-ID und der Transaktionssequenznummer bestimmen.

Schlüssel	Beschreibung
Schlüssel zur Vorbereitung von Kartendaten	Masterschlüssel, Schlüssel und Verifizierungswerte des Ausstellers von Ausstellern sowie Datendateischutzschlüssel zur Kartenpersonalisierung werden verwendet, um Daten für einzelne Karten zur Verwendung durch einen Kartenpersonalisierungsanbieter zu erstellen. Diese Schlüssel und kryptografischen Validierungsdaten werden auch von ausstellenden Banken oder Ausstellern für die Authentifizierung von Kartendaten im Rahmen der Autorisierung von Transaktionen verwendet.
Schlüssel zur Vorbereitung von Kartendaten	Masterschlüssel, Schlüssel und Verifizierungswerte des Ausstellers von Ausstellern sowie Datendateischutzschlüssel zur Kartenpersonalisierung werden verwendet, um Daten für einzelne Karten zur Verwendung durch einen Kartenpersonalisierungsanbieter zu erstellen. Diese Schlüssel und kryptografischen Validierungsdaten werden auch von ausstellenden Banken oder Ausstellern für die Authentifizierung von Kartendaten im Rahmen der Autorisierung von Transaktionen verwendet.
Zahlungsnetzwerk-Arbeitsschlüssel	Wird häufig als Aussteller-Arbeitsschlüssel oder Erwerber-Arbeitsschlüssel bezeichnet. Dies sind die Schlüssel, mit denen Transaktionen verschlüsselt werden, die an Zahlungsnetzwerke gesendet oder von diesen empfangen werden. Diese Schlüssel werden häufig vom Netzwerk rotiert, oft täglich oder stündlich. Dies sind PIN-Verschlüsselungsschlüssel (PEK) für PIN/Debit-Transaktionen.

Schlüssel	Beschreibung
Verschlüsselungsschlüssel für persönliche Identifikationsnummern (PIN) (PEK)	Anwendungen, die PIN-Blöcke erstellen oder entschlüsseln, verwenden PEK, um die Speicherung oder Übertragung von Klartext-PINs zu verhindern.

Interne Operationen

In diesem Thema werden interne Anforderungen beschrieben, die der Service zum Schutz von Kundenschlüsseln und kryptografischen Operationen für einen global verteilten und skalierbaren Zahlungskryptografie- und Schlüsselverwaltungsservice implementiert.

HSM-Spezifikationen und Lebenszyklus

AWS Payment Cryptography verwendet eine Flotte kommerziell verfügbarer HSM. Die HSMs sind FIPS 140-2 Level 3-validiert und verwenden außerdem Firmware-Versionen und die Sicherheitsrichtlinie, die in der Liste der vom PCI Security Standards Europe genehmigten PCI-PTS-Geräte als PCI-HSM-v3-Beschwerde aufgeführt ist. Der PCI PTS HSM-Standard enthält zusätzliche Anforderungen für die Herstellung, Sendung, Bereitstellung, Verwaltung und Beseitigung von HSM-Hardware, die für die Zahlungssicherheit und -konformität wichtig sind, aber nicht von FIPS 140 erfüllt werden.

Alle HSMs werden im PCI-Modus betrieben und mit der PCI PTS HSM-Sicherheitsrichtlinie konfiguriert. Es sind nur Funktionen aktiviert, die zur Unterstützung von Anwendungsfällen für AWS Payment Cryptography erforderlich sind. AWS Payment Cryptography bietet keine Funktionen zum Drucken, Anzeigen oder Zurückgeben von Klartext-PINs.

Physische Sicherheit des HSM-Geräts

Nur HSMs, deren Geräteschlüssel vor der Lieferung von einer AWS Payment Cryptography Certificate Authority (CA) des Herstellers signiert wurden, können vom Service verwendet werden. Die AWS Payment Cryptography ist eine Unterzertifizierungsstelle der CA des Herstellers, die die Vertrauensstellung für HSM-Hersteller und Gerätezertifikate darstellt. Die CA des Herstellers implementiert ANSI TR 34 und hat die Einhaltung von PCI PIN Security Anhang A und PCI P2PE Anhang A bestätigt. Der Hersteller überprüft, ob alle HSMs mit Geräteschlüsseln, die von der AWS

Payment Cryptography CA signiert wurden, an den von AWS designierten Empfänger geliefert werden.

