Panduan Pengguna

AWS Kriptografi Pembayaran

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Kriptografi Pembayaran: Panduan Pengguna

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau mungkin tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Apa itu Kriptografi AWS Pembayaran?

AWS Kriptografi Pembayaran adalah AWS layanan terkelola yang menyediakan akses ke fungsi kriptografi dan manajemen kunci yang digunakan dalam pemrosesan pembayaran sesuai dengan standar industri kartu pembayaran (PCI) tanpa perlu Anda mendapatkan instans HSM pembayaran khusus. AWS Kriptografi Pembayaran memberi pelanggan yang melakukan fungsi pembayaran seperti pengakuisisi, fasilitator pembayaran, jaringan, sakelar, prosesor, dan bank kemampuan untuk memindahkan operasi kriptografi pembayaran mereka lebih dekat ke aplikasi di cloud dan meminimalkan ketergantungan pada pusat data tambahan atau fasilitas kolokasi yang berisi pembayaran khusus. HSMs

Layanan ini dirancang untuk memenuhi aturan industri yang berlaku termasuk PIN PCI, PCI P2PE, dan PCI DSS, dan layanan ini memanfaatkan perangkat keras yang disertifikasi PCI PTS HSM V3 dan FIPS 140-2 Level 3. Ini dirancang untuk mendukung latensi rendah dan tingkat up-time dan ketahanan yang tinggi. AWS Kriptografi Pembayaran sepenuhnya elastis dan menghilangkan banyak persyaratan operasional di tempat HSMs, seperti kebutuhan untuk menyediakan perangkat keras, mengelola materi kunci dengan aman, dan untuk menjaga cadangan darurat di fasilitas yang aman. AWS Kriptografi Pembayaran juga memberi Anda opsi untuk berbagi kunci dengan mitra Anda secara elektronik, menghilangkan kebutuhan untuk berbagi komponen paper clear text.

Anda dapat menggunakan AWS Payment Cryptography Control Plane API untuk membuat dan mengelola kunci.

Anda dapat menggunakan AWS Payment Cryptography Data Plane API untuk menggunakan kunci enkripsi untuk pemrosesan transaksi terkait pembayaran dan operasi kriptografi terkait.

AWS Kriptografi Pembayaran menyediakan fitur penting yang dapat Anda gunakan untuk mengelola kunci Anda:

- Buat dan kelola kunci Kriptografi AWS Pembayaran simetris dan asimetris, termasuk kunci TDES, AES, dan RSA dan tentukan tujuan yang dimaksudkan seperti untuk pembuatan CVV atau derivasi kunci DUKPT.
- Secara otomatis menyimpan kunci Kriptografi AWS Pembayaran Anda dengan aman, dilindungi oleh modul keamanan perangkat keras (HSMs) sambil menegakkan pemisahan kunci antara kasus penggunaan.
- Buat, hapus, daftar, dan perbarui alias, yang merupakan “nama ramah” yang dapat digunakan untuk mengakses atau mengontrol akses ke kunci Kriptografi AWS Pembayaran Anda.

- Tandai kunci Kriptografi AWS Pembayaran Anda untuk identifikasi, pengelompokan, otomatisasi, kontrol akses, dan pelacakan biaya.
- Impor dan ekspor kunci simetris antara Kriptografi AWS Pembayaran dan HSM Anda (atau pihak ke-3) menggunakan Kunci Enkripsi Kunci (KEK) mengikuti TR-31 (Spesifikasi Blok Kunci Pertukaran Kunci Aman yang Dapat Dioperasikan).
- Impor dan ekspor Kunci Enkripsi Kunci simetris (KEK) antara Kriptografi AWS Pembayaran dan sistem lain menggunakan pasangan kunci asimetris berikut dengan menggunakan sarana elektronik seperti TR-34 (Metode Untuk Distribusi Kunci Simetris Menggunakan Teknik Asimetris).

Anda dapat menggunakan kunci Kriptografi AWS Pembayaran Anda dalam operasi kriptografi, seperti:

- Mengenkripsi, mendekripsi, dan mengenkripsi ulang data dengan kunci Kriptografi Pembayaran simetris atau asimetris. AWS
- Terjemahkan data sensitif dengan aman (seperti pin pemegang kartu) di antara kunci enkripsi tanpa mengekspos teks yang jelas sesuai dengan aturan PIN PCI.
- Menghasilkan atau memvalidasi data pemegang kartu seperti CVV, atau ARQC. CVV2
- Buat dan validasi pin pemegang kartu.
- Menghasilkan atau memvalidasi tanda tangan MAC.

Konsep

Pelajari istilah dan konsep dasar yang digunakan dalam Kriptografi AWS Pembayaran dan bagaimana Anda dapat menggunakannya untuk membantu Anda melindungi data Anda.

Alias

> Nama yang mudah digunakan yang dikaitkan dengan kunci Kriptografi AWS Pembayaran. Alias dapat digunakan secara bergantian dengan ARN kunci di banyak operasi API Kriptografi Pembayaran. AWS Alias memungkinkan kunci diputar atau diubah tanpa memengaruhi kode aplikasi Anda. Nama alias adalah satu string berisi hingga 256 karakter. Ini secara unik mengidentifikasi kunci Kriptografi AWS Pembayaran terkait dalam akun dan wilayah. Dalam Kriptografi AWS Pembayaran, nama alias selalu dimulai dengan. `alias/`
>
> Format nama alias adalah sebagai berikut:

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

Sebagai contoh:

```
alias/sampleAlias2
```

ARN kunci

ARN kunci adalah Nama Sumber Daya Amazon (ARN) dari entri kunci dalam Kriptografi Pembayaran. AWS Ini adalah pengidentifikasi unik dan sepenuhnya memenuhi syarat untuk kunci Kriptografi AWS Pembayaran. ARN kunci mencakup Akun AWS, wilayah, dan ID yang dihasilkan secara acak. ARN tidak terkait atau berasal dari bahan kunci. Karena mereka secara otomatis ditetapkan selama operasi membuat atau mengimpor, nilai-nilai ini tidak idempoten. Mengimpor kunci yang sama beberapa kali akan menghasilkan beberapa kunci ARNs dengan siklus hidupnya sendiri.

Format ARN kunci adalah sebagai berikut:

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

Berikut ini adalah contoh kunci ARN:

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

Pengidentifikasi Kunci

Pengenal Kunci adalah referensi ke kunci dan satu (atau lebih) dari mereka adalah input khas untuk operasi Kriptografi AWS Pembayaran. Pengidentifikasi kunci yang valid bisa berupa Key Arn a Key Alias.

AWS Kunci Kriptografi Pembayaran

AWS Kunci Kriptografi Pembayaran (kunci) digunakan untuk semua fungsi kriptografi. Kunci dihasilkan secara langsung oleh Anda menggunakan perintah create key atau ditambahkan ke sistem dengan memanggil key import. Asal kunci dapat ditentukan dengan meninjau atribut KeyOrigin. AWS Kriptografi Pembayaran juga mendukung kunci turunan atau perantara yang digunakan selama operasi kriptografi seperti yang digunakan oleh DUKPT.

Kunci-kunci ini memiliki atribut yang tidak dapat diubah dan dapat diubah yang ditentukan pada saat pembuatan. Atribut, seperti algoritma, panjang, dan penggunaan didefinisikan pada saat

pembuatan dan tidak dapat diubah. Lainnya, seperti tanggal efektif atau tanggal kedaluwarsa, dapat dimodifikasi. Lihat Referensi API Kriptografi AWS Pembayaran untuk daftar lengkap atribut Kunci Kriptografi AWS Pembayaran.

AWS Kunci Kriptografi Pembayaran memiliki tipe kunci, terutama didefinisikan oleh ANSI X9 TR 31, yang membatasi penggunaannya untuk tujuan yang dimaksudkan sebagaimana ditentukan dalam PCI PIN v3.1 Persyaratan 19.

Atribut terikat ke kunci menggunakan blok kunci saat disimpan, dibagikan dengan akun lain, atau diekspor seperti yang ditentukan dalam PCI PIN v3.1 Persyaratan 18-3.

Kunci diidentifikasi dalam platform Kriptografi AWS Pembayaran menggunakan nilai unik yang dikenal sebagai nama sumber daya Amazon utama (ARN).

> **Note**
>
> Kunci ARN dihasilkan ketika kunci awalnya dibuat atau diimpor ke layanan Kriptografi AWS Pembayaran. Jadi, jika menambahkan materi kunci yang sama beberapa kali menggunakan fungsionalitas kunci impor, materi kunci yang sama akan ditempatkan di bawah beberapa kunci ARNS tetapi masing-masing dengan siklus hidup kunci yang berbeda.

Terminologi industri

Topik

- Jenis kunci umum
- Istilah lain

Jenis kunci umum

AWS Kunci Kriptografi Pembayaran

Kunci Kriptografi AWS Pembayaran ada dalam satu AWS Region. Ini terdiri dari metadata kunci dan materi yang disimpan dalam Layanan Kriptografi AWS Pembayaran. Kunci dapat diimpor dari sumber eksternal sebagai blok kunci TR-31 atau dihasilkan oleh Layanan Kriptografi AWS Pembayaran.

AWK

Acquirer working key (AWK) adalah kunci yang biasanya digunakan untuk bertukar data antara acquirer/acquirer prosesor dan jaringan (seperti Visa atau Mastercard). Secara historis AWK memanfaatkan 3DES untuk enkripsi dan akan direpresentasikan sebagai _P0_PIN_ENCRYPTION_KEY. TR31

BDK

Kunci derivasi dasar (BDK) adalah kunci kerja yang digunakan untuk menurunkan kunci berikutnya dan biasanya digunakan sebagai bagian dari proses PCI PIN dan PCI P2PE DUKPT. Hal ini dilambangkan sebagai TR31 _B0_BASE_DERIVATION_KEY.

CMK

Kunci master kartu (CMK) adalah satu atau lebih kunci spesifik kartu yang biasanya berasal dari Kunci Master Penerbit, PAN dan PSN dan biasanya merupakan kunci 3DES. Kunci-kunci ini disimpan di EMV Chip selama personalisasi. Contohnya CMKs termasuk kunci AC, SMI dan SMC.

CMK-AC

Kunci kriptogram aplikasi (AC) digunakan sebagai bagian dari transaksi EMV untuk menghasilkan kriptogram transaksi dan merupakan jenis kunci master kartu.

CMK-SMI

Kunci integritas pesan aman (SMI) digunakan sebagai bagian dari EMV untuk memverifikasi integritas muatan yang dikirim ke kartu menggunakan MAC seperti skrip pembaruan pin. Ini adalah jenis kunci master kartu.

CMK-SMC

Kunci kerahasiaan pesan aman (SMC) digunakan sebagai bagian dari EMV untuk mengenkripsi data yang dikirim ke kartu seperti pembaruan pin. Ini adalah jenis kunci master kartu.

CVK

Kunci verifikasi kartu (CVK) adalah kunci yang digunakan untuk menghasilkan CVV, CVV2 dan nilai serupa menggunakan algoritma yang ditentukan serta memvalidasi input. Hal ini dilambangkan sebagai TR31 _C0_CARD_VERIFICATION_KEY.

IMK

Sebuah issuer master key (IMK) adalah kunci master yang digunakan sebagai bagian dari personalisasi kartu chip EMV. Biasanya akan ada 3 IMKs - satu masing-masing untuk kunci

AC (cryptogram), SMI (script master key forintegrity/signature), and SMC (script master key for confidentiality/encryption).

IK

Kunci awal (IK) adalah kunci pertama yang digunakan dalam proses DUKPT dan berasal dari Kunci Derivasi Dasar (BDK). Tidak ada transaksi yang diproses pada kunci ini, tetapi digunakan untuk mendapatkan kunci future yang akan digunakan untuk transaksi. Metode derivasi untuk membuat IK didefinisikan dalam X9. 24-1:2017. Ketika TDES BDK digunakan, X9. 24-1:2009 adalah standar yang berlaku dan IK diganti dengan Initial Pin Encryption Key (IPEK).

IPEK

Kunci enkripsi PIN awal (IPEK) adalah kunci awal yang digunakan dalam proses DUKPT dan berasal dari Kunci Derivasi Dasar (BDK). Tidak ada transaksi yang diproses pada kunci ini, tetapi digunakan untuk mendapatkan kunci future yang akan digunakan untuk transaksi. IPEK adalah keliru karena kunci ini juga dapat digunakan untuk mendapatkan enkripsi data dan kunci mac. Metode derivasi untuk membuat IPEK didefinisikan dalam X9. 24-1:2009. Ketika AES BDK digunakan, X9. 24-1:2017 adalah standar yang berlaku dan IPEK diganti dengan Initial Key (IK).

IWK

Kunci kerja penerbit (IWK) adalah kunci yang biasanya digunakan untuk bertukar data antara issuer/issuer prosesor dan jaringan (seperti Visa atau Mastercard). Secara historis IWK memanfaatkan 3DES untuk enkripsi dan direpresentasikan sebagai _P0_PIN_ENCRYPTION_KEY. TR31

KBPK

Kunci enkripsi blok kunci (KBPK) adalah jenis kunci simetris yang digunakan untuk melindungi blok kunci dan dengan demikian kunci lainnya. wrap/encrypt KBPK mirip dengan KEK tetapi KEK secara langsung melindungi materi kunci sedangkan dalam TR-31 dan skema serupa, KBPK hanya secara tidak langsung melindungi kunci kerja. Saat menggunakan TR-31, TR31_K1_KEY_BLOCK_PROTECTION_KEY adalah tipe kunci yang benar, meskipun _K0_KEY_ENCRYPTION_KEY didukung secara bergantian untuk tujuan historis. TR31

KEK

Kunci enkripsi kunci (KEK) adalah kunci yang digunakan untuk mengenkripsi kunci lain baik untuk transmisi atau penyimpanan. Kunci yang dimaksudkan untuk melindungi kunci lain biasanya memiliki KeyUsage TR31_K0_KEY_ENCRYPTION_KEY sesuai dengan standar. TR-31

PEK

Kunci enkripsi PIN (PEK) adalah jenis kunci kerja yang digunakan untuk mengenkripsi PINs baik untuk penyimpanan atau transmisi antara dua pihak. IWK dan AWK adalah dua contoh penggunaan spesifik kunci enkripsi pin. Kunci ini direpresentasikan sebagai TR31_P0_PIN_ENCRYPTION_KEY.

PGK

PGK (Pin Generation Key) adalah nama lain untuk Kunci Verifikasi Pin. Ini sebenarnya tidak digunakan untuk menghasilkan pin (yang secara default adalah angka acak kriptografis) tetapi digunakan untuk menghasilkan nilai verifikasi seperti PVV.

PRK

Kunci Wilayah Utama adalah sumber replikasi otoritatif untuk kunci Kriptografi Pembayaran tertentu yang replikasi telah diaktifkan. PRK adalah referensi ke peran kunci Kriptografi Pembayaran sumber dalam konfigurasi replikasi kunci Multi-Region. Ketika replikasi diaktifkan pada kunci Kriptografi Pembayaran, itu disebut sebagai PRK untuk konfigurasi replikasi kunci tertentu.

PVK

Kunci verifikasi PIN (PVK) adalah jenis kunci kerja yang digunakan untuk menghasilkan nilai verifikasi PIN seperti PVV. Dua jenis yang paling umum adalah TR31_V1_ _PIN_VERIFICATION_KEY digunakan untuk menghasilkan nilai offset dan IBM3624 _V2_VISA_PIN_VERIFICATION_KEY digunakan untuk nilai verifikasi. IBM3624 TR31 Visa/ABA Ini juga bisa dikenal sebagai Pin Generation Key.

RRK

Kunci Replica Region adalah bahan kunci yang direplikasi dan metadata yang disalin dengan aman dari PRK ke replika yang dikonfigurasi. AWS Region RRK adalah replika hanya baca dari kunci Kriptografi Pembayaran. RRK adalah referensi peran yang dimainkan kunci tertentu dalam konfigurasi replikasi kunci Multi-Region. Setiap perubahan metadata kunci, termasuk pengaturan replikasi harus diterapkan ke PRK.

Istilah lain

ARQC

Authorization Request Cryptogram (ARQC) adalah kriptogram yang dihasilkan pada waktu transaksi oleh kartu chip standar EMV (atau implementasi tanpa kontak yang setara). Biasanya, ARQC dihasilkan oleh kartu chip dan diteruskan ke penerbit atau agen mereka untuk memverifikasi pada waktu transaksi.

CVV

Nilai verifikasi kartu adalah nilai rahasia statis yang secara tradisional tertanam pada strip magnetik dan digunakan untuk memvalidasi keaslian transaksi. Algoritma ini juga digunakan untuk tujuan lain seperti iCVV, CAVV,. CVV2 Ini mungkin tidak disematkan dengan cara ini untuk kasus penggunaan lainnya.

CVV2

Nilai verifikasi kartu 2 adalah nilai rahasia statis yang secara tradisional dicetak di bagian depan (atau belakang) kartu pembayaran dan digunakan untuk memverifikasi keaslian kartu yang tidak ada pembayaran (seperti di telepon atau online). Ini menggunakan algoritma yang sama dengan CVV tetapi kode layanan diatur ke 000.

iCVV

iCvv adalah nilai CVV2 -like tetapi disematkan dengan data setara track2 pada kartu EMV (Chip). Nilai ini dihitung menggunakan kode layanan 999 dan berbeda dari CVV1/CVV2 untuk mencegah informasi yang dicuri digunakan untuk membuat kredensyal pembayaran baru dari jenis yang berbeda. Misalnya, jika data transaksi chip diperoleh, tidak mungkin menggunakan data ini untuk menghasilkan strip magnetik (CVV1) atau untuk pembelian online (CVV2).

Ini menggunakan ??? kunci

DUKPT

Derived Unique Key Per Transaction (DUKPT) adalah standar manajemen kunci yang biasanya digunakan untuk menentukan penggunaan kunci enkripsi sekali pakai pada POS/POI fisik. Secara historis DUKPT memanfaatkan 3DES untuk enkripsi. Standar industri untuk DUKPT didefinisikan dalam ANSI X9.24-3-2017.

ECC

ECC (Elliptic Curve Cryptography) adalah sistem kriptografi kunci publik yang menggunakan matematika kurva elips untuk membuat kunci enkripsi. ECC menyediakan tingkat keamanan

yang sama dengan metode tradisional seperti RSA tetapi dengan panjang kunci yang jauh lebih pendek, memberikan keamanan yang setara dengan cara yang lebih efisien. Ini sangat relevan untuk kasus penggunaan di mana RSA bukan solusi praktis (panjang kunci RSA > 4096 bit). AWS Kriptografi Pembayaran mendukung kurva yang ditentukan oleh NIST untuk digunakan dalam operasi ECDH.

ECDH

ECDH (Elliptic Curve Diffie-Hellman) adalah protokol perjanjian kunci yang memungkinkan dua pihak untuk membangun rahasia bersama (seperti KEK atau PEK). Dalam ECDH, Pihak A dan B masing-masing memiliki pasangan kunci publik-pribadi mereka sendiri dan bertukar kunci publik satu sama lain (dalam bentuk sertifikat untuk Kriptografi AWS Pembayaran) serta metadata derivasi kunci (metode derivasi, jenis hash dan info bersama). Kedua belah pihak mengalikan kunci privat mereka dengan kunci publik yang lain dan karena sifat kurva elips, kedua belah pihak dapat memperoleh (menghasilkan) kunci yang dihasilkan.

EMV

EMV (awalnya Europay, Mastercard, Visa) adalah badan teknis yang bekerja dengan pemangku kepentingan pembayaran untuk menciptakan standar dan teknologi pembayaran yang dapat dioperasikan. Salah satu contoh standar adalah untuk chip/contactless kartu dan terminal pembayaran yang berinteraksi dengan mereka, termasuk kriptografi yang digunakan. Derivasi kunci EMV mengacu pada metode menghasilkan kunci unik untuk setiap kartu pembayaran berdasarkan set kunci awal seperti `IMK`

HSM

Modul Keamanan Perangkat Keras (HSM) adalah perangkat fisik yang melindungi operasi kriptografi (misalnya, enkripsi, dekripsi, dan tanda tangan digital) serta kunci yang mendasari yang digunakan untuk operasi ini.

KCAAS

A Key Custodian As A Service (KCAAS) menyediakan berbagai layanan yang berkaitan dengan manajemen kunci. Untuk kunci pembayaran, mereka biasanya dapat mengonversi komponen kunci berbasis kertas ke formulir elektronik yang didukung oleh Kriptografi AWS Pembayaran atau mengonversi kunci yang dilindungi secara elektronik menjadi komponen berbasis kertas yang mungkin diperlukan oleh vendor tertentu. Mereka juga dapat menyediakan layanan escrow utama untuk kunci yang kerugiannya akan merugikan opeasi Anda yang sedang berlangsung. Vendor KCAAS dapat membantu pelanggan melepaskan beban operasional pengelolaan materi utama di luar layanan yang aman seperti Kriptografi AWS Pembayaran dengan cara yang sesuai dengan standar PCI DSS, PCI PIN, dan PCI P2PE.

KCV

Key Check Value (KCV) mengacu pada berbagai metode checksum primer yang digunakan untuk membandingkan kunci satu sama lain tanpa memiliki akses ke materi kunci yang sebenarnya. KCV juga telah digunakan untuk validasi integritas (terutama ketika bertukar kunci), meskipun peran ini sekarang disertakan sebagai bagian dari format blok kunci seperti. TR-31 Untuk kunci TDES, KCV dihitung dengan mengenkripsi 8 byte, masing-masing dengan nilai nol, dengan kunci yang akan diperiksa dan mempertahankan 3 byte urutan tertinggi dari hasil terenkripsi. Untuk kunci AES, KCV dihitung menggunakan algoritma CMAC di mana data input adalah 16 byte nol dan mempertahankan 3 byte urutan tertinggi dari hasil terenkripsi.

KDH

Key Distribution Host (KDH) adalah perangkat atau sistem yang mengirim kunci dalam proses pertukaran kunci seperti TR-34. Saat mengirim kunci dari Kriptografi AWS Pembayaran, itu dianggap sebagai KDH.

KIF

Fasilitas Injeksi Kunci (KIF) adalah fasilitas aman yang digunakan untuk menginisialisasi terminal pembayaran termasuk memuatnya dengan kunci enkripsi.

KRD

Perangkat Penerima Kunci (KRD) adalah perangkat yang menerima kunci dalam proses pertukaran kunci seperti TR-34. Saat mengirim kunci ke Kriptografi AWS Pembayaran, itu dianggap sebagai KRD.

KSN

Key Serial Number (KSN) adalah nilai yang digunakan sebagai masukan untuk enkripsi/dekripsi DUKPT untuk membuat kunci enkripsi unik per transaksi. KSN biasanya terdiri dari pengenal BDK, ID terminal semi-unik serta penghitung transaksi yang meningkat pada setiap transisi yang diproses pada terminal pembayaran tertentu. Per X9.24, untuk TDES 10 byte KSN biasanya terdiri dari 24 bit untuk Key Set ID, 19 bit untuk ID terminal dan 21 bit untuk penghitung transaksi meskipun batas antara Key Set ID dan ID terminal tidak berdampak pada fungsi Kriptografi Pembayaran. AWS Untuk AES, KSN 12 byte biasanya terdiri dari 32 bit untuk ID BDK, 32 bit untuk pengenal derivasi (ID) dan 32 bit untuk penghitung transaksi.

MPoC

MPoC (Mobile Point of Sale on Commercial Hardware) adalah standar PCI yang membahas persyaratan keamanan untuk solusi yang memungkinkan pedagang menerima pembayaran

pemegang kartu PINs atau tanpa kontak menggunakan smartphone atau perangkat seluler komersial off-the-shelf (COTS) lainnya.

PANCI

Nomor Akun Utama (PAN) adalah pengenal unik untuk akun seperti kartu kredit atau debit. Biasanya panjangnya 13-19 digit. 6-8 digit pertama mengidentifikasi jaringan dan bank penerbit.

Blok PIN

Sebuah blok data yang berisi PIN selama pemrosesan atau transmisi serta elemen data lainnya. Format blok PIN menstandarisasi konten blok PIN dan bagaimana hal itu dapat diproses untuk mengambil PIN. Sebagian besar blok PIN terdiri dari PIN, panjang PIN, dan sering berisi sebagian atau seluruh PAN. AWS Kriptografi Pembayaran mendukung format ISO 9564-1 0, 1, 3 dan 4. Format 4 diperlukan untuk kunci AES. Saat memverifikasi atau menerjemahkan PINs, ada kebutuhan untuk menentukan blok PIN dari data yang masuk atau keluar.

POI

Point of Interaction (POI), juga sering digunakan secara anonim dengan Point of Sale (POS), adalah perangkat keras yang berinteraksi dengan pemegang kartu untuk menunjukkan kredensi pembayaran mereka. Contoh POI adalah terminal fisik di lokasi pedagang. Untuk daftar terminal PCI PTS POI bersertifikat, lihat situs web PCI.

PSN

PAN Sequence Number (PSN) adalah nilai numerik yang digunakan untuk membedakan beberapa kartu yang dikeluarkan dengan PAN yang sama.

Kunci publik

Ketika menggunakan cipher asimetris (RSA, ECC), kunci publik adalah komponen publik dari public-private key pair. Kunci publik dapat dibagi dan didistribusikan ke entitas yang perlu mengenkripsi data untuk pemilik pasangan kunci publik-privat. Untuk operasi tanda tangan digital, pasangan kunci publik digunakan untuk memverifikasi tanda tangan.

Kunci privat

Ketika menggunakan cipher asimetris (RSA, ECC), kunci pribadi adalah komponen pribadi dari public-private key pair. Kunci privat digunakan untuk mendekripsi data atau membuat tanda tangan digital. Mirip dengan kunci Kriptografi AWS Pembayaran simetris, kunci pribadi dibuat dengan aman oleh. HSMs Mereka didekripsi hanya ke dalam memori volatile HSM dan hanya untuk waktu yang diperlukan untuk memproses permintaan kriptografi Anda.

PVV

Nilai verifikasi PIN (PVV) adalah jenis output kriptografi yang dapat digunakan untuk memverifikasi pin tanpa menyimpan pin yang sebenarnya. Meskipun merupakan istilah umum, dalam konteks Kriptografi AWS Pembayaran, PVV mengacu pada metode PVV Visa atau ABA. PVV ini adalah nomor empat digit yang inputnya adalah nomor kartu, nomor urut pan, pan itu sendiri dan kunci verifikasi PIN. Selama tahap validasi, Kriptografi AWS Pembayaran secara internal membuat ulang PVV menggunakan data transaksi dan membandingkannya lagi nilai yang telah disimpan oleh pelanggan Kriptografi Pembayaran. AWS Dengan cara ini, secara konseptual mirip dengan hash kriptografi atau MAC.

Bungkus/Buka RSA

RSA wrap menggunakan kunci asimetris untuk membungkus kunci simetris (seperti kunci TDES) untuk transmisi ke sistem lain. Hanya sistem dengan kunci pribadi yang cocok yang dapat mendekripsi muatan dan memuat kunci simetris. Sebaliknya, RSA membuka, akan mendekripsi kunci yang dienkripsi dengan aman menggunakan RSA dan kemudian memuat kunci ke dalam Kriptografi Pembayaran. AWS RSA wrap adalah metode pertukaran kunci tingkat rendah dan tidak mengirimkan kunci dalam format blok kunci dan tidak menggunakan penandatanganan payload oleh pihak pengirim. Kontrol alternatif harus dipertimbangkan untuk memastikan pemeliharaan dan atribut kunci tidak bermutasi.

TR-34 juga menggunakan RSA secara internal, tetapi merupakan format terpisah dan tidak dapat dioperasikan.

TR-31

TR-31 (secara formal didefinisikan sebagai ANSI X9 TR 31) adalah format blok kunci yang didefinisikan oleh American National Standards Institute (ANSI) untuk mendukung mendefinisikan atribut kunci dalam struktur data yang sama dengan data kunci itu sendiri. Format blok kunci TR-31 mendefinisikan satu set atribut kunci yang terikat ke kunci sehingga mereka disatukan. AWS Kriptografi Pembayaran menggunakan persyaratan standar TR-31 bila memungkinkan untuk memastikan pemisahan kunci yang tepat dan tujuan utama. TR-31 telah digantikan oleh ANSI X9.143-2022.

TR-34

TR-34 adalah implementasi ANSI X9.24-2 yang menggambarkan protokol untuk mendistribusikan kunci simetris dengan aman (seperti 3DES dan AES) menggunakan teknik asimetris (seperti RSA). AWS Kriptografi Pembayaran menggunakan metode TR-34 untuk mengizinkan impor dan ekspor kunci yang aman.

X9.143

X9.143 adalah format blok kunci yang didefinisikan oleh American National Standards Institute (ANSI) untuk mendukung pengamanan atribut kunci dan kunci dalam struktur data yang sama. Format blok kunci mendefinisikan satu set atribut kunci yang terikat ke kunci sehingga mereka disatukan. AWS Kriptografi Pembayaran menggunakan persyaratan standar X9.143 bila memungkinkan untuk memastikan pemisahan kunci yang tepat dan tujuan utama. X9.143 menggantikan proposal TR-31 sebelumnya meskipun dalam banyak kasus mereka kompatibel mundur dan maju dan istilah sering digunakan secara bergantian.

Layanan terkait

AWS Key Management Service

AWS Key Management Service (AWS KMS) adalah layanan terkelola yang memudahkan Anda membuat dan mengontrol kunci kriptografi yang digunakan untuk melindungi data Anda. AWS KMS menggunakan modul keamanan perangkat keras (HSMs) untuk melindungi dan memvalidasi kunci AWS KMS Anda.

AWS CloudHSM

AWS CloudHSM menyediakan instans HSM tujuan umum khusus kepada pelanggan di Cloud. AWS AWS CloudHSM dapat menyediakan berbagai fungsi kriptografi seperti membuat kunci, penandatanganan data atau mengenkripsi dan mendekripsi data.

Untuk informasi selengkapnya

- Untuk mempelajari tentang istilah dan konsep yang digunakan dalam Kriptografi AWS Pembayaran, lihat Konsep Kriptografi AWS Pembayaran.
- Untuk informasi tentang AWS Payment Cryptography Control Plane API, lihat Referensi API Pesawat Kontrol Kriptografi AWS Pembayaran.
- Untuk informasi tentang API Pesawat Data Kriptografi AWS Pembayaran, lihat Referensi API Pesawat Data Kriptografi AWS Pembayaran.
- Untuk informasi teknis terperinci tentang bagaimana Kriptografi AWS Pembayaran menggunakan kriptografi dan mengamankan kunci Kriptografi AWS Pembayaran, lihat detail Kriptografi.

Endpoint untuk AWS Payment Cryptography

Untuk terhubung secara terprogram ke AWS Payment Cryptography, Anda menggunakan titik akhir, URL titik masuk untuk layanan. Alat AWS SDKs dan baris perintah secara otomatis menggunakan titik akhir default untuk layanan AWS Region berdasarkan konteks wilayah permintaan, jadi biasanya tidak perlu secara eksplisit menetapkan nilai-nilai ini. Bila diperlukan, Anda dapat menentukan titik akhir yang berbeda untuk permintaan API Anda.

Titik akhir bidang kendali

Nama Wilayah	Wilayah	Titik Akhir	Protokol	
AS Timur (Ohio)	us–east-2	controlplane.payment-cryptography.us-east-2.amazonaws.com controlplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
AS Timur (Virginia Utara)	us-east-1	controlplane.payment-cryptography.us-east-1.amazonaws.com controlplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
US West (Oregon)	us-west-2	controlplane.payment-cryptography.us-west-2.amazonaws.com controlplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS	
Africa (Cape Town)	af-south-1	controlplane.payment-cryptography.af-south-1.amazonaws.com controlplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Asia Pasifik	ap-south-2	controlplane.payment-cryptography.ap-south-2.amazonaws.com	HTTPS	

Nama Wilayah	Wilayah	Titik Akhir	Protokol
(Hyderabad)		controlplane.payment-cryptography.ap-south-2.api.aws	HTTPS
Asia Pasifik (Mumbai)	ap-south-1	controlplane.payment-cryptography.ap-south-1.amazonaws.com controlplane.payment-cryptography.ap-south-1.api.aws	HTTPS HTTPS
Asia Pasifik (Osaka)	ap-northeast-3	controlplane.payment-cryptography.ap-northeast-3.amazonaws.com controlplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Asia Pasifik (Singapura)	ap-southeast-1	controlplane.payment-cryptography.ap-southeast-1.amazonaws.com controlplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Asia Pacific (Sydney)	ap-southeast-2	controlplane.payment-cryptography.ap-southeast-2.amazonaws.com controlplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Asia Pacific (Tokyo)	ap-northeast-1	controlplane.payment-cryptography.ap-northeast-1.amazonaws.com controlplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
Canada (Central)	ca-central-1	controlplane.payment-cryptography.ca-central-1.amazonaws.com controlplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS
Eropa (Frankfurt)	eu-central-1	controlplane.payment-cryptography.eu-central-1.amazonaws.com controlplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS
Eropa (Irlandia)	eu-west-1	controlplane.payment-cryptography.eu-west-1.amazonaws.com controlplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS
Europe (London)	eu-west-2	controlplane.payment-cryptography.eu-west-2.amazonaws.com controlplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS
Eropa (Paris)	eu-west-3	controlplane.payment-cryptography.eu-west-3.amazonaws.com controlplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS

Titik akhir bidang data

Nama Wilayah	Wilayah	Titik Akhir	Protokol
AS Timur (Ohio)	us–east-2	dataplane.payment-cryptography.us-east-2.amazonaws.com dataplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS
AS Timur (Virginia Utara)	us-east-1	dataplane.payment-cryptography.us-east-1.amazonaws.com dataplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS
US West (Oregon)	us-west-2	dataplane.payment-cryptography.us-west-2.amazonaws.com dataplane.payment-cryptography.us-west-2.api.aws	HTTPS HTTPS
Africa (Cape Town)	af-south-1	dataplane.payment-cryptography.af-south-1.amazonaws.com dataplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS
Asia Pasifik (Hyderabad)	ap-south-2	dataplane.payment-cryptography.ap-south-2.amazonaws.com dataplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS
Asia Pasifik (Mumbai)	ap-south-1	dataplane.payment-cryptography.ap-south-1.amazonaws.com	HTTPS HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
		dataplane.payment-cryptography.ap-south-1.api.aws	
Asia Pasifik (Osaka)	ap-northeast-3	dataplane.payment-cryptography.ap-northeast-3.amazonaws.com dataplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS
Asia Pasifik (Singapura)	ap-southeast-1	dataplane.payment-cryptography.ap-southeast-1.amazonaws.com dataplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS
Asia Pacific (Sydney)	ap-southeast-2	dataplane.payment-cryptography.ap-southeast-2.amazonaws.com dataplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS
Asia Pacific (Tokyo)	ap-northeast-1	dataplane.payment-cryptography.ap-northeast-1.amazonaws.com dataplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS
Canada (Central)	ca-central-1	dataplane.payment-cryptography.ca-central-1.amazonaws.com dataplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS

Nama Wilayah	Wilayah	Titik Akhir	Protokol
Eropa (Frankfurt)	eu-central-1	dataplane.payment-cryptography.eu-central-1.amazonaws.com dataplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS
Eropa (Irlandia)	eu-west-1	dataplane.payment-cryptography.eu-west-1.amazonaws.com dataplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS
Europe (London)	eu-west-2	dataplane.payment-cryptography.eu-west-2.amazonaws.com dataplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS
Eropa (Paris)	eu-west-3	dataplane.payment-cryptography.eu-west-3.amazonaws.com dataplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS

Memulai Kriptografi AWS Pembayaran

Untuk memulai dengan Kriptografi AWS Pembayaran, pertama-tama Anda ingin membuat kunci dan kemudian menggunakannya dalam berbagai operasi kriptografi. Tutorial di bawah ini menyediakan kasus penggunaan sederhana untuk menghasilkan kunci yang akan digunakan untuk generating/verifying CVV2 nilai. Untuk mencoba contoh lain dan menjelajahi pola penerapan dalam AWS, silakan coba Workshop Kriptografi AWS Pembayaran berikut atau jelajahi proyek sampel kami yang tersedia di GitHub

Tutorial ini memandu Anda melalui pembuatan satu kunci dan melakukan operasi kriptografi menggunakan kunci. Setelah itu, Anda menghapus kunci jika Anda tidak lagi menginginkannya, yang melengkapi siklus hidup kunci.