Gemäß PCI PIN Security stellt der Hersteller eine Liste von Seriennummern über einen anderen Kommunikationskanal als die HSM-Sendung bereit. Diese Seriennummern werden bei jedem Schritt der HSM-Installation in AWS-Rechenzentren überprüft. Schließlich validieren die Operatoren für AWS Zahlungskryptografie die Liste der installierten HSM anhand der Liste des Herstellers, bevor sie die Seriennummer zur Liste der HSMs hinzufügen, die AWS Zahlungskryptografieschlüssel erhalten dürfen.

HSMs befinden sich jederzeit im sicheren Speicher oder unter doppelter Kontrolle, einschließlich:

- Lieferung vom Hersteller an eine AWS-Rack-Einrichtung.
- Während der Rack-Baugruppe.
- Transport von der Rack-Einrichtung zu einem Rechenzentrum.
- Empfang und Installation in einem sicheren Rechenzentrumsverarbeitungsraum. HSM-Racks erzwingen die duale Kontrolle mit kartenzugriffsgesteuerten Sperren, alarmierten Eingangsmeldern und Kameras.
- Während des Betriebs.
- Während der Außerbetriebnahme und Beseitigung.

Für jedes HSM wird ein vollständiger chain-of-custodymit individueller Rechenschaftspflicht gepflegt und überwacht.

HSM-Initialisierung

Ein HSM wird erst als Teil der AWS Payment-Cryptography-Flotte initialisiert, nachdem seine Identität und Integrität durch Seriennummern, vom Hersteller installierte Geräteschlüssel und Firmware-Prüfsumme validiert wurden. Nach der Authentizität und Integrität eines HSM wird es konfiguriert, einschließlich der Aktivierung des PCI-Modus. Dann werden die Hauptschlüssel der Region AWS Payment Cryptography und die Profil-Hauptschlüssel eingerichtet und das HSM ist für den Service verfügbar.

HSM-Service und Reparatur

HSM verfügen über servicefähige Komponenten, die keinen Verstoß gegen die kryptografische Grenze des Geräts erfordern. Zu diesen Komponenten gehören wärmende Kabel, Netzteile und

Kabel. Wenn ein HSM oder ein anderes Gerät innerhalb des HSM-Racks gewartet werden muss, wird die doppelte Kontrolle während des gesamten Zeitraums aufrechterhalten, in dem das Rack geöffnet ist.

Außerbetriebnahme von HSM

Die Außerbetriebnahme erfolgt aufgrund end-of-life oder eines Ausfalls eines HSM. HSM werden logisch nullisiert, bevor sie aus ihrem Rack entfernt werden, falls sie funktionsfähig sind, und dann in sicheren Verarbeitungsräumen von AWS-Rechenzentren zerstört. Sie werden niemals zur Reparatur an den Hersteller zurückgegeben, für einen anderen Zweck verwendet oder vor der Beseitigung anderweitig aus einem sicheren Verarbeitungsraum entfernt.

HSM-Firmware-Update

HSM-Firmware-Updates werden angewendet, wenn dies erforderlich ist, um die Abstimmung mit den in PCI PTS HSM und FIPS 140-2 (oder FIPS 140-3) aufgeführten Versionen aufrechtzuerhalten, wenn ein Update sicherheitsbezogen ist oder festgestellt wird, dass Kunden von Funktionen in einer neuen Version profitieren können. AWS HSMs für Zahlungskryptografie führen off-the-shelf Firmware aus, die mit Versionen übereinstimmt, die auf der PCI-PTS-HSM-Liste aufgeführt sind. Neue Firmware-Versionen werden mit den PCI- oder FIPS-zertifizierten Firmware-Versionen auf Integrität überprüft und dann vor dem Rollout auf allen HSMs auf Funktionalität getestet.

Operatorzugriff

Operatoren können zur Fehlerbehebung in seltenen Fällen nicht Konsolenzugriff auf HSM haben, wenn die vom HSM während des normalen Betriebs gesammelten Informationen nicht ausreicht, um ein Problem zu identifizieren oder eine Änderung zu planen. Die folgenden Schritte werden ausgeführt:

- Fehlerbehebungsaktivitäten werden entwickelt und genehmigt und die Sitzung außerhalb der Konsole ist geplant.
- Ein HSM wird aus dem Kundenverarbeitungsservice entfernt.
- Hauptschlüssel werden unter Dual-Kontrolle gelöscht.
- Der Operator darf unter doppelter Kontrolle nicht auf die Konsole zugreifen, um genehmigte Fehlerbehebungsaktivitäten durchzuführen.
  - Nach Beendigung der Nicht-Konsolensitzung wird der erste Bereitstellungsprozess auf dem HSM durchgeführt, wobei die Standardfirmware und -konfiguration zurückgegeben und dann der Hauptschlüssel synchronisiert wird, bevor das HSM an die Kunden zurückgegeben wird.

- Datensätze der Sitzung werden in der Änderungsnachverfolgung aufgezeichnet.
- Die aus der Sitzung abgerufenen Informationen werden zur Planung zukünftiger Änderungen verwendet.

Alle Datensätze, die nicht auf die Konsole zugreifen, werden auf Prozess-Compliance und potenzielle Änderungen an der HSM-Überwachung, dem non-console-access Verwaltungsprozess oder dem Training von Operatoren überprüft.

Schlüsselverwaltung

Alle HSMs in einer Region werden mit einem regionalen Hauptschlüssel synchronisiert. Ein Regions-Hauptschlüssel schützt mindestens einen Profil-Hauptschlüssel. Ein Profil-Hauptschlüssel schützt Kundenschlüssel.

Alle Hauptschlüssel werden von einem HSM generiert und durch symmetrische Schlüsselverteilung mithilfe asymmetrischer Techniken an verteilt, die auf ANSI X9 TR 34 und PCI PIN Anhang A ausgerichtet sind.

Themen

Generation

AES 256-Bit-Hauptschlüssel werden auf einem der für die Service-HSM-Flotte bereitgestellten HSMs mithilfe des Zufallszahlengenerators von PCI PTS HSM generiert.

Hauptschlüsselsynchronisierung der Region

Die HSM-Regions-Hauptschlüssel werden vom Service in der gesamten regionalen Flotte mit Mechanismen synchronisiert, die durch ANSI X9 TR-34 definiert sind, darunter:

- Gegenseitige Authentifizierung mit Schlüsselverteilungs-Host (KDH)- und Schlüsselempfangsgerät (KRD)-Schlüsseln und -Zertifikaten zur Authentifizierung und Integrität von für öffentliche Schlüssel.
- Zertifikate werden von einer Zertifizierungsstelle (CA) signiert, die die Anforderungen von PCI PIN Anhang A2 erfüllt, mit Ausnahme asymmetrischer Algorithmen und Schlüsselstärken, die für den Schutz von AES-256-Bit-Schlüsseln geeignet sind.
- Identifizierung und Schlüsselschutz für die verteilten symmetrischen Schlüssel, die mit ANSI X9 TR-34 und PCI PIN Anhang A1 übereinstimmen, mit Ausnahme asymmetrischer Algorithmen und Schlüsselstärken, die für den Schutz von AES-256-Bit-Schlüsseln geeignet sind.

Regions-Hauptschlüssel werden für HSMs eingerichtet, die für eine Region authentifiziert und bereitgestellt wurden von:

- Auf einem HSM in der Region wird ein Hauptschlüssel generiert. Dieses HSM wird als Schlüsselverteilungs-Host bezeichnet.
- Alle bereitgestellten HSMs in der Region generieren KRD-Authentifizierungstoken, die den öffentlichen Schlüssel des HSM und nicht wiederholbare Authentifizierungsinformationen enthalten.
- KRD-Token werden der KDH-Zulassungsliste hinzugefügt, nachdem der KDH die Identität und Berechtigung des HSM zum Empfangen von Schlüsseln validiert hat.
- Das KDH erzeugt ein authentifiziertes Hauptschlüsseltoken für jedes HSM. Token enthalten KDH-Authentifizierungsinformationen und verschlüsselten Hauptschlüssel, der nur auf einem HSM geladen werden kann, für das er erstellt wurde.
- Jedem HSM wird das dafür erstellte Hauptschlüssel-Token gesendet. Nach der Validierung der eigenen Authentifizierungsinformationen des HSM und der KDH-Authentifizierungsinformationen wird der Hauptschlüssel durch den privaten KRD-Schlüssel entschlüsselt und in den Hauptschlüssel geladen.