> ⚠ Warning
>
> Contoh di seluruh panduan pengguna ini dapat menggunakan nilai sampel. Kami sangat menyarankan untuk tidak menggunakan nilai sampel dalam lingkungan produksi seperti nomor seri kunci.

Topik

Prasyarat

Sebelum Anda mulai, pastikan bahwa:

- Anda memiliki izin untuk mengakses layanan. Untuk informasi selengkapnya, lihat kebijakan IAM.

- Anda telah AWS CLImenginstal. Anda juga dapat menggunakan AWS SDKsatau AWS APIsmengakses Kriptografi AWS Pembayaran, tetapi instruksi dalam tutorial ini menggunakan. AWS CLI

Langkah 1: Buat kunci

Langkah pertama adalah membuat kunci. Untuk tutorial ini, Anda membuat kunci 3DES (2KEY TDES) panjang ganda CVK untuk menghasilkan dan memverifikasi nilai CVV/. CVV2

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (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"
    }
}
```

Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi. Anda membutuhkannya di langkah berikutnya.

Langkah 2: Hasilkan CVV2 nilai menggunakan kunci

Pada langkah ini, Anda menghasilkan CVV2 untuk tanggal tertentu PAN dan kedaluwarsa menggunakan kunci dari langkah 1.

```
$  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"
}
```

Perhatikan`cardDataValue`, dalam hal ini angka 3 digit 144. Anda membutuhkannya di langkah berikutnya.

Langkah 3: Verifikasi nilai yang dihasilkan pada langkah 2

Dalam contoh ini, Anda memvalidasi CVV2 dari langkah 2 menggunakan kunci yang Anda buat di langkah 1.

Jalankan perintah berikut untuk memvalidasi. CVV2

```
$  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"
}
```

Layanan mengembalikan respon HTTP 200 untuk menunjukkan bahwa itu memvalidasi. CVV2

Langkah 4: Lakukan tes negatif

Pada langkah ini, Anda membuat tes negatif di mana tidak CVV2 benar dan tidak memvalidasi. Anda mencoba untuk memvalidasi yang salah CVV2 menggunakan kunci yang Anda buat di langkah 1. Ini adalah operasi yang diharapkan misalnya jika pemegang kartu salah memasukkan CVV2 saat checkout.

```
$  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.
```

Layanan mengembalikan respons HTTP 400 dengan pesan “Verifikasi data validasi kartu gagal” dan alasan INVALID_VALIDATION_DATA.

Langkah 5: (Opsional) Bersihkan

Sekarang Anda dapat menghapus kunci yang Anda buat di langkah 1. Untuk meminimalkan perubahan yang tidak dapat dipulihkan, periode penghapusan kunci default adalah tujuh hari.

```
$ 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_C0_CARD_VERIFICATION_KEY"
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "DELETE_PENDING",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```


Catat dua bidang dalam output. `deletePendingTimestamp`Ini diatur ke tujuh hari di masa depan secara default. KeyState diatur ke`DELETE_PENDING`. Anda dapat membatalkan penghapusan ini kapan saja sebelum waktu penghapusan yang dijadwalkan dengan menelepon. `restore-key`

Mengelola kunci

Untuk memulai dengan Kriptografi AWS Pembayaran, buat kunci Kriptografi AWS Pembayaran.

Bagian ini menjelaskan cara membuat dan mengelola berbagai jenis kunci Kriptografi AWS Pembayaran sepanjang siklus hidupnya. Anda akan belajar cara membuat, melihat, dan mengedit kunci, serta cara menandai kunci, membuat alias kunci, dan mengaktifkan atau menonaktifkan kunci.

Kunci Kriptografi AWS Pembayaran adalah sumber daya regional. Jika Anda bermaksud menggunakan kunci tertentu dalam beberapa Wilayah AWS, Anda dapat mengaktifkan replikasi kunci Multi-Region yang menyalin materi kunci dan metadata dengan aman ke dalam Partisi dan Wilayah AWS Akun yang sama. AWS Kunci sumber dalam replikasi kunci Multi-Region dikenal sebagai kunci Wilayah Primer (PRK) dan ini tetap menjadi sumber otoritatif untuk semua kegiatan manajemen kunci. Kunci yang direplikasi dikenal sebagai Replica Region key (RRK) dan ini adalah replika read-only PRK. Anda harus mempertimbangkan untuk menggunakan kunci Multi-Wilayah dengan kunci Anda untuk memenuhi tujuan desain seputar ketersediaan, pemulihan bencana, dan latensi rendah.

Topik

Membuat kunci

Anda dapat membuat kunci Kriptografi AWS Pembayaran menggunakan operasi CreateKey API. Saat Anda membuat kunci, Anda menentukan atribut seperti algoritme kunci, penggunaan kunci,

operasi yang diizinkan, dan apakah itu dapat diekspor. Anda tidak dapat mengubah properti ini setelah Anda membuat kunci Kriptografi AWS Pembayaran.

> **Note**
>
> Jika replikasi kunci Multi-Region diaktifkan untuk Anda Akun AWS dan Anda membuat kunci Kriptografi Pembayaran, kunci ini akan secara otomatis menjadi kunci Wilayah Utama (PRK). PRK direplikasi bahkan jika Anda tidak menentukan `--replication-regions` parameter dalam perintah. CreateKey Untuk informasi selengkapnya, lihat Cara kerja replikasi kunci Multi-Region.

Contoh

- Membuat kunci derivasi dasar 3KEY TDES
- Membuat kunci TDES 2KEY untuk CVV/ CVV2
- Membuat kunci HMAC
- Membuat kunci AES-256
- Membuat Kunci Enkripsi PIN (PEK)
- Membuat kunci asimetris (RSA)
- Membuat Kunci Nilai Verifikasi PIN (PVV)
- Membuat kunci ECC asimetris

Membuat kunci derivasi dasar 3KEY TDES

Example

Perintah ini menciptakan kunci derivasi 3KEY TDES yang akan direplikasi ke wilayah AS Timur (Ohio) dan AS Barat (Oregon). Respons tersebut mencakup parameter reques, Amazon Resource Name (ARN) untuk panggilan berikutnya, dan Key Check Value (KCV).

```
$ aws payment-cryptography create-key --exportable --key-attributes \
     "KeyUsage=TR31_B0_BASE_DERIVATION_KEY, \
     KeyClass=SYMMETRIC_KEY,KeyAlgorithm=TDES_3KEY, \
     KeyModesOfUse={NoRestrictions=true}" \
     --replication-regions us-east-2 --region us-west-2
```

Contoh output:


```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "FE23D3",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_3KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": true,
                "Wrap": false
            },
            "KeyUsage": "TR31_B0_BASE_DERIVATION_KEY"
        },
        "KeyCheckValue": "FE23D3",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-26T16:04:11.559000-07:00"
}
```

Membuat kunci TDES 2KEY untuk CVV/ CVV2

Example

Perintah ini membuat kunci TDES 2KEY untuk menghasilkan dan memverifikasi nilai CVV2 CVV/. Respons tersebut mencakup parameter permintaan, Nama Sumber Daya Amazon (ARN) untuk panggilan berikutnya, dan Nilai Pemeriksaan Kunci (KCV).

```
$ 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}'
```

Contoh output:

```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
        "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": "AEA5CD",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-26T16:04:11.559000-07:00"
    }
}
```

Membuat kunci HMAC

Example

Kunci HMAC digunakan untuk menghasilkan atau memverifikasi kode otentikasi pesan hash (HMAC). Dengan kunci HMAC, jenis hash ditetapkan pada saat pembuatan kunci (seperti HMAC_ dan HMAC_) SHA224 dan tidak dapat dimodifikasiSHA512.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=HMAC_SHA512,KeyUsage=TR31_M7_HMAC_KEY,KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Gene
 = true,Verify = true}'
```

Contoh output:

```
{
 "Key": {
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/qnobl5lghrzunce6",
 "KeyAttributes": {
 "KeyUsage": "TR31_M7_HMAC_KEY",
 "KeyClass": "SYMMETRIC_KEY",
 "KeyAlgorithm": "HMAC_SHA512",
 "KeyModesOfUse": {
 "Encrypt": false,
 "Decrypt": false,
 "Wrap": false,
 "Unwrap": false,
 "Generate": true,
 "Sign": false,
 "Verify": true,
 "DeriveKey": false,
 "NoRestrictions": false
 }
 },
 "KeyCheckValue": "2976E7",
 "KeyCheckValueAlgorithm": "HMAC",
 "Enabled": true,
 "Exportable": true,
 "KeyState": "CREATE_COMPLETE",
 "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
 "CreateTimestamp": "2025-07-30T10:06:12.142000-07:00",
 "UsageStartTimestamp": "2025-07-30T10:06:12.128000-07:00"
 }
}
```

Membuat kunci AES-256

Example

Perintah ini membuat kunci simetris AES-256 untuk enkripsi dan dekripsi data. Kunci AES menyediakan enkripsi yang kuat untuk data sensitif dan biasanya digunakan dalam pemrosesan pembayaran untuk mengenkripsi data pemegang kartu dan informasi sensitif lainnya, namun TDES lebih umum digunakan untuk kasus penggunaan penerbit seperti EMV.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=AES_256,KeyUsage=TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY,KeyClass=SYMMETRIC_KEY,Key
```

Contoh output:

```
{
    "Key": {
        "CreateTimestamp": "2025-02-02T10:15:30.142000-08:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyAlgorithm": "AES_256",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": true,
                "DeriveKey": false,
                "Encrypt": true,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": true,
                "Verify": false,
                "Wrap": true
            },
            "KeyUsage": "TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY"
        },
        "KeyCheckValue": "2976F5",
        "KeyCheckValueAlgorithm": "CMAC",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2025-02-02T10:15:30.128000-08:00"
    }
}
```

Membuat Kunci Enkripsi PIN (PEK)

Example

Perintah ini membuat kunci TDES 3KEY untuk mengenkripsi nilai PIN meskipun tombol pin juga dapat berupa AES tergantung pada kebutuhan Anda akan interoperabilitas. Anda dapat menggunakan kunci ini untuk menyimpan PINs atau mendekripsi dengan aman PINs selama verifikasi, seperti dalam transaksi. Respons termasuk parameter permintaan, ARN untuk panggilan berikutnya, dan KCV.

```
$ 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}'
```

Contoh output:

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T08:27:51.795000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
        "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": "7CC9E2",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T08:27:51.753000-07:00"
    }
}
```

Membuat kunci asimetris (RSA)

Example

Perintah ini menghasilkan key pair asimetris RSA 2048-bit baru. Ini menciptakan kunci pribadi baru dan kunci publik yang cocok. Anda dapat mengambil kunci publik menggunakan getPublicCertificateAPI.

```
$ 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}'
```

Contoh output:

```
{
    "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": "SHA-1",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-11-15T11:15:42.182000-08:00"
    }
}
```

Membuat Kunci Nilai Verifikasi PIN (PVV)

Example

Perintah ini menciptakan kunci TDES 3KEY untuk menghasilkan nilai PVV. Anda dapat menggunakan kunci ini untuk menghasilkan PVV yang dapat dibandingkan dengan PVV yang dihitung selanjutnya. Respons termasuk parameter permintaan, ARN untuk panggilan berikutnya, dan KCV.

```
$ 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}'
```

Contoh output:

```
{
    "Key": {
        "CreateTimestamp": "2022-10-27T10:22:59.668000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
        "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": "7F2363",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2022-10-27T10:22:59.614000-07:00"
    }
}
```

Membuat kunci ECC asimetris

Example

Perintah ini menghasilkan key pair ECC untuk membuat perjanjian kunci ECDH (Elliptic Curve Diffie-Hellman) antara dua pihak. Dengan ECDH, masing-masing pihak menghasilkan key pair ECC sendiri dengan tujuan utama K3 dan mode penggunaan X, dan mereka bertukar kunci publik. Kedua belah pihak kemudian menggunakan kunci pribadi mereka dan kunci publik yang diterima untuk membuat kunci turunan bersama.

Untuk mempertahankan prinsip penggunaan tunggal kunci kriptografi dalam pembayaran, kami merekomendasikan untuk tidak menggunakan kembali pasangan kunci ECC untuk berbagai tujuan, seperti derivasi dan penandatanganan kunci ECDH.

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT, \
    KeyClass=ASYMMETRIC_KEY_PAIR,KeyModesOfUse='{DeriveKey=true}'
```

Contoh output

```
{
    "Key": {
        "CreateTimestamp": "2024-10-17T01:31:55.908000+00:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
wc3rjsssguhxtilv",
        "KeyAttributes": {
            "KeyAlgorithm": "ECC_NIST_P256",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": false,
                "Wrap": false
            },
            "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT"
        },
        "KeyCheckValue": "7E34F19F",
        "KeyCheckValueAlgorithm": "SHA-1",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "UsageStartTimestamp": "2024-10-17T01:31:55.866000+00:00"
    }
}
```

Kunci daftar

Gunakan ListKeys operasi untuk mendapatkan daftar kunci yang dapat diakses oleh Anda di akun dan Wilayah Anda.

Example

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

Contoh output:

```
{
    "Keys": [
        {
            "CreateTimestamp": "2022-10-12T10:58:28.920000-07:00",
            "Enabled": false,
            "Exportable": true,
            "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/37y2tsl45p5zjbh2",
            "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": "7F2363",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "KeyState": "CREATE_COMPLETE",
            "UsageStopTimestamp": "2022-10-27T14:19:42.488000-07:00"
        }
    ]
}
```

Mengaktifkan dan menonaktifkan kunci

Anda dapat menonaktifkan dan mengaktifkan kembali kunci Kriptografi AWS Pembayaran. Saat Anda membuat kunci, itu diaktifkan secara default. Jika Anda menonaktifkan kunci, itu tidak dapat digunakan dalam operasi kriptografi apa pun sampai Anda mengaktifkannya kembali. Start/stop Perintah penggunaan segera berlaku, jadi disarankan agar Anda meninjau penggunaan sebelum membuat perubahan seperti itu. Anda juga dapat mengatur perubahan (mulai atau menghentikan penggunaan) agar berlaku di masa mendatang menggunakan `timestamp` parameter opsional.

Karena bersifat sementara dan mudah dibatalkan, menonaktifkan kunci Kriptografi AWS Pembayaran adalah alternatif yang lebih aman untuk menghapus kunci Kriptografi AWS Pembayaran, tindakan yang merusak dan tidak dapat diubah. Jika Anda mempertimbangkan untuk menghapus kunci Kriptografi AWS Pembayaran, nonaktifkan terlebih dahulu dan pastikan bahwa Anda tidak perlu menggunakan kunci untuk mengenkripsi atau mendekripsi data di masa mendatang.

Topik

- Mulai penggunaan kunci
- Hentikan penggunaan kunci

Mulai penggunaan kunci

Penggunaan kunci harus diaktifkan untuk menggunakan kunci untuk operasi kriptografi. Jika kunci tidak diaktifkan, Anda dapat menggunakan operasi ini untuk membuatnya dapat digunakan. Bidang `UsageStartTimestamp` akan mewakili ketika kunci became/will menjadi aktif. Ini akan menjadi masa lalu untuk token yang diaktifkan, dan di masa depan jika tertunda aktivasi.

Example

Dalam contoh ini, kunci diminta untuk diaktifkan untuk penggunaan kunci. Respons mencakup informasi kunci dan flag enable telah dialihkan ke true. Ini juga akan tercermin dalam objek respons list-keys.

```
$ 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"
  }
}
```

Hentikan penggunaan kunci

Jika Anda tidak lagi berencana untuk menggunakan kunci, Anda dapat menghentikan penggunaan kunci untuk mencegah operasi kriptografi lebih lanjut. Operasi ini tidak permanen, sehingga Anda dapat membalikkannya menggunakan penggunaan kunci awal. Anda juga dapat mengatur kunci untuk dinonaktifkan di masa depan. Bidang `UsageStopTimestamp` akan mewakili ketika kunci became/will menjadi dinonaktifkan.

Example

Dalam contoh ini, diminta untuk menghentikan penggunaan kunci di masa mendatang. Setelah eksekusi, kunci ini tidak dapat digunakan untuk operasi kriptografi kecuali diaktifkan kembali melalui penggunaan kunci awal Respons mencakup informasi kunci dan tanda aktifkan telah dialihkan ke false. Ini juga akan tercermin dalam objek respons list-keys.

```
$ 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"
  }
}
```

Replikasi Kriptografi AWS Pembayaran Kriptografi

AWS Kriptografi Pembayaran mendukung replikasi kunci Multi-Wilayah, memungkinkan Anda untuk mendistribusikan materi dan metadata kunci dengan aman dari Kunci Kriptografi AWS Pembayaran yang diberikan ke satu atau lebih Wilayah AWS dalam partisi dan akun yang sama. AWS

Kunci sumber dikenal sebagai kunci Wilayah Primer (PRK) dan tetap menjadi sumber otoritatif untuk semua aktivitas manajemen kunci sementara kunci PRK dan Replica Region (RRK) dapat digunakan untuk operasi kriptografi masing-masing. Wilayah AWS

Manfaat replikasi kunci Multi-Region

Berikut ini menguraikan beberapa manfaat replikasi kunci Multi-Region.

- Pengaturan yang lebih mudah untuk aplikasi yang sangat tersedia - Kriptografi AWS Pembayaran menangani distribusi kunci untuk Anda sehingga Anda dapat menggunakan kunci dalam beberapa Wilayah AWS tanpa perlu membuat salinan terpisah dari kunci yang diberikan.
- Ketersediaan tinggi dan kunci latensi rendah - Dengan replikasi kunci Multi-Region, Anda dapat mengakses kunci Anda dalam beberapa Wilayah AWS sehingga sangat tersedia, menghasilkan latensi yang lebih rendah.
- Daya tahan material utama - Kunci Wilayah Replika adalah replika kunci lengkap dan dapat digunakan secara independen dari kunci Wilayah Primer mereka dalam operasi kriptografi. RRK menyediakan replika yang tahan lama jika terjadi bencana kehilangan data PRK.

Cara kerja replikasi kunci Multi-Region

Ketika replikasi kunci Multi-Region diaktifkan, layanan Kriptografi AWS Pembayaran menggunakan mekanisme distribusi kunci yang aman untuk menyalin materi kunci dan metadata ke replika yang Anda tentukan. Wilayah AWS Perubahan pada metadata kunci Wilayah Utama, seperti atribut kunci, status, dan pemberdayaan, secara otomatis direplikasi ke kunci Wilayah Replika.

Pertimbangan dan batasan

Berikut ini adalah beberapa batasan dan pertimbangan replikasi kunci Multi-Region.

- Anda harus mengaktifkan fitur ini untuk kunci Kriptografi Pembayaran AWS Region atau tertentu.
  - Jika fitur ini diaktifkan untuk AWS Region, semua kunci Kriptografi AWS Pembayaran yang dibuat setelah pengaktifan akan direplikasi ke yang ditentukan. AWS Region Kunci yang dibuat

di Wilayah ini akan menjadi kunci Wilayah Utama. Kunci yang ada di Wilayah ini tidak akan direplikasi secara otomatis. Anda dapat mengaktifkan replikasi kunci Multi-Region untuk kunci yang ada AWS Region di dalam level kunci.

  - Masing-masing AWS Region dapat memiliki pengaturan replikasi kunci Multi-Region yang unik.
  - Pengaturan replikasi Multi-Region kunci lebih diutamakan daripada pengaturan replikasi kunci AWS Region Multi-Region.
- Kunci Replica Region tidak dapat dikonfigurasi untuk mereplikasi ke yang lain. Wilayah AWS
- Replikasi kunci Multi-Region tersedia untuk kunci Kriptografi Pembayaran simetris seperti Triple Data Encryption Standard (3DES), Advanced Encryption Standard (AES), dan Hash Based Message Authentication Code (HMAC).
- Kunci Kriptografi Pembayaran Asimetris tidak mendukung replikasi kunci Multi-Region.
- Kunci Replica Region adalah kunci read-only. Semua perubahan pada kunci Wilayah Utama akan diterapkan ke kunci Wilayah Replika.
- Perubahan kunci Wilayah Primer pada akhirnya konsisten dengan kunci Wilayah Replika.
- Kunci Kriptografi Pembayaran hanya dapat direplikasi dengan AWS partisi dan akun yang sama.
- Jumlah kunci Wilayah Replika terhadap batas Kriptografi AWS Pembayaran Akun AWS level Anda.
- Kunci Wilayah Utama dan kunci Wilayah Replika menggunakan pengenal kunci yang sama yang memungkinkan Anda mereferensikan kedua kunci dengan ARN yang sama dalam kebijakan IAM.

Mengaktifkan replikasi kunci Multi-Region

Ada dua cara Anda dapat mengaktifkan replikasi kunci Multi-Region untuk kunci Kriptografi AWS Pembayaran.

1. AWS Region: Replikasi kunci Multi-Region diterapkan ke semua kunci baru yang dibuat AWS Region saat diaktifkan. Metode ini memberikan replikasi yang konsisten untuk semua kunci.
2. Kunci Kriptografi AWS Pembayaran Khusus: Anda dapat mengelola replikasi kunci Multi-Wilayah untuk kunci individual yang memungkinkan tingkat kontrol yang lebih terperinci.

Setelah replikasi kunci Multi-Region diaktifkan, kunci Kriptografi Pembayaran Anda akan mereplikasi ke yang Anda tentukan. Wilayah AWS

> **Important**
>
> Replikasi kunci Multi-Region tidak dapat dijeda. Kunci Anda secara otomatis direplikasi ke yang Wilayah AWS Anda tentukan setelah replikasi diaktifkan. Replikasi kunci Multi-Region dapat dinonaktifkan untuk kunci tertentu AWS Region atau Kriptografi Pembayaran. Anda harus menghapus AWS Region sebagai wilayah replikasi dari kunci Wilayah Utama untuk menghapus kunci Wilayah Replika.
> Atau, Anda dapat memanggil perintah `StopKeyUsage`API atau stop-key-usageCLI di PRK Anda untuk menghentikan penggunaan PRK dan semua yang terkait. RRKs Anda tidak akan dapat menggunakan kunci ini dalam operasi kriptografi. Menggunakan perintah `StopKeyUsage` API atau stop-key-usage CLI tidak akan menghentikan replikasi kunci Multi-Region yang sedang berlangsung yang diaktifkan untuk PRK Anda.

Anda dapat memeriksa pengaturan replikasi kunci Multi-Region untuk kunci Kriptografi AWS Pembayaran secara spesifik AWS Region dengan memanggil perintah API `GetDefaultKeyReplicationRegions` atau CLIget-default-key-replication-regions. Kunci di AWS Region tempat Anda memanggil tindakan atau perintah API ini akan menjadi PRK Anda.

Gunakan prosedur berikut untuk mengaktifkan replikasi kunci Multi-Region.

For AWS Region

- Gunakan perintah berikut untuk mengaktifkan replikasi kunci Multi-Region untuk yang AWS Region Anda tentukan. Dalam contoh ini, replikasi kunci Multi-Region diaktifkan di US East (Ohio) dan US West (Oregon). Untuk menggunakan perintah ini, ganti perintah *`italicized placeholder text`* in the example dengan informasi Anda sendiri.

```
aws payment-cryptography enable-default-key-replication-regions \
    --replication-regions us-east-2 us-west-2
```

> **Note**
>
> Mengaktifkan replikasi kunci Multi-Region untuk sebuah tidak AWS Region akan mengubah konfigurasi replikasi kunci Kriptografi Pembayaran yang ada AWS .
> Anda dapat mengaktifkan fitur ini untuk kunci yang ada di tingkat kunci. Hanya kunci

yang dibuat setelah replikasi kunci Multi-Region diaktifkan untuk AWS Region akan menggunakan pengaturan replikasi wilayah.

For specific AWS Payment Cryptography keys

- Gunakan perintah berikut untuk mengaktifkan replikasi kunci Multi-Region untuk kunci Kriptografi Pembayaran tertentu. Dalam contoh ini, replikasi kunci Multi-Region diaktifkan di US East (Ohio). Untuk menggunakan perintah ini, ganti perintah *italicized placeholder text* in the example dengan informasi Anda sendiri.

```
aws payment-cryptography add-key-replication-regions \
    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --replication-regions us-east-2
```

Atau, Anda dapat membuat kunci Kriptografi Pembayaran baru dengan fitur ini diaktifkan dengan memasukkan replikasi Wilayah AWS dalam permintaan kunci buat Anda.

> **Note**
>
> Pengaturan replikasi kunci lebih diutamakan daripada pengaturan replikasi. AWS Region

Menonaktifkan replikasi kunci Multi-Region

Jika Anda ingin menonaktifkan replikasi kunci Multi-Region, Anda dapat memanggil perintah atau disable-default-key-replication remove-key-replication-regions CLI, tergantung pada bagaimana replikasi kunci Multi-Region diaktifkan. Anda harus menentukan ARN kunci dan untuk menonaktifkan replikasi kunci Multi-Region. AWS Region

Pertimbangan-pertimbangan

Penghapusan kunci wilayah replikasi akhirnya konsisten.

Anda dapat memeriksa pengaturan replikasi kunci Multi-Region untuk kunci Kriptografi AWS Pembayaran secara spesifik AWS Region dengan memanggil perintah API `GetDefaultKeyReplicationRegions` atau CLIget-default-key-replication-regions.

Gunakan prosedur berikut untuk menonaktifkan replikasi kunci Multi-Region.

For AWS Region

- Gunakan perintah berikut untuk menonaktifkan replikasi kunci Multi-Region untuk yang AWS Region Anda tentukan. Dalam contoh ini, replikasi kunci Multi-Region dinonaktifkan di US East (Ohio). Untuk menggunakan perintah ini, ganti perintah *italicized placeholder text* in the example dengan informasi Anda sendiri.

```
aws payment-cryptography disable-default-key-replication-regions \
    --replication-regions us-east-2
```

For specific AWS Payment Cryptography keys

- Gunakan perintah berikut untuk menonaktifkan replikasi kunci Multi-Region untuk kunci Kriptografi Pembayaran tertentu. Dalam contoh ini, replikasi kunci Multi-Region dinonaktifkan di US East (Ohio). Untuk menggunakan perintah ini, ganti perintah *italicized placeholder text* in the example dengan informasi Anda sendiri.

```
aws payment-cryptography remove-key-replication-regions \
    --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --replication-regions us-east-2
```

Pertimbangan keamanan

Berikut ini adalah pertimbangan keamanan saat menggunakan replikasi kunci Multi-Region untuk kunci Kriptografi Pembayaran Anda. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan untuk Kriptografi AWS Pembayaran.

- Batasi berbagi materi utama.
- Ikuti prinsip izin hak istimewa paling sedikit saat membuat kebijakan IAM.
- Anda tidak dapat membuat perubahan pada kunci Replica Region karena ini adalah kunci hanya-baca.

Praktik terbaik

Berikut ini adalah beberapa praktik terbaik saat menggunakan replikasi kunci Multi-Region dengan kunci Kriptografi AWS Pembayaran.

- Pastikan aplikasi Anda terus berfungsi meskipun replikasi kunci Multi-Region ke yang AWS Region ditentukan tidak langsung. Jika Anda perlu tahu kapan replikasi kunci Multi-Region selesai, Anda dapat memantau dengan tindakan GetKeyAPI. Anda dapat memantau peristiwa replikasi kunci dengan AWS CloudTrail.
- Uji dan terapkan proses penerapan otomatis jika terjadi kegagalan dari satu Wilayah ke Wilayah lain AWS Region .

Harga

Anda dikenakan biaya untuk kunci Wilayah Replika yang Anda buat dengan Kriptografi AWS Pembayaran. Kunci-kunci ini dikenakan biaya per AWS Region. Untuk informasi harga Kriptografi Pembayaran terbaru, lihat halaman harga Kriptografi AWS Pembayaran.

Menghapus kunci

Menghapus kunci Kriptografi AWS Pembayaran menghapus materi kunci dan semua metadata yang terkait dengan kunci dan tidak dapat diubah kecuali salinan kunci tersedia di luar Kriptografi Pembayaran. AWS Setelah kunci dihapus, Anda tidak dapat lagi mendekripsi data yang dienkripsi di bawah kunci itu, yang berarti bahwa data mungkin menjadi tidak dapat dipulihkan. Anda harus menghapus kunci hanya ketika Anda yakin bahwa Anda tidak perlu menggunakannya lagi dan tidak ada pihak lain yang menggunakan kunci ini. Jika Anda tidak yakin, pertimbangkan untuk menghentikan penggunaan kunci alih-alih menghapusnya. Anda dapat mengaktifkan kembali kunci yang dinonaktifkan jika Anda perlu menggunakannya lagi nanti, tetapi Anda tidak dapat memulihkan kunci Kriptografi AWS Pembayaran yang dihapus kecuali Anda dapat mengimpornya kembali dari sumber lain.

Sebelum menghapus kunci, Anda harus memastikan bahwa Anda tidak lagi membutuhkan kunci. AWS Kriptografi Pembayaran tidak menyimpan hasil operasi kriptografi seperti CVV2 dan tidak dapat menentukan apakah kunci diperlukan untuk materi kriptografi yang persisten.

AWS Kriptografi Pembayaran tidak pernah menghapus kunci milik AWS akun aktif kecuali Anda secara eksplisit menjadwalkannya untuk dihapus dan masa tunggu wajib berakhir.

Namun, Anda dapat memilih untuk menghapus kunci Kriptografi AWS Pembayaran karena satu atau beberapa alasan berikut:

- Untuk menyelesaikan siklus hidup kunci untuk kunci yang tidak lagi Anda perlukan
- Untuk menghindari overhead manajemen yang terkait dengan pemeliharaan kunci Kriptografi AWS Pembayaran yang tidak digunakan

> **Note**
>
> Jika Anda menutup atau menghapus Akun AWS, kunci Kriptografi AWS Pembayaran Anda menjadi tidak dapat diakses. Anda tidak perlu menjadwalkan penghapusan kunci Kriptografi AWS Pembayaran Anda terpisah dari penutupan akun.

AWS Kriptografi Pembayaran mencatat entri di AWS CloudTraillog Anda ketika Anda menjadwalkan penghapusan kunci Kriptografi AWS Pembayaran dan ketika kunci Kriptografi AWS Pembayaran benar-benar dihapus.

Saat menggunakan replikasi kunci Multi-Region, menghapus kunci Kriptografi Pembayaran yang merupakan kunci Wilayah Utama (PRK), kunci Wilayah Replika (RRK) juga akan dihapus secara otomatis. RRK tidak dapat dihapus seperti PRK. Jika Anda ingin menghapus RRK, Anda harus memodifikasi wilayah replikasi untuk PRK Anda.

Tentang masa tunggu

Karena menghapus kunci tidak dapat diubah, Kriptografi AWS Pembayaran mengharuskan Anda untuk menetapkan masa tunggu antara 3-180 hari. Masa tunggu default adalah tujuh hari.

Namun, masa tunggu sebenarnya mungkin hingga 24 jam lebih lama dari yang Anda jadwalkan. Untuk mendapatkan tanggal dan waktu aktual ketika kunci Kriptografi AWS Pembayaran akan dihapus, gunakan GetKey operasi. Pastikan untuk mencatat zona waktu.

Selama masa tunggu, status kunci Kriptografi AWS Pembayaran dan status kunci adalah Penghapusan tertunda.

> **Note**
>
> Kunci Kriptografi AWS Pembayaran yang tertunda penghapusan tidak dapat digunakan dalam operasi kriptografi apa pun.

Setelah masa tunggu berakhir, Kriptografi AWS Pembayaran menghapus kunci Kriptografi AWS Pembayaran, aliasnya, dan semua metadata Kriptografi Pembayaran terkait AWS .

Gunakan masa tunggu untuk memastikan bahwa Anda tidak memerlukan kunci Kriptografi AWS Pembayaran sekarang atau di masa depan. Jika Anda menemukan bahwa Anda membutuhkan kunci selama masa tunggu, Anda dapat membatalkan penghapusan kunci sebelum masa tunggu berakhir. Setelah masa tunggu berakhir, Anda tidak dapat membatalkan penghapusan kunci, dan layanan menghapus kunci.

Example

Dalam contoh ini, kunci diminta untuk dihapus. Selain informasi kunci dasar, dua bidang yang relevan adalah bahwa status kunci telah diubah menjadi DELETE_PENDING dan deletePendingTimestamp mewakili kapan kunci saat ini dijadwalkan untuk dihapus.

```
$ 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": "0A3674",
        "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

Dalam contoh ini, penghapusan yang tertunda dibatalkan. Setelah berhasil diselesaikan, kunci tidak akan lagi dihapus sesuai jadwal sebelumnya. Respons berisi informasi kunci dasar; selain itu, dua bidang yang relevan telah berubah - `KeyState` dan`deletePendingTimestamp`. `KeyState`dikembalikan ke nilai CREATE_COMPLETE, sementara `DeletePendingTimestamp` dihapus.