Für den Fall, dass ein einzelnes HSM erneut mit einer Region synchronisiert werden muss:

- Es wird erneut validiert und mit Firmware und Konfiguration bereitgestellt.
- Wenn es neu in der Region ist:
  - Das HSM generiert ein KRD-Authentifizierungstoken.

- Der KDH fügt das Token zu seiner Zulassungsliste hinzu.
- Das KDH generiert ein Hauptschlüssel-Token für das HSM.
- Das HSM lädt den Hauptschlüssel.
- Das HSM wird dem Service zur Verfügung gestellt.

Dadurch wird sichergestellt, dass:

- Nur HSM, das für die Verarbeitung von AWS Zahlungskryptografie innerhalb einer Region validiert wurde, kann den Masterschlüssel dieser Region erhalten.
- Nur ein Masterschlüssel aus einem AWS Payment Cryptography HSM kann an ein HSM in der Flotte verteilt werden.

Rotation des regionalen Hauptschlüssels

Regions-Hauptschlüssel werden nach Ablauf des Kryptozeitraums, im unwahrscheinlichen Fall einer mutmaßlichen Schlüsselkompromittierung oder nach Änderungen am Service rotiert, die sich auf die Sicherheit des Schlüssels auswirken.

Ein neuer Regions-Hauptschlüssel wird wie bei der ersten Bereitstellung generiert und verteilt. Gespeicherte Profil-Hauptschlüssel müssen in den neuen regionalen Hauptschlüssel übersetzt werden.

Die Rotation des regionalen Hauptschlüssels wirkt sich nicht auf die Kundenverarbeitung aus.

Profil der Hauptschlüsselsynchronisierung

Profil-Hauptschlüssel sind durch Regions-Hauptschlüssel geschützt. Dadurch wird ein Profil auf eine bestimmte Region beschränkt.

Profil-Hauptschlüssel werden entsprechend bereitgestellt:

- Ein Profil-Hauptschlüssel wird auf einem HSM generiert, auf dem der Regions-Hauptschlüssel synchronisiert ist.
- Der Hauptschlüssel des Profils wird gespeichert und mit der Profilkonfiguration und anderem Kontext verschlüsselt.
- Das Profil wird von jedem HSM in der Region mit dem regionalen Hauptschlüssel für kryptografische Kundenfunktionen verwendet.

Profil-Hauptschlüsselrotation

Profil-Hauptschlüssel werden nach Ablauf des Kryptozeitraums, nach einer mutmaßlichen Schlüsselkompromittierung oder nach Änderungen am Service rotiert, die sich auf die Sicherheit des Schlüssels auswirken.

Schritte zur Rotation:

- Ein neuer Profil-Hauptschlüssel wird generiert und als ausstehender Hauptschlüssel wie bei der ersten Bereitstellung verteilt.
- Ein Hintergrundprozess übersetzt Kundenschlüsselmaterial vom festgelegten Profil-Hauptschlüssel in den ausstehenden Hauptschlüssel.
- Wenn alle Kundenschlüssel mit dem ausstehenden Schlüssel verschlüsselt wurden, wird der ausstehende Schlüssel zum Profil-Hauptschlüssel hochgestuft.
- Ein Hintergrundprozess löscht Kundenschlüsselmaterial, das durch den abgelaufenen Schlüssel geschützt ist.

Die Rotation des Profil-Hauptschlüssels wirkt sich nicht auf die Kundenverarbeitung aus.

Schutz

Schlüssel hängen aus Schutzgründen nur von der Schlüsselhierarchie ab. Der Schutz von Hauptschlüsseln ist entscheidend, um Verlust oder Kompromittierung aller Kundenschlüssel zu verhindern.

Regions-Hauptschlüssel können nur aus der Sicherung auf HSM authentifiziert und für den Service bereitgestellt werden. Diese Schlüssel können nur als gegenseitig authentifizierte, verschlüsselte Hauptschlüssel-Token aus einem bestimmten KDH für ein bestimmtes HSM gespeichert werden.

Profilmasterschlüssel werden mit Profilkonfiguration und Kontextinformationen gespeichert, die nach Region verschlüsselt sind.

Kundenschlüssel werden in Schlüsselblöcken gespeichert, die durch einen Profilmasterschlüssel geschützt sind.

Alle Schlüssel existieren ausschließlich in einem HSM oder werden durch einen anderen Schlüssel gleicher oder stärkerer kryptografischer Stärke geschützt gespeichert.

Haltbarkeit

Kundenschlüssel für Transaktionskryptografie und Geschäftsfunktionen müssen auch in extremen Situationen verfügbar sein, die in der Regel zu Ausfällen führen würden. AWS Payment Cryptography nutzt ein Redundanzmodell auf mehreren Ebenen über Availability Zones und AWS Regionen hinweg. Kunden, die eine höhere Verfügbarkeit und Beständigkeit für kryptografische Zahlungsvorgänge benötigen als die, die der Service bietet, sollten Architekturen mit mehreren Regionen implementieren.

HSM-Authentifizierung und Hauptschlüssel-Token werden gespeichert und können verwendet werden, um einen Hauptschlüssel wiederherzustellen oder mit einem neuen Hauptschlüssel zu synchronisieren, falls ein HSM zurückgesetzt werden muss. Die Token werden archiviert und bei Bedarf nur unter doppelter Kontrolle verwendet.

Kommunikationssicherheit

Extern

AWS Payment-Cryptography-API-Endpunkte erfüllen AWS Sicherheitsstandards, einschließlich TLS bei oder über 1.2 und Signature Version 4 für die Authentifizierung und Integrität von Anfragen.

Eingehende TLS-Verbindungen werden auf Network Load Balancern beendet und über interne TLS-Verbindungen an API-Handler weitergeleitet.

Intern

Die interne Kommunikation zwischen Servicekomponenten und zwischen Servicekomponenten und anderen AWS-Services wird durch TLS unter Verwendung starker Kryptografie geschützt.

HSM befinden sich in einem privaten, nicht virtuellen Netzwerk, das nur von Servicekomponenten aus erreichbar ist. Alle Verbindungen zwischen HSM und Servicekomponenten werden mit gegenseitiger TLS (mTLS) bei oder über TLS 1.2 gesichert. Interne Zertifikate für TLS und mTLS werden von Amazon Certificate Manager mithilfe einer AWS Private Certificate Authority verwaltet. Interne VPCs und das HSM-Netzwerk werden auf ungewöhnliche Aktivitäten und Konfigurationsänderungen überwacht.

Verwaltung von Kundenschlüsseln

Bei hat AWS Kundenvertrauen unsere oberste Priorität. Sie behalten die volle Kontrolle über Ihre Schlüssel, die Sie im Service unter Ihrem AWS-Konto hochladen oder erstellen, und sind für die Konfiguration des Zugriffs auf Schlüssel verantwortlich.

AWS Payment Cryptography hat die volle Verantwortung für die physische HSM-Compliance und Schlüsselverwaltung für Schlüssel, die vom Service verwaltet werden. Dies erfordert den Besitz und die Verwaltung von HSM-Hauptschlüsseln und die Speicherung geschützter Kundenschlüssel in der AWS Payment-Cryptography-Schlüsseldatenbank.

Trennung des Kundenschlüsselraums

AWS Payment Cryptography erzwingt Schlüsselrichtlinien für die gesamte Schlüsselverwendung, einschließlich der Beschränkung von Prinzipalen auf das Konto, das den Schlüssel besitzt, es sei denn, ein Schlüssel wird explizit mit einem anderen Konto geteilt.

Sicherung und Wiederherstellung

Schlüssel und Schlüsselinformationen für eine Region werden von in verschlüsselten Archiven gesichert AWS. Archive erfordern AWS für die Wiederherstellung eine duale Kontrolle durch .

Schlüsselblöcke

Alle Schlüssel werden in Schlüsselblöcken im ANSI X9 TR-31-Format gespeichert.

Schlüssel können aus Kryptogrammen oder anderen von unterstützten Schlüsselblockformaten in den Service importiert werden ImportKey. Ebenso können Schlüssel, sofern sie exportierbar sind, in andere Schlüsselblockformate oder Kryptogramme exportiert werden, die von Schlüsselexportprofilen unterstützt werden.

Schlüsselverwendung

Die Schlüsselnutzung ist auf das beschränkt, das KeyUsage vom Service konfiguriert wurde. Der Service schlägt alle Anfragen mit unangemessener Schlüsselnutzung, Verwendungsmodus oder Algorithmus für die angeforderte kryptografische Operation fehl.