```
$ 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": "0A3674",
        "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"
    }
}
```

Mengimpor dan mengekspor kunci

Anda dapat mengimpor kunci Kriptografi AWS Pembayaran dari solusi lain dan mengekspornya ke solusi lain, seperti HSMs. Banyak pelanggan bertukar kunci dengan penyedia layanan menggunakan fungsionalitas impor dan ekspor. Kami merancang Kriptografi AWS Pembayaran untuk menggunakan pendekatan elektronik modern untuk manajemen kunci yang membantu Anda mempertahankan kepatuhan dan kontrol. Sebaiknya gunakan pertukaran kunci elektronik berbasis standar alih-alih komponen kunci berbasis kertas.

Kekuatan kunci minimum dan pengaruhnya terhadap fungsi impor dan ekspor

PCI membutuhkan kekuatan kunci minimum khusus untuk operasi kriptografi, penyimpanan kunci, dan transmisi kunci. Persyaratan ini dapat berubah ketika standar PCI direvisi. Aturan menentukan bahwa kunci pembungkus yang digunakan untuk penyimpanan atau transportasi harus setidaknya sekuat kunci yang dilindungi. Kami menerapkan persyaratan ini secara otomatis selama ekspor dan mencegah kunci dilindungi oleh kunci yang lebih lemah, seperti yang ditunjukkan pada tabel berikut.

Tabel berikut menunjukkan kombinasi yang didukung dari kunci pembungkus, kunci untuk melindungi, dan metode perlindungan.

	Kunci Pembungkus											
Kunci Untuk Melindungi	TDES CI	TDES CI	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Catatan
TDES_2KU CI	TR-3	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
TDES_3KU CI	✗ Tidak diduk	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	
AES_128	✗ Tidak diduk	✗ Tidak diduk	TR-3	TR-3	TR-3	✗ Tidak diduk	TR-3 RSA	TR-3 RSA	ECDI	ECDI	ECDI	

	Kunci Pembungkus											
Kunci Untuk Melindungi	TDES CI	TDES CI	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Catatan
AES_192	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	TR-3	TR-3	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	ECDH	ECDH	
AES_256	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	TR-3	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	✗ Tidak diduk	ECDH	

Untuk informasi selengkapnya, lihat Lampiran D - Ukuran dan Kekuatan Kunci Minimum dan Setara untuk Algoritma yang Disetujui dalam standar PCI HSM.

Pertukaran Kunci Enkripsi Kunci (KEK)

Kami merekomendasikan menggunakan standar ANSI X9.24 TR-34. Jenis kunci awal ini dapat disebut Key Encryption Key (KEK), Zone Master Key (ZMK), atau Zone Control Master Key (ZCMK). Jika sistem atau mitra Anda tidak mendukung TR-34, Anda dapat menggunakan RSA Wrap/Unwrap. Jika kebutuhan Anda termasuk menukar kunci AES-256, Anda dapat menggunakan ECDH.

Jika Anda perlu terus memproses komponen kunci paper sampai semua mitra mendukung pertukaran kunci elektronik, pertimbangkan untuk menggunakan HSM offline atau menggunakan kustodian kunci pihak ketiga sebagai layanan.

> **Note**
>
> Untuk mengimpor kunci pengujian Anda sendiri atau untuk menyinkronkan kunci dengan yang ada HSMs, silakan lihat kode contoh Kriptografi AWS Pembayaran di. GitHub

Pertukaran Kunci Kerja (WK)

Kami menggunakan standar industri (ANSI X9.24 TR 31-2018 dan X9.143) untuk bertukar kunci kerja. Ini mengharuskan Anda telah menukar KEK menggunakan TR-34, RSA Wrap, ECDH atau skema serupa. Pendekatan ini memenuhi persyaratan PIN PCI untuk mengikat materi kunci secara kriptografis ke jenis dan penggunaannya setiap saat. Kunci kerja termasuk kunci kerja pengakuisisi, kunci kerja penerbit, BDK, dan IPEK.

Topik

Kunci impor

> **Important**
>
> Contoh memerlukan versi terbaru AWS CLI V2. Sebelum memulai, pastikan Anda telah meningkatkan ke versi terbaru.

Daftar Isi

Pengantar kunci impor

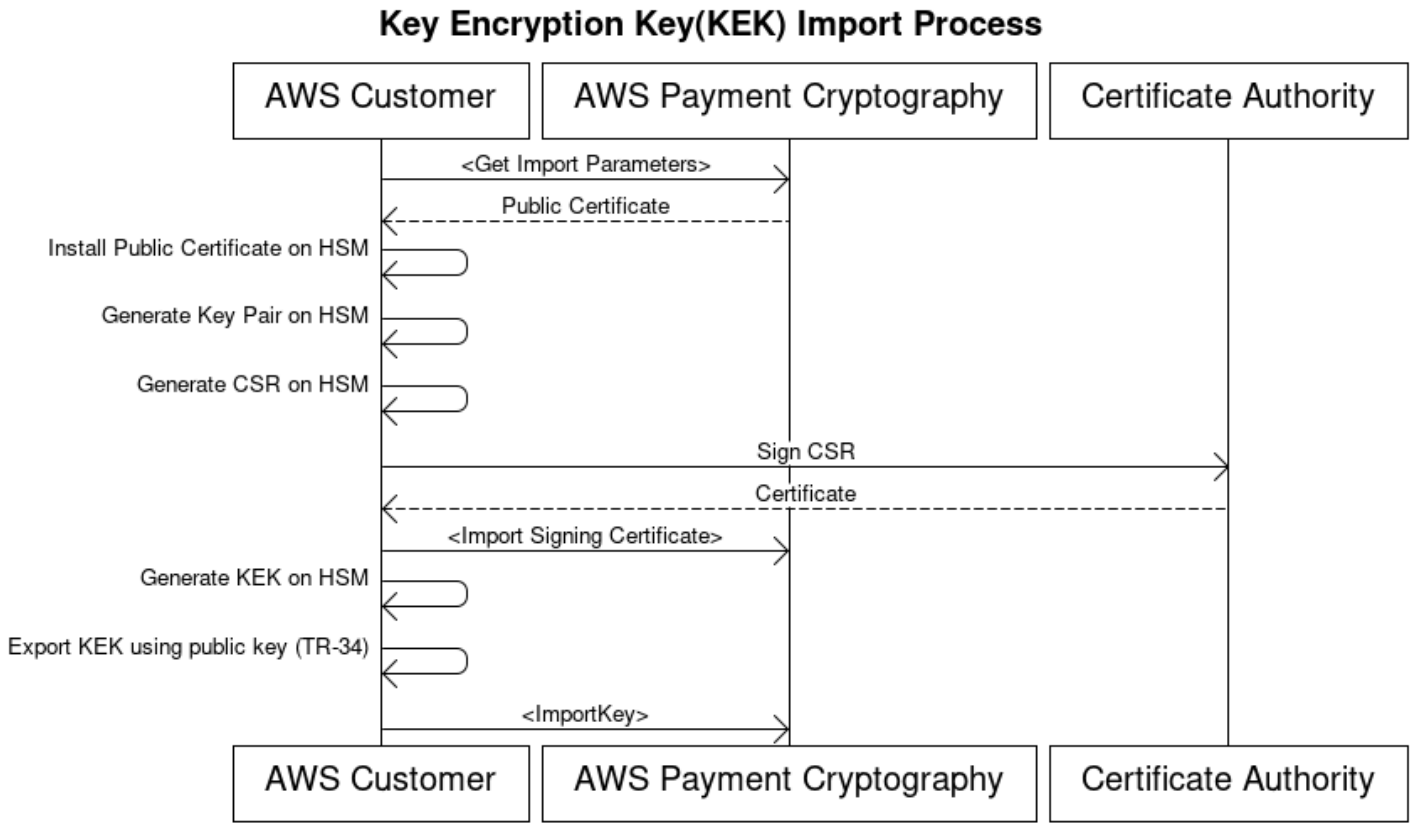
> **Note**
>
> Saat mengimpor kunci menggunakan blok kunci X9.143, TR-31 atau TR-34, Kriptografi AWS Pembayaran biasanya mempertahankan (tetapi tidak menggunakan) header opsional apa pun. Header HM (HMAC hash type) digunakan selama operasi kriptografi. Header KP (KCV dari kunci pembungkus) khusus untuk proses impor dan tidak dipertahankan.

Ketika bertukar kunci dengan rekanan, biasanya untuk pertama menukar kunci pertukaran kunci (KEK). Kunci ini kemudian akan digunakan untuk melindungi kunci berikutnya. Menggunakan format elektronik, KEK dapat ditukar menggunakan teknik asimetris seperti TR-34, ECDH atau RSA wrap. Kunci selanjutnya akan ditukar menggunakan pertukaran kunci simetris seperti TR-31. KEK ini akan berumur panjang dan hanya dapat diperbarui setiap beberapa tahun berdasarkan kebijakan dan periode kripto yang ditentukan.

Jika hanya satu atau dua kunci yang dipertukarkan, Anda juga dapat memilih untuk menggunakan teknik asimetris untuk langsung menukar kunci tersebut seperti BDK. AWS Kriptografi Pembayaran mendukung kedua metode pertukaran kunci.

Mengimpor kunci simetris

Kunci impor menggunakan teknik asimetris (TR-34)



TR-34 menggunakan kriptografi asimetris RSA untuk mengenkripsi dan menandatangani kunci simetris untuk pertukaran. Ini memastikan kerahasiaan (enkripsi) dan integritas (tanda tangan) dari kunci yang dibungkus.

Untuk mengimpor kunci Anda sendiri, lihat proyek sampel Kriptografi AWS Pembayaran di GitHub. Untuk petunjuk tentang cara import/export mengunci dari platform lain, kode sampel tersedia di GitHubatau lihat panduan pengguna untuk platform tersebut.

1. Panggil perintah Inisialisasi Impor

   Panggilan `get-parameters-for-import` untuk menginisialisasi proses impor. API ini menghasilkan key pair untuk impor kunci, menandatangani kunci, dan mengembalikan sertifikat dan root sertifikat. Enkripsi kunci yang akan diekspor menggunakan kunci ini. Dalam terminologi TR-34, ini dikenal sebagai Sertifikat KRD. Sertifikat ini dikodekan base64, berumur pendek, dan dimaksudkan hanya untuk tujuan ini. Simpan `ImportToken` nilainya.

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

```
{
    "ImportToken": "import-token-bwxli6ocftypneu5",
    "ParametersValidUntilTimestamp": 1698245002.065,
    "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0....",
    "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0....",
    "WrappingKeyAlgorithm": "RSA_2048"
}
```

2. Instal sertifikat publik pada sistem sumber utama

   Dengan sebagian besar HSMs, Anda perlu menginstal, memuat, atau mempercayai sertifikat publik yang dihasilkan pada langkah 1 untuk mengekspor kunci yang menggunakannya. Ini dapat mencakup seluruh rantai sertifikat atau hanya sertifikat root dari langkah 1, tergantung pada HSM.

3. Hasilkan key pair pada sistem sumber dan berikan rantai sertifikat ke AWS Payment Cryptography

   Untuk memastikan integritas muatan yang ditransmisikan, pihak pengirim (Key Distribution Host atau KDH) menandatanganinya. Buat kunci publik untuk tujuan ini dan buat sertifikat kunci publik (X509) untuk memberikan kembali Kriptografi AWS Pembayaran.

   Saat mentransfer kunci dari HSM, buat key pair pada HSM itu. HSM, pihak ketiga, atau layanan seperti AWS Private CA dapat menghasilkan sertifikat.

   Muat sertifikat root ke Kriptografi AWS Pembayaran menggunakan `importKey` perintah dengan KeyMaterialType dari `RootCertificatePublicKey` dan KeyUsageType dari. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

   Untuk sertifikat perantara, gunakan `importKey` perintah dengan KeyMaterialType dari `TrustedCertificatePublicKey` dan KeyUsageType dari`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Ulangi proses ini untuk beberapa sertifikat perantara. Gunakan sertifikat impor terakhir dalam rantai sebagai masukan ke perintah impor berikutnya. `KeyArn`

> **Note**
>
> Jangan mengimpor sertifikat daun. Berikan langsung selama perintah impor.

4. Ekspor kunci dari sistem sumber

   Banyak HSMs dan sistem terkait mendukung kunci ekspor menggunakan norma TR-34. Tentukan kunci publik dari langkah 1 sebagai sertifikat KRD (enkripsi) dan kunci dari langkah 3 sebagai sertifikat KDH (penandatanganan). Untuk mengimpor ke Kriptografi AWS Pembayaran, tentukan formatnya sebagai format dua pass TR-34.2012 non-CMS, yang juga dapat disebut sebagai format TR-34 Diebold.

5. Panggil Kunci Impor

   Panggil ImportKey API dengan file KeyMaterialType . `TR34_KEY_BLOCK` Gunakan keYarn dari CA terakhir yang diimpor pada langkah 3 untuk`certificate-authority-public-key-identifier`, bahan kunci yang dibungkus dari langkah 4 untuk`key-material`, dan sertifikat daun dari langkah 3 untuk`signing-key-certificate`. Sertakan token impor dari langkah 1.

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr34KeyBlock": { \
    "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/zabouwe3574jysdl", \
    "ImportToken": "import-token-bwxli6ocftypneu5", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "SigningKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJ...", \
    "WrappedKeyBlock":
 "308205A106092A864886F70D010702A08205923082058E020101310D300B0609608648016503040201308203.
 \
    }'
```

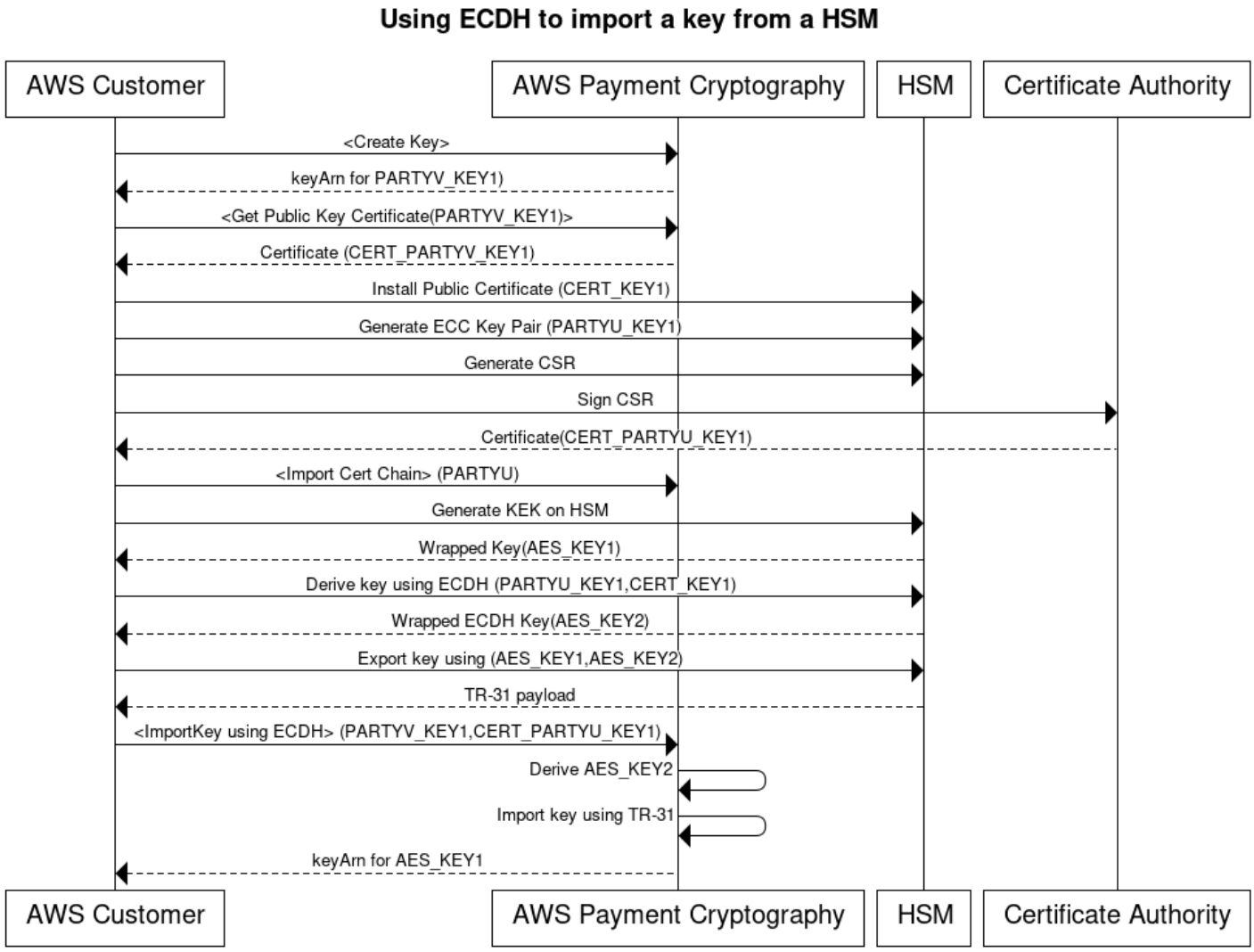
```
{
  "Key": {
    "CreateTimestamp": "2023-06-13T16:52:52.859000-04:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "KeyAttributes": {
      "KeyAlgorithm": "TDES_3KEY",
```

```
      "KeyClass": "SYMMETRIC_KEY",
      "KeyModesOfUse": {
        "Decrypt": true,
        "DeriveKey": false,
        "Encrypt": true,
        "Generate": false,
        "NoRestrictions": false,
        "Sign": false,
        "Unwrap": true,
        "Verify": false,
        "Wrap": true
      },
      "KeyUsage": "TR31_K1_KEY_ENCRYPTION_KEY"
    },
    "KeyCheckValue": "CB94A2",
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "KeyOrigin": "EXTERNAL",
    "KeyState": "CREATE_COMPLETE",
    "UsageStartTimestamp": "2023-06-13T16:52:52.859000-04:00"
  }
}
```

6. Gunakan kunci yang diimpor untuk operasi kriptografi atau impor berikutnya

   Jika yang diimpor KeyUsage adalah TR31 _K0_KEY_ENCRYPTION_KEY, Anda dapat menggunakan kunci ini untuk impor kunci berikutnya menggunakan TR-31. Untuk jenis kunci lainnya (seperti TR31 _D0_SYMMETRIC_DATA_ENCRYPTION_KEY), Anda dapat menggunakan kunci secara langsung untuk operasi kriptografi.

Kunci impor menggunakan teknik asimetris (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) menggunakan kriptografi asimetris ECC untuk membuat kunci bersama antara dua pihak tanpa memerlukan kunci yang telah dipertukarkan sebelumnya. Kunci ECDH bersifat sementara, jadi Kriptografi AWS Pembayaran tidak menyimpannya. Dalam proses ini, KBPK/KEK satu kali diturunkan menggunakan ECDH. Kunci turunan itu segera digunakan untuk membungkus kunci sebenarnya yang ingin Anda transfer, yang bisa berupa KBPK lain, kunci IPEK, atau jenis kunci lainnya.

Saat mengimpor, sistem pengiriman umumnya dikenal sebagai Pihak U (Inisiator) dan Kriptografi AWS Pembayaran dikenal sebagai Party V (Responder).

> **Note**
>
> Meskipun ECDH dapat digunakan untuk menukar jenis kunci simetris apa pun, ini adalah satu-satunya pendekatan yang dapat mentransfer kunci AES-256 dengan aman.

1. Hasilkan Pasangan Kunci ECC

   Panggilan `create-key` untuk membuat key pair ECC untuk proses ini. API ini menghasilkan key pair untuk impor atau ekspor kunci. Saat pembuatan, tentukan jenis kunci apa yang dapat diturunkan menggunakan kunci ECC ini. Saat menggunakan ECDH untuk menukar (membungkus) kunci lainnya, gunakan nilai. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

   > **Note**
   >
   > Meskipun ECDH tingkat rendah menghasilkan kunci turunan yang dapat digunakan untuk tujuan apa pun, Kriptografi AWS Pembayaran membatasi penggunaan kembali kunci yang tidak disengaja untuk berbagai tujuan dengan mengizinkan kunci hanya digunakan untuk satu jenis kunci turunan.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT,KeyClass=ASYM
 --derive-key-usage "TR31_K1_KEY_BLOCK_PROTECTION_KEY"
```

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
                    "KeyAttributes": {
                        "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT",
                        "KeyClass": "ASYMMETRIC_KEY_PAIR",
                        "KeyAlgorithm": "ECC_NIST_P256",
                        "KeyModesOfUse": {
                            "Encrypt": false,
                            "Decrypt": false,
                            "Wrap": false,
                            "Unwrap": false,
                            "Generate": false,
```

```
                            "Sign": false,
                            "Verify": false,
                            "DeriveKey": true,
                            "NoRestrictions": false
                        }
                    },
                    "KeyCheckValue": "2432827F",
                    "KeyCheckValueAlgorithm": "CMAC",
                    "Enabled": true,
                    "Exportable": true,
                    "KeyState": "CREATE_COMPLETE",
                    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                    "CreateTimestamp": "2025-03-28T22:03:41.087000-07:00",
                    "UsageStartTimestamp": "2025-03-28T22:03:41.068000-07:00"
                }
            }
```

2. Dapatkan Sertifikat Kunci Publik

   Hubungi `get-public-key-certificate` untuk menerima kunci publik sebagai sertifikat X.509 yang ditandatangani oleh CA akun Anda yang khusus untuk Kriptografi AWS Pembayaran di wilayah tertentu.

   Example

```
$ aws payment-cryptography get-public-key-certificate \
                --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv
```

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

3. Instal sertifikat publik pada sistem rekanan (Partai U)

   Dengan banyak HSMs, Anda perlu menginstal, memuat, atau mempercayai sertifikat publik yang dihasilkan pada langkah 1 untuk mengekspor kunci menggunakannya. Ini dapat mencakup seluruh rantai sertifikat atau hanya sertifikat root dari langkah 1, tergantung pada HSM. Konsultasikan dokumentasi HSM Anda untuk informasi lebih lanjut.

4. Hasilkan key pair ECC pada sistem sumber dan berikan rantai sertifikat ke AWS Payment Cryptography

   Dalam ECDH, masing-masing pihak menghasilkan key pair dan menyetujui common key. Untuk Kriptografi AWS Pembayaran untuk mendapatkan kunci, diperlukan kunci publik rekanan dalam format kunci publik X.509.

   Saat mentransfer kunci dari HSM, buat key pair pada HSM itu. Untuk blok kunci dukungan HSMs itu, header kunci akan terlihat mirip dengan`D0144K3EX00E0000`. Saat membuat sertifikat, Anda biasanya menghasilkan CSR pada HSM dan kemudian HSM, pihak ketiga, atau layanan seperti AWS Private CA dapat menghasilkan sertifikat.

   Muat sertifikat root ke Kriptografi AWS Pembayaran menggunakan `importKey` perintah dengan KeyMaterialType dari `RootCertificatePublicKey` dan KeyUsageType dari. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

   Untuk sertifikat perantara, gunakan `importKey` perintah dengan KeyMaterialType dari `TrustedCertificatePublicKey` dan KeyUsageType dari`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Ulangi proses ini untuk beberapa sertifikat perantara. Gunakan sertifikat impor terakhir dalam rantai sebagai masukan ke perintah impor berikutnya. `KeyArn`

   > **Note**
   >
   > Jangan mengimpor sertifikat daun. Berikan langsung selama perintah impor.

5. Dapatkan kunci satu kali menggunakan ECDH pada Party U HSM

   Banyak HSMs dan sistem terkait mendukung pembuatan kunci menggunakan ECDH. Tentukan kunci publik dari langkah 1 sebagai kunci publik dan kunci dari langkah 3 sebagai kunci pribadi. Untuk opsi yang diizinkan, seperti metode derivasi, lihat panduan API.

   > **Note**
   >
   > Parameter derivasi seperti tipe hash harus sama persis di kedua sisi. Jika tidak, Anda akan menghasilkan kunci yang berbeda.

6. Ekspor kunci dari sistem sumber

Terakhir, ekspor kunci yang ingin Anda bawa ke Kriptografi AWS Pembayaran menggunakan perintah TR-31 standar. Tentukan kunci turunan ECDH sebagai KBPK. Kunci yang akan diekspor dapat berupa kunci TDES atau AES yang tunduk pada kombinasi TR-31 yang valid, selama kunci pembungkus setidaknya sekuat kunci yang akan diekspor.

7. Panggil Kunci Impor

   Panggil `import-key` API dengan KeyMaterialType file`DiffieHellmanTr31KeyBlock`. Gunakan keYarn dari CA terakhir yang diimpor pada langkah 3 untuk`certificate-authority-public-key-identifier`, bahan kunci yang dibungkus dari langkah 4 untuk`key-material`, dan sertifikat daun dari langkah 3 untuk. `public-key-certificate` Sertakan kunci pribadi ARN dari langkah 1.

```
$ aws payment-cryptography import-key \
          --key-material='{
            "DiffieHellmanTr31KeyBlock": {
              "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-2:111122223333:key/swseahwtq2oj6zi5",
              "DerivationData": {
                "SharedInformation": "1234567890"
              },
              "DeriveKeyAlgorithm": "AES_256",
              "KeyDerivationFunction": "NIST_SP800",
              "KeyDerivationHashAlgorithm": "SHA_256",
              "PrivateKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
              "PublicKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN....",
              "WrappedKeyBlock":
 "D0112K1TB00E0000D603CCA8ACB71517906600FF8F0F195A38776A7190A0EF0024F088A5342DB98E2735084A7
            }
          }'
```

```
{
        "Key": {
          "CreateTimestamp": "2025-03-13T16:52:52.859000-04:00",
          "Enabled": true,
          "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
          "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_K1_KEY_ENCRYPTION_KEY"
          },
          "KeyCheckValue": "CB94A2",
          "KeyCheckValueAlgorithm": "ANSI_X9_24",
          "KeyOrigin": "EXTERNAL",
          "KeyState": "CREATE_COMPLETE",
          "UsageStartTimestamp": "2025-03-13T16:52:52.859000-04:00"
        }
      }
```

8. Gunakan kunci yang diimpor untuk operasi kriptografi atau impor berikutnya

   Jika yang diimpor KeyUsage adalah TR31 _K0_KEY_ENCRYPTION_KEY, Anda dapat menggunakan kunci ini untuk impor kunci berikutnya menggunakan TR-31. Untuk jenis kunci lainnya (seperti TR31 _D0_SYMMETRIC_DATA_ENCRYPTION_KEY), Anda dapat menggunakan kunci secara langsung untuk operasi kriptografi.

Kunci impor menggunakan teknik asimetris (RSA Unwrap)

Ikhtisar: Kriptografi AWS Pembayaran mendukung RSA wrap/unwrap untuk pertukaran kunci ketika TR-34 tidak layak. Seperti TR-34, teknik ini menggunakan kriptografi asimetris RSA untuk mengenkripsi kunci simetris untuk pertukaran. Namun, tidak seperti TR-34, metode ini tidak memiliki pihak pengirim yang menandatangani payload. Selain itu, teknik pembungkus RSA ini tidak menjaga integritas metadata kunci selama transfer karena tidak menyertakan blok kunci.

> **Note**
>
> Anda dapat menggunakan bungkus RSA untuk mengimpor atau mengekspor kunci TDES dan AES-128.

1. Panggil perintah Inisialisasi Impor

   Panggilan get-parameters-for-import untuk menginisialisasi proses impor dengan `KeyMaterialType` dari. `KEY_CRYPTOGRAM` Gunakan `RSA_2048` untuk `WrappingKeyAlgorithm` saat bertukar tombol TDES. Gunakan `RSA_3072` atau `RSA_4096` saat menukar tombol TDES atau AES-128. API ini menghasilkan key pair untuk impor kunci, menandatangani kunci menggunakan root sertifikat, dan mengembalikan sertifikat dan root sertifikat. Enkripsi kunci yang akan diekspor menggunakan kunci ini. Sertifikat ini berumur pendek dan dimaksudkan hanya untuk tujuan ini.

   ```
   $ 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. Instal sertifikat publik pada sistem sumber utama

   Dengan banyak HSMs, Anda perlu menginstal, memuat, atau mempercayai sertifikat publik (dan/atau akarnya) yang dihasilkan pada langkah 1 untuk mengekspor kunci menggunakannya.

3. Ekspor kunci dari sistem sumber

   Banyak HSMs dan sistem terkait mendukung kunci ekspor menggunakan bungkus RSA. Tentukan kunci publik dari langkah 1 sebagai sertifikat enkripsi (`WrappingKeyCertificate`). Jika Anda membutuhkan rantai kepercayaan, gunakan `WrappingKeyCertificateChain` dari langkah 1. Saat mengekspor kunci dari HSM Anda, tentukan formatnya sebagai RSA, dengan Mode Padding = PKCS #1 v2.2 OAEP (dengan SHA 256 atau SHA 512).

4. Panggilan import-key

   Panggil import-key API dengan `KeyMaterialType` file`KeyMaterial`. Anda membutuhkan `ImportToken` dari langkah 1 dan `key-material` (bahan kunci yang dibungkus) dari langkah 3. Berikan parameter kunci (seperti Key Usage) karena RSA wrap tidak menggunakan blok kunci.

```
$ 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
```

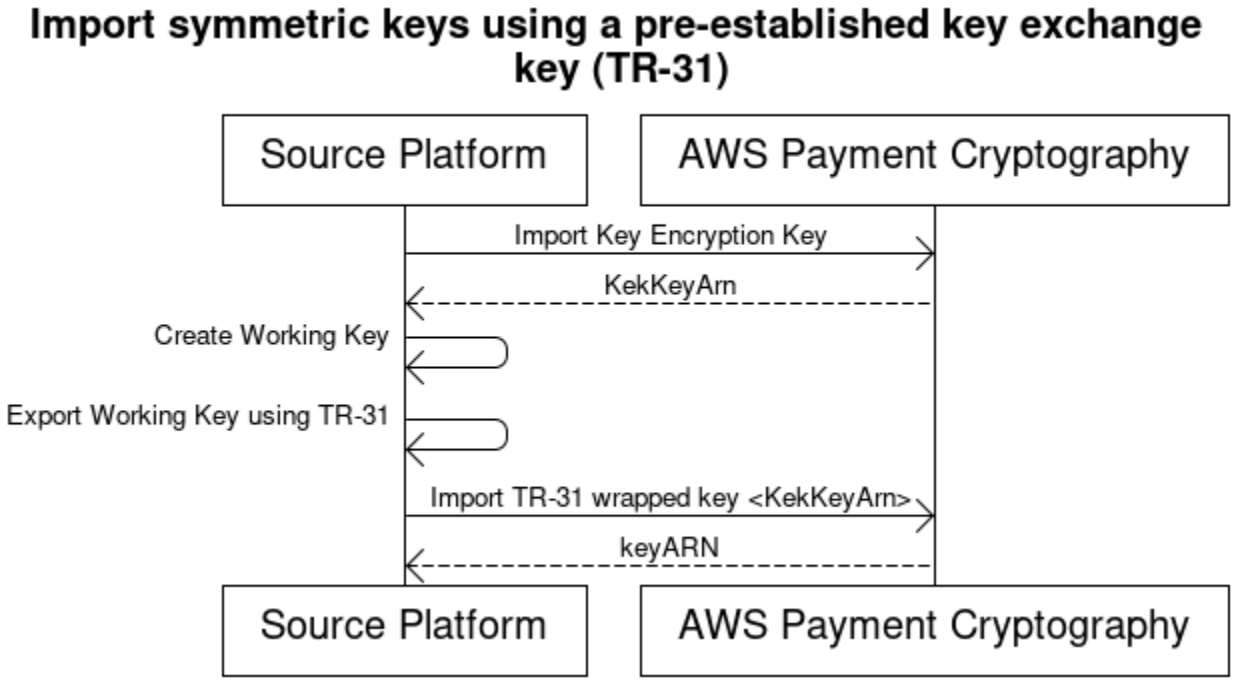
```
{
  "Key": {
    "KeyOrigin": "EXTERNAL",
    "Exportable": true,
    "KeyCheckValue": "DA1ACF",
    "UsageStartTimestamp": 1697643478.92,
    "Enabled": true,
```

```
    "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. Gunakan kunci yang diimpor untuk operasi kriptografi atau impor berikutnya

   Jika diimpor `KeyUsage` adalah `TR31_K0_KEY_ENCRYPTION_KEY` atau`TR31_K1_KEY_BLOCK_PROTECTION_KEY`, Anda dapat menggunakan kunci ini untuk impor kunci berikutnya menggunakan TR-31. Jika jenis kunci adalah jenis lain (seperti`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`), Anda dapat menggunakan kunci secara langsung untuk operasi kriptografi.

Impor kunci simetris menggunakan kunci pertukaran kunci yang telah ditetapkan sebelumnya (TR-31)



Saat bertukar beberapa kunci atau mendukung rotasi kunci, mitra biasanya pertama kali menukar kunci enkripsi kunci awal (KEK). Anda dapat melakukan ini menggunakan teknik seperti komponen kunci paper atau, untuk Kriptografi AWS Pembayaran, menggunakan TR-34.

Setelah membuat KEK, Anda dapat menggunakannya untuk mengangkut kunci berikutnya (termasuk yang lain KEKs). AWS Kriptografi Pembayaran mendukung pertukaran kunci ini menggunakan ANSI TR-31, yang banyak digunakan dan didukung oleh vendor HSM.

1. Kunci Enkripsi Kunci Impor (KEK)

   Pastikan Anda telah mengimpor KEK Anda dan memiliki keYarn (atau KeyAlias) yang tersedia.

2. Buat kunci pada platform sumber

   Jika kunci tidak ada, buat di platform sumber. Atau, Anda dapat membuat kunci pada Kriptografi AWS Pembayaran dan menggunakan export perintah.

3. Ekspor kunci dari platform sumber

   Saat mengekspor, tentukan format ekspor sebagai TR-31. Platform sumber akan meminta kunci untuk mengekspor dan kunci enkripsi kunci untuk digunakan.

4. Impor ke Kriptografi AWS Pembayaran

Saat memanggil import-key perintah, gunakan keYarn (atau alias) kunci enkripsi kunci Anda untuk. `WrappingKeyIdentifier` Gunakan output dari platform sumber untuk`WrappedKeyBlock`.