Schlüsselaustauschbeziehungen

PCI PIN Security und PCI P2PE erfordern, dass Organisationen, die Schlüssel teilen, die PINs verschlüsseln, einschließlich des KEK, der zur Freigabe dieser Schlüssel verwendet wurde, diese Schlüssel nicht mit anderen Organisationen teilen. Es hat sich bewährt, symmetrische Schlüssel nur zwischen zwei Parteien gemeinsam zu nutzen, auch innerhalb derselben Organisation. Dadurch werden die Auswirkungen von verdächtigen Schlüsselkompromittierungen minimiert, die das Ersetzen der betroffenen Schlüssel erzwingen.

Selbst Geschäftsfälle, die die gemeinsame Nutzung von Schlüsseln zwischen mehr als zwei Parteien erfordern, sollten die Anzahl der Parteien auf die Mindestanzahl beschränken.

AWS Payment Cryptography bietet Schlüssel-Tags, mit denen die Schlüsselnutzung innerhalb dieser Anforderungen verfolgt und erzwungen werden kann.

Beispielsweise können KEK undK für verschiedene Einrichtungen zur Schlüsselinjektion identifiziert werden, indem ein „KIF“ = „POSStation“ für alle Schlüssel festgelegt wird, die mit diesem Serviceanbieter geteilt werden. Ein weiteres Beispiel wäre, Schlüssel, die mit Zahlungsnetzwerken geteilt werden, mit „Netzwerk“=“ zu markierenPayCard. Mit Tagging können Sie Zugriffskontrollen erstellen und Prüfungsberichte erstellen, um Ihre wichtigsten Verwaltungspraktiken durchzusetzen und zu demonstrieren.

Schlüssellöschung

DeleteKey markiert Schlüssel in der Datenbank nach einem vom Kunden konfigurierbaren Zeitraum zum Löschen. Nach diesem Zeitraum wird der Schlüssel unwiederbringlich gelöscht. Dies ist ein Sicherheitsmechanismus, um das versehentliche oder böswillige Löschen eines Schlüssels zu verhindern. Schlüssel, die zum Löschen markiert sind, sind für keine Aktionen außer verfügbar RestoreKey.

Gelöschte Schlüssel bleiben nach dem Löschen 7 Tage lang in Service-Backups. Sie können während dieses Zeitraums nicht wiederhergestellt werden.

Schlüssel, die zu geschlossenen AWS-Konten gehören, werden zum Löschen markiert. Wenn das Konto vor Ablauf des Löschzeitraums erneut aktiviert wird, werden Schlüssel, die zum Löschen markiert sind, wiederhergestellt, aber deaktiviert. Sie müssen von Ihnen erneut aktiviert werden, um sie für kryptografische Operationen verwenden zu können.

Schlüsselfreigabe

Schlüssel können mithilfe von AWS Resource Access Manager (https://docs.aws.amazon.com/ARG/index.html) für andere Konten innerhalb oder außerhalb Ihrer Organisation freigegeben werden. Schlüssel können in einer Ressourcenfreigabe gruppiert und dann mit einem -Konto oder bestimmten IAM-Benutzern und -Rollen innerhalb eines -Kontos geteilt werden. Sie geben Nutzungsberechtigungen für jede Ressourcenfreigabe an. Freigabeberechtigungen werden durch eine Schlüsselressourcenrichtlinie eingeschränkt. Ein freigegebener Schlüssel erlaubt keine Aktion, die durch eine eigene Richtlinie eingeschränkt wird. Die Freigabeberechtigung kann jederzeit aufgehoben werden.

Protokollierung und Überwachung

Zu den internen Serviceprotokollen gehören:

- CloudTrail Protokolle von AWS-Service-Aufrufen, die vom Service getätigt wurden
- CloudWatch Protokolle beider Ereignisse, die direkt in CloudWatch Protokollen oder Ereignissen von HSM protokolliert werden
- Protokolldateien von HSM- und Servicesystemen
- Protokollarchive

Alle Protokollquellen überwachen und filtern vertrauliche Informationen, einschließlich Schlüsseln. Protokolle werden systematisch überprüft, um sicherzustellen, dass sie keine sensiblen Kundeninformationen enthalten.

Der Zugriff auf Protokolle ist auf Personen beschränkt, die für die Ausführung von Auftragsrollen benötigt werden.

Alle Protokolle werden im Einklang mit den AWS-Richtlinien zur Aufbewahrung von Protokollen aufbewahrt.