Example

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza", \
    "WrappedKeyBlock":
 "D0112B0AX00E00002E0A3D58252CB67564853373D1EBCC1E23B2ADE7B15E967CC27B85D5999EF58E11662991F
 \
    }'
```

```
{
  "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"
  }
}
```

Mengimpor kunci publik asimetris (RSA, ECC)

Semua sertifikat yang diimpor harus setidaknya sekuat sertifikat penerbitannya (pendahulunya) dalam rantai. Ini berarti bahwa RSA_2048 CA hanya dapat digunakan untuk melindungi sertifikat daun RSA_2048 dan sertifikat ECC harus dilindungi oleh sertifikat ECC lain dengan kekuatan setara. Sertifikat ECC P384 hanya dapat diterbitkan oleh P384 atau P521 CA. Semua sertifikat harus belum kedaluwarsa pada saat impor.

Mengimpor kunci publik RSA

AWS Kriptografi Pembayaran mendukung impor kunci RSA publik sebagai sertifikat X.509. Untuk mengimpor sertifikat, pertama-tama impor sertifikat akarnya. Semua sertifikat harus belum kedaluwarsa pada saat impor. Sertifikat harus dalam format PEM dan base64 dikodekan.

1. Impor Sertifikat Root ke Kriptografi AWS Pembayaran

   Gunakan perintah berikut untuk mengimpor sertifikat root:

Example

2. Impor Sertifikat Kunci Publik ke Kriptografi AWS Pembayaran

   Anda sekarang dapat mengimpor kunci publik. Karena TR-34 dan ECDH mengandalkan lulus sertifikat daun saat run-time, opsi ini hanya digunakan saat mengenkripsi data menggunakan kunci publik dari sistem lain. KeyUsage akan diatur ke TR31 _D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION.

Example

```
$ aws payment-cryptography import-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza", \
    "WrappedKeyBlock":
 "D0112B0AX00E00002E0A3D58252CB67564853373D1EBCC1E23B2ADE7B15E967CC27B85D5999EF58E11662991F
 \
    }'
```

```
{
  "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"
  }
}
```

Mengimpor kunci publik ECC

AWS Kriptografi Pembayaran mendukung impor kunci ECC publik sebagai sertifikat X.509. Untuk mengimpor sertifikat, pertama-tama impor sertifikat CA root dan sertifikat perantara apa pun. Semua sertifikat harus belum kedaluwarsa pada saat impor. Sertifikat harus dalam format PEM dan base64 dikodekan.

1. Impor Sertifikat Root ECC ke AWS Kriptografi Pembayaran

   Gunakan perintah berikut untuk mengimpor sertifikat root:

Example

2. Impor Sertifikat Menengah ke Kriptografi AWS Pembayaran

   Gunakan perintah berikut untuk mengimpor sertifikat perantara:

Example

3. Impor Sertifikat Kunci Publik (Daun) ke dalam Kriptografi AWS Pembayaran

   Meskipun Anda dapat mengimpor sertifikat ECC daun, saat ini tidak ada fungsi yang ditentukan dalam Kriptografi AWS Pembayaran untuk itu selain penyimpanan. Ini karena saat menggunakan fungsi ECDH, sertifikat daun diteruskan saat runtime.

Kunci ekspor

Daftar Isi

Ekspor kunci simetris

> **Important**
>
> Pastikan Anda memiliki versi terbaru AWS CLI sebelum memulai. Untuk memutakhirkan, lihat Menginstal file AWS CLI.

Kunci ekspor menggunakan teknik asimetris (TR-34)

TR-34 menggunakan kriptografi asimetris RSA untuk mengenkripsi dan menandatangani kunci simetris untuk pertukaran. Enkripsi melindungi kerahasiaan, sementara tanda tangan memastikan integritas. Saat Anda mengekspor kunci, Kriptografi AWS Pembayaran bertindak sebagai host distribusi kunci (KDH), dan sistem target Anda menjadi perangkat penerima kunci (KRD).

> **Note**
>
> Jika HSM Anda mendukung ekspor TR-34 tetapi bukan impor TR-34, kami sarankan Anda terlebih dahulu membuat KEK bersama antara HSM Anda dan Kriptografi Pembayaran menggunakan TR-34. AWS Anda kemudian dapat menggunakan TR-31 untuk mentransfer kunci yang tersisa.

1. Inisialisasi proses ekspor

   Jalankan get-parameters-for-export untuk menghasilkan key pair untuk ekspor kunci. Kami menggunakan key pair ini untuk menandatangani payload TR-34. Dalam terminologi TR-34, ini adalah sertifikat penandatanganan KDH. Sertifikat berumur pendek dan hanya berlaku untuk durasi yang ditentukan dalam`ParametersValidUntilTimestamp`.

   > **Note**
   >
   > Semua sertifikat dalam pengkodean base64.

   Example

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

   ```
   {
     "SigningKeyCertificate":
    "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFTZ0F3SUJ...",
     "SigningKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS....",
     "SigningKeyAlgorithm": "RSA_2048",
     "ExportToken": "export-token-au7pvkbsq4mbup6i",
     "ParametersValidUntilTimestamp": "2023-06-13T15:40:24.036000-07:00"
   }
   ```

2. Impor sertifikat Kriptografi AWS Pembayaran ke sistem penerima Anda

   Impor rantai sertifikat dari langkah 1 ke sistem penerima Anda.

3. Siapkan sertifikat sistem penerimaan Anda

Untuk melindungi muatan yang ditransmisikan, pihak pengirim (KDH) mengenkripsinya. Sistem penerima Anda (biasanya HSM Anda atau HSM mitra Anda) perlu menghasilkan kunci publik dan membuat sertifikat kunci publik X.509. Anda dapat menggunakan AWS Private CA untuk menghasilkan sertifikat, tetapi Anda dapat menggunakan otoritas sertifikat apa pun.

Setelah Anda memiliki sertifikat, impor sertifikat root ke Kriptografi AWS Pembayaran menggunakan ImportKey perintah. Atur `KeyMaterialType` ke `RootCertificatePublicKey` dan `KeyUsageType` ke `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Kami menggunakan `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` sebagai `KeyUsageType` karena ini adalah kunci root yang menandatangani sertifikat daun. Anda tidak perlu mengimpor sertifikat daun ke dalam Kriptografi AWS Pembayaran — Anda dapat meneruskannya secara inline.

> **Note**
>
> Jika sebelumnya Anda mengimpor sertifikat root, lewati langkah ini. Untuk sertifikat perantara, gunakan`TrustedCertificatePublicKey`.

4. Ekspor kunci Anda

   Panggil ExportKey API dengan `KeyMaterialType` set ke`TR34_KEY_BLOCK`. Anda perlu menyediakan:

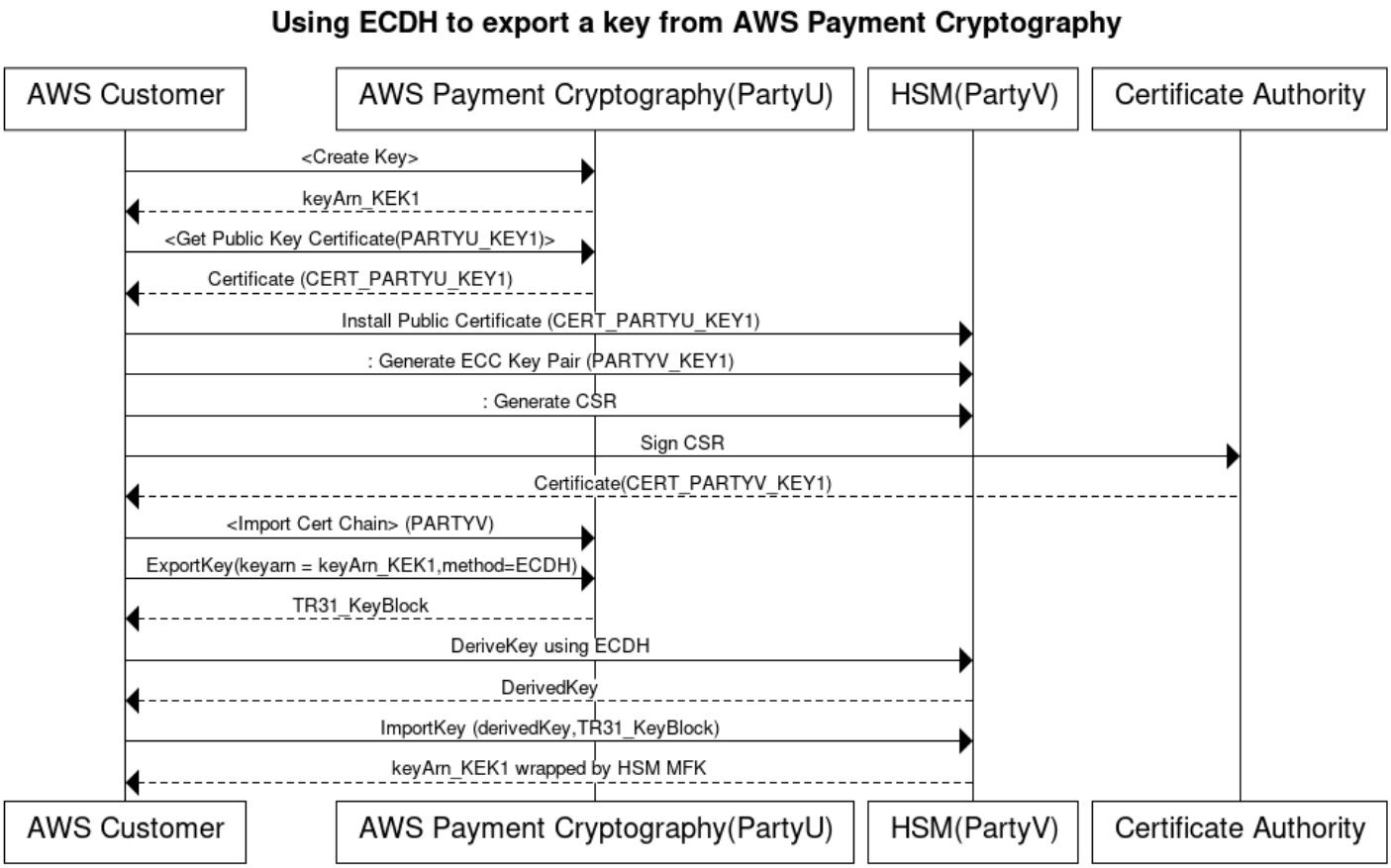
   - KeYarn dari akar CA dari langkah 3 sebagai `CertificateAuthorityPublicKeyIdentifier`
   - Sertifikat daun dari langkah 3 sebagai `WrappingKeyCertificate`
   - KeYarn (atau alias) dari kunci yang ingin Anda ekspor sebagai `--export-key-identifier`
   - Token ekspor dari langkah 1

Example

```
$ aws payment-cryptography export-key \
    --export-key-identifier "example-export-key" \
    --key-material '{"Tr34KeyBlock": { \
    "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/4kd6xud22e64wcbk", \
    "ExportToken": "export-token-au7pvkbsq4mbup6i", \
    "KeyBlockFormat": "X9_TR34_2012", \
    "WrappingKeyCertificate":
 "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV2RENDQXFXZ0F3SUJBZ0lSQ..."} \
    }'
```

```
{
  "WrappedKey": {
    "KeyMaterial": "308205A106092A864886F70D010702A08205923082058...",
    "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK"
  }
}
```

Kunci ekspor menggunakan teknik asimetris (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) menggunakan kriptografi asimetris ECC untuk membuat kunci bersama antara dua pihak tanpa memerlukan kunci yang telah dipertukarkan sebelumnya. Kunci ECDH bersifat sementara, jadi Kriptografi AWS Pembayaran tidak menyimpannya. Dalam proses ini, KBPK/KEK satu kali diturunkan menggunakan ECDH. Kunci turunan itu segera digunakan untuk membungkus kunci yang ingin Anda transfer, yang bisa berupa KBPK lain, BDK, kunci IPEK, atau jenis kunci lainnya.

Saat mengekspor, Kriptografi AWS Pembayaran disebut sebagai Pihak U (Inisiator) dan sistem penerima dikenal sebagai Party V (Responder).

> **Note**
>
> ECDH dapat digunakan untuk menukar jenis kunci simetris apa pun, tetapi ini adalah satu-satunya pendekatan yang dapat digunakan untuk mentransfer kunci AES-256 jika KEK belum ditetapkan.

1. Hasilkan Pasangan Kunci ECC

Panggilan `create-key` untuk membuat key pair ECC untuk proses ini. API ini menghasilkan key pair untuk impor atau ekspor kunci. Saat pembuatan, tentukan jenis kunci apa yang dapat diturunkan menggunakan kunci ECC ini. Saat menggunakan ECDH untuk menukar (membungkus) kunci lainnya, gunakan nilai. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

> **Note**
>
> Meskipun ECDH tingkat rendah menghasilkan kunci turunan yang dapat digunakan untuk tujuan apa pun, Kriptografi AWS Pembayaran membatasi penggunaan kembali kunci yang tidak disengaja untuk berbagai tujuan dengan mengizinkan kunci hanya digunakan untuk satu jenis kunci turunan.

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=ECC_NIST_P256,KeyUsage=TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT,KeyClass=ASYM
 --derive-key-usage "TR31_K1_KEY_BLOCK_PROTECTION_KEY"
```

```
{
        "Key": {
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
wc3rjsssguhxtilv",
            "KeyAttributes": {
                "KeyUsage": "TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT",
                "KeyClass": "ASYMMETRIC_KEY_PAIR",
                "KeyAlgorithm": "ECC_NIST_P256",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "2432827F",
```

```
            "KeyCheckValueAlgorithm": "CMAC",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2025-03-28T22:03:41.087000-07:00",
            "UsageStartTimestamp": "2025-03-28T22:03:41.068000-07:00"
        }
    }
```

2. Dapatkan Sertifikat Kunci Publik

   Hubungi `get-public-key-certificate` untuk menerima kunci publik sebagai sertifikat X.509 yang ditandatangani oleh CA akun Anda yang khusus untuk Kriptografi AWS Pembayaran di wilayah tertentu.

   Example

```
$ aws payment-cryptography get-public-key-certificate \
          --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv
```

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

3. Instal sertifikat publik pada sistem rekanan (Partai V)

   Dengan banyak HSMs, Anda perlu menginstal, memuat, atau mempercayai sertifikat publik yang dihasilkan pada langkah 1 untuk membuat kunci. Ini dapat mencakup seluruh rantai sertifikat atau hanya sertifikat root, tergantung pada HSM. Konsultasikan dokumentasi HSM Anda untuk instruksi khusus.

4. Hasilkan key pair ECC pada sistem sumber dan berikan rantai sertifikat ke AWS Payment Cryptography

   Dalam ECDH, masing-masing pihak menghasilkan key pair dan menyetujui common key. Untuk Kriptografi AWS Pembayaran untuk mendapatkan kunci, diperlukan kunci publik rekanan dalam format kunci publik X.509.

Saat mentransfer kunci dari HSM, buat key pair pada HSM itu. Untuk blok kunci dukungan HSMs itu, header kunci akan terlihat mirip dengan`D0144K3EX00E0000`. Saat membuat sertifikat, Anda biasanya menghasilkan CSR di HSM, dan kemudian HSM, pihak ketiga, atau layanan seperti AWS Private CA dapat menghasilkan sertifikat.

Muat sertifikat root ke Kriptografi AWS Pembayaran menggunakan `importKey` perintah dengan KeyMaterialType dari `RootCertificatePublicKey` dan KeyUsageType dari. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

Untuk sertifikat perantara, gunakan `importKey` perintah dengan KeyMaterialType dari `TrustedCertificatePublicKey` dan KeyUsageType dari`TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`. Ulangi proses ini untuk beberapa sertifikat perantara. Gunakan sertifikat impor terakhir dalam rantai sebagai masukan ke perintah ekspor berikutnya. `KeyArn`

> **Note**
>
> Jangan mengimpor sertifikat daun. Berikan langsung selama perintah ekspor.

5. Dapatkan kunci dan kunci ekspor dari Kriptografi AWS Pembayaran

Saat mengekspor, layanan memperoleh kunci menggunakan ECDH dan kemudian segera menggunakannya sebagai KBPK untuk membungkus kunci ekspor menggunakan TR-31. Kunci yang akan diekspor dapat berupa kunci TDES atau AES yang tunduk pada kombinasi TR-31 yang valid, selama kunci pembungkus setidaknya sekuat kunci yang akan diekspor.

```
$ aws payment-cryptography export-key \
            --export-key-identifier arn:aws:payment-cryptography:us-
west-2:529027455495:key/e3a65davqhbpjm4h \
            --key-material='{
              "DiffieHellmanTr31KeyBlock": {
                "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-2:111122223333:key/swseahwtq2oj6zi5",
                "DerivationData": {
                  "SharedInformation": "ADEF567890"
                },
                "DeriveKeyAlgorithm": "AES_256",
                "KeyDerivationFunction": "NIST_SP800",
                "KeyDerivationHashAlgorithm": "SHA_256",
```

```
                "PrivateKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/wc3rjsssguhxtilv",
                "PublicKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FUR..."
              }
            }'
```

```
{
            "WrappedKey": {
                "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK",
                "KeyMaterial":
 "D0112K1TB00E00007012724C0FAAF64DA50E2FF4F9A94DF50441143294E0E995DB2171554223EAA56D078C4CF
                "KeyCheckValue": "E421AD",
                "KeyCheckValueAlgorithm": "ANSI_X9_24"
            }
        }
```

6. Dapatkan kunci satu kali menggunakan ECDH pada Party V HSM

   Banyak HSMs dan sistem terkait mendukung pembuatan kunci menggunakan ECDH. Tentukan kunci publik dari langkah 1 sebagai kunci publik dan kunci dari langkah 3 sebagai kunci pribadi. Untuk opsi yang diizinkan, seperti metode derivasi, lihat panduan API.

   > **Note**
   >
   > Parameter derivasi seperti tipe hash harus sama persis di kedua sisi. Jika tidak, Anda akan menghasilkan kunci yang berbeda.

7. Kunci impor ke sistem target

   Terakhir, impor kunci dari Kriptografi AWS Pembayaran menggunakan perintah TR-31 standar. Tentukan kunci turunan ECDH sebagai KBPK dan gunakan blok kunci TR-31 yang sebelumnya diekspor dari Kriptografi Pembayaran. AWS

Kunci ekspor menggunakan teknik asimetris (RSA Wrap)

Ketika TR-34 tidak tersedia, Anda dapat menggunakan RSA wrap/unwrap untuk pertukaran kunci. Seperti TR-34, metode ini menggunakan kriptografi asimetris RSA untuk mengenkripsi kunci simetris. Namun, bungkus RSA tidak termasuk:

- Penandatanganan muatan oleh pihak pengirim

- Blok kunci yang menjaga integritas metadata kunci selama transportasi

> **Note**
>
> Anda dapat menggunakan bungkus RSA untuk mengekspor tombol TDES dan AES-128.

1. Buat kunci RSA dan sertifikat pada sistem penerima Anda

   Buat atau identifikasi kunci RSA untuk menerima kunci yang dibungkus. Kami membutuhkan kunci dalam format sertifikat X.509. Pastikan sertifikat ditandatangani oleh root certificate yang dapat Anda impor ke AWS Payment Cryptography.

2. Impor sertifikat publik root ke Kriptografi AWS Pembayaran

   Gunakan import-key dengan `--key-material` opsi untuk mengimpor sertifikat

```
$ 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": "LS0tLS1CRUdJTiBDRV..."} \
    }'
```

```
{
  "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. Ekspor kunci Anda

   Beri tahu Kriptografi AWS Pembayaran untuk mengekspor kunci Anda menggunakan sertifikat daun Anda. Anda perlu menentukan:

   - ARN untuk sertifikat root yang Anda impor di langkah 2
   - Sertifikat daun untuk ekspor
   - Kunci simetris untuk mengekspor

   Outputnya adalah versi biner berbungkus (terenkripsi) hex-encode dari kunci simetris Anda.

Example Contoh - Mengekspor kunci

```
$ 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": "LS0tLS1CRUdJTiBDEXAMPLE...",
      "WrappingSpec": "RSA_OAEP_SHA_256"
    }
  }
}
```

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

```
{
  "WrappedKey": {
    "KeyMaterial":
 "18874746731E9E1C4562E4116D1C2477063FCB08454D757D81854AEAEE0A52B1F9D303FA29C02DC82AE778535
    "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM"
  }
}
```

4. Impor kunci ke sistem penerima Anda

   Banyak HSMs dan sistem terkait mendukung kunci impor menggunakan RSA unwrap (termasuk Kriptografi AWS Pembayaran). Saat mengimpor, tentukan:

   - Kunci publik dari langkah 1 sebagai sertifikat enkripsi
   - Format sebagai RSA
   - Mode Padding sebagai PKCS #1 v2.2 OAEP (dengan SHA 256)

> **Note**
>
> Kami menampilkan kunci yang dibungkus dalam format HexBinary. Anda mungkin perlu mengonversi format jika sistem Anda memerlukan representasi biner yang berbeda, seperti base64.

Ekspor kunci simetris menggunakan kunci pertukaran kunci yang telah ditetapkan sebelumnya (TR-31)

Saat bertukar beberapa kunci atau mendukung rotasi kunci, Anda biasanya menukar kunci enkripsi kunci awal (KEK) terlebih dahulu menggunakan komponen kunci kertas atau, dengan Kriptografi AWS Pembayaran, menggunakan TR-34. Setelah membuat KEK, Anda dapat menggunakannya untuk mengangkut kunci berikutnya, termasuk yang lain KEKs. Kami mendukung pertukaran kunci ini menggunakan ANSI TR-31, yang didukung secara luas oleh vendor HSM.

1. Siapkan Kunci Enkripsi Kunci (KEK)

   Pastikan Anda telah menukar KEK Anda dan memiliki keYarn (atau KeyAlias) yang tersedia.

2. Buat kunci Anda pada Kriptografi AWS Pembayaran

   Buat kunci Anda jika belum ada. Atau, Anda dapat membuat kunci pada sistem Anda yang lain dan menggunakan perintah impor.

3. Ekspor kunci Anda dari Kriptografi AWS Pembayaran

   Saat mengekspor dalam format TR-31, tentukan kunci yang ingin Anda ekspor dan kunci pembungkus yang akan digunakan.

Example Contoh - Mengekspor kunci menggunakan blok TR31 kunci

```
$ 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":
 "D0144K0AB00E0000A24D3ACF3005F30A6E31D533E07F2E1B17A2A003B338B1E79E5B3AD4FBF7850FACF9A3784
    "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK"
  }
}
```

4. Impor kunci ke sistem Anda

   Gunakan implementasi kunci impor sistem Anda untuk mengimpor kunci.

Kunci Awal DUKPT Ekspor (IPEK/IK)

Saat menggunakan DUKPT, Anda dapat menghasilkan Base Derivation Key (BDK) tunggal untuk armada terminal. Terminal tidak memiliki akses langsung ke BDK. Sebagai gantinya, setiap terminal menerima kunci terminal awal yang unik, yang dikenal sebagai IPEK atau Initial Key (IK). Setiap IPEK berasal dari BDK menggunakan Key Serial Number (KSN) yang unik.

Struktur KSN bervariasi menurut jenis enkripsi:

- Untuk TDES: KSN 10-byte meliputi:
  - 24 bit untuk Key Set ID
  - 19 bit untuk ID terminal
  - 21 bit untuk penghitung transaksi
- Untuk AES: KSN 12-byte meliputi:
  - 32 bit untuk ID BDK

- 32 bit untuk pengenal derivasi (ID)
- 32 bit untuk penghitung transaksi

Kami menyediakan mekanisme untuk menghasilkan dan mengekspor kunci awal ini. Anda dapat mengekspor kunci yang dihasilkan menggunakan metode pembungkus TR-31, TR-34, atau RSA. Perhatikan bahwa kunci IPEK tidak bertahan dan tidak dapat digunakan untuk operasi selanjutnya pada Kriptografi AWS Pembayaran.

Kami tidak memberlakukan pemisahan antara dua bagian pertama KSN. Jika Anda ingin menyimpan pengenal derivasi dengan BDK, Anda dapat menggunakan tag. AWS

> **Note**
>
> Bagian counter dari KSN (32 bit untuk AES DUKPT) tidak digunakan untuk derivasi IPEK/IK. Misalnya, input 12345678901234560001 dan 12345678901234569999 akan menghasilkan IPEK yang sama.

```
$ 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"
}
}
```

Tentukan header blok kunci untuk ekspor

Anda dapat memodifikasi atau menambahkan informasi blok kunci saat mengekspor dalam format ASC TR-31 atau TR-34. Tabel berikut menjelaskan format blok kunci TR-31 dan elemen mana yang dapat Anda modifikasi selama ekspor.

Atribut Blok Kunci	Tujuan	Bisakah Anda memodifikasi selama ekspor?	Catatan
ID Versi	Mendefinisikan metode yang digunakan untuk melindungi bahan utama. Standar meliputi: • Versi A dan C (varian kunci - usang) • Versi B (derivasi menggunakan TDES) • Versi D (derivasi kunci menggunakan AES)	Tidak	Kami menggunakan versi B untuk tombol pembungkus TDES dan versi D untuk tombol pembungkus AES. Kami mendukung versi A dan C hanya untuk operasi impor.
Panjang Blok Kunci	Menentukan panjang pesan yang tersisa	Tidak	Kami menghitung nilai ini secara otomatis. Panjangnya mungkin tampak salah sebelum mendekripsi muatan karena kami dapat menambahkan padding kunci seperti yang dipersyaratkan oleh spesifikasi.

Atribut Blok Kunci	Tujuan	Bisakah Anda memodifikasi selama ekspor?	Catatan
Penggunaan Kunci	Mendefinisikan tujuan yang diizinkan untuk kunci, seperti: • C0 (Verifikasi Kartu) • B0 (Kunci Derivasi Dasar)	Tidak	
Algoritme	Menentukan algoritma kunci yang mendasarinya. Kami mendukung: • T (TDES) • H (HMAC) • A (AES)	Tidak	Kami mengekspor nilai ini apa adanya.
Penggunaan Kunci	Mendefinisikan operasi yang diizinkan, seperti: • Hasilkan dan Verifikasi (C) • Encrypt/Decrypt/Wrap/Unwrap(B)	Ya*	
Versi Kunci	Menunjukkan nomor versi untuk penggantian/rotasi kunci. Default ke 00 jika tidak ditentukan.	Ya - Dapat menambahkan	

Atribut Blok Kunci	Tujuan	Bisakah Anda memodifikasi selama ekspor?	Catatan
Ekspor Kunci	Mengontrol apakah kunci dapat diekspor: • N - Tidak Ada Ekspor • E - Ekspor sesuai dengan X9.24 (blok kunci) • S - Ekspor di bawah blok kunci atau format blok non-kunci	Ya*	
Blok Kunci Opsional	Ya - Dapat menambahkan	Blok kunci opsional adalah name/value pasangan yang terikat secara kriptografis ke kunci. Misalnya, KeySet ID untuk kunci DUKPT. Kami secara otomatis menghitung jumlah blok, panjang setiap blok, dan blok padding (PB) berdasarkan input name/value pasangan Anda.	

*Saat memodifikasi nilai, nilai baru Anda harus lebih ketat daripada nilai saat ini dalam AWS Kriptografi Pembayaran. Contoh:

- Jika mode penggunaan kunci saat ini adalah Generate=True, Verify=True, Anda dapat mengubahnya menjadi Generate=True, Verify=False
- Jika kunci sudah disetel ke tidak dapat diekspor, Anda tidak dapat mengubahnya menjadi ekspor

Saat Anda mengekspor kunci, kami secara otomatis menerapkan nilai saat ini dari kunci yang diekspor. Namun, Anda mungkin ingin memodifikasi atau menambahkan nilai-nilai tersebut sebelum mengirim ke sistem penerima. Berikut adalah beberapa skenario umum:

- Saat mengekspor kunci ke terminal pembayaran, atur ekspornya `Not Exportable` karena terminal biasanya hanya mengimpor kunci dan tidak boleh mengekspornya.
- Saat Anda perlu meneruskan metadata kunci terkait ke sistem penerima, gunakan header opsional TR-31 untuk mengikat metadata secara kriptografis ke kunci alih-alih membuat muatan khusus.
- Atur Versi Kunci menggunakan `KeyVersion` bidang untuk melacak rotasi kunci.

TR-31/X9.143 mendefinisikan header umum, tetapi Anda dapat menggunakan header lain selama mereka memenuhi parameter Kriptografi AWS Pembayaran dan sistem penerima Anda dapat menerimanya. Untuk informasi selengkapnya tentang header blok kunci selama ekspor, lihat Header Blok Kunci di Panduan API.

Berikut adalah contoh mengekspor kunci BDK (misalnya, ke KIF) dengan spesifikasi berikut:

- Versi kunci: 02
- KeyExportability: TIDAK DAPAT DIEKSPOR
- KeySetID: 00ABCDEFAB (00 menunjukkan kunci TDES, ABCDEFABCD adalah kunci awal)

Karena kita tidak menentukan mode penggunaan kunci, kunci ini mewarisi mode penggunaan dari arn:aws:payment-cryptography:us-east- 2:111122223333:key/5rplquuwozodpwsp (= true). DeriveKey

> **Note**
>
> Bahkan ketika Anda menyetel exportability ke Not Exportable dalam contoh ini, KIF masih dapat:
>
> - Turunkan kunci seperti IPEK/IK yang digunakan dalam DUKPT
> - Ekspor kunci turunan ini untuk dipasang di perangkat

Ini secara khusus diizinkan oleh standar.

```
$ aws payment-cryptography export-key \
    --key-material='{"Tr31KeyBlock": { \
    "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza", \
    "KeyBlockHeaders": { \
    "KeyModesOfUse": { \
    "Derive": true}, \
    "KeyExportability": "NON_EXPORTABLE", \
    "KeyVersion": "02", \
    "OptionalBlocks": { \
    "BI": "00ABCDEFABCD"}}} \
    }' \
    --export-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp
```

```
{
"WrappedKey": {
    "WrappedKeyMaterialFormat": "TR31_KEY_BLOCK",
    "KeyMaterial": "EXAMPLE_KEY_MATERIAL_TR31",
    "KeyCheckValue": "A4C9B3",
    "KeyCheckValueAlgorithm": "ANSI_X9_24"
    }
}
```

Header Umum

X9.143 mendefinisikan header tertentu untuk kasus penggunaan umum. Dengan pengecualian header HM (HMAC Hash), Kriptografi AWS Pembayaran tidak mengurai atau menggunakan header ini.

Nama Header	Tujuan	Validasi Khas	Catatan
BI	Pengidentifikasi Kunci Derivasi Dasar untuk DUKPT	2 karakter hex (00 untuk TDES, 11 untuk AES) kemudian 10 karakter hex	Berisi (ID BDK, untuk AES DUKPT) atau Pengenal Set Kunci (KSI, untuk TDES

Nama Header	Tujuan	Validasi Khas	Catatan
		untuk TDES KSI atau 8 karakter hex untuk BDK ID (AES DUKPT).	DUKPT). Dapat digunakan saat menukar ID BDK atau KSI, tetapi tidak perlu bertukar data lain yang terdapat dalam blok IK dan KS. Biasanya BI digunakan saat mentransmisikan ke KIF sedangkan IK atau KS digunakan saat menyuntikkan ke terminal itu sendiri.
HM	Menentukan jenis hash untuk operasi HMAC	• 10 — SHA-1 • 20 — SHA-224 • 21 — SHA-256 • 22 — SHA-384 • 23 — SHA-512 • 24 — SHA-512/224 • 25 — SHA-512/256 • 30 — SHA3 -224 • 31 — SHA3 -256 • 32 — SHA3 -384 • 33 — SHA3 -512 • 40 — SHAKE128 • 41 — SHAKE256	Layanan secara otomatis mengisi bidang ini pada ekspor dan akan menguraikannya pada impor. Jenis hash yang tidak didukung oleh layanan seperti SHAKE128 dapat diimpor tetapi mungkin tidak dapat digunakan untuk fungsi kriptografi.

Nama Header	Tujuan	Validasi Khas	Catatan
IK	Nomor Seri Kunci Awal untuk AES DUKPT	16 karakter hex	Nilai ini digunakan untuk membuat instance penggunaan kunci DUKPT Awal pada perangkat penerima dan mengidentifikasi Kunci Awal yang berasal dari BDK. Bidang ini biasanya berisi data derivasi tetapi tidak ada penghitung. Gunakan KS untuk TDES DUKPT.
KS	Nomor Seri Kunci Awal untuk TDES DUKPT	20 karakter hex	Nilai ini digunakan untuk membuat instance penggunaan kunci DUKPT Awal pada perangkat penerima dan mengidentifikasi Kunci Awal yang berasal dari BDK. Bidang ini biasanya berisi data derivasi+ nilai penghitung nol. Gunakan IK untuk AES DUKPT.

Nama Header	Tujuan	Validasi Khas	Catatan
KP	KCV dari kunci pembungkus	2 karakter hex mewakili metode KCV (00 untuk metode X9.24 dan 01 untuk metode CMAC). Diikuti oleh nilai KCV yang biasanya 6 karakter hex. Misalnya 010 FA329 mewakili KCV dari 0 FA329 dihitung menggunakan 01 (CMAC) metode.	Nilai ini digunakan untuk membuat instance penggunaan kunci DUKPT Awal pada perangkat penerima dan mengidentifikasi Kunci Awal yang berasal dari BDK. Bidang ini biasanya berisi data derivasi+ nilai penghitung nol. Gunakan IK untuk AES DUKPT.
PB	Blok bantalan	karakter ASCII acak yang dapat dicetak	Layanan secara otomatis mengisi bidang ini pada ekspor untuk memastikan header opsional adalah kelipatan panjang blok enkripsi

Ekspor kunci asimetris (RSA)

Untuk mengekspor kunci publik dalam bentuk sertifikat, gunakan get-public-key-certificate perintah. Perintah ini mengembalikan:

- Sertifikat
- Sertifikat root

Kedua sertifikat dalam pengkodean base64.

> **Note**
>
> Operasi ini tidak idempoten—panggilan berikutnya mungkin menghasilkan sertifikat yang berbeda bahkan saat menggunakan kunci dasar yang sama.

Example

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

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

Topik Lanjutan

Bagian ini mencakup skenario pertukaran kunci lanjutan dan konfigurasi.

Topik

- Bawa Otoritas Sertifikat Anda Sendiri (BYOCA)

Bawa Otoritas Sertifikat Anda Sendiri (BYOCA)

Secara default, ketika sertifikat kunci publik diperlukan untuk kunci asimetris (RSA, ECC) yang dibuat dalam layanan, sertifikat ini dikeluarkan oleh Kriptografi AWS Pembayaran dan otoritas sertifikat unik akun (CA). Ini dimaksudkan untuk memudahkan penggunaan X.509 tanpa beban mengidentifikasi atau menyiapkan CA atau mengelola Permintaan Penandatanganan Sertifikat (CSR).

AWS Kriptografi Pembayaran juga menyediakan kemampuan untuk menggunakan CA Anda sendiri ketika diperlukan karena alasan kebijakan atau kepatuhan.

Ikhtisar

Fitur BYOCA memungkinkan Anda untuk menggunakan Otoritas Sertifikat Anda sendiri di mana pun sertifikat digunakan, termasuk impor/ekspor TR-34, RSA Unwrap, dan transfer kunci berbasis

ECDH. Ini berguna ketika Anda perlu mempertahankan rantai sertifikat yang konsisten di seluruh organisasi Anda atau ketika bekerja dengan mitra yang memerlukan sertifikat CA tertentu. Contoh berikut menunjukkan alur kerja BYOCA menggunakan TR-34 kunci ekspor.

Tiga perbedaan utama dibandingkan dengan aliran ekspor TR-34 standar adalah:

1. Kunci RSA penandatanganan secara eksplisit dibuat menggunakan. CreateKey Sebelumnya, itu secara implisit dibuat melalui. GetParametersForExport
2. API baru akan GetCertificateSigningRequestmembuat Certificate Signing Request (CSR) yang dapat ditandatangani oleh CA eksternal Anda.
3. ExportKeyAPI diperluas untuk memungkinkan sertifikat disediakan saat runtime. Sebelumnya, ini secara implisit disediakan oleh`import-token`, yang menjadi bidang opsional.

⚠ Pertimbangan Penting

- Contoh-contoh ini menggunakan kunci RSA-2048 dan membungkus kunci TDES-2KEY. Saat mengekspor AES-128, pastikan semua kunci adalah RSA-3072 atau RSA-4096.
- Kesalahan yang paling umum adalah bahwa kunci diwakili oleh `SigningKeyIdentifier` dan `SigningKeyCertificate` tidak cocok.

Alur Kerja BYOCA

Langkah-langkah berikut menunjukkan alur kerja BYOCA lengkap untuk ekspor TR-34.

Langkah-langkah

- Langkah 1: Buat Kunci RSA
- Langkah 2: Hasilkan Permintaan Penandatanganan Sertifikat
- Langkah 3: Tinjau CSR (Opsional)
- Langkah 4: Tanda tangani CSR dengan Otoritas Sertifikat
- Langkah 5: Impor Sertifikat CA
- Langkah 6: Dapatkan Sertifikat Enkripsi KRD
- Langkah 7: Kunci Ekspor dengan BYOCA

Langkah 1: Buat Kunci RSA

Pertama, buat Pasangan Kunci RSA yang pada akhirnya akan menjadi Sertifikat Penandatanganan KDH. Anda dapat menambahkan tag untuk mengidentifikasi tujuan kunci.

Example Buat Kunci RSA untuk Penandatanganan

```
$ aws payment-cryptography create-key --exportable \
    --key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE,KeyClass=ASYMMETRI
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xgmq6fs6uow736uc",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyAlgorithm": "RSA_2048",
            "KeyModesOfUse": {
                "Sign": true
            }
        },
        "KeyCheckValue": "41E3723C",
        "KeyCheckValueAlgorithm": "SHA_1",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY"
    }
}
```

Catat `KeyArn` karena Anda akan membutuhkannya di langkah berikutnya.

Langkah 2: Hasilkan Permintaan Penandatanganan Sertifikat

Buat Permintaan Penandatanganan Sertifikat (CSR) untuk ditandatangani oleh CA eksternal Anda menggunakan GetCertificateSigningRequestAPI. Outputnya adalah file PEM yang dikodekan base64. Jika Anda base64 memecahkan kode konten dan menyimpannya, Anda akan memiliki CSR yang valid dalam format PEM.

Example Hasilkan CSR

```
$ aws payment-cryptography-data get-certificate-signing-request \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xgmq6fs6uow736uc \
    --signing-algorithm SHA512 \
    --certificate-subject '{
        "CommonName": "MyCertificateAWSUSEAST",
        "Organization": "Amazon",
        "OrganizationUnit": "PaymentCryptography",
        "Country": "US",
        "StateOrProvince": "Virginia",
        "City": "Arlington"
    }'
```

```
{
    "CertificateSigningRequest": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..."
}
```

`CertificateSigningRequest`Kolom berisi CSR berenkode base64 yang akan Anda kirim ke CA untuk ditandatangani.

Langkah 3: Tinjau CSR (Opsional)

Anda dapat secara opsional menggunakan OpenSSL untuk meninjau konten CSR dan memastikannya valid dan seperti yang diharapkan.

Example Tinjau CSR dengan OpenSSL

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d | openssl req -
text
```

Langkah 4: Tanda tangani CSR dengan Otoritas Sertifikat

Setelah membuat CSR, Anda harus menandatanganinya oleh Certificate Authority (CA). Dalam lingkungan produksi, Anda biasanya akan menggunakan AWS Private CA atau infrastruktur CA yang didirikan organisasi Anda. Untuk tujuan pengujian, Anda dapat menggunakan OpenSSL untuk membuat sertifikat yang ditandatangani sendiri.

Menggunakan AWS Private CA

Untuk menandatangani CSR menggunakan AWS Private CA, pertama-tama dekode CSR yang dikodekan base64 dan simpan ke file, lalu gunakan API. IssueCertificate

Example Menandatangani CSR dengan AWS Private CA

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > csr.pem

$ aws acm-pca issue-certificate \
    --certificate-authority-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012 \
    --csr fileb://csr.pem \
    --signing-algorithm SHA256WITHRSA \
    --validity Value=365,Type=DAYS
```

```
{
    "CertificateArn": "arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012/certificate/abcdef1234567890"
}
```

Kemudian ambil sertifikat yang ditandatangani:

Example Ambil Sertifikat yang Ditandatangani

```
$ aws acm-pca get-certificate \
    --certificate-authority-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012 \
    --certificate-arn arn:aws:acm-pca:us-east-1:111122223333:certificate-
authority/12345678-1234-1234-1234-123456789012/certificate/abcdef1234567890
```

```
{
    "Certificate": "-----BEGIN CERTIFICATE-----\nMIID...\n-----END CERTIFICATE-----",
    "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIID...\n-----END
 CERTIFICATE-----"
}
```

Simpan konten sertifikat untuk digunakan dalam langkah ekspor. Anda harus mengkodekannya dengan base64 saat menyediakannya ke API. `ExportKey`

Menggunakan OpenSSL untuk Pengujian

Untuk tujuan pengujian, Anda dapat menggunakan OpenSSL untuk membuat CA yang ditandatangani sendiri dan menandatangani CSR. Pertama, buat kunci pribadi CA dan sertifikat yang ditandatangani sendiri:

Example Buat Test CA dengan OpenSSL

```
$ # Generate CA private key
openssl genrsa -out ca-key.pem 4096

$ # Create self-signed CA certificate
openssl req -new -x509 -days 3650 -key ca-key.pem -out ca-cert.pem \
    -subj "/C=US/ST=Virginia/L=Arlington/O=TestOrg/CN=Test CA"
```

Kemudian dekode CSR dari langkah sebelumnya dan tandatangani dengan CA pengujian Anda:

Example Menandatangani CSR dengan OpenSSL

```
$ # Decode the base64-encoded CSR
echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > csr.pem

$ # Sign the CSR with the CA
openssl x509 -req -in csr.pem -CA ca-cert.pem -CAkey ca-key.pem \
    -CAcreateserial -out signed-cert.pem -days 365 -sha512
```

```
Certificate request self-signature ok
subject=C=US, ST=Virginia, L=Arlington, O=Amazon, OU=PaymentCryptography,
 CN=MyCertificateAWSUSEAST
```

Sertifikat yang ditandatangani sekarang masuk`signed-cert.pem`. Anda harus meng-base64-encode sertifikat ini saat memberikannya ke API: `ExportKey`

Example Base64 Mengkodekan Sertifikat yang Ditandatangani

```
$ cat signed-cert.pem | base64 -w 0
```

Langkah 5: Impor Sertifikat CA

Setiap CA yang digunakan harus dipercaya terlebih dahulu untuk mencegah sertifikat arbitrer digunakan. Impor sertifikat root CA eksternal Anda menggunakan ImportKeyAPI. Jika menggunakan

CA perantara, panggil `import-key` lagi tetapi tentukan `TrustedPublicKey` alih-alih `RootCertificatePublicKey` dan tentukan root CA ARN.

Example Impor Sertifikat Root CA

```
$ 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": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t..."
    }
}'
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xivpaqy7qbbm7cdw",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096",
            "KeyModesOfUse": {
                "Verify": true
            }
        },
        "Enabled": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL"
    }
}
```

Perhatikan CA `KeyArn` untuk digunakan dalam langkah ekspor.

Langkah 6: Dapatkan Sertifikat Enkripsi KRD

Dalam contoh ini, kami mengimpor kembali ke Kriptografi AWS Pembayaran, jadi kami memanggil layanan untuk menerima sertifikat kunci publik KRD menggunakan API. GetParametersForImport

Dalam skenario nyata, ini akan disediakan oleh sistem lain, seperti HSM, ATM, terminal pembayaran atau sistem manajemen terminal pembayaran.

Example Dapatkan Parameter untuk Impor

```
$ aws payment-cryptography-data get-parameters-for-import \
    --key-material-type "TR34_KEY_BLOCK" \
    --wrapping-key-algorithm RSA_2048
```

```
{
    "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "WrappingKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "WrappingKeyAlgorithm": "RSA_2048",
    "ImportToken": "import-token-v2rxpl6drxeptn7w",
    "ParametersValidUntilTimestamp": "2025-11-01T18:45:31.271000-07:00"
}
```

Langkah 7: Kunci Ekspor dengan BYOCA

Terakhir, ekspor kunci menggunakan TR-34 dengan sertifikat bertanda CA Anda sendiri menggunakan API. ExportKey Berikan sertifikat penandatanganan yang ditandatangani oleh CA eksternal Anda.

Example TR-34 Ekspor dengan BYOCA

```
$ aws payment-cryptography-data export-key \
    --export-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
iox73p5f4c4yjiod \
    --key-material '{
        "Tr34KeyBlock": {
            "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-
cryptography:us-east-1:111122223333:key/j625deyfqlwctu57",
            "SigningKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/xgmq6fs6uow736uc",
            "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
            "KeyBlockFormat": "X9_TR34_2012",
            "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t..."
        }
    }'
```

```
{
```

```
    "WrappedKey": {
        "WrappedKeyMaterialFormat": "TR34_KEY_BLOCK",
        "KeyMaterial": "3082055A06092A864886F70D010702A082054B30820547...",
        "KeyCheckValue": "3DCA31",
        "KeyCheckValueAlgorithm": "ANSI_X9_24"
    }
}
```

Blok kunci yang diekspor sekarang dapat diimpor oleh sistem penerima menggunakan proses impor TR-34 standar.

Catatan Tambahan

- Contoh-contoh ini ditampilkan menggunakan AWS CLI. Fungsionalitas yang sama tersedia di semua AWS SDKs termasuk Java, Python, Go, dan Rust.
- Jika Anda menguji dengan CA yang ditandatangani sendiri, Anda dapat menggunakan OpenSSL untuk membuat CA pengujian dan menandatangani CSR. Dalam produksi, gunakan infrastruktur CA yang sudah mapan organisasi Anda.

Menggunakan alias

Alias adalah nama yang ramah untuk kunci Kriptografi AWS Pembayaran. Misalnya, alias memungkinkan Anda merujuk ke kunci sebagai `alias/test-key` ganti. `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`

Anda dapat menggunakan alias untuk mengidentifikasi kunci di sebagian besar operasi manajemen kunci (bidang kontrol), dan dalam operasi kriptografi (bidang data).

Anda juga dapat mengizinkan dan menolak akses ke kunci Kriptografi AWS Pembayaran berdasarkan alias mereka tanpa mengedit kebijakan atau mengelola hibah. Fitur ini merupakan bagian dari dukungan layanan untuk kontrol akses berbasis atribut (ABAC).

Sebagian besar kekuatan alias berasal dari kemampuan Anda untuk mengubah kunci yang terkait dengan alias kapan saja. Alias dapat membuat kode Anda lebih mudah ditulis dan dipelihara. Misalnya, Anda menggunakan alias untuk merujuk ke kunci Kriptografi AWS Pembayaran tertentu dan Anda ingin mengubah kunci Kriptografi AWS Pembayaran. Dalam hal ini, cukup kaitkan alias dengan kunci yang berbeda. Anda tidak perlu mengubah kode atau konfigurasi aplikasi Anda.

Alias juga mempermudah menggunakan kembali kode yang sama di Wilayah AWS berbeda. Buat alias dengan nama yang sama di beberapa Wilayah dan kaitkan setiap alias dengan kunci Kriptografi

AWS Pembayaran di Wilayahnya. Ketika kode berjalan di setiap Wilayah, alias mengacu pada kunci Kriptografi AWS Pembayaran terkait di Wilayah tersebut.

Anda dapat membuat alias untuk kunci Kriptografi AWS Pembayaran dengan menggunakan API. `CreateAlias`

API Kriptografi AWS Pembayaran memberikan kontrol penuh atas alias di setiap akun dan Wilayah. API mencakup operasi untuk membuat alias (CreateAlias), melihat nama alias dan keYarn yang ditautkan (list-alias), mengubah kunci Kriptografi AWS Pembayaran yang terkait dengan alias (update-alias), dan menghapus alias (delete-alias).

Topik

Tentang alias

Pelajari cara kerja alias dalam Kriptografi AWS Pembayaran.

Alias adalah sumber daya independen AWS

> Alias bukan milik kunci Kriptografi AWS Pembayaran. Tindakan yang Anda lakukan pada alias tidak memengaruhi kunci terkait. Anda dapat membuat alias untuk kunci Kriptografi AWS Pembayaran dan kemudian memperbarui alias sehingga dikaitkan dengan kunci Kriptografi AWS Pembayaran yang berbeda. Anda bahkan dapat menghapus alias tanpa efek apa pun pada kunci Kriptografi AWS Pembayaran terkait. Jika Anda menghapus kunci Kriptografi AWS Pembayaran, semua alias yang terkait dengan kunci tersebut akan menjadi tidak ditetapkan.
>
> Jika Anda menentukan alias sebagai sumber daya dalam kebijakan IAM, kebijakan mengacu pada alias, bukan ke kunci Kriptografi AWS Pembayaran terkait.

Setiap alias memiliki nama yang ramah

> Saat Anda membuat alias, Anda menentukan nama alias yang diawali oleh. `alias/` Sebagai contoh `alias/test_1234`

Setiap alias dikaitkan dengan satu kunci Kriptografi AWS Pembayaran pada satu waktu

> Alias dan kunci Kriptografi AWS Pembayarannya harus berada di akun dan Wilayah yang sama.

Kunci Kriptografi AWS Pembayaran dapat dikaitkan dengan lebih dari satu alias secara bersamaan, tetapi setiap alias hanya dapat dipetakan ke satu kunci

Misalnya, `list-aliases` output ini menunjukkan bahwa `alias/sampleAlias1` alias dikaitkan dengan tepat satu kunci Kriptografi AWS Pembayaran target, yang diwakili oleh properti. `KeyArn`

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

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

Beberapa alias dapat dikaitkan dengan kunci Kriptografi AWS Pembayaran yang sama

Misalnya, Anda dapat mengaitkan `alias/sampleAlias1;` dan `alias/sampleAlias2` alias dengan kunci yang sama.

```
$ 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"
```

```
            }
        ]
    }
```

Alias harus unik untuk akun dan Wilayah tertentu

Misalnya, Anda hanya dapat memiliki satu alias `alias/sampleAlias1` di setiap akun dan Wilayah. Alias peka huruf besar/kecil, tetapi kami merekomendasikan untuk tidak menggunakan alias yang hanya berbeda dalam kapitalisasi karena dapat rentan terhadap kesalahan. Anda tidak dapat mengubah nama alias. Namun, Anda dapat menghapus alias dan membuat alias baru dengan nama yang diinginkan.

Anda dapat membuat alias dengan nama yang sama di Wilayah yang berbeda

Misalnya, Anda dapat memiliki alias `alias/sampleAlias2` di AS Timur (Virginia N.) dan alias `alias/sampleAlias2` di AS Barat (Oregon). Setiap alias akan dikaitkan dengan kunci Kriptografi AWS Pembayaran di Wilayahnya. Jika kode Anda merujuk pada nama alias seperti `alias/finance-key`, Anda dapat menjalankannya di beberapa Wilayah. Di setiap Wilayah, ia menggunakan alias/SampleAlias2 yang berbeda. Lihat perinciannya di Menggunakan alias dalam aplikasi Anda.

Anda dapat mengubah kunci Kriptografi AWS Pembayaran yang terkait dengan alias

Anda dapat menggunakan `UpdateAlias` operasi untuk mengaitkan alias dengan kunci Kriptografi AWS Pembayaran yang berbeda. Misalnya, jika `alias/sampleAlias2` alias dikaitkan dengan kunci Kriptografi `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` AWS Pembayaran, Anda dapat memperbaruinya sehingga dikaitkan dengan kunci. `arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi`

> **Warning**
>
> AWS Kriptografi Pembayaran tidak memvalidasi bahwa kunci lama dan baru memiliki semua atribut yang sama seperti penggunaan kunci. Memperbarui dengan jenis kunci yang berbeda dapat mengakibatkan masalah dalam aplikasi Anda.

Beberapa kunci tidak memiliki alias

Alias adalah fitur opsional dan tidak semua kunci akan memiliki alias kecuali Anda memilih untuk mengoperasikan lingkungan Anda dengan cara ini. Kunci dapat dikaitkan dengan Alias

menggunakan `create-alias` perintah. Selain itu, Anda dapat menggunakan operasi update-alias untuk mengubah kunci Kriptografi AWS Pembayaran yang terkait dengan alias dan operasi hapus-alias untuk menghapus alias. Akibatnya, beberapa kunci Kriptografi AWS Pembayaran mungkin memiliki beberapa alias, dan beberapa mungkin tidak memilikinya.

Memetakan kunci ke alias

Anda dapat memetakan kunci (diwakili oleh ARN) ke satu atau lebih alias menggunakan perintah. `create-alias` Perintah ini tidak idempoten - untuk memperbarui alias, gunakan perintah 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"
    }
}
```

Menggunakan alias dalam aplikasi Anda

Anda dapat menggunakan alias untuk mewakili kunci Kriptografi AWS Pembayaran dalam kode aplikasi Anda. `key-identifier`Parameter dalam operasi data Kriptografi AWS Pembayaran serta operasi lain seperti List Keys menerima nama alias atau 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}
```

Saat menggunakan alias ARN, ingatlah bahwa alias pemetaan ke AWS kunci Kriptografi Pembayaran didefinisikan dalam akun yang memiliki kunci Kriptografi Pembayaran dan mungkin AWS berbeda di setiap Wilayah.

Salah satu penggunaan alias yang paling kuat adalah pada aplikasi yang berjalan dalam beberapa Wilayah AWS.

Anda dapat membuat versi aplikasi yang berbeda di setiap Wilayah atau menggunakan kamus, konfigurasi, atau pernyataan sakelar untuk memilih kunci Kriptografi AWS Pembayaran yang tepat untuk setiap Wilayah. Tetapi mungkin lebih mudah untuk membuat alias dengan nama alias yang sama di setiap Wilayah. Ingat bahwa nama alias peka terhadap huruf besar-kecil.

Terkait APIs

Tanda

Tag adalah pasangan kunci dan nilai yang bertindak sebagai metadata untuk mengatur kunci Kriptografi AWS Pembayaran Anda. Mereka dapat digunakan untuk mengidentifikasi kunci secara fleksibel atau mengelompokkan satu atau lebih kunci bersama-sama.

Dapatkan kunci

Kunci Kriptografi AWS Pembayaran mewakili satu unit bahan kriptografi dan hanya dapat digunakan untuk operasi kriptografi untuk layanan ini. GetKeys API mengambil KeyIdentifier sebagai input dan mengembalikan metadata kunci termasuk atribut, status, dan stempel waktu, tetapi tidak mengembalikan materi kunci kriptografi yang sebenarnya.

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"
  }
}
```

Dapatkan publik key/certificate yang terkait dengan key pair

Get Public Key/Certificate mengembalikan kunci publik yang ditunjukkan oleh`KeyArn`. Ini bisa menjadi bagian kunci publik dari key pair yang dihasilkan pada AWS Payment Cryptography atau public key yang sebelumnya diimpor. Kasus penggunaan yang paling umum adalah menyediakan kunci publik ke layanan luar yang akan mengenkripsi data. Data tersebut kemudian dapat diteruskan ke aplikasi yang memanfaatkan Kriptografi AWS Pembayaran dan data dapat didekripsi menggunakan kunci pribadi yang diamankan dalam Kriptografi Pembayaran. AWS

Layanan mengembalikan kunci publik sebagai sertifikat publik. Hasil API berisi CA dan sertifikat kunci publik. Kedua elemen data dikodekan base64.

> **Note**
>
> Sertifikat publik yang dikembalikan dimaksudkan untuk berumur pendek dan tidak dimaksudkan untuk menjadi idempoten. Anda mungkin menerima sertifikat yang berbeda pada setiap panggilan API bahkan kunci publik itu sendiri tidak berubah.

Example

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

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

Tombol penandaan

Dalam Kriptografi AWS Pembayaran, Anda dapat menambahkan tag ke kunci Kriptografi AWS Pembayaran saat Anda membuat kunci, dan menandai atau menghapus tag kunci yang ada kecuali mereka menunggu penghapusan. Tanda adalah opsional, tetapi tanda bisa sangat berguna.

Untuk informasi umum tentang tag, termasuk praktik terbaik, strategi penandaan, serta format dan sintaks tag, lihat Menandai AWS sumber daya di. Referensi Umum Amazon Web

Topik

Tentang tag dalam Kriptografi AWS Pembayaran

Tag adalah label metadata opsional yang dapat Anda tetapkan (atau AWS dapat ditetapkan) ke sumber daya. AWS Setiap tanda terdiri dari kunci tanda dan nilai tanda, keduanya adalah string peka huruf besar/kecil. Nilai tanda bisa berupa string kosong (null). Setiap tag pada sumber daya harus memiliki kunci tag yang berbeda, tetapi Anda dapat menambahkan tag yang sama ke beberapa AWS sumber daya. Setiap sumber daya dapat memiliki hingga 50 tanda yang dibuat pengguna.

Jangan sertakan informasi rahasia atau sensitif dalam kunci tag atau nilai tag. Tag dapat diakses oleh banyak orang Layanan AWS, termasuk penagihan.

Dalam Kriptografi AWS Pembayaran, Anda dapat menambahkan tag ke kunci saat Anda membuat kunci, dan menandai atau menghapus tag kunci yang ada kecuali mereka menunggu penghapusan. Anda tidak dapat menandai alias. Tanda adalah opsional, tetapi tanda bisa sangat berguna.

Misalnya, Anda dapat menambahkan `"Project"="Alpha"` tag ke semua kunci Kriptografi AWS Pembayaran dan bucket Amazon S3 yang Anda gunakan untuk proyek Alpha. Contoh lain adalah menambahkan `"BIN"="20130622"` tag ke semua kunci yang terkait dengan nomor identifikasi bank tertentu (BIN).

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

Untuk informasi umum tentang tag, termasuk format dan sintaks, lihat Menandai AWS sumber daya di. Referensi Umum Amazon Web

Tanda membantu Anda melakukan hal berikut:

- Identifikasi dan atur AWS sumber daya Anda. Banyak AWS layanan mendukung penandaan, sehingga Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait. Misalnya, Anda dapat menetapkan tag yang sama ke kunci Kriptografi AWS Pembayaran dan volume atau rahasia Amazon Elastic Block Store (Amazon EBS) EBS). AWS Secrets Manager Anda juga dapat menggunakan tag untuk mengidentifikasi kunci untuk otomatisasi.
- Lacak AWS biaya Anda. Saat menambahkan tag ke AWS sumber daya, buat AWS laporan alokasi biaya dengan penggunaan dan biaya yang dikumpulkan berdasarkan tag. Anda dapat menggunakan fitur ini untuk melacak biaya Kriptografi AWS Pembayaran untuk proyek, aplikasi, atau pusat biaya.

  Untuk informasi selengkapnya tentang penggunaan tanda untuk alokasi biaya, lihat Menggunakan Tanda Alokasi Biaya dalam Panduan Pengguna AWS Billing . Untuk informasi tentang aturan untuk kunci tanda dan nilai tanda, lihat Pembatasan Tanda Ditetapkan Pengguna di AWS Billing Panduan Pengguna.
- Kontrol akses ke AWS sumber daya Anda. Mengizinkan dan menolak akses ke kunci berdasarkan tag mereka adalah bagian dari dukungan Kriptografi AWS Pembayaran untuk kontrol akses berbasis atribut (ABAC). Untuk informasi tentang mengendalikan akses ke Kriptografi AWS Pembayaran berdasarkan tag mereka, lihatOtorisasi berdasarkan tag Kriptografi AWS Pembayaran. Untuk informasi umum selengkapnya tentang penggunaan tag untuk mengontrol akses ke AWS sumber daya, lihat Mengontrol Akses ke AWS Sumber Daya Menggunakan Tag Sumber Daya di Panduan Pengguna IAM.

AWS Kriptografi Pembayaran menulis entri ke AWS CloudTrail log Anda saat Anda menggunakan TagResource, UntagResource, atau ListTagsForResource operasi.

Melihat tag kunci di konsol

Untuk melihat tag di konsol, Anda memerlukan izin penandaan pada kunci dari kebijakan IAM yang menyertakan kunci. Anda memerlukan izin ini selain izin untuk melihat kunci di konsol.

Mengelola tag kunci dengan operasi API

Anda dapat menggunakan AWS Payment Cryptography API untuk menambahkan, menghapus, dan mencantumkan tag untuk kunci yang Anda kelola. Contoh-contoh ini menggunakan AWS Command Line Interface (AWS CLI), tetapi Anda dapat menggunakan bahasa pemrograman yang didukung. Anda tidak dapat menandai Kunci yang dikelola AWS.

Untuk menambahkan, mengedit, melihat, dan menghapus tag untuk kunci, Anda harus memiliki izin yang diperlukan. Lihat perinciannya di Pengontrolan akses ke tanda.

Topik

- CreateKey: Tambahkan tag ke kunci baru
- TagResource: Menambahkan atau mengubah tag untuk kunci
- ListResourceTags: Dapatkan tag untuk kunci
- UntagResource: Hapus tag dari kunci

CreateKey: Tambahkan tag ke kunci baru

Anda dapat menambahkan tag saat membuat kunci. Untuk menentukan tag, gunakan `Tags` parameter CreateKeyoperasi.

Untuk menambahkan tag saat membuat kunci, pemanggil harus memiliki `payment-cryptography:TagResource` izin dalam kebijakan IAM. Minimal, izin harus mencakup semua kunci di akun dan Wilayah. Lihat perinciannya di Pengontrolan akses ke tanda.

Nilai dari parameter `Tags` dari `CreateKey` adalah kumpulan kunci tanda peka huruf besar/kecil dan pasangan nilai kunci. Setiap tag pada kunci harus memiliki nama tag yang berbeda. Nilai tanda bisa berupa string kosong atau null.

Misalnya, AWS CLI perintah berikut membuat kunci enkripsi simetris dengan `Project:Alpha` tag. Saat menentukan lebih dari satu pasangan nilai kunci, gunakan spasi untuk memisahkan setiap pasangan.

```
$ 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"}]'
```

Ketika perintah ini berhasil, ia mengembalikan `Key` objek dengan informasi tentang kunci baru. Namun, `Key` tidak termasuk tanda. Untuk mendapatkan tag, gunakan ListResourceTagsoperasi.

TagResource: Menambahkan atau mengubah tag untuk kunci

TagResourceOperasi menambahkan satu atau lebih tag ke kunci. Anda tidak dapat menggunakan operasi ini untuk menambah atau mengedit tanda dalam Akun AWS berbeda.

Untuk menambahkan tanda, tentukan kunci tanda baru dan nilai tanda. Untuk mengedit tanda, tentukan kunci tanda yang sudah ada dan nilai tanda baru. Setiap tag pada kunci harus memiliki kunci tag yang berbeda. Nilai tanda bisa berupa string kosong atau null.

Misalnya, perintah berikut menambahkan **`UseCase`** dan **`BIN`** tag ke kunci contoh.

```
$ 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"}]'
```

Ketika perintah ini berhasil, maka tidak mengembalikan output apa pun. Untuk melihat tag pada kunci, gunakan ListResourceTagsoperasi.

Anda juga dapat menggunakan TagResource untuk mengubah nilai tanda dari tanda yang ada. Untuk mengganti nilai tanda, tentukan kunci tanda yang sama dengan nilai yang berbeda. Tag yang tidak tercantum dalam perintah modifikasi tidak diubah atau dihapus.

Sebagai contoh, perintah ini mengubah nilai tanda `Project` dari `Alpha` ke `Noe`.

Perintah akan mengembalikan http/200 tanpa konten. Untuk melihat perubahan Anda, gunakan `ListTagsForResource`

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

ListResourceTags: Dapatkan tag untuk kunci

ListResourceTagsOperasi mendapatkan tag untuk kunci. Parameter `ResourceArn` (keYarn atau KeyAlias) diperlukan. Anda tidak dapat menggunakan operasi ini untuk melihat tag pada kunci yang berbeda Akun AWS.

Misalnya, perintah berikut mendapatkan tag untuk kunci contoh.

```
$ 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: Hapus tag dari kunci

UntagResourceOperasi menghapus tag dari kunci. Untuk mengidentifikasi tanda yang akan dihapus, tentukan kunci tanda. Anda tidak dapat menggunakan operasi ini untuk menghapus tag dari kunci yang berbeda Akun AWS.

Ketika berhasil, operasi `UntagResource` tidak mengembalikan output apa pun. Juga, jika kunci tag yang ditentukan tidak ditemukan pada kunci, itu tidak membuang pengecualian atau mengembalikan respons. Untuk mengonfirmasi bahwa operasi berhasil, gunakan ListResourceTagsoperasi.

Misalnya, perintah ini menghapus **`Purpose`** tag dan nilainya dari kunci yang ditentukan.

```
$ aws payment-cryptography untag-resource \
```

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

Pengontrolan akses ke tanda

Untuk menambah, melihat, dan menghapus tag dengan menggunakan API, prinsipal memerlukan izin penandaan dalam kebijakan IAM.

Anda juga dapat membatasi izin ini dengan menggunakan kunci kondisi AWS global untuk tag. Dalam Kriptografi AWS Pembayaran, kondisi ini dapat mengontrol akses ke operasi penandaan, seperti TagResourcedan. UntagResource

Untuk kebijakan contoh dan informasi lebih lanjut, lihat Mengontrol Akses Berdasarkan Kunci Tanda di Panduan Pengguna IAM.

Izin untuk membuat dan mengelola tanda bekerja sebagai berikut.

pembayaran-kriptografi: TagResource

Memungkinkan prinsipal untuk menambah atau mengedit tanda. Untuk menambahkan tag saat membuat kunci, prinsipal harus memiliki izin dalam kebijakan IAM yang tidak terbatas pada kunci tertentu.

pembayaran-kriptografi: ListTagsForResource

Memungkinkan prinsipal untuk melihat tag pada kunci.

pembayaran-kriptografi: UntagResource

Memungkinkan prinsipal untuk menghapus tag dari kunci.

Izin tanda dalam kebijakan

Anda dapat memberikan izin penandaan dalam kebijakan kunci atau kebijakan IAM. Misalnya, kebijakan kunci contoh berikut memberikan izin penandaan pengguna tertentu pada kunci. Ini memberikan semua pengguna yang dapat mengasumsikan contoh peran Administrator atau Developer izin untuk melihat tanda.

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Id": "example-key-policy",
  "Statement": [
    {
      "Sid": "EnableIAMUserPermissions",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "payment-cryptography:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowAllTaggingPermissions",
      "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:ListTagsForResource",
      "Resource": "*"
    }
  ]
}
```

Untuk memberikan izin penandaan prinsipal pada beberapa kunci, Anda dapat menggunakan kebijakan IAM. Agar kebijakan ini efektif, kebijakan kunci untuk setiap kunci harus mengizinkan akun menggunakan kebijakan IAM untuk mengontrol akses ke kunci.

Misalnya, kebijakan IAM berikut memungkinkan prinsipal untuk membuat kunci. Ini juga memungkinkan mereka untuk membuat dan mengelola tag pada semua kunci di akun yang ditentukan. Kombinasi ini memungkinkan prinsipal untuk menggunakan parameter tag CreateKeyoperasi untuk menambahkan tag ke kunci saat mereka membuatnya.

JSON

```
{
  "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/*"
    }
  ]
}
```

Membatasi izin tanda

Anda dapat membatasi izin penandaan dengan menggunakan ketentuan kebijakan. Kondisi kebijakan berikut dapat diterapkan ke izin `payment-cryptography:TagResource` dan `payment-cryptography:UntagResource`. Misalnya, Anda dapat menggunakan kondisi `aws:RequestTag/`*`tag-key`* mengizinkan prinsipal untuk menambahkan hanya tanda tertentu, atau mencegah prinsipal menambahkan tanda dengan kunci tanda tertentu.

- aws: RequestTag
- aws:ResourceTag/tag-key (hanya kebijakan IAM)

- aws: TagKeys

Sebagai praktik terbaik saat Anda menggunakan tag untuk mengontrol akses ke kunci, gunakan tombol `aws:RequestTag/tag-key` atau `aws:TagKeys` kondisi untuk menentukan tag (atau kunci tag) mana yang diizinkan.

Sebagai contoh, kebijakan IAM berikut ini mirip dengan yang sebelumnya. Namun, kebijakan ini memungkinkan prinsipal untuk membuat tanda (`TagResource`) dan menghapus tanda `UntagResource` hanya untuk tanda dengan kunci tanda `Project`.

Karena `TagResource` dan `UntagResource` permintaan dapat menyertakan beberapa tag, Anda harus menentukan operator `ForAllValues` atau `ForAnyValue` set dengan TagKeys kondisi aws:. Operator `ForAnyValue` mensyaratkan bahwa setidaknya salah satu kunci tanda dalam permintaan cocok dengan salah satu kunci tanda dalam kebijakan. Operator `ForAllValues` mensyaratkan bahwa semua kunci tanda dalam permintaan cocok dengan salah satu kunci tanda dalam kebijakan. `ForAllValues`Operator juga kembali `true` jika tidak ada tag dalam permintaan, tetapi TagResource dan UntagResource gagal ketika tidak ada tag yang ditentukan. Untuk detail tentang operator set, lihat Gunakan beberapa kunci dan nilai di Panduan Pengguna IAM.

JSON

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

Menggunakan tag untuk mengontrol akses ke tombol

Anda dapat mengontrol akses ke Kriptografi AWS Pembayaran berdasarkan tag pada kunci. Misalnya, Anda dapat menulis kebijakan IAM yang memungkinkan prinsipal mengaktifkan dan menonaktifkan hanya kunci yang memiliki tag tertentu. Atau Anda dapat menggunakan kebijakan IAM untuk mencegah prinsipal menggunakan kunci dalam operasi kriptografi kecuali kunci memiliki tag tertentu.

Fitur ini merupakan bagian dari dukungan Kriptografi AWS Pembayaran untuk kontrol akses berbasis atribut (ABAC). Untuk informasi tentang penggunaan tag untuk mengontrol akses ke AWS sumber daya, lihat Untuk apa ABAC? AWS dan Mengontrol Akses ke AWS Sumber Daya Menggunakan Tag Sumber Daya di Panduan Pengguna IAM.

AWS Kriptografi Pembayaran mendukung kunci konteks kondisi global aws:ResourceTag/tag-key, yang memungkinkan Anda mengontrol akses ke kunci berdasarkan tag pada kunci. Karena beberapa kunci dapat memiliki tag yang sama, fitur ini memungkinkan Anda menerapkan izin ke satu set kunci tertentu. Anda juga dapat dengan mudah mengubah kunci di set dengan mengubah tag mereka.

Dalam Kriptografi AWS Pembayaran, kunci `aws:ResourceTag/`*`tag-key`* kondisi hanya didukung dalam kebijakan IAM. Ini tidak didukung dalam kebijakan utama, yang hanya berlaku untuk satu kunci, atau pada operasi yang tidak menggunakan kunci tertentu, seperti ListKeysatau ListAliasesoperasi.

Mengontrol akses dengan tanda menyediakan cara sederhana, dapat diskalakan, dan fleksibel untuk mengelola izin. Namun, jika tidak dirancang dan dikelola dengan benar, itu dapat mengizinkan atau menolak akses ke kunci Anda secara tidak sengaja. Jika Anda menggunakan tanda untuk mengontrol akses, pertimbangkan praktik berikut.

- Gunakan tanda untuk memperkuat praktik terbaik dari akses dengan keistimewaan terkecil. Berikan kepala sekolah IAM hanya izin yang mereka butuhkan hanya pada kunci yang harus mereka gunakan atau kelola. Misalnya, gunakan tag untuk memberi label kunci yang digunakan untuk proyek. Kemudian berikan izin kepada tim proyek untuk hanya menggunakan kunci dengan tag proyek.
- Berhati-hatilah tentang memberikan prinsipal izin `payment-cryptography:TagResource` dan `payment-cryptography:UntagResource` yang memungkinkan mereka menambahkan, mengedit, dan menghapus tanda. Saat Anda menggunakan tag untuk mengontrol akses ke kunci, mengubah tag dapat memberikan izin kepada prinsipal untuk menggunakan kunci yang tidak diizinkan untuk digunakan. Itu juga dapat menolak akses ke kunci yang diperlukan oleh kepala sekolah lain untuk melakukan pekerjaan mereka. Administrator kunci yang tidak memiliki izin untuk mengubah kebijakan kunci atau membuat hibah dapat mengontrol akses ke kunci jika mereka memiliki izin untuk mengelola tag.

  Jika memungkinkan, gunakan kondisi kebijakan, seperti `aws:RequestTag/`*`tag-key`* atau `aws:TagKeys` untuk membatasi izin penandaan prinsipal untuk tag atau pola tag tertentu pada kunci tertentu.
- Tinjau prinsipal di Anda Akun AWS yang saat ini memiliki izin penandaan dan pembatalan tag dan sesuaikan, jika perlu. Kebijakan IAM mungkin mengizinkan izin tag dan untag pada semua kunci. Misalnya, kebijakan terkelola Admin memungkinkan prinsipal untuk menandai, menghapus tag, dan mencantumkan tag pada semua kunci.
- Sebelum menetapkan kebijakan yang bergantung pada tag, tinjau tag pada kunci di tag Anda Akun AWS. Pastikan bahwa kebijakan Anda hanya berlaku untuk tanda yang ingin Anda sertakan. Gunakan CloudTrail log dan CloudWatch alarm untuk mengingatkan Anda untuk menandai perubahan yang mungkin memengaruhi akses ke kunci Anda.
- Kondisi kebijakan berbasis tanda menggunakan pencocokan pola; mereka tidak terikat pada instans tertentu dari tanda. Kebijakan yang menggunakan kunci kondisi berbasis tanda memengaruhi semua tanda baru dan yang sudah ada yang cocok dengan pola. Jika Anda menghapus dan membuat ulang tanda yang cocok dengan kondisi kebijakan, kondisi berlaku untuk tanda baru, seperti halnya pada tanda lama.