Kundenbetrieb

AWS Payment Cryptography hat die volle Verantwortung für die physische HSM-Compliance gemäß PCI-Standards. Der Service bietet auch einen sicheren Schlüsselspeicher und stellt sicher, dass Schlüssel nur für die Zwecke verwendet werden können, die gemäß PCI-Standards zulässig sind und von Ihnen während der Erstellung oder des Imports angegeben werden. Sie sind für die Konfiguration der wichtigsten Attribute und des Zugriffs verantwortlich, um die Sicherheits- und Compliance-Funktionen des Services zu nutzen.

Themen

Generieren von Schlüsseln

Beim Erstellen von Schlüsseln legen Sie die Attribute fest, die der Service verwendet, um die konforme Verwendung des Schlüssels zu erzwingen:

- Algorithmus und Schlüssellänge
- Verwendung
- Verfügbarkeit und Ablauf

Tags, die für die attributbasierte Zugriffskontrolle (ABAC) verwendet werden, um Schlüssel für die Verwendung mit bestimmten Partnern oder Anwendungen einzuschränken, sollten auch während der Erstellung festgelegt werden. Achten Sie darauf, Richtlinien einzubeziehen, um Rollen einzuschränken, die Tags löschen oder ändern dürfen.

Sie sollten sicherstellen, dass die Richtlinien, die die Rollen bestimmen, die den Schlüssel verwenden und verwalten können, vor der Erstellung des Schlüssels festgelegt werden.

> **Note**
>
> IAM-Richtlinien für die CreateKey Befehle können verwendet werden, um die duale Kontrolle für die Schlüsselgenerierung zu erzwingen und zu demonstrieren.

Importieren von Schlüsseln

Beim Importieren von Schlüsseln werden die Attribute zum Erzwingen der konformen Verwendung des Schlüssels vom Service unter Verwendung der kryptografisch begrenzten Informationen im Schlüsselblock festgelegt. Der Mechanismus zum Festlegen des grundlegenden Schlüsselkontexts besteht darin, Schlüsselblöcke zu verwenden, die mit dem Quell-HSM erstellt und durch einen gemeinsamen oder asymmetrischen KEK geschützt wurden. Dies entspricht den PCI-PIN-Anforderungen und behält Nutzung, Algorithmus und Schlüsselstärke der Quellanwendung bei.

Wichtige Schlüsselattribute, Tags und Zugriffskontrollrichtlinien müssen beim Import zusätzlich zu den Informationen im Schlüsselblock eingerichtet werden.

Beim Importieren von Schlüsseln mit Kryptogrammen werden keine Schlüsselattribute aus der Quellanwendung übertragen. Mit diesem Mechanismus müssen Sie die Attribute entsprechend festlegen.

Oft werden Schlüssel mit Klartextkomponenten ausgetauscht, von Schlüssel-Custodianen übertragen und dann mit einer Zeremony geladen, die Dual Control in einem sicheren Raum implementiert. Dies wird von AWS Payment Cryptography nicht direkt unterstützt. Die API exportiert einen öffentlichen Schlüssel mit einem Zertifikat, das von Ihrem eigenen HSM importiert werden kann, um

einen Schlüsselblock zu exportieren, der vom Service importiert werden kann. Die ermöglicht die Verwendung Ihres eigenen HSM zum Laden von Klartextkomponenten.

Sie sollten Schlüsselprüfungswerte (KCV) verwenden, um zu überprüfen, ob importierte Schlüssel mit Quellschlüsseln übereinstimmen.

IAM-Richtlinien für die ImportKey -API können verwendet werden, um die duale Kontrolle für den Schlüsselimport zu erzwingen und zu demonstrieren.

Exportieren von Schlüsseln

Die Freigabe von Schlüsseln mit Partnern oder On-Premises-Anwendungen erfordert möglicherweise den Export von Schlüsseln. Die Verwendung von Schlüsselblöcken für Exporte behält den grundlegenden Schlüsselkontext mit dem verschlüsselten Schlüsselmaterial bei.

Schlüssel-Tags können verwendet werden, um den Export von Schlüsseln nach KEK zu beschränken, die dasselbe Tag und denselben Wert haben.