Misalnya, pertimbangkan contoh kebijakan IAM berikut. Ini memungkinkan kepala sekolah untuk memanggil operasi Dekripsi hanya pada kunci di akun Anda yang merupakan Wilayah AS Timur (Virginia N.) dan memiliki tag. `"Project"="Alpha"` Anda mungkin melampirkan kebijakan ini ke peran dalam contoh proyek Alpha.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IAMPolicyWithResourceTag",
      "Effect": "Allow",
      "Action": [
        "payment-cryptography:DecryptData"
      ],
      "Resource": "arn:aws:payment-cryptography:us-east-1:111122223333:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "Alpha"
        }
      }
    }
  ]
}
```

Contoh berikut kebijakan IAM memungkinkan prinsipal untuk menggunakan kunci apa pun dalam akun untuk operasi kriptografi tertentu. Tapi itu melarang prinsipal menggunakan operasi kriptografi ini pada kunci dengan tag atau tanpa tag. `"Type"="Reserved"` `"Type"`

JSON

```
{
  "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"
        }
      }
    }
  ]
}
```

Memahami atribut kunci untuk kunci Kriptografi AWS Pembayaran

Prinsip manajemen kunci yang tepat adalah bahwa kunci dicakup dengan tepat dan hanya dapat digunakan untuk operasi yang diizinkan. Dengan demikian, kunci tertentu hanya dapat dibuat dengan mode penggunaan kunci tertentu. Bila memungkinkan, ini sejalan dengan mode penggunaan yang tersedia seperti yang didefinisikan oleh TR-31.

Meskipun Kriptografi AWS Pembayaran akan mencegah Anda membuat kunci yang tidak valid, kombinasi yang valid disediakan di sini untuk kenyamanan Anda.

Tombol Simetris

- TR31_B0_BASE_DERIVATION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_C0_CARD_VERIFICATION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY*, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E1_EMV_MKEY_KERAHASIAAN
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E2_EMV_MKEY_INTEGRITY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_E5_EMV_MKEY_CARD_PERSONALISASI
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}

- TR31_E6_EMV_MKEY_LAINNYA
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}, { NoRestrictions = true}
- TR31_K0_KEY_ENCRYPTION_KEY
  - Disarankan untuk menggunakan TR31 _K1_KEY_BLOCK_PROTECTION_KEY. Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_K1_KEY_BLOCK_PROTECTION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {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
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M3_ISO_9797_3_MAC_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M6_ISO_9797_5_CMAC_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_M7_HMAC_KUNCI
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_P0_PIN_ENCRYPTION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256

  - Kombinasi mode penggunaan kunci yang diizinkan: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_V1_ IBM3624 _PIN_VERIFICATION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}
- TR31_V2_VISA_PIN_VERIFICATION_KEY
  - Algoritma Kunci yang Diizinkan: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Kombinasi mode penggunaan kunci yang diizinkan: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, { NoRestrictions = true}

Tombol Asimetris

- TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENKRIPSI
  - Algoritma Kunci yang Diizinkan: RSA_2048, RSA_3072, RSA_4096
  - Kombinasi mode penggunaan kunci yang diizinkan: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}
  - CATATAN:: {Encrypt = true, Wrap = true} adalah satu-satunya opsi yang valid saat mengimpor kunci publik yang dimaksudkan untuk mengenkripsi data atau membungkus kunci
- TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE
  - Algoritma Kunci yang Diizinkan: RSA_2048, RSA_3072, RSA_4096
  - Kombinasi mode penggunaan kunci yang diizinkan: {Sign = true}, {Verify = true}
  - CATATAN:: {Verify = true} adalah satu-satunya opsi yang valid saat mengimpor kunci yang dimaksudkan untuk ditandatangani, seperti sertifikat root, sertifikat perantara, atau sertifikat penandatanganan untuk TR-34.
- TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT
  - Digunakan untuk algoritma perjanjian utama seperti ECDH
  - Algoritma Kunci yang Diizinkan: ECC_NIST_P256, ECC_NIST_P384, ECC_NIST_P521
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}.
  - CATATAN: DeriveKeyUsage digunakan untuk menentukan jenis kunci apa yang akan diturunkan dari kunci dasar ini. Ini diperbaiki pada pembuatan/impor utama.

- TR31_K2_ TR34 _ASYMMETRIC_KEY
  - Kunci asimetris yang digunakan untuk mekanisme pertukaran kunci yang kompatibel dengan X9.24 seperti TR-34
  - Algoritma Kunci yang Diizinkan: RSA_2048, RSA_3072, RSA_4096
  - Kombinasi mode penggunaan kunci yang diizinkan: { DeriveKey = true}.
  - Kombinasi mode penggunaan kunci yang diizinkan: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}
  - CATATAN:: {Encrypt = true, Wrap = true} adalah satu-satunya opsi yang valid saat mengimpor kunci publik yang dimaksudkan untuk mengenkripsi data atau membungkus kunci

* Kombinasi algorithm/key jenis ini saat ini tidak didukung oleh operasi kriptografi apa pun

Operasi data

Setelah Anda membuat kunci Kriptografi AWS Pembayaran, itu dapat digunakan untuk melakukan operasi kriptografi. Operasi yang berbeda melakukan berbagai jenis aktivitas mulai dari enkripsi, hashing, serta algoritme spesifik domain seperti CVV2 pembuatan.

Data terenkripsi tidak dapat didekripsi tanpa kunci dekripsi yang cocok (kunci simetris atau kunci pribadi tergantung pada jenis enkripsi). Algoritma hashing dan domain spesifik juga tidak dapat diverifikasi tanpa kunci simetris atau kunci publik.

Untuk informasi tentang jenis kunci yang valid untuk operasi tertentu, silakan lihat Kunci yang valid untuk operasi kriptografi

> **Note**
>
> Sebaiknya gunakan data uji saat berada di lingkungan non-produksi. Menggunakan kunci dan data produksi (PAN, ID BDK, dll.) di lingkungan non-produksi dapat memengaruhi cakupan kepatuhan Anda seperti untuk PCI DSS dan PCI P2PE.

Topik

Enkripsi, Dekripsi, dan Enkripsi Ulang Data

Metode enkripsi dan dekripsi dapat digunakan untuk mengenkripsi atau mendekripsi data menggunakan berbagai teknik simetris dan asimetris termasuk TDES, AES dan RSA. Metode ini juga mendukung kunci yang diturunkan menggunakan teknik DUKPT dan EMV. Untuk kasus penggunaan di mana Anda ingin mengamankan data di bawah kunci baru tanpa mengekspos data yang mendasarinya, ReEncrypt perintah juga dapat digunakan.

> **Note**
>
> Saat menggunakan encrypt/decrypt fungsi, semua input diasumsikan berada di HexBinary - misalnya nilai 1 akan dimasukkan sebagai 31 (hex) dan huruf kecil t direpresentasikan sebagai 74 (hex). Semua output ada di HexBinary juga.

Untuk detail tentang semua opsi yang tersedia, silakan baca Panduan API untuk Enkripsi, Dekripsi, dan Enkripsi Ulang.

Topik

Enkripsi data

`Encrypt Data`API digunakan untuk mengenkripsi data menggunakan kunci enkripsi data simetris dan asimetris serta kunci turunan DUKPT dan EMV. Berbagai algoritma dan variasi didukung termasuk`TDES`, `RSA` dan`AES`.

Input utama adalah kunci enkripsi yang digunakan untuk mengenkripsi data, data teks biasa dalam format HexBinary yang akan dienkripsi dan atribut enkripsi seperti vektor inisialisasi dan mode untuk sandi blok seperti TDES. Data plaintext harus dalam kelipatan 8 byte untuk`TDES`, 16 byte untuk `AES` dan panjang kunci dalam kasus. `RSA` Input kunci simetris (TDES, AES, DUKPT, EMV) harus empuk dalam kasus di mana data input tidak memenuhi persyaratan ini. Tabel berikut menunjukkan panjang maksimum plaintext untuk setiap jenis kunci dan jenis padding yang Anda tentukan `EncryptionAttributes` untuk kunci RSA.

Jenis bantalan	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

Jenis bantalan	RSA_2048	RSA_3072	RSA_4096
None	488	744	1000

Output utama termasuk data terenkripsi sebagai ciphertext dalam format HexBinary dan nilai checksum untuk kunci enkripsi. Untuk detail tentang semua opsi yang tersedia, silakan baca Panduan API untuk Enkripsi.

Contoh

- Enkripsi data menggunakan kunci simetris AES
- Enkripsi data menggunakan kunci DUKPT
- Enkripsi data menggunakan kunci simetris turunan EMV
- Enkripsi data menggunakan kunci RSA

Enkripsi data menggunakan kunci simetris AES

> **Note**
>
> Semua contoh mengasumsikan kunci yang relevan sudah ada. Kunci dapat dibuat menggunakan CreateKeyoperasi atau diimpor menggunakan ImportKeyoperasi.

Example

Dalam contoh ini, kita akan mengenkripsi data plaintext menggunakan kunci simetris yang telah dibuat menggunakan CreateKeyOperasi atau diimpor menggunakan Operasi. ImportKey Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `Encrypt` dan KeyUsage disetel ke`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya.

```
$ 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"
}
```

Enkripsi data menggunakan kunci DUKPT

Example

Dalam contoh ini, kita akan mengenkripsi data plaintext menggunakan kunci DUKPT. AWS Dukungan Kriptografi Pembayaran `TDES` dan kunci `AES` DUKPT. Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `DeriveKey` dan KeyUsage disetel ke`TR31_B0_BASE_DERIVATION_KEY`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya.

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi
 --plain-text 31323334313233343132333431323334 --encryption-attributes
 'Dukpt={KeySerialNumber=FFFF9876543210E00001}'
```

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

Enkripsi data menggunakan kunci simetris turunan EMV

Example

Dalam contoh ini, kita akan mengenkripsi data teks yang jelas menggunakan kunci simetris turunan EMV yang telah dibuat. Anda dapat menggunakan perintah seperti ini untuk mengirim data ke kartu EMV. Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `Derive` dan KeyUsage disetel ke `TR31_E1_EMV_MKEY_CONFIDENTIALITY` atau`TR31_E6_EMV_MKEY_OTHER`. Silakan lihat Kunci untuk Operasi Kriptografi untuk lebih jelasnya.

```
$ 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"
}
```

Enkripsi data menggunakan kunci RSA

Example

Dalam contoh ini, kita akan mengenkripsi data plaintext menggunakan kunci publik RSA yang telah diimpor menggunakan operasi. ImportKey Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `Encrypt` dan KeyUsage disetel ke`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya.

Untuk PKCS #7 atau skema padding lainnya yang saat ini tidak didukung, mohon terapkan sebelum memanggil layanan dan pilih no padding dengan menghilangkan indikator padding 'Asymmetric= {} '

```
$ aws payment-cryptography-data encrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/thfezpmsalcfwmsg
 --plain-text 31323334313233343132333431323334 --encryption-attributes
 'Asymmetric={PaddingType=OAEP_SHA256}'
```

```
{
 "CipherText":
 "12DF6A2F64CC566D124900D68E8AFEAA794CA819876E258564D525001D00AC93047A83FB13 \
 E73F06329A100704FA484A15A49F06A7A2E55A241D276491AA91F6D2D8590C60CDE57A642BC64A897F4832A3930
 \
 0FAEC7981102CA0F7370BFBF757F271EF0BB2516007AB111060A9633D1736A9158042D30C5AE11F8C5473EC70F067
 \
 72590DEA1638E2B41FAE6FB1662258596072B13F8E2F62F5D9FAF92C12BB70F42F2ECDCF56AADF0E311D4118FE3591
 \
 FB672998CCE9D00FFFE05D2CD154E3120C5443C8CF9131C7A6A6C05F5723B8F5C07A4003A5A6173E1B425E2B5E42AD
 \
 7A2966734309387C9938B029AFB20828ACFC6D00CD1539234A4A8D9B94CDD4F23A",
 "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/5dza7xqd6soanjtb",
 "KeyCheckValue": "FF9DE9CE"
}
```

Dekripsi data

`Decrypt Data`API digunakan untuk mendekripsi data menggunakan kunci enkripsi data simetris dan asimetris serta kunci turunan DUKPT dan EMV. Berbagai algoritma dan variasi didukung termasuk`TDES`, `RSA` dan`AES`.

Input utama adalah kunci dekripsi yang digunakan untuk mendekripsi data, data ciphertext dalam format HexBinary yang akan didekripsi dan atribut dekripsi seperti vektor inisialisasi, mode sebagai cipher blok dll. Output utama termasuk data yang didekripsi sebagai plaintext dalam format HexBinary dan nilai checksum untuk kunci dekripsi. Untuk detail tentang semua opsi yang tersedia, silakan baca Panduan API untuk Dekripsi.

Contoh

- Dekripsi data menggunakan kunci simetris AES
- Dekripsi data menggunakan kunci DUKPT
- Dekripsi data menggunakan kunci simetris turunan EMV
- Dekripsi data menggunakan kunci RSA

Dekripsi data menggunakan kunci simetris AES

Example

Dalam contoh ini, kita akan mendekripsi data ciphertext menggunakan kunci simetris. Contoh ini menunjukkan `AES` kunci tetapi `TDES_2KEY` dan `TDES_3KEY` juga didukung. Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `Decrypt` dan KeyUsage disetel ke`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya.

```
$ 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"
}
```

Dekripsi data menggunakan kunci DUKPT

> **Note**
>
> Menggunakan data dekripsi dengan DUKPT untuk transaksi P2PE dapat mengembalikan PAN kartu kredit dan data pemegang kartu lainnya ke aplikasi Anda yang perlu dipertanggungjawabkan saat menentukan cakupan PCI DSS-nya.

Example

Dalam contoh ini, kita akan mendekripsi data ciphertext menggunakan kunci DUKPT yang telah dibuat menggunakan CreateKeyOperasi atau diimpor menggunakan Operasi. ImportKey Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `DeriveKey` dan KeyUsage disetel ke`TR31_B0_BASE_DERIVATION_KEY`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya. Bila Anda menggunakanDUKPT, untuk `TDES` algoritma, panjang data ciphertext harus kelipatan 16 byte. Untuk `AES` algoritma, panjang data ciphertext harus kelipatan 32 byte.

```
$ 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": "313233343132333431323334313233343132333431323334"
}
```

Dekripsi data menggunakan kunci simetris turunan EMV

Example

Dalam contoh ini, kita akan mendekripsi data ciphertext menggunakan kunci simetris turunan EMV yang telah dibuat menggunakan operasi atau diimpor menggunakan operasi. CreateKeyImportKey Untuk operasi ini, kunci harus KeyModesOfUse disetel ke `Derive` dan KeyUsage disetel ke `TR31_E1_EMV_MKEY_CONFIDENTIALITY` atau`TR31_E6_EMV_MKEY_OTHER`. Silakan lihat Kunci untuk Operasi Kriptografi untuk lebih jelasnya.

```
$ 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"
}
```

Dekripsi data menggunakan kunci RSA

Example

Dalam contoh ini, kita akan mendekripsi data ciphertext menggunakan key pair RSA yang telah dibuat menggunakan operasi. CreateKey Untuk operasi ini, kunci harus KeyModesOfUse disetel untuk mengaktifkan `Decrypt` dan KeyUsage mengatur ke`TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION`. Silakan lihat Kunci untuk Operasi Kriptografi untuk opsi lainnya.

Untuk PKCS #7 atau skema padding lainnya yang saat ini tidak didukung, pilih no padding dengan menghilangkan indikator padding 'Asymmetric= {} 'dan hapus padding setelah memanggil layanan.

```
$ 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-1:111122223333:key/5dza7xqd6soanjtb",
    "KeyCheckValue": "FF9DE9CE",
    "PlainText": "313233343132333431323334313233343132333431323334"
}
```

Menghasilkan dan memverifikasi data kartu

Menghasilkan dan memverifikasi data kartu menggabungkan data yang berasal dari data kartu, misalnya CVV,, CVC dan DCVV. CVV2

Topik

- Hasilkan data kartu
- Verifikasi data kartu

Hasilkan data kartu

`Generate Card Data`API digunakan untuk menghasilkan data kartu menggunakan algoritma seperti CVV, CVV2 atau Dynamic. CVV2 Untuk melihat kunci apa yang dapat digunakan untuk perintah ini, silakan lihat Kunci yang valid untuk operasi kriptografi bagian.

Banyak nilai kriptografi seperti CVV,, ICVV CVV2, CAVV V7 menggunakan algoritma kriptografi yang sama tetapi memvariasikan nilai input. Misalnya CardVerificationValue1 memiliki input ServiceCode, Nomor Kartu dan Tanggal Kedaluwarsa. Sementara CardVerificationValue2 hanya memiliki dua input ini, ini karena untuk CVV2/CVC2, ServiceCode ditetapkan pada 000. Demikian pula, untuk iCVV ServiceCode ditetapkan pada 999. Beberapa algoritma dapat menggunakan kembali bidang yang ada seperti CAVV V8 dalam hal ini Anda perlu berkonsultasi dengan manual penyedia Anda untuk nilai input yang benar.

> **Note**
>
> Tanggal kedaluwarsa harus dimasukkan dalam format yang sama (seperti MMYY vs YYMM) untuk pembuatan dan validasi untuk menghasilkan hasil yang benar.

Menghasilkan CVV2

Example

Dalam contoh ini, kami akan menghasilkan CVV2 untuk PAN tertentu dengan input PAN dan tanggal kedaluwarsa kartu. Ini mengasumsikan bahwa Anda memiliki kunci verifikasi kartu yang dihasilkan.

```
$ 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"
  }
```

Menghasilkan iCVV

Example

Dalam contoh ini, kami akan menghasilkan iCVV untuk PAN tertentu dengan inputPAN, kode layanan 999 dan tanggal kedaluwarsa kartu. Ini mengasumsikan bahwa Anda memiliki kunci verifikasi kartu yang dihasilkan.

Untuk semua parameter yang tersedia, lihat CardVerificationValue1 di panduan referensi API.

```
$ 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"
}
```

Verifikasi data kartu

`Verify Card Data`digunakan untuk memverifikasi data yang telah dibuat menggunakan algoritma pembayaran yang mengandalkan prinsip enkripsi seperti. `DISCOVER_DYNAMIC_CARD_VERIFICATION_CODE`

Nilai input biasanya diberikan sebagai bagian dari transaksi masuk ke penerbit atau mitra platform pendukung. Untuk memverifikasi kriptogram ARQC (digunakan untuk kartu chip EMV), silakan lihat Verifikasi ARQC.

Untuk informasi selengkapnya, lihat VerifyCardValidationDatadi panduan API.

Jika nilainya diverifikasi, maka api akan mengembalikan http/200. Jika nilainya tidak diverifikasi, itu akan mengembalikan http/400.

Verifikasi CVV2

Example

Dalam contoh ini, kita akan memvalidasi CVV/ CVV2 untuk PAN tertentu. Biasanya CVV2 disediakan oleh pemegang kartu atau pengguna selama waktu transaksi untuk validasi. Untuk memvalidasi input mereka, nilai-nilai berikut akan diberikan saat runtime - Kunci untuk Digunakan untuk validasi (CVK),PAN, tanggal kedaluwarsa kartu dan dimasukkan. CVV2 Format kedaluwarsa kartu harus sesuai dengan yang digunakan dalam pembuatan nilai awal.

Untuk semua parameter yang tersedia, lihat CardVerificationValue2 di panduan referensi API.

```
$ 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"
  }
```

Verifikasi iCvv

Example

Dalam contoh ini, kami akan memverifikasi iCVV untuk PAN tertentu dengan input Key to Use for validation (CVK),, kode layanan 999PAN, tanggal kedaluwarsa kartu dan iCVV yang disediakan oleh transaksi untuk memvalidasi.

iCVV bukan nilai yang dimasukkan pengguna (seperti CVV2) tetapi disematkan pada kartu EMV. Pertimbangan harus diberikan apakah harus selalu memvalidasi saat disediakan.

Untuk semua parameter yang tersedia, lihat, CardVerificationValue1 di panduan referensi API.

```
$ 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
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999} --validation-data 801
```

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

Menghasilkan, menerjemahkan, dan memverifikasi data PIN

Fungsi data PIN memungkinkan Anda untuk menghasilkan pin acak, nilai verifikasi pin (PVV) dan memvalidasi pin terenkripsi masuk terhadap PVV atau PIN Offset.

Terjemahan pin memungkinkan Anda menerjemahkan pin dari satu kunci kerja ke yang lain tanpa mengekspos pin dalam teks yang jelas seperti yang ditentukan oleh Persyaratan PIN PCI 1.

> **Note**
>
> Karena pembuatan dan validasi PIN biasanya merupakan fungsi penerbit dan terjemahan PIN adalah fungsi pengakuisisi yang khas, kami menyarankan Anda mempertimbangkan

akses yang paling tidak diprivileged dan menetapkan kebijakan dengan tepat untuk kasus penggunaan sistem Anda.

Topik

- Terjemahkan data PIN
- Hasilkan data PIN
- Verifikasi data PIN

Terjemahkan data PIN

Fungsi data PIN Translate digunakan untuk menerjemahkan data PIN terenkripsi dari satu set kunci ke yang lain tanpa data terenkripsi meninggalkan HSM. Ini digunakan untuk enkripsi P2PE di mana kunci kerja harus berubah tetapi sistem pemrosesan tidak perlu, atau tidak diizinkan untuk, mendekripsi data. Input utama adalah data terenkripsi, kunci enkripsi yang digunakan untuk mengenkripsi data, parameter yang digunakan untuk menghasilkan nilai input. Kumpulan input lainnya adalah parameter output yang diminta seperti kunci yang akan digunakan untuk mengenkripsi output dan parameter yang digunakan untuk membuat output itu. Output utama adalah dataset yang baru dienkripsi serta parameter yang digunakan untuk menghasilkannya.

> **Note**
>
> Untuk kepatuhan PCI, PrimaryAccountNumber nilai masuk dan keluar harus cocok. Menerjemahkan PIN dari satu PAN ke PAN lainnya tidak diizinkan.

Topik

- PIN dari PEK ke DUKPT
- PIN dari PEK ke PEK

PIN dari PEK ke DUKPT

Example

Dalam contoh ini, kami akan menerjemahkan PIN dari Blok PIN ISO 4 AES menggunakan enkripsi DUKPT ke PEK TDES menggunakan blok PIN ISO 0. Ini umum terjadi di mana terminal pembayaran mengenkripsi pin dalam ISO 4 dan kemudian dapat diterjemahkan kembali ke TDES untuk pemrosesan hilir jika koneksi berikutnya belum mendukung AES.

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "AC17DC148BDA645E" --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": "1F4209C670E49F83E75CC72E81B787D9",
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
            "KeyCheckValue": "7CC9E2"
        }
```

PIN dari PEK ke PEK

Example

Dalam contoh ini, kami menerjemahkan PIN yang dienkripsi di bawah satu PEK (PIN Encryption Key) ke PEK lain. Ini biasanya digunakan saat merutekan transaksi antara sistem atau mitra yang berbeda yang menggunakan kunci enkripsi yang berbeda, sambil mempertahankan kepatuhan PIN PCI dengan menjaga PIN terenkripsi selama proses berlangsung. Kedua kunci menggunakan enkripsi TDES 3KEY dalam contoh ini, tetapi berbagai opsi tersedia termasuk AES ISO-4 ke TDES ISO-0, DUKPT ke PEK, atau ke PEK. AS2805

```
$ 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-translation-attributes
 IsoFormat0='{PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh
```

```
{
    "PinBlock": "E8F2A6C4D1B93E7F",
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
    "KeyCheckValue": "9A325B"
}
```

Blok PIN keluaran sekarang dienkripsi di bawah PEK kedua dan dapat dikirim dengan aman ke sistem hilir yang memegang kunci yang sesuai.

Hasilkan data PIN

Menghasilkan fungsi data PIN digunakan untuk menghasilkan nilai terkait PIN, seperti PVV dan offset blok pin yang digunakan untuk memvalidasi entri pin oleh pengguna selama waktu transaksi atau otorisasi. API ini juga dapat menghasilkan pin acak baru menggunakan berbagai algoritma.

Hasilkan pin acak dan pencocokan Visa PVV

Example

Dalam contoh ini, kami akan menghasilkan pin baru (acak) di mana output akan dienkripsi `PIN block` (. PinData PinBlock) dan a PVV (pindata.offset). Input kuncinya adalah`PAN`, the`Pin Verification Key`, the `Pin Encryption Key` and the. `PIN block format`

Perintah ini mengharuskan kuncinya bertipe`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"
            }
        }
```

Hasilkan PVV Visa untuk pin yang dikenal

Example

Dalam contoh ini, kami akan menghasilkan PVV untuk pin yang diberikan (terenkripsi). Pin terenkripsi dapat diterima di hulu seperti dari terminal pembayaran atau dari pemegang kartu menggunakan aliran pin yang dapat dipilih pengguna. Input kuncinya adalahPAN, thePin Verification Key, thePin Encryption Key, the `Encrypted Pin Block` and the. `PIN block format`

```
$ 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
 VisaPinVerificationValue={PinVerificationKeyIndex=1,EncryptedPinBlock=AA584CED31790F37}
```

```
{
            "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"
            }
        }
```

Hasilkan offset IBM3624 pin untuk pin

IBM 3624 PIN Offset juga kadang-kadang disebut metode IBM. Metode ini menghasilkan natural/intermediate PIN menggunakan data validasi (biasanya PAN) dan Kunci PIN (PVK). Pin alami secara efektif merupakan nilai turunan dan deterministik sangat efisien untuk ditangani oleh penerbit karena tidak ada data pin yang perlu disimpan pada tingkat pemegang kartu. Kontra yang paling jelas adalah bahwa skema ini tidak memperhitungkan pin yang dapat dipilih atau acak pemegang kartu. Untuk memungkinkan jenis pin tersebut, algoritma offset ditambahkan ke skema. Offset mewakili perbedaan antara pin yang dipilih pengguna (atau acak) dan kunci alami. Nilai offset disimpan oleh penerbit kartu atau prosesor kartu. Pada saat transaksi, layanan Kriptografi AWS Pembayaran

secara internal menghitung ulang pin alami dan menerapkan offset untuk menemukan pin. Kemudian membandingkan ini dengan nilai yang diberikan oleh otorisasi transaksi.

Ada beberapa opsi untuk IBM3624:

- `Ibm3624NaturalPin`akan menampilkan pin alami dan blok pin terenkripsi
- `Ibm3624PinFromOffset`akan menghasilkan blok pin terenkripsi yang diberi offset
- `Ibm3624RandomPin`akan menghasilkan pin acak dan kemudian blok pin offset dan terenkripsi yang cocok.
- `Ibm3624PinOffset`menghasilkan offset pin yang diberikan pin yang dipilih pengguna.

Internal Kriptografi AWS Pembayaran, langkah-langkah berikut dilakukan:

- Pad panci yang disediakan hingga 16 karakter. Jika <16 disediakan, pad di sisi kanan menggunakan karakter padding yang disediakan.
- Mengenkripsi data validasi menggunakan kunci pembuatan PIN.
- Dekimalisasi data terenkripsi menggunakan tabel desimalisasi. Ini memetakan digit heksidesimal ke digit desimal misalnya 'A' dapat memetakan ke 9 dan 1 dapat memetakan ke 1.
- Dapatkan 4 digit pertama dari representasi heksidesimal output. Ini adalah pin alami.
- Jika pin yang dipilih pengguna atau acak dihasilkan, modulo kurangi pin alami dengan pin pelanggan. Hasilnya adalah offset pin.

Contoh

- Contoh: Hasilkan offset IBM3624 pin untuk pin

Contoh: Hasilkan offset IBM3624 pin untuk pin

Dalam contoh ini, kami akan menghasilkan pin baru (acak) di mana output akan dienkripsi `PIN block` (. PinData PinBlock) dan nilai `IBM3624` offset (pindata.offset). Inputnya adalah`PAN`, data validasi (biasanya pan), karakter padding,`Pin Verification Key`, dan. `Pin Encryption Key` `PIN block format`

Perintah ini mensyaratkan bahwa kunci pembuatan pin adalah tipe `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` dan kunci enkripsi bertipe `TR31_P0_PIN_ENCRYPTION_KEY`

Example

Contoh berikut menunjukkan menghasilkan pin acak kemudian mengeluarkan blok pin terenkripsi dan nilai IBM3624 offset menggunakan 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"
              }
          }
```

Verifikasi data PIN

Verifikasi fungsi data PIN digunakan untuk memverifikasi apakah pin sudah benar. Ini biasanya melibatkan membandingkan nilai pin yang sebelumnya disimpan dengan apa yang dimasukkan oleh pemegang kartu di POI. Fungsi-fungsi ini membandingkan dua nilai tanpa mengekspos nilai yang mendasari dari salah satu sumber.

Validasi PIN terenkripsi menggunakan metode PVV

Example

Dalam contoh ini, kita akan memvalidasi PIN untuk PAN tertentu. PIN biasanya disediakan oleh pemegang kartu atau pengguna selama waktu transaksi untuk validasi dan dibandingkan dengan nilai pada file (input dari pemegang kartu diberikan sebagai nilai terenkripsi dari terminal atau penyedia hulu lainnya). Untuk memvalidasi input ini, nilai berikut juga akan diberikan saat runtime: Kunci yang digunakan untuk mengenkripsi pin input (ini sering merupakan`IWK`), PAN dan nilai untuk memverifikasi terhadap (baik a `PVV` atau). `PIN offset`

Jika Kriptografi AWS Pembayaran dapat memvalidasi pin, http/200 dikembalikan. Jika pin tidak divalidasi, itu akan mengembalikan http/400.

```
$ 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",

        }
```

Validasi PIN terenkripsi menggunakan metode PVV - kesalahan pin buruk

Example

Dalam contoh ini, kami akan mencoba untuk memvalidasi PIN untuk PAN tertentu tetapi akan gagal karena pin yang salah.

Saat menggunakan SDKs, ini muncul sebagai {"Pesan” :"Verifikasi blok pin gagal.” , "Alasan” :"INVALID_PIN "}

```
$ 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=9999}" --
encrypted-pin-block AC17DC148BDA645E
```

```
       An error occurred (VerificationFailedException) when calling the VerifyPinData
 operation: Pin block verification failed.
```

Validasi PIN terenkripsi menggunakan metode PVV - kesalahan input buruk

Example

Dalam contoh ini, kami akan mencoba untuk memvalidasi PIN untuk PAN tertentu tetapi akan gagal karena input yang buruk dan data yang masuk bukan pin yang valid. Penyebab umum adalah: 1/ kunci yang salah digunakan 2/parameter input seperti format pan atau pin block yang salah 3/pin blok rusak.

```
$ aws payment-cryptography-data verify-pin-data --verification-key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/37y2tsl45p5zjbh2
 --encryption-key-identifier  --primary-account-number 171234567890123
 --pin-block-format ISO_FORMAT_0  --verification-attributes
 VisaPin="{PinVerificationKeyIndex=1,VerificationValue=9999}" --encrypted-pin-block
 AC17DC148BDA645E
```

```
       An error occurred (ValidationException) when calling the VerifyPinData
operation: Pin block provided is invalid. Please check your input to ensure all field
values are correct.
```

Validasi PIN terhadap offset pin yang disimpan IBM3624 sebelumnya

Dalam contoh ini, kami akan memvalidasi PIN yang diberikan pemegang kartu terhadap offset pin yang disimpan pada file dengan penerbit/prosesor kartu. Input serupa ??? dengan tambahan pin terenkripsi yang disediakan oleh terminal pembayaran (atau penyedia hulu lainnya seperti jaringan kartu). Jika pin cocok, api akan mengembalikan http 200. di mana output akan dienkripsi `PIN block` (. PinData PinBlock) dan nilai `IBM3624` offset (pindata.offset).

Perintah ini mensyaratkan bahwa kunci pembuatan pin adalah tipe `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` dan kunci enkripsi bertipe `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"
}
}
```

Verifikasi kriptogram permintaan autentikasi (ARQC)

API kriptogram permintaan autentikasi verifikasi digunakan untuk memverifikasi ARQC. Generasi ARQC berada di luar cakupan Kriptografi AWS Pembayaran dan biasanya dilakukan pada Kartu Chip EMV (atau setara digital seperti dompet seluler) selama waktu otorisasi transaksi. ARQC unik untuk setiap transaksi dan dimaksudkan untuk menunjukkan validitas kartu secara kriptografis serta untuk memastikan bahwa data transaksi sama persis dengan transaksi saat ini (yang diharapkan).

AWS Kriptografi Pembayaran menyediakan berbagai opsi untuk memvalidasi ARQC dan menghasilkan nilai ARPC opsional termasuk yang didefinisikan dalam EMV 4.4 Buku 2 dan skema lain yang digunakan oleh Visa dan Mastercard. Untuk daftar lengkap semua opsi yang tersedia, silakan lihat VerifyCardValidationData bagian di Panduan API.

Kriptogram ARQC biasanya memerlukan input berikut (meskipun ini mungkin berbeda berdasarkan implementasi):

- PAN - Ditentukan di PrimaryAccountNumber lapangan
- Nomor Urutan PAN (PSN) - ditentukan di lapangan PanSequenceNumber

- Metode Derivasi Kunci seperti Common Session Key (CSK) - Ditentukan dalam SessionKeyDerivationAttributes
- Mode Derivasi Kunci Master (seperti Opsi EMV A) - Ditentukan dalam MajorKeyDerivationMode
- Data transaksi - serangkaian berbagai transaksi, terminal dan data kartu seperti Jumlah dan Tanggal - ditentukan dalam TransactionData bidang
- Penerbit Master Key - kunci utama yang digunakan untuk mendapatkan kunci kriptogram (AC) yang digunakan untuk melindungi transaksi individu dan ditentukan di lapangan KeyIdentifier

Topik

Membangun data transaksi

Konten (dan urutan) yang tepat dari bidang data transaksi bervariasi menurut implementasi dan skema jaringan tetapi bidang minimum yang direkomendasikan (dan urutan penggabungan) didefinisikan dalam EMV 4.4 Buku 2 Bagian 8.1.1 - Pemilihan Data. Jika tiga bidang pertama adalah jumlah (17.00), jumlah lain (0.00) dan negara pembelian, yang akan menghasilkan data transaksi dimulai sebagai berikut:

- 000000001700 - jumlah - 12 posisi tersirat dua digit desimal
- 000000000000 - jumlah lainnya - 12 posisi tersirat dua digit desimal
- 0124 - kode negara empat digit
- Data Transaksi Keluaran (sebagian) - 00000000170000000000000000000124

Padding data transaksi

Data transaksi harus empuk sebelum dikirim ke layanan. Sebagian besar skema menggunakan padding ISO 9797 Metode 2, di mana string hex ditambahkan oleh hex 80 diikuti oleh 00 hingga bidang adalah kelipatan dari ukuran blok enkripsi; 8 byte atau 16 karakter untuk TDES dan 16 byte atau 32 karakter untuk AES. Alternatif (metode 1) tidak umum tetapi hanya menggunakan 00 sebagai karakter padding.

ISO 9797 Metode 1 Padding

Tidak empuk:
0000000017000000000000000840008000800080000840160517000000000938000000B03011203
(74 karakter atau 37 byte)

Empuk:
0000000017000000000000000840008000800080000840160517000000000938000000B03011203
000000 (80 karakter atau 40 byte)

ISO 9797 Metode 2 Padding

Tidak empuk:
0000000017000000000000000840008000800080000840160517000000000938000000B1F220103000000
(80 karakter atau 40 byte)

Empuk:
0000000017000000000000000840008400080000840160517000000000938000000B1F220103000000
80000000000000 (88 karakter atau 44 byte)

Contoh

Visa CVN1 0

Example

Dalam contoh ini, kami akan memvalidasi ARQC yang dihasilkan menggunakan Visa 0. CVN1

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. Jika kemudian ARCQ (Authorization Request Cryptogram) tidak divalidasi, itu akan mengembalikan respons http/400.

```
$ 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
 00000000170000000000000008400080008000084016051700000000093800000B03011203000000 \
--session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
,"PrimaryAccountNumber":"9137631040001422"}}'
```

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

Visa CVN18 dan Visa CVN22

Example

Dalam contoh ini, kami akan memvalidasi ARQC yang dihasilkan menggunakan Visa atau. CVN18 CVN22 Operasi kriptografi adalah sama antara CVN18 dan CVN22 tetapi data yang terkandung dalam data transaksi bervariasi. Dibandingkan dengan CVN1 0, kriptogram yang sama sekali berbeda dihasilkan bahkan dengan input yang sama.

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. Jika ARCQ tidak divalidasi, itu akan mengembalikan http/400.

```
$ 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"
}
```

Hasilkan dan verifikasi MAC

Kode Otentikasi Pesan (MAC) biasanya digunakan untuk mengautentikasi integritas pesan (apakah sudah dimodifikasi). Hash kriptografi seperti HMAC (Hash Based Message Authentication Code), CBC-MAC dan CMAC (Cipher-based Message Authentication Code) memberikan jaminan tambahan kepada pengirim MAC dengan memanfaatkan kriptografi. HMAC didasarkan pada fungsi hash sementara CMAC didasarkan pada blok cipher. Layanan ini juga mendukung ISO9797 Algoritma 1 dan 3 yang merupakan jenis CBC-. MACs

Semua algoritma MAC dari layanan ini menggabungkan fungsi hash kriptografi dan kunci rahasia bersama. Mereka mengambil pesan dan kunci rahasia, seperti materi kunci dalam kunci, dan mengembalikan tag atau mac unik. Jika bahkan satu karakter pesan berubah, atau jika kunci rahasia berubah, tag yang dihasilkan sama sekali berbeda. Dengan membutuhkan kunci rahasia, kriptografi MACs juga memberikan keaslian; tidak mungkin untuk menghasilkan mac identik tanpa kunci rahasia. Kriptografi kadang-kadang MACs disebut tanda tangan simetris, karena mereka bekerja seperti tanda tangan digital, tetapi menggunakan satu kunci untuk penandatanganan dan verifikasi.

AWS Kriptografi Pembayaran mendukung beberapa jenis MACs:

ISO9797 ALGORITMA 1

Ditandai dengan `KeyUsage` dari _. ISO9797 ALGORITHM1 Jika bidang bukan kelipatan ukuran blok (8 byte/16 karakter hex untuk TDES, 16 byte/32 karakter untuk AES, Kriptografi Pembayaran secara otomatis menerapkan Metode Padding 1. AWS ISO9797 Jika metode padding lain diperlukan, Anda dapat menerapkannya sebelum menelepon layanan.

ISO9797 ALGORITMA 3 (MAC Eceran)

Ditandai dengan `KeyUsage` dari _. ISO9797 ALGORITHM3 Aturan padding yang sama berlaku sebagai Algoritma 1

ISO9797 ALGORITMA 5 (CMAC)

Ditandai dengan `KeyUsage` _M6_ISO_9797_5_CMAC_KEY TR31

HMAC

Dilambangkan dengan `KeyUsage` TR31 _M7_HMAC_KEY termasuk HMAC_, HMAC_, HMAC_ dan HMAC_ SHA224 SHA256 SHA384 SHA512

AS2805.4.1 MAC

Ditandai dengan `KeyUsage` TR31 _M0_ISO_16609_MAC_KEY. Untuk detail lebih lanjut tentang AS2805, lihat ???

DUKPT MAC

DUKPT MAC biasanya digunakan untuk mengkonfirmasi sumber dan muatan terminal to/from pembayaran pesan. Ini memperoleh kunci menggunakan teknik derivasi DUKPT dan kemudian melakukan MAC. Kunci yang digunakan dengan opsi ini dilambangkan dengan `KeyUsage` TR31 _B0_BASE_DERIVATION_KEY.

EMV MAC

EMV MAC biasanya disebut sebagai kunci integritas dalam dokumentasi EMV. Ini memperoleh kunci menggunakan teknik derivasi EMV dan kemudian menggunakan ISO9797 _ secara internal. ALGORITHM3 Ini biasanya digunakan untuk mengirim skrip penerbit ke kartu chip untuk pemrograman ulang. Kunci yang digunakan dengan opsi ini dilambangkan dengan `KeyUsage` _E2_EMV_MKEY_INTEGRITY TR31. Jika Anda berdua mengirim skrip dan memperbarui pin offline, lihat GenerateMacEmvPinChangeyang melakukan kedua operasi ini.

Topik

- Hasilkan MAC
- Verifikasi MAC

Hasilkan MAC

Generate MAC API digunakan untuk mengautentikasi data terkait kartu, seperti melacak data dari strip magnetik kartu, dengan menggunakan kunci kriptografi yang dikenal untuk menghasilkan MAC (Message Authentication Code) untuk validasi data antara pihak pengirim dan penerima. Data yang digunakan untuk menghasilkan MAC termasuk data pesan, kunci enkripsi MAC rahasia dan algoritma MAC untuk menghasilkan nilai MAC yang unik untuk transmisi. Pihak penerima MAC akan menggunakan data pesan MAC yang sama, kunci enkripsi MAC, dan algoritma untuk mereproduksi nilai MAC lain untuk perbandingan dan otentikasi data. Bahkan jika satu karakter pesan berubah atau kunci MAC yang digunakan untuk verifikasi tidak identik, nilai MAC yang dihasilkan berbeda. API mendukung ISO 9797-1 Algoritma 1 dan ISO 9797-1 Algoritma 3 MAC (menggunakan kunci MAC statis dan kunci DUKPT turunan), kunci enkripsi HMAC dan EMV MAC untuk operasi ini.

Nilai masukan untuk `message-data` harus data HexBinary.

Untuk informasi selengkapnya tentang semua opsi untuk API ini, lihat GenerateMacdan VerifyMac.

Parameter opsional mac-length memungkinkan Anda untuk memotong nilai output (meskipun ini juga dapat dilakukan dalam kode Anda). Panjang 8 mengacu pada 8 byte atau 16 karakter hex.

Kunci MAC dapat dibuat dengan Kriptografi AWS Pembayaran dengan menelepon CreateKeyatau diimpor dengan menelepon ImportKey.

> **Note**
>
> Algoritma CMAC dan HMAC tidak memerlukan padding. Semua yang lain mengharuskan data dimasukkan ke ukuran blok algoritme, yang merupakan kelipatan 8 byte (16 karakter hex) untuk TDES dan 16 byte (32 karakter hex) untuk AES.

Contoh

- Menghasilkan HMAC
- Hasilkan MAC menggunakan ISO 9797-1 Algoritma 3
- Hasilkan MAC menggunakan CMAC
- Hasilkan MAC menggunakan DUKPT CMAC

Menghasilkan HMAC

Dalam contoh ini, kita akan menghasilkan HMAC (Hash Based Message Authentication Code) untuk otentikasi data kartu menggunakan algoritma `HMAC_SHA256` HMAC dan kunci enkripsi HMAC. Kuncinya harus KeyUsage disetel ke `TR31_M7_HMAC_KEY` dan KeyModesOfUse ke`Generate`. Panjang hash (misalnya 256) didefinisikan ketika kunci dibuat dan tidak dapat dimodifikasi.

Parameter panjang mac-opsional akan memangkas output MAC, meskipun ini dapat dilakukan di luar layanan juga. Nilai ini dalam byte, sehingga nilai 16 akan mengharapkan string hex dengan panjang 32.

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
```

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

Hasilkan MAC menggunakan ISO 9797-1 Algoritma 3

Dalam contoh ini, kami akan menghasilkan MAC menggunakan ISO 9797-1 Algorithm 3 (Retail MAC) untuk otentikasi data kartu. Kuncinya harus KeyUsage disetel ke `TR31_M3_ISO_9797_3_MAC_KEY` dan KeyModesOfUse ke`Generate`.

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes="Algorithm=ISO9797_ALGORITHM3"
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kwapwa6qaifllw2h",
    "KeyCheckValue": "2976EA",
    "Mac": "A8F7A73DAF87B6D0"
}
```

Hasilkan MAC menggunakan CMAC

CMAC paling sering digunakan ketika kuncinya adalah AES tetapi juga mendukung TDES. Dalam contoh ini, kami akan menghasilkan MAC menggunakan CMAC (ISO 9797-1 Algorithm 5) untuk otentikasi data kartu dengan kunci AES. Kuncinya harus KeyUsage disetel ke `TR31_M6_ISO_9797_5_CMAC_KEY` dan KeyModesOfUse ke`Generate`.

Example

```
$ aws payment-cryptography-data generate-mac \
    --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
    --message-data
 "3b3130383834393030313037333934313533d32343038323236303030373030303f33" \
    --generation-attributes Algorithm="CMAC"
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "C1EB8F",
    "Mac": "1F8C36E63F91E4E93DF7842BF5E2E5F7"
}
```

Hasilkan MAC menggunakan DUKPT CMAC

Dalam contoh ini, kita akan menghasilkan MAC menggunakan DUKPT (Derived Unique Key Per Transaction) dengan CMAC untuk otentikasi data kartu. Kunci harus KeyUsage disetel ke `TR31_B0_BASE_DERIVATION_KEY` dan KeyModesOfUse `DeriveKey` disetel ke true. Kunci DUKPT memperoleh kunci unik untuk setiap transaksi menggunakan Kunci Derivasi Dasar (BDK) dan Nomor Seri Kunci (KSN).

Example

```
$ aws payment-cryptography-data generate-mac --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/qnobl5lghrzunce6 --message-data
 "3b31303838343930303130373339343153d32343038323236303030373030303f33" --generation-
attributes="DukptCmac={KeySerialNumber="932A6E954ABB32DD00000001",Direction=BIDIRECTIONAL}"
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "C1EB8F"
}
```

Verifikasi MAC

Verifikasi MAC API digunakan untuk memverifikasi MAC (Kode Otentikasi Pesan) untuk otentikasi data terkait kartu. Itu harus menggunakan kunci enkripsi yang sama yang digunakan selama menghasilkan MAC untuk menghasilkan kembali nilai MAC untuk otentikasi. Kunci enkripsi MAC dapat dibuat dengan Kriptografi AWS Pembayaran dengan menelepon CreateKeyatau diimpor dengan menelepon ImportKey. API mendukung kunci enkripsi DUKPT MAC, HMAC dan EMV MAC untuk operasi ini.

Jika nilai diverifikasi, maka parameter respons `MacDataVerificationSuccessful` akan kembali`Http/200`, jika tidak `Http/400` dengan pesan yang menunjukkan itu`Mac verification failed`.

Contoh

- Verifikasi HMAC
- Verifikasi MAC menggunakan DUKPT CMAC

Verifikasi HMAC

Dalam contoh ini, kami akan memverifikasi HMAC (Hash Based Message Authentication Code) untuk otentikasi data kartu menggunakan algoritma `HMAC_SHA256` HMAC dan kunci enkripsi HMAC. Kunci harus KeyUsage disetel ke `TR31_M7_HMAC_KEY` dan KeyModesOfUse `Verify` disetel ke true.

Example

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

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

Verifikasi MAC menggunakan DUKPT CMAC

Dalam contoh ini, kami akan memverifikasi MAC menggunakan DUKPT (Derived Unique Key Per Transaction) dengan CMAC untuk otentikasi data kartu. Kunci harus KeyUsage disetel ke `TR31_B0_BASE_DERIVATION_KEY` dan KeyModesOfUse `DeriveKey` disetel ke true. Kunci DUKPT memperoleh kunci unik untuk setiap transaksi menggunakan Kunci Derivasi Dasar (BDK) dan Nomor Seri Kunci (KSN). Nilai DukptKeyVariant harus cocok antara pengirim dan penerima. REQUEST biasanya akan digunakan dari terminal ke backend, VERIFIKASI dari backend ke terminal dan BIDIRECTIONAL ketika satu kunci digunakan di kedua arah.

Example

```
$ aws payment-cryptography-data verify-mac \
     --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi \
     --message-data
 "3b343038383439303031303733393431353d323430383232363030303730303033f33" \
     --mac D8E804EE74BF1D909A2C01C0BDE8EF34 \
     --verification-attributes
 DukptCmac='{"KeySerialNumber":"932A6E954ABB32DD00000001","DukptKeyVariant":"BIDIRECTIONAL"}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
    "KeyCheckValue": "C1EB8F"
}
```

Kunci yang valid untuk operasi kriptografi

Kunci tertentu hanya dapat digunakan untuk operasi tertentu. Selain itu, beberapa operasi dapat membatasi mode penggunaan kunci untuk kunci. Silakan lihat tabel berikut untuk kombinasi yang diizinkan.

> **Note**
>
> Kombinasi tertentu, meskipun diizinkan, dapat menciptakan situasi yang tidak dapat digunakan seperti menghasilkan kode CVV (`generate`) tetapi kemudian tidak dapat memverifikasinya. (`verify`)

Topik

- GenerateCardData
- VerifyCardData
- GeneratePinData (untuk VISA/ABA skema)
- GeneratePinData (untukIBM3624)
- VerifyPinData (untuk VISA/ABA skema)

- VerifyPinData (untukIBM3624)
- Dekripsi Data
- Enkripsi Data
- Terjemahkan Pin Data
- Hasilkan/Verifikasi MAC
- GenerateMacEmvPinChange
- VerifyAuthRequestCryptogram
- Kunci Impor/Ekspor
- Jenis kunci yang tidak digunakan

GenerateCardData

Titik Akhir API	Operasi atau Algoritma Kriptografi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
GenerateCardData	• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KUNCI • TDES_3KUNCI	{Hasilkan = benar}, {Hasilkan = benar, Verifikasi = benar}
GenerateCardData	• CARD_VERIFICATION_VALUE_1 • CARD_VERIFICATION_VALUE_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KUNCI	{Hasilkan = benar}, {Hasilkan = benar, Verifikasi = benar}

Titik Akhir API	Operasi atau Algoritma Kriptografi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
GenerateCardData	• CARDHOLDER_AUTHENTICATION_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_LAINNYA	• TDES_2KUNCI	{ DeriveKey = benar}
GenerateCardData	• DYNAMIC_CARD_VERIFICATION_CODE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KUNCI	{ DeriveKey = benar}
GenerateCardData	• DYNAMIC_CARD_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_LAINNYA	• TDES_2KUNCI	{ DeriveKey = benar}

VerifyCardData

Operasi atau Algoritma Kriptografi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KUNCI • TDES_3KUNCI	{Hasilkan = benar}, {Hasilkan = benar, Verifikasi = benar}

Operasi atau Algoritma Kriptografi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
• CARD_VERIFICATION_VALUE_1 • CARD_VERIFICATION_VALUE_2	TR31_C0_CARD_VERIFICATION_KEY	• TDES_2KUNCI	{Hasilkan = benar}, {Hasilkan = benar, Verifikasi = benar}
• CARDHOLDER_AUTHENTICATION_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_LAINNYA	• TDES_2KUNCI	{ DeriveKey = benar}
• DYNAMIC_CARD_VERIFICATION_CODE	TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS	• TDES_2KUNCI	{ DeriveKey = benar}
• DYNAMIC_CARD_VERIFICATION_VALUE	TR31_E6_EMV_MKEY_LAINNYA	• TDES_2KUNCI	{ DeriveKey = benar}

GeneratePinData (untuk VISA/ABA skema)

`VISA_PIN or VISA_PIN_VERIFICATION_VALUE`

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Enkripsi PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI	• {Encrypt = true, Wrap = true} • {Encrypt = true, Decrypt = true,

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Kunci Pembuatan PIN	TR31_V2_VISA_PIN_VERIFICATION_KEY	• TDES_3KUNCI	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar}

GeneratePinData (untuk**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Enkripsi PIN	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI	Untuk IBM3624 _NATURAL_PIN, _RANDOM_PIN, _PIN_FROM_OFFSET IBM3624 IBM3624 • {Encrypt = true, Wrap = true} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			• { NoRestrictions = benar} Untuk IBM3624 _PIN_OFFSET • {Encrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Kunci Pembuatan PIN	TR31_V1_ IBM3624 _PIN_VERI FICATION_KEY	• TDES_3KUNCI	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar}

VerifyPinData (untuk VISA/ABA skema)

`VISA_PIN`

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Enkripsi PIN	TR31_P0_P IN_ENCRYP TION_KEY	• TDES_2KUNCI • TDES_3KUNCI	• {Dekripsi = benar, Buka bungkus = benar}

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			• {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Kunci Pembuatan PIN	TR31_V2_V ISA_PIN_VERIFICATI ON_KEY	• TDES_3KUNCI	• {Verifikasi = benar} • {Hasilkan = benar, Verifikasi = benar}

VerifyPinData (untuk**IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Enkripsi PIN	TR31_P0_P IN_ENCRYP TION_KEY	• TDES_2KUNCI • TDES_3KUNCI	Untuk IBM3624 _NATURAL_PIN, _RANDOM_P IN, _PIN_FROM _OFFSET IBM3624 IBM3624 • {Dekripsi = benar, Buka bungkus = benar} • {Encrypt = true, Decrypt = true,

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Kunci Verifikasi PIN	TR31_V1_ IBM3624 _PIN_VERI FICATION_KEY	• TDES_3KUNCI	• {Verifikasi = benar} • {Hasilkan = benar, Verifikasi = benar}

Dekripsi Data

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
DUKPT	TR31_B0_B ASE_DERIV ATION_KEY	• TDES_2KUNCI • AES_128 • AES_192 • AES_256	• { DeriveKey = benar} • { NoRestrictions = benar}
EMV	TR31_E1_E MV_MKEY_K ERAHASIAAN TR31_E6_E MV_MKEY_LAINNYA	• TDES_2KUNCI	• { DeriveKey = benar}
RSA	TR31_D1_A SYMMETRIC _KEY_FOR_ DATA_ENKRIPSI	• RSA_2048 • RSA_3072 • RSA_4096	• {Dekripsi = benar, buka bungkus = Benar} • {encrypt=True, wrap=True,

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			Dekripsi = true, buka bungkus=Benar}
Tombol simetris	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Dekripsi = benar, buka bungkus = Benar} • {encrypt=True, wrap=True, Dekripsi = true, buka bungkus=Benar} • { NoRestrictions = benar}

Enkripsi Data

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KUNCI • AES_128 • AES_192 • AES_256	• { DeriveKey = benar} • { NoRestrictions = benar}
EMV	TR31_E1_EMV_MKEY_KERAHASIAAN TR31_E6_EMV_MKEY_LAINNYA	• TDES_2KUNCI	• { DeriveKey = benar}

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
RSA	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENKRIPSI	• RSA_2048 • RSA_3072 • RSA_4096	• {Enkripsi = benar, bungkus = benar} • {encrypt=True, wrap=True, Dekripsi = true, buka bungkus=Benar}
Tombol simetris	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Enkripsi = benar, bungkus = benar} • {encrypt=True, wrap=True, Dekripsi = true, buka bungkus=Benar} • { NoRestrictions = benar}

Terjemahkan Pin Data

Arahan	Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Sumber Data Masuk	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KUNCI • AES_128 • AES_192	• { DeriveKey = benar} • { NoRestrictions = benar}

Arahan	Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
			• AES_256	
Sumber Data Masuk	Non-DUKPT (PEK, AWK, IWK, dll)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Dekripsi = benar, Buka bungkus = benar} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Target Data Keluar	DUKPT	TR31_B0_BASE_DERIVATION_KEY	• TDES_2KUNCI • AES_128 • AES_192 • AES_256	• { DeriveKey = benar} • { NoRestrictions = benar}
Target Data Keluar	Non-DUKPT (PEK, IWK, AWK, dll)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap = true} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}

Hasilkan/Verifikasi MAC

Kunci MAC digunakan untuk membuat hash kriptografi message/body dari data. Tidak disarankan untuk membuat kunci dengan mode penggunaan kunci terbatas karena Anda tidak akan dapat melakukan operasi pencocokan. Namun, Anda mungkin import/export kunci dengan hanya satu operasi jika sistem lain dimaksudkan untuk melakukan setengah lainnya dari pasangan operasi.

Penggunaan Kunci yang Diizinkan	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci MAC	TR31_M1_ISO_9797_1_MAC_KEY	• TDES_2KUNCI • TDES_3KUNCI	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar} • {Verifikasi = benar} • {Menghasilkan = benar}
Kunci MAC (MAC Ritel)	TR31_M1_ISO_9797_3_MAC_KEY	• TDES_2KUNCI • TDES_3KUNCI	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar} • {Verifikasi = benar} • {Menghasilkan = benar}
Kunci MAC (CMAC)	TR31_M6_ISO_9797_5_CMAC_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar} • {Verifikasi = benar} • {Menghasilkan = benar}

Penggunaan Kunci yang Diizinkan	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci MAC (HMAC)	TR31_M7_HMAC_KUNCI	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar} • {Verifikasi = benar}
Kunci MAC (AS2805)	TR31_M0_ISO_16609_MAC_KEY	• TDES_2KUNCI • TDES_3KUNCI	• {Menghasilkan = benar} • {Hasilkan = benar, Verifikasi = benar} • {Verifikasi = benar}

GenerateMacEmvPinChange

GenerateMacEmvPinChange menggabungkan pembuatan MAC dan enkripsi PIN untuk operasi perubahan PIN offline EMV. Operasi ini memerlukan dua jenis kunci yang berbeda: kunci integritas untuk pembuatan MAC dan kunci kerahasiaan untuk enkripsi PIN.

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Integritas Pesan Aman	TR31_E2_EMV_MKEY_INTEGRITY	• TDES_2KUNCI	• { NoRestrictions = benar}
Kunci Kerahasiaan Pesan Aman	TR31_E1_EMV_MKEY_KERAHASIAAN	• TDES_2KUNCI	• { DeriveKey = benar}

Tipe Kunci	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
PIN PEK saat ini (Kunci Enkripsi PIN)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Dekripsi = benar, Buka bungkus = benar} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}
PIN PEK Baru (Kunci Enkripsi PIN)	TR31_P0_PIN_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Dekripsi = benar, Buka bungkus = benar} • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true} • { NoRestrictions = benar}
Kunci ARQC **Note** Hanya berlaku untuk skema derivasi Visa dan Amex.	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• TDES_2KUNCI	• { DeriveKey = benar}

VerifyAuthRequestCryptogram

Penggunaan Kunci yang Diizinkan	Opsi EMV	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
• OPSI A • OPSI B	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• TDES_2KUNCI	• { DeriveKey = benar}

Kunci Impor/Ekspor

Tipe operasi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
Kunci Pembungkus TR-31	TR31_K1_KEY_BLOCK_PROTECTION_KEY TR31_K0_KEY_ENCRYPTION_KEY	• TDES_2KUNCI • TDES_3KUNCI • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap = true} (hanya ekspor) • {Decrypt = true, Unwrap = true} (hanya impor) • {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}
Impor CA tepercaya	TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE	• RSA_2048 • RSA_3072 • RSA_4096	• {Verifikasi = benar}
Impor sertifikat kunci publik untuk enkripsi asimetris	TR31_D1_ASYMMETRIC	• RSA_2048 • RSA_3072	• {Encrypt=true, wrap=True}

Tipe operasi	Penggunaan Kunci yang Diizinkan	Algoritma Kunci yang Diizinkan	Kombinasi yang diizinkan dari mode penggunaan utama
	_KEY_FOR_ DATA_ENKRIPSI	• RSA_4096	
Kunci yang digunakan untuk algoritma kesepakatan kunci seperti ECDH	TR31_K3_A SYMMETRIC _KEY_FOR_ KEY_AGREEMENT	• ECC_NIST_P256 • ECC_NIST_P384 • ECC_NIST_P521	• { DeriveKey = benar}

Jenis kunci yang tidak digunakan

Jenis kunci berikut saat ini tidak digunakan oleh Kriptografi AWS Pembayaran

- TR31_P1_PIN_GENERATION_KEY

Kasus penggunaan umum

AWS Kriptografi Pembayaran mendukung banyak operasi kriptografi pembayaran yang khas. Topik berikut bertindak sebagai panduan tentang cara menggunakan operasi ini untuk kasus penggunaan umum yang umum. Untuk daftar semua perintah, silakan tinjau API Kriptografi AWS Pembayaran.

Topik

Emiten dan prosesor penerbit

Kasus penggunaan penerbit biasanya terdiri dari beberapa bagian. Bagian ini diatur oleh fungsi (seperti bekerja dengan pin). Dalam sistem produksi, kunci biasanya dicakup ke bin kartu tertentu dan dibuat selama pengaturan bin daripada sebaris seperti yang ditunjukkan di sini.

Topik

Fungsi Umum

Topik

Hasilkan pin acak dan PVV terkait lalu verifikasi nilainya

Topik

Buat kunci (s)

Untuk menghasilkan pin acak dan PVV, Anda memerlukan dua kunci, Kunci Verifikasi Pin (PVK) untuk menghasilkan PVV dan Kunci Enkripsi Pin untuk mengenkripsi pin. Pin itu sendiri dihasilkan secara acak dengan aman di dalam layanan dan tidak terkait dengan salah satu kunci secara kriptografi.

PGK harus menjadi kunci algoritma TDES_2KEY berdasarkan algoritma PVV itu sendiri. PEK dapat berupa TDES_2KEY, TDES_3KEY atau AES_128. Dalam hal ini, karena PEK ditujukan untuk penggunaan internal dalam sistem Anda, AES_128 akan menjadi pilihan yang baik. Jika PEK digunakan untuk pertukaran dengan sistem lain (misalnya jaringan kartu, pengakuisisi, ATMs) atau sedang dipindahkan sebagai bagian dari migrasi, TDES_2KEY mungkin menjadi pilihan yang lebih tepat untuk alasan kompatibilitas.

Buat PEK

```
$ aws payment-cryptography create-key \
               --exportable
               --key-attributes
 KeyAlgorithm=AES_128,KeyUsage=TR31_P0_PIN_ENCRYPTION_KEY,\
                   KeyClass=SYMMETRIC_KEY,\
                   KeyModesOfUse='{Encrypt=true,Decrypt=true,Wrap=true,Unwrap=true}' --
tags='[{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
               "Key": {
                   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
                   "KeyAttributes": {
```

```
                    "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "AES_128",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "7CC9E2",
                "KeyCheckValueAlgorithm": "CMAC",
                "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"
            }
        }
```

Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt. Anda membutuhkannya di langkah berikutnya.

Buat PVK

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyMo
  --tags='[{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza",
```

```
                    "KeyAttributes": {
                        "KeyUsage": "TR31_V2_VISA_PIN_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": "51A200",
                    "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"
                }
            }
```


Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza. Anda membutuhkannya di langkah berikutnya.

Hasilkan pin acak, hasilkan PVV dan kembalikan PIN dan PVV terenkripsi

Example

Dalam contoh ini, kami akan menghasilkan pin 4 digit baru (acak) di mana output akan dienkripsi `PIN block` (. PinData PinBlock) dan a PVV (pinData. VerificationValue). Input kuncinya adalahPAN, `Pin Verification Key` (juga dikenal sebagai kunci pembuatan pin), `Pin Encryption Key` dan format Blok PIN.

Perintah ini mengharuskan kuncinya bertipe`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"
            }
        }
```

Validasi PIN terenkripsi menggunakan metode PVV

Example

Dalam contoh ini, kita akan memvalidasi PIN untuk PAN tertentu. PIN biasanya disediakan oleh pemegang kartu atau pengguna selama waktu transaksi untuk validasi dan dibandingkan dengan nilai pada file (input dari pemegang kartu diberikan sebagai nilai terenkripsi dari terminal atau penyedia hulu lainnya). Untuk memvalidasi input ini, nilai berikut juga akan diberikan saat runtime - Pin terenkripsi, kunci yang digunakan untuk mengenkripsi pin input (sering disebut sebagai IWK), PAN dan nilai untuk memverifikasi terhadap (baik a atau). PVV `PIN offset`

Jika Kriptografi AWS Pembayaran dapat memvalidasi pin, http/200 dikembalikan. Jika pin tidak divalidasi, itu akan mengembalikan http/400.

```
$ 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",
}
```

Menghasilkan atau memverifikasi CVV untuk kartu tertentu

CVV atau CVV1 merupakan nilai yang secara tradisional tertanam dalam strip magnetik kartu. Ini tidak sama dengan CVV2 (terlihat oleh pemegang kartu dan untuk digunakan untuk pembelian online).

Langkah pertama adalah membuat kunci. Untuk tutorial ini, Anda membuat kunci 3DES (2KEY TDES) panjang ganda CVK.

> **Note**
>
> CVV, CVV2 dan iCVV semuanya menggunakan algoritma yang serupa jika tidak identik tetapi memvariasikan data input. Semua menggunakan jenis kunci yang sama TR31 _C0_CARD_VERIFICATION_KEY tetapi disarankan untuk menggunakan kunci terpisah untuk setiap tujuan. Ini dapat dibedakan menggunakan and/or tag alias seperti pada contoh di bawah ini.

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr",
```

```
            "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": "DE89F9",
            "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"
        }
    }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east -2:111122223333:key/r52o3wbqxyf6qlqr. Anda membutuhkannya di langkah berikutnya.

Menghasilkan CVV

Example

Dalam contoh ini, kami akan menghasilkan CVV untuk PAN tertentu dengan inputPAN, kode layanan (sebagaimana didefinisikan oleh ISO/IEC 7813) dari 121 dan tanggal kedaluwarsa kartu.

Untuk semua parameter yang tersedia, lihat CardVerificationValue1 di panduan referensi API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=121}'
```

```
                {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/r52o3wbqxyf6qlqr",
                    "KeyCheckValue": "DE89F9",
                    "ValidationData": "801"
                }
```

Validasi CVV

Example

Dalam contoh ini, kami akan memverifikasi CVV untuk PAN tertentu dengan input CVK,, kode layanan 121PAN, tanggal kedaluwarsa kartu dan CVV yang disediakan selama transaksi untuk divalidasi.

Untuk semua parameter yang tersedia, lihat, CardVerificationValue1 di panduan referensi API.

> **Note**
>
> CVV bukan nilai yang dimasukkan pengguna (seperti CVV2) tetapi biasanya disematkan pada magstripe. Pertimbangan harus diberikan apakah harus selalu memvalidasi saat disediakan.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/r52o3wbqxyf6qlqr
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=121} --validation-data 801
```

```
{
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
r52o3wbqxyf6qlqr",
                    "KeyCheckValue": "DE89F9",
                    "ValidationData": "801"
```

```
}
```

Menghasilkan atau memverifikasi CVV2 untuk kartu tertentu

CVV2adalah nilai yang secara tradisional disediakan di bagian belakang kartu dan digunakan untuk pembelian online. Untuk kartu virtual, mungkin juga ditampilkan di aplikasi atau layar. Secara kriptografis, itu sama dengan CVV1 tetapi dengan nilai kode layanan yang berbeda.

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVV2"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
               "Key": {
                   "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                   "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": "AEA5CD",
                   "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"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/7f7g4spf3xcklhzu. Anda membutuhkannya di langkah berikutnya.

Menghasilkan CVV2

Example

Dalam contoh ini, kami akan menghasilkan CVV2untuk PAN tertentu dengan input PAN dan tanggal kedaluwarsa kartu.

Untuk semua parameter yang tersedia, lihat CardVerificationValue2 di panduan referensi API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/7f7g4spf3xcklhzu
 --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue2='{CardExpiryDate=1127}'
```

```
                {
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                    "KeyCheckValue": "AEA5CD",
                    "ValidationData": "321"
                }
```

Validasi CVV2

Example

Dalam contoh ini, kami akan memverifikasi PAN yang diberikan dengan input CVK, PAN dan tanggal kedaluwarsa kartu dan CVV yang disediakan CVV2selama transaksi untuk divalidasi.

Untuk semua parameter yang tersedia, lihat, CardVerificationValue2 di panduan referensi API.

> **Note**
>
> CVV2 dan input lainnya adalah nilai yang dimasukkan pengguna. Dengan demikian, ini belum tentu merupakan tanda masalah yang gagal divalidasi secara berkala.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/7f7g4spf3xcklhzu
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue2='{CardExpiryDate=1127} --validation-data 321
```

```
{
                      "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu",
                      "KeyCheckValue": "AEA5CD",
                      "ValidationData": "801"
   }
```

Buat atau verifikasi iCVV untuk kartu tertentu

ICVV menggunakan algoritma yang sama dengan CVV2 CVV/tetapi iCVV tertanam di dalam kartu chip. Kode layanannya adalah 999.

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"ICVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
              "Key": {
                  "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
c7dsi763r6s7lfp3",
                  "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": "1201FB",
                "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"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/c7dsi763r6s7lfp3. Anda membutuhkannya di langkah berikutnya.

Menghasilkan iCVV

Example

Dalam contoh ini, kami akan menghasilkan iCVV untuk PAN tertentu dengan inputPAN, kode layanan (sebagaimana didefinisikan oleh ISO/IEC 7813) dari 999 dan tanggal kedaluwarsa kartu.

Untuk semua parameter yang tersedia, lihat CardVerificationValue1 di panduan referensi API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
c7dsi763r6s7lfp3 --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999}'
```

```
                    {
                        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/c7dsi763r6s7lfp3",
                        "KeyCheckValue": "1201FB",
                        "ValidationData": "532"
                    }
```

Validasi iCVV

Example

Untuk validasi, inputnya adalah CVK, kode layanan 999PAN, tanggal kedaluwarsa kartu dan iCVV yang disediakan selama transaksi untuk divalidasi.

Untuk semua parameter yang tersedia, lihat, CardVerificationValue1 di panduan referensi API.

> **Note**
>
> iCVV bukan nilai yang dimasukkan pengguna (seperti CVV2) tetapi biasanya disematkan pada kartu. EMV/chip Pertimbangan harus diberikan apakah harus selalu memvalidasi saat disediakan.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/c7dsi763r6s7lfp3
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=1127,ServiceCode=999} --validation-data 532
```

```
{
                    "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/c7dsi763r6s7lfp3",
                    "KeyCheckValue": "1201FB",
                    "ValidationData": "532"
   }
```

Verifikasi EMV ARQC dan hasilkan ARPC

ARQC (Authorization Request Cryptogram) adalah kriptogram yang dihasilkan oleh kartu EMV (chip) dan digunakan untuk memvalidasi detail transaksi serta penggunaan kartu resmi. Ini menggabungkan data dari kartu, terminal dan transaksi itu sendiri.

Pada waktu validasi di backend, input yang sama diberikan ke Kriptografi AWS Pembayaran, kriptogram dibuat ulang secara internal dan ini dibandingkan dengan nilai yang diberikan dengan transaksi. Dalam hal ini, ini mirip dengan MAC. EMV 4.4 Buku 2 mendefinisikan tiga aspek fungsi ini - metode derivasi kunci (dikenal sebagai kunci sesi umum - CSK) untuk menghasilkan kunci transaksi satu kali, muatan minimum dan metode untuk menghasilkan respons (ARPC).

Skema kartu individu dapat menentukan bidang transaksional tambahan untuk dimasukkan atau urutan bidang tersebut muncul. Skema derivasi spesifik lainnya (umumnya tidak digunakan lagi) juga ada dan tercakup di tempat lain dalam dokumentasi ini.

Untuk informasi selengkapnya, lihat VerifyCardValidationDatadi panduan API.

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN18"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                    "KeyAttributes": {
                        "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                        "KeyClass": "SYMMETRIC_KEY",
                        "KeyAlgorithm": "TDES_2KEY",
                        "KeyModesOfUse": {
                            "Encrypt": false,
                            "Decrypt": false,
                            "Wrap": false,
                            "Unwrap": false,
```

```
                        "Generate": false,
                        "Sign": false,
                        "Verify": false,
                        "DeriveKey": true,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "08D7B4",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Anda membutuhkannya di langkah berikutnya.

Menghasilkan ARQC

ARQC dihasilkan secara eksklusif oleh kartu EMV. Dengan demikian, Kriptografi AWS Pembayaran tidak memiliki fasilitas untuk menghasilkan muatan seperti itu. Untuk tujuan pengujian, sejumlah perpustakaan tersedia secara online yang dapat menghasilkan muatan yang sesuai serta nilai yang diketahui yang umumnya disediakan oleh berbagai skema.

Validasi ARQC

Example

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. ARPC (respons) secara opsional dapat diberikan dan dimasukkan dalam respons setelah ARQC divalidasi.

```
$ 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
 00000000170000000000000008400080008000084016051700000000093800000B1F220103000000000000000000000
 --session-key-derivation-attributes='{"EmvCommon":
{"ApplicationTransactionCounter":"000B",
```

```
 "PanSequenceNumber":"01","PrimaryAccountNumber":"9137631040001422"}}' --auth-response-
attributes='{"ArpcMethod2":{"CardStatusUpdate":"12345678"}}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4",
    "AuthResponseValue":"2263AC85"
}
```

Hasilkan dan Verifikasi EMV MAC

EMV MAC adalah MAC menggunakan input dari kunci turunan EMV dan kemudian melakukan ISO9797 -3 (Retail) MAC atas data yang dihasilkan. EMV MAC biasanya digunakan untuk mengirim perintah ke kartu EMV seperti skrip buka blokir.

> **Note**
>
> AWS Kriptografi Pembayaran tidak memvalidasi isi skrip. Silakan berkonsultasi dengan skema atau manual kartu Anda untuk detail tentang perintah tertentu untuk disertakan.

Untuk informasi selengkapnya, lihat MacAlgorithmEmvdi panduan API.

Topik

- Buat kuncinya
- Menghasilkan EMV MAC

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E2_EMV_MKEY_INTEGRITY,KeyClass=SYMMETRIC_KEY,KeyModesOfUs
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN18"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
```

```
        "Key": {
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
            "KeyAttributes": {
                "KeyUsage": "TR31_E2_EMV_MKEY_INTEGRITY",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyAlgorithm": "TDES_2KEY",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "08D7B4",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
        }
    }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Anda membutuhkannya di langkah berikutnya.

Menghasilkan EMV MAC

Aliran tipikal adalah bahwa proses backend akan menghasilkan skrip EMV (seperti membuka blokir kartu), menandatanganinya menggunakan perintah ini (yang memperoleh kunci satu kali khusus untuk satu kartu tertentu) dan kemudian mengembalikan MAC. Kemudian perintah+MAC dikirim ke kartu yang akan diterapkan. Mengirim perintah ke kartu berada di luar cakupan Kriptografi AWS Pembayaran.

> **Note**
>
> Perintah ini dimaksudkan untuk perintah ketika tidak ada data terenkripsi (seperti PIN) yang dikirim. EMV Encrypt dapat dikombinasikan dengan perintah ini untuk menambahkan data terenkripsi ke skrip penerbit sebelum memanggil perintah ini

Data Pesan

Data pesan termasuk header dan perintah APDU. Meskipun ini dapat bervariasi berdasarkan implementasi, contoh ini adalah header APDU untuk membuka blokir (84 24 00 00 08), diikuti oleh ATC (0007) dan kemudian ARQC dari transaksi sebelumnya (999E57 F47CACE). FD0 Layanan tidak memvalidasi isi bidang ini.