AWS Payment Cryptography bietet oder zeigt keine Klartext-Schlüsselkomponenten an. Dies erfordert direkten Zugriff von Schlüsseladministratoren auf PCI PTS HSM oder ISO 13491-getestete sichere kryptografische Geräte (SCD) zum Anzeigen oder Drucken. Sie können einen asymmetrischen KEK oder einen symmetrischen KEK mit Ihrer SCD einrichten, um die Erstellungszeremony für Klartextschlüsselkomponenten unter dualer Kontrolle durchzuführen.

Schlüsselprüfungswerte (KCV) sollten verwendet werden, um zu überprüfen, ob sie von den Ziel-HSM-Übereinstimmungs-Quellschlüsseln importiert wurden.

Löschen von Schlüsseln

Sie können die Löschschlüssel-API verwenden, um Schlüssel nach einem von Ihnen konfigurierten Zeitraum zum Löschen zu planen. Vor dieser Zeit sind Schlüssel wiederherstellbar. Sobald Schlüssel gelöscht wurden, werden sie dauerhaft aus dem Service entfernt.

IAM-Richtlinien für die DeleteKey -API können verwendet werden, um die doppelte Kontrolle für das Löschen von Schlüsseln zu erzwingen und zu demonstrieren.

Rotieren von -Schlüsseln

Die Auswirkung der Schlüsselrotation kann mithilfe des Schlüsselalias implementiert werden, indem ein neuer Schlüssel erstellt oder importiert wird, und dann der Schlüsselalias so geändert wird, dass

er auf den neuen Schlüssel verweist. Der alte Schlüssel würde je nach Ihren Verwaltungspraktiken gelöscht oder deaktiviert.

Kontingente für AWS Payment Cryptography

Das AWS-Konto verfügt über Standardkontingente (früher als Limits bezeichnet) für jeden AWS-Service. Sofern nicht anders angegeben, ist jedes Kontingent regionsspezifisch. Sie können Erhöhungen für einige Kontingente beantragen und andere Kontingente können nicht erhöht werden.

Name	Standard	Anpas	Beschreibung
Aliasnamen	Jede unterstützte Region: 2.000	Ja	Die maximale Anzahl von Aliassen, die Sie in diesem Konto in der aktuellen Region haben können.
Kombinierte Rate von Anforderungen auf Steuerebene	Jede unterstützte Region: 5 pro Sekunde	Ja	Die maximale Anzahl von Anforderungen auf Steuerebene pro Sekunde, die Sie in diesem Konto in der aktuellen Region vornehmen können. Dieses Kontingent gilt für alle Operationen der Steuerebene zusammengenommen.
Kombinierte Rate von Anforderungen auf Datenebene (asymmetrisch)	Jede unterstützte Region: 20 pro Sekunde	Ja	Die maximale Anzahl von Anforderungen pro Sekunde für Operationen auf Datenebene mit einem asymmetrischen Schlüssel, die Sie in diesem Konto in der aktuellen Region vornehmen können. Dieses Kontingent gilt für alle Operationen auf

Name	Standard	Anpas	Beschreibung
			Datenebene zusammeng enommen.
Kombinierte Rate von Anforderungen auf Datenebene (symmetrisch)	Jede unterstützte Region: 500 pro Sekunde	Ja	Die maximale Anzahl von Anforderungen pro Sekunde für Operation en auf Datenebene mit einem symmetris chen Schlüssel, die Sie in diesem Konto in der aktuellen Region vornehmen können. Dieses Kontingent gilt für alle Operationen auf Datenebene zusammeng enommen.
Schlüssel	Jede unterstützte Region: 2.000	Ja	Die maximale Anzahl von Schlüsseln, die Sie in diesem Konto in der aktuellen Region haben können, ausgenommen gelöschte Schlüssel.

Dokumentenverlauf für das AWS Payment Cryptography User Guide

In der folgenden Tabelle werden die Dokumentationsversionen für AWS Payment Cryptography beschrieben.

Änderung	Beschreibung	Datum
Veröffentlichung der Funktionen	Hinzufügen von Informationen zu VPC-Endpunkten (PrivateLink) und iCVV-Beispielen.	30. Mai 2024
Veröffentlichung der Funktion	Es wurden Informationen zu neuen Funktionen rund um den Import/Export von Schlüsseln mit RSA und den Export von DUKPT IPEK/IK-Schlüsseln hinzugefügt.	15. Januar 2024
Erstversion	Erste Veröffentlichung des Benutzerleitfadens AWS zur Zahlungskryptografie	08. Juni 2023

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.