Mode Derivasi Kunci Sesi

Bidang ini mendefinisikan bagaimana kunci sesi dihasilkan. EMV_COMMON_SESSION_KEY umumnya digunakan untuk implementasi baru, sedangkan EMV2000 | AMEX | MASTERCARD_SESSION_KEY | VISA dapat digunakan juga.

MajorKeyDerivationMode

EMV Mendefinisikan Mode A, B atau C. Mode A adalah yang paling umum dan Kriptografi AWS Pembayaran saat ini mendukung mode A atau mode B.

PANCI

Nomor rekening, biasanya tersedia di bidang chip 5A atau ISO8583 bidang 2 tetapi juga dapat diambil dari sistem kartu.

PSN

Nomor urutan kartu. Jika tidak digunakan, masukkan 00.

SessionKeyDerivationValue

Ini adalah data derivasi per sesi. Ini bisa berupa ARQC terakhir (ApplicationCryptogram) dari bidang 9F26 atau ATC terakhir dari 9F36 tergantung pada skema derivasi.

Bantalan

Padding diterapkan secara otomatis dan menggunakan metode padding ISO/IEC 9797-1 2.

Example

```
$ aws payment-cryptography-data generate-mac --message-data
 842400000800079 99E57FD0F47CACE --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk --message-
data 8424000008999E57FD0F47CACE0007 --generation-attributes
 EmvMac="{MajorKeyDerivationMode=EMV_OPTION_A,PanSequenceNumber='00',PrimaryAccountNumber='2235
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk",
"KeyCheckValue": "08D7B4",
"Mac":"5652EEDF83EA0D84"
}
```

Hasilkan EMV MAC untuk Perubahan PIN

Perubahan PIN EMV menggabungkan dua operasi: menghasilkan MAC untuk skrip penerbit dan mengenkripsi PIN baru untuk perubahan PIN offline pada kartu chip EMV. Perintah ini hanya diperlukan di negara-negara tertentu di mana pin disimpan pada kartu chip (ini umum untuk negara-negara Eropa). Ini biasanya digunakan ketika pemegang kartu perlu mengubah PIN mereka dan PIN baru harus dikirim dengan aman ke kartu bersama dengan MAC untuk memverifikasi keaslian perintah.

> **Note**
>
> Jika Anda hanya perlu mengirim perintah ke kartu tetapi tidak mengubah PIN, pertimbangkan untuk menggunakan perintah ARPC CSU atau Generate EMV MAC sebagai gantinya.

Untuk informasi selengkapnya, lihat GenerateMacEmvPinChangedi panduan API.

Hasilkan EMV MAC dan PIN terenkripsi untuk perubahan PIN

Operasi ini memerlukan dua kunci: kunci integritas EMV (: TR31 _E2_EMV_MKEY_INTEGRITY) untuk pembuatan MAC dan kunci kerahasiaan EMV (KeyUsage: _E4_EMV_MKEY_CONFIDENTIALITY) untuk enkripsi PIN. KeyUsage TR31 Aliran tipikal adalah bahwa proses backend akan menghasilkan skrip perubahan PIN EMV, yang mencakup MAC untuk skrip penerbit dan PIN baru yang dienkripsi. Perintah dan PIN terenkripsi kemudian dikirim ke kartu untuk memperbarui PIN offline. Mengirim perintah ke kartu berada di luar cakupan Kriptografi AWS Pembayaran.

Data Pesan

Data pesan termasuk perintah APDU untuk skrip penerbit. Layanan tidak memvalidasi isi bidang ini.

Blok PIN Terenkripsi Baru

Blok PIN terenkripsi baru yang akan dikirim ke kartu. Ini harus disediakan sebagai nilai terenkripsi menggunakan kunci enkripsi PIN.

Pengenal PIN PEK Baru

Kunci yang digunakan untuk mengenkripsi PIN baru sebelum diteruskan ke API ini.

Kunci Integritas Pesan Aman

Kunci integritas EMV (KeyUsage: TR31 _E2_EMV_MKEY_INTEGRITY) digunakan untuk pembuatan MAC.

Kunci Kerahasiaan Pesan Aman

Kunci kerahasiaan EMV (KeyUsage: TR31 _E4_EMV_MKEY_CONFIDENTIALITY) digunakan untuk enkripsi PIN.

MajorKeyDerivationMode

EMV mendefinisikan Mode A, B, atau C. Mode A adalah yang paling umum dan Kriptografi AWS Pembayaran saat ini mendukung mode A atau mode B.

Modus

Mode enkripsi, biasanya CBC untuk operasi perubahan PIN.

PANCI

Nomor rekening, biasanya tersedia di bidang chip 5A atau ISO8583 bidang 2 tetapi juga dapat diambil dari sistem kartu.

PanSequenceNumber

Nomor urutan kartu. Jika tidak digunakan, masukkan 00.

ApplicationCryptogram

Ini adalah data derivasi per sesi, biasanya ARQC terakhir dari bidang 9F26.

PinBlockLengthPosition

Menentukan di mana panjang blok PIN dikodekan. Biasanya diatur ke NONE. Periksa spesifikasi skema kartu Anda jika Anda tidak yakin.

PinBlockPaddingType

Menentukan jenis padding untuk blok PIN. Biasanya diatur ke NO_PADDING. Periksa spesifikasi skema kartu Anda jika Anda tidak yakin.

Example

```
$ aws payment-cryptography-data generate-mac-emv-pin-change \
    --message-data 00A4040008A000000004101080D8050000000101 0A04000000000000 \
    --new-encrypted-pin-block 67FB27C75580EFE7 \
    --new-pin-pek-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_0 \
    --secure-messaging-confidentiality-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi \
    --secure-messaging-integrity-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk \
    --derivation-method-attributes
 'EmvCommon={ApplicationCryptogram=1234567890123457,MajorKeyDerivationMode=EMV_OPTION_A,Mode=CB
```

```
{
    "SecureMessagingIntegrityKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk",
    "SecureMessagingIntegrityKeyCheckValue": "08D7B4",
    "SecureMessagingConfidentialityKeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/tqv5yij6wtxx64pi",
    "SecureMessagingConfidentialityKeyCheckValue": "C1EB8F",
    "Mac": "5652EEDF83EA0D84",
    "EncryptedPinBlock": "F1A2B3C4D5E6F7A8"
}
```

Fungsi spesifik jaringan

Topik

- Fungsi khusus visa
- Fungsi khusus Mastercard

- Fungsi spesifik American Express
- Fungsi khusus JCB

Fungsi khusus visa

Topik

- ARQC -/ CVN18CVN22
- ARQC - CVN10
- 3DS CAVV V7
- dCVV (Nilai Verifikasi Kartu Dinamis) - CVN17

ARQC -/ CVN18CVN22

CVN18 dan CVN22 memanfaatkan metode CSK derivasi kunci. Data transaksi yang tepat bervariasi antara kedua metode ini - silakan lihat dokumentasi skema untuk detail tentang pembuatan bidang data transaksi.

ARQC - CVN10

CVN10 adalah metode Visa lama untuk transaksi EMV yang menggunakan derivasi kunci per kartu daripada derivasi sesi (per transaksi) dan juga menggunakan muatan yang berbeda. Untuk informasi tentang isi muatan, silakan hubungi skema untuk detailnya.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN10"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                    "KeyClass": "SYMMETRIC_KEY",
```

```
                "KeyAlgorithm": "TDES_2KEY",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "08D7B4",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
        }
    }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Anda membutuhkannya di langkah berikutnya.

Validasi ARQC

Example

Dalam contoh ini, kami akan memvalidasi ARQC yang dihasilkan menggunakan Visa. CVN10

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. Jika arqc tidak divalidasi, itu akan mengembalikan respons http/400.

```
$ 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
 0000000017000000000000000840008000800008401605170000000093800000B03011203000000 \
```

```
  --session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
  ,"PrimaryAccountNumber":"9137631040001422"}}'
```

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

3DS CAVV V7

Untuk transaksi Visa Secure (3DS), CAVV (Nilai Verifikasi Otentikasi Pemegang Kartu) dihasilkan oleh penerbit Access Control Server (ACS). CAVV adalah bukti bahwa otentikasi pemegang kartu terjadi, unik untuk setiap transaksi otentikasi dan disediakan oleh pihak pengakuisisi dalam pesan otorisasi. CAVV v7 mengikat data tambahan tentang transaksi dengan persetujuan termasuk elemen seperti nama pedagang, jumlah pembelian, dan tanggal pembelian. Dengan cara ini, ini secara efektif merupakan hash kriptografi dari muatan transaksi.

Secara kriptografis, CAVV V7 menggunakan algoritma CVV tetapi masukannya semuanya adalah changed/repurposed. Please consult appropriate third party/Visa dokumentasi tentang cara menghasilkan input untuk menghasilkan muatan CAVV V7.

Buat kuncinya

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CAVV-V7"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
                "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": "F3FB13",
            "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"
        }
    }
```

Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-2:111122223333:key/dnaeyrjgdjjtw6dk. Anda membutuhkannya di langkah berikutnya.

Hasilkan CAVV V7

Example

Dalam contoh ini, kami akan menghasilkan CAVV V7 untuk transaksi tertentu dengan input sebagaimana ditentukan dalam spesifikasi. Perhatikan bahwa untuk algoritme ini, bidang dapat digunakan kembali/digunakan kembali, jadi tidak boleh diasumsikan bahwa label bidang cocok dengan input.

Untuk semua parameter yang tersedia, lihat CardVerificationValue1 di panduan referensi API.

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk --primary-account-number=171234567890123 --generation-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=431}'
```

```
            {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
                "KeyCheckValue": "F3FB13",
                "ValidationData": "491"
            }
```

Validasi CAVV V7

Example

Untuk validasi, inputnya adalah CVK, nilai input yang dihitung dan CAVV yang disediakan selama transaksi untuk divalidasi.

Untuk semua parameter yang tersedia, lihat, CardVerificationValue1 di panduan referensi API.

> **Note**
>
> CAVV bukan nilai yang dimasukkan pengguna (seperti CVV2) tetapi dihitung oleh penerbit ACS. Pertimbangan harus diberikan apakah harus selalu memvalidasi saat disediakan.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/dnaeyrjgdjjtw6dk
 --primary-account-number=171234567890123 --verification-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=431} --validation-data 491
```

```
{
            "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
            "KeyCheckValue": "F3FB13",
            "ValidationData": "491"
}
```

dCVV (Nilai Verifikasi Kartu Dinamis) - CVN17

DCVV (Dynamic Card Verification Value) adalah kriptogram dinamis khusus Visa yang digunakan untuk transaksi EMV nirkontak. Ini dikenal sebagai EMV awal dan memberikan keamanan yang

ditingkatkan dengan menghasilkan nilai verifikasi unik untuk setiap transaksi. DCVV menggunakan input termasuk Primary Account Number (PAN), PAN Sequence Number (PSN), Application Transaction Counter (ATC), nomor tak terduga, dan data track. Ini masih digunakan di beberapa tempat, tetapi sebagian besar telah digantikan oleh algoritma lain seperti CVN18.

Untuk semua parameter yang tersedia, lihat DynamicCardVerificationValuedi panduan referensi API.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"DCVV"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS",
                    "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": "A8E4D2",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
```

```
                "CreateTimestamp": "2025-02-02T11:45:30.648000-08:00",
                "UsageStartTimestamp": "2025-02-02T11:45:30.626000-08:00"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc. Anda membutuhkannya di langkah berikutnya.

Menghasilkan DCVV

Example

Dalam contoh ini, kita akan menghasilkan dCVV untuk transaksi EMV contactless. Input termasuk PAN, Nomor Urutan PAN, Penghitung Transaksi Aplikasi, nomor tak terduga, dan data trek.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=5111112627662122 \
    --generation-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,TrackData
 \
    --validation-data-length 5
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2",
    "ValidationData": "36667"
}
```

Validasi dCVV

Example

Dalam contoh ini, kami akan memvalidasi dCVV yang disediakan selama transaksi. Masukan yang sama yang digunakan untuk pembuatan harus disediakan untuk validasi.

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
```

```
    --primary-account-number=5111112627662122 \
    --validation-data=36667 \
    --verification-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,TrackData
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2"
}
```

Fungsi khusus Mastercard

Topik

- DCVC3
- ARQC -/ CVN14CVN15
- ARQC -/ CVN12CVN13
- 3DS AAV SPA2

DCVC3

DCVC3 mendahului CVN12 skema EMV CSK dan Mastercard dan merupakan pendekatan lain untuk memanfaatkan kunci dinamis. Kadang-kadang digunakan kembali untuk kasus penggunaan lain juga. Dalam skema ini, inputnya adalah data PAN, PSN, Track1/Track2, nomor tak terduga dan penghitung transaksi (ATC).

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"DCVC3"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
hrh6qgbi3sk4y3wq",
```

```
            "KeyAttributes": {
                "KeyUsage": "TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS",
                "KeyClass": "SYMMETRIC_KEY",
                "KeyAlgorithm": "TDES_2KEY",
                "KeyModesOfUse": {
                    "Encrypt": false,
                    "Decrypt": false,
                    "Wrap": false,
                    "Unwrap": false,
                    "Generate": false,
                    "Sign": false,
                    "Verify": false,
                    "DeriveKey": true,
                    "NoRestrictions": false
                }
            },
            "KeyCheckValue": "08D7B4",
            "KeyCheckValueAlgorithm": "ANSI_X9_24",
            "Enabled": true,
            "Exportable": true,
            "KeyState": "CREATE_COMPLETE",
            "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
            "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
        }
    }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east -2:111122223333:key/hrh6qgbi3sk4y3wq. Anda membutuhkannya di langkah berikutnya.

Menghasilkan DCVC3

Example

Meskipun DCVC3 biasanya dihasilkan oleh kartu chip, itu juga dapat dibuat secara manual seperti dalam contoh ini

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk
 --primary-account-number=5413123456784808 --generation-attributes
 DynamicCardVerificationCode='{ApplicationTransactionCounter=0000,TrackData=524106000000069D13
```

```
{
```

```
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
    "KeyCheckValue": "08D7B4",
    "ValidationData": "865"
  }
```

Validasi DCVC3

Example

Dalam contoh ini, kami akan memvalidasi. DCVC3 Perhatikan bahwa ATC harus disediakan sebagai nomor hex misalnya penghitung 11 harus direpresentasikan sebagai 000B. Layanan mengharapkan 3 digit DCVC3, jadi jika Anda telah menyimpan nilai 4 (atau 5) digit, cukup potong karakter kiri hingga Anda memiliki 3 digit (misalnya 15321 akan menghasilkan nilai validasi-data 321).

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk
 --primary-account-number=5413123456784808 --verification-attributes
 DynamicCardVerificationCode='{ApplicationTransactionCounter=000B,TrackData=524106000000069D13
 --validation-data 398
```

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

ARQC -/ CVN14CVN15

CVN14 dan CVN15 memanfaatkan metode EMV CSK derivasi kunci. Data transaksi yang tepat bervariasi antara kedua metode ini - silakan lihat dokumentasi skema untuk detail tentang pembuatan bidang data transaksi.

ARQC -/ CVN12CVN13

CVN12 dan CVN13 merupakan metode khusus Mastercard yang lebih tua untuk transaksi EMV yang menggabungkan angka tak terduga ke dalam derivasi per transaksi dan juga menggunakan muatan yang berbeda. Untuk informasi tentang isi muatan, silakan hubungi skema.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN12"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk",
                "KeyAttributes": {
                    "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "TDES_2KEY",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": false,
                        "Sign": false,
                        "Verify": false,
                        "DeriveKey": true,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "08D7B4",
                "KeyCheckValueAlgorithm": "ANSI_X9_24",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Anda membutuhkannya di langkah berikutnya.

Validasi ARQC

Example

Dalam contoh ini, kami akan memvalidasi ARQC yang dihasilkan menggunakan Mastercard. CVN12

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. Jika arqc tidak divalidasi, itu akan mengembalikan respons http/400.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram 31BE5D49F14A5F01 \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
  --major-key-derivation-mode EMV_OPTION_A \
  --transaction-data
 00000000170000000000000008400080008000084016051700000000093800000B1F220103000000000000000000000
 \
  --session-key-derivation-attributes='{"MastercardSessionKey":
{"ApplicationTransactionCounter":"000B","PanSequenceNumber":"01","PrimaryAccountNumber":"541312
```

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

3DS AAV SPA2

SPA2 AAV (Account Authentication Value) digunakan untuk transaksi Mastercard 3DS (juga dikenal sebagai Mastercard Idenity Check). Ini menyediakan otentikasi kriptografi untuk transaksi e-commerce menggunakan generasi MAC berbasis HMAC. AAV dihasilkan menggunakan data spesifik transaksi dan kunci rahasia bersama.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=HMAC_SHA256,KeyUsage=TR31_M7_HMAC_KEY,KeyClass=SYMMETRIC_KEY,KeyModesOfUse='{Gene
 --tags='[{"Key":"KEY_PURPOSE","Value":"SPA2_AAV"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
                "KeyAttributes": {
                    "KeyUsage": "TR31_M7_HMAC_KEY",
                    "KeyClass": "SYMMETRIC_KEY",
                    "KeyAlgorithm": "HMAC_SHA256",
                    "KeyModesOfUse": {
                        "Encrypt": false,
                        "Decrypt": false,
                        "Wrap": false,
                        "Unwrap": false,
                        "Generate": true,
                        "Sign": false,
                        "Verify": true,
                        "DeriveKey": false,
                        "NoRestrictions": false
                    }
                },
                "KeyCheckValue": "C661F9",
                "KeyCheckValueAlgorithm": "HMAC",
                "Enabled": true,
                "Exportable": true,
                "KeyState": "CREATE_COMPLETE",
                "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
            }
        }
```


Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-west-2:111122223333:key/q5vjtshsg67cz5gn. Anda membutuhkannya di langkah berikutnya.

Hasilkan SPA2 AAV

Example

Dalam contoh ini, kami akan menghasilkan komponen Nilai Otentikasi Penerbit (IAV) dari SPA2 AAV menggunakan generasi MAC HMAC. Data pesan berisi informasi spesifik transaksi yang akan diautentikasi. Format data pesan harus mengikuti SPA2 spesifikasi Mastercard dan tidak tercakup dalam contoh ini.

> **Note**
>
> Harap tinjau spesifikasi Mastercard Anda untuk pemformatan untuk memasukkan IAV ke dalam nilai AAV.

```
$ aws payment-cryptography-data generate-mac --key-identifier arn:aws:payment-
cryptography:us-west-2:111122223333:key/q5vjtshsg67cz5gn --message-data
 "2226400099919520FFFFd8b448be65694fe7b42f836bad396e9d" --generation-attributes
 Algorithm=HMAC --region us-west-2
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
    "KeyCheckValue": "C661F9",
    "Mac": "6FB2405E9D8A4C1F7B173F73ADD1A6DC358531CAB0E9994FC5B62012ADDE91FC"
}
```

Verifikasi SPA2 AAV

Example

Dalam contoh ini, kami akan memverifikasi SPA2 AAV. Data pesan dan nilai MAC yang sama disediakan untuk verifikasi.

Jika Kriptografi AWS Pembayaran dapat memvalidasi MAC, http/200 dikembalikan. Jika MAC tidak divalidasi, itu akan mengembalikan respons http/400.

```
$ aws payment-cryptography-data verify-mac --key-identifier arn:aws:payment-
cryptography:us-west-2:111122223333:key/q5vjtshsg67cz5gn --message-
data "2226400099919520FFFFd8b448be65694fe7b42f836bad396e9d" --mac
 "6FB2405E9D8A4C1F7B173F73ADD1A6DC358531CAB0E9994FC5B62012ADDE91FC" --verification-
attributes Algorithm=HMAC --region us-west-2
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-west-2:111122223333:key/
q5vjtshsg67cz5gn",
    "KeyCheckValue": "C661F9"
}
```

Fungsi spesifik American Express

Topik

- CSC1
- CSC2
- ICSc
- 3DS AEVV

CSC1

CSC Versi 1 juga dikenal sebagai Algoritma CSC Klasik. Layanan ini dapat menyediakannya sebagai angka 3,4 atau 5 digit.

Untuk semua parameter yang tersedia, lihat AmexCardSecurityCodeVersion1 di panduan referensi API.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC1"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
                "Key": {
                    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
                    "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": "8B5077",
                "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"
            }
        }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east -2:111122223333:key/esh6hn7pxdtttzgq. Anda membutuhkannya di langkah berikutnya.

Menghasilkan CSC1

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion1='{CardExpiryDate=1224}' --validation-data-length 4
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
    "KeyCheckValue": "8B5077",
    "ValidationData": "3938"
  }
```

Validasi CSC1

Example

Dalam contoh ini, kami akan memvalidasi a CSC1.

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/esh6hn7pxdtttzgq
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion1='{CardExpiryDate=1224}''  --validation-data 3938
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
esh6hn7pxdtttzgq",
    "KeyCheckValue": "8B5077"
  }
```

CSC2

CSC Version 2 juga dikenal sebagai Enhanced CSC Algorithm. Layanan ini dapat menyediakannya sebagai angka 3,4 atau 5 digit. Kode layanan untuk CSC2 biasanya 000.

Untuk semua parameter yang tersedia, lihat AmexCardSecurityCodeVersion2 di panduan referensi API.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC2"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
            "Key": {
                "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
erlm445qvunmvoda",
                "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": "BF1077",
            "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"
        }
    }
```

Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda. Anda membutuhkannya di langkah berikutnya.

Menghasilkan CSC2

Dalam contoh ini, kita akan menghasilkan a CSC2 dengan panjang 4. CSC dapat dihasilkan dengan panjang 3,4 atau 5. Untuk American Express, PANs harus 15 digit dan mulai dengan 34 atau 37. Tanggal kedaluwarsa biasanya diformat sebagai YYMM. Kode layanan dapat bervariasi - tinjau manual Anda tetapi nilai tipikal adalah 000, 201 atau 702

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
erlm445qvunmvoda --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=2412,ServiceCode=000}' --validation-
data-length 4
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda",
```

```
"KeyCheckValue": "BF1077",
"ValidationData": "3982"
}
```

Validasi CSC2

Example

Dalam contoh ini, kami akan memvalidasi a CSC2.

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=2412,ServiceCode=000}' --validation-data
 3982
```

```
{
"KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/erlm445qvunmvoda",
"KeyCheckValue": "BF1077"
}
```

ICSc

ICSC juga dikenal sebagai Algoritma CSC statis dan dihitung menggunakan CSC Versi 2. Layanan ini dapat menyediakannya sebagai angka 3,4 atau 5 digit.

Gunakan kode layanan 999 untuk menghitung ICSC untuk kartu kontak. Gunakan kode layanan 702 untuk menghitung ICSC untuk kartu nirkontak.

Untuk semua parameter yang tersedia, lihat AmexCardSecurityCodeVersion2 di panduan referensi API.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"CSC1"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).


```
{
        "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/7vrybrbvjcvwtunv",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
            "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
            },
        },
        "KeyCheckValue": "7121C7",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "CreateTimestamp": "2025-01-29T09:19:21.209000-05:00",
        "UsageStartTimestamp": "2025-01-29T09:19:21.192000-05:00"
        }
    }
```


Perhatikan `KeyArn` yang mewakili kunci, misalnya arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv. Anda membutuhkannya di langkah berikutnya.

Menghasilkan ICSc

Dalam contoh ini, kami akan menghasilkan ICSC dengan panjang 4, untuk kartu nirkontak menggunakan kode layanan 702. CSC dapat dihasilkan dengan panjang 3,4 atau 5. Untuk American Express, PANs harus 15 digit dan mulai dengan 34 atau 37.

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv
 --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1224,ServiceCode=702}' --validation-
data-length 4
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv,
    "KeyCheckValue": 7121C7,
    "ValidationData": "2365"
}
```

Validasi ICSC

Example

Dalam contoh ini, kami akan memvalidasi ICSc.

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv
 --primary-account-number=344131234567848 --verification-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1224,ServiceCode=702}' --validation-data
 2365
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-1:111122223333:key/7vrybrbvjcvwtunv,
    "KeyCheckValue": 7121C7
}
```

3DS AEVV

3DS AEVV (Nilai Verifikasi Akun Aman 3-D) digunakan untuk otentikasi American Express 3-D Secure. Ini menggunakan algoritma yang sama CSC2 tetapi dengan parameter input yang berbeda. Bidang tanggal kedaluwarsa harus diisi dengan nomor tak terduga (acak), dan kode layanan terdiri

dari Kode Hasil Otentikasi AEVV (1 digit) ditambah Kode Otentikasi Faktor Kedua (2 digit). Panjang output harus 3 digit.

Untuk semua parameter yang tersedia, lihat AmexCardSecurityCodeVersion2 di panduan referensi API.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_C0_CARD_VERIFICATION_KEY,KeyClass=SYMMETRIC_KEY,KeyModesO
 --tags='[{"Key":"KEY_PURPOSE","Value":"3DS_AEVV"},
{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
        "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
kw8djn5qxvfh3ztm",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY"
            "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
            },
        },
        "KeyCheckValue": "8F3A21",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "KeyState": "CREATE_COMPLETE",
        "CreateTimestamp": "2025-02-02T10:30:15.209000-05:00",
```

```
        "UsageStartTimestamp": "2025-02-02T10:30:15.192000-05:00"
        }
    }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm. Anda membutuhkannya di langkah berikutnya.

Hasilkan AEVV 3DS

Dalam contoh ini, kami akan menghasilkan 3DS AEVV dengan panjang 3. Bidang tanggal kedaluwarsa berisi nomor tak terduga (acak) (misalnya, 1234), dan kode layanan terdiri dari Kode Hasil Otentikasi AEVV (1 digit) ditambah Kode Otentikasi Faktor Kedua (2 digit), misalnya 543 di mana 5 adalah Kode Hasil Otentikasi dan 43 adalah Kode Otentikasi Faktor Kedua. Untuk American Express, PANs harus 15 digit dan mulai dengan 34 atau 37.

Example

```
$ aws payment-cryptography-data generate-card-validation-data --key-
identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
kw8djn5qxvfh3ztm --primary-account-number=344131234567848 --generation-attributes
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1234,ServiceCode=543}' --validation-
data-length 3
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm,
    "KeyCheckValue": 8F3A21,
    "ValidationData": "921"
}
```

Validasi 3DS AEVV

Example

Dalam contoh ini, kami akan memvalidasi 3DS AEVV.

Jika Kriptografi AWS Pembayaran dapat memvalidasi, http/200 dikembalikan. Jika nilai tidak divalidasi, itu akan mengembalikan respon http/400.

```
$ aws payment-cryptography-data verify-card-validation-data  --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm
 --primary-account-number=344131234567848 --verification-attributes
```

```
 AmexCardSecurityCodeVersion2='{CardExpiryDate=1234,ServiceCode=543}' --validation-data
 921
```

```
{
    "KeyArn": arn:aws:payment-cryptography:us-east-2:111122223333:key/kw8djn5qxvfh3ztm,
    "KeyCheckValue": 8F3A21
}
```

Fungsi khusus JCB

Topik

- ARQC - CVN04
- ARQC - CVN01

ARQC - CVN04

JCB CVN04 menggunakan metode CSK derivasi kunci. Silakan lihat dokumentasi skema untuk detail tentang pembuatan bidang data transaksi.

ARQC - CVN01

CVN01 adalah metode JCB yang lebih lama untuk transaksi EMV yang menggunakan derivasi kunci per kartu daripada derivasi sesi (per transaksi) dan juga menggunakan muatan yang berbeda. Pesan ini juga digunakan oleh Visa sehingga nama elemen memiliki nama itu meskipun itu juga digunakan untuk JCB. Untuk informasi tentang isi payload, silakan hubungi dokumentasi skema.

Buat kunci

```
$ aws payment-cryptography create-key --exportable --key-attributes
 KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS,KeyClass=SYMMETRIC_KEY,KeyMod
 --tags='[{"Key":"KEY_PURPOSE","Value":"CVN10"},{"Key":"CARD_BIN","Value":"12345678"}]'
```

Respons menggemakan kembali parameter permintaan, termasuk ARN untuk panggilan berikutnya serta Nilai Pemeriksaan Kunci (KCV).

```
{
                    "Key": {
                        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/pw3s6nl62t5ushfk",
                        "KeyAttributes": {
```

```
                        "KeyUsage": "TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS",
                        "KeyClass": "SYMMETRIC_KEY",
                        "KeyAlgorithm": "TDES_2KEY",
                        "KeyModesOfUse": {
                            "Encrypt": false,
                            "Decrypt": false,
                            "Wrap": false,
                            "Unwrap": false,
                            "Generate": false,
                            "Sign": false,
                            "Verify": false,
                            "DeriveKey": true,
                            "NoRestrictions": false
                        }
                    },
                    "KeyCheckValue": "08D7B4",
                    "KeyCheckValueAlgorithm": "ANSI_X9_24",
                    "Enabled": true,
                    "Exportable": true,
                    "KeyState": "CREATE_COMPLETE",
                    "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
                    "CreateTimestamp": "2024-03-07T06:41:46.648000-07:00",
                    "UsageStartTimestamp": "2024-03-07T06:41:46.626000-07:00"
                }
            }
```

Perhatikan yang mewakili kunci, misalnya `KeyArn` arn:aws:payment-cryptography:us-east-2:111122223333:key/pw3s6nl62t5ushfk. Anda membutuhkannya di langkah berikutnya.

Validasi ARQC

Example

Dalam contoh ini, kami akan memvalidasi ARQC yang dihasilkan menggunakan JCB. CVN01 Ini menggunakan opsi yang sama dengan metode Visa, oleh karena itu nama parameternya.

Jika Kriptografi AWS Pembayaran dapat memvalidasi ARQC, http/200 dikembalikan. Jika arqc tidak divalidasi, itu akan mengembalikan respons http/400.

```
$ 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
 0000000017000000000000000840008000800008401605170000000093800000B03011203000000 \
            --session-key-derivation-attributes='{"Visa":{"PanSequenceNumber":"01" \
            ,"PrimaryAccountNumber":"9137631040001422"}}'
```

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

Fasilitator perolehan dan pembayaran

Acquirers, PSPs dan Payment Facilator biasanya memiliki seperangkat persyaratan kriptografi yang berbeda dari penerbit. Kasus penggunaan umum meliputi:

Dekripsi Data

Data (terutama data pan) dapat dienkripsi oleh terminal pembayaran dan perlu didekripsi oleh backend. Dekripsi Data dan Enkripsi Data mendukung berbagai metode termasuk teknik derivasi TDES, AES dan DUKPT. Layanan Kriptografi AWS Pembayaran itu sendiri juga sesuai dengan PCI P2PE dan terdaftar sebagai komponen dekripsi PCI P2PE.

TranslatePin

Untuk menjaga kepatuhan PIN PCI, sistem perolehan tidak boleh memiliki pin pemegang kartu yang jelas setelah dimasukkan pada perangkat yang aman. Oleh karena itu, untuk meneruskan pin dari terminal ke sistem hilir (seperti jaringan pembayaran atau penerbit), ada kebutuhan untuk mengenkripsi ulang menggunakan kunci yang berbeda dari yang digunakan terminal pembayaran. Translate Pin menyelesaikannya dengan mengonversi pin terenkripsi dari satu kunci ke kunci lainnya secara aman dengan servicebbb. Dengan menggunakan perintah ini, pin dapat dikonversi antara berbagai skema seperti derivasi TDES, AES dan DUKPT dan format blok pin seperti ISO-0, ISO-3 dan ISO-4.

VerifyMac

Data dari terminal pembayaran mungkin MAC untuk memastikan bahwa data belum dimodifikasi dalam perjalanan. Verifikasi Mac dan GenerateMac dukung berbagai teknik menggunakan kunci simetris termasuk teknik derivasi TDES, AES dan DUKPT untuk digunakan dengan algoritma ISO-9797-1 1, algoritma ISO-9797-1 3 (Retail MAC) dan teknik CMAC.

Topik Tambahan

- Menggunakan Tombol Dinamis

Menggunakan Tombol Dinamis

Dynamic Keys memungkinkan kunci penggunaan satu kali atau terbatas untuk digunakan untuk operasi kriptografi seperti. `EncryptData` Aliran ini dapat digunakan ketika materi kunci sering berputar (seperti pada setiap transaksi kartu) dan ada keinginan untuk menghindari mengimpor materi kunci ke dalam layanan. Kunci berumur pendek dapat digunakan sebagai bagian dari SoftPOS/MPOC atau solusi lainnya.

> **Note**
>
> Ini dapat digunakan sebagai pengganti aliran tipikal menggunakan Kriptografi AWS Pembayaran, di mana kunci kriptografi dibuat atau diimpor ke layanan dan kunci ditentukan menggunakan alias kunci atau kunci arn.

Operasi berikut mendukung Dynamic Keys:

- EncryptData
- DecryptData
- ReEncryptData
- TranslatePin

Mendekripsi Data

Contoh berikut menunjukkan menggunakan Dynamic Keys bersama dengan perintah dekripsi. Pengidentifikasi kunci dalam hal ini adalah kunci pembungkus (KEK) yang mengamankan kunci dekripsi (yang disediakan dalam parameter kunci yang dibungkus dalam format TR-31). Kunci yang dibungkus harus menjadi tujuan utama D0 untuk digunakan dengan perintah dekripsi bersama dengan mode penggunaan B atau D.

Example

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza
```

```
 --cipher-text 12341234123412341234123412341234123A --decryption-attributes
 'Symmetric={Mode=CBC,InitializationVector=1234123412341234}'   --wrapped-key
 WrappedKeyMaterial={"Tr31KeyBlock"="D0112D0TN00E0000B05A6E82D7FC68B95C84306634B0000DA4701BE9BC
```

```
{
   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
   "KeyCheckValue": "0A3674",
   "PlainText": "2E138A746A0032023BEF5B85BA5060BA"
}
```

Menerjemahkan pin

Contoh berikut menunjukkan penggunaan Dynamic Keys bersama dengan perintah translate pin untuk menerjemahkan dari kunci dinamis ke kunci kerja pengakuisisi semi-statis (AWK). Pengidentifikasi kunci yang masuk dalam hal ini adalah kunci pembungkus (KEK) yang melindungi kunci enkripsi pin dinamis (PEK) yang disediakan dalam format TR-31. Kunci yang dibungkus harus menjadi tujuan utama `P0` bersama dengan mode penggunaan B atau D. Pengidentifikasi kunci keluar adalah kunci tipe `TR31_P0_PIN_ENCRYPTION_KEY` dan mode penggunaan enkripsi = True, Wrap=True

Example

```
$ aws payment-cryptography-data translate-pin-data --encrypted-pin-block
 "C7005A4C0FA23E02" --incoming-translation-
attributes=IsoFormat0='{PrimaryAccountNumber=171234567890123}'
 --incoming-key-identifier alias/PARTNER1_KEK  --outgoing-key-
identifier alias/ACQUIRER_AWK_PEK --outgoing-translation-attributes
 IsoFormat0="{PrimaryAccountNumber=171234567890123}"  --incoming-wrapped-key
 WrappedKeyMaterial={"Tr31KeyBlock"="D0112P0TB00S0000EB5D8E63076313162B04245C8CE351C956EA4A16CC
```

```
{
   "PinBlock": "2E66192BDA390C6F",
   "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
   "KeyCheckValue": "0A3674"
}
```

Fitur khusus wilayah untuk Kriptografi AWS Pembayaran

Fitur tertentu mungkin spesifik wilayah dan tidak digunakan sebaliknya. Fitur-fitur tersebut dijelaskan secara lebih rinci di bagian ini.

AS2805

Standar Australia 2805 (AS2805) adalah standar untuk transfer dana elektronik yang digunakan terutama untuk transaksi pembayaran berbasis kartu. Hal ini dikelola oleh Standar Australia. Standar ini terdiri dari 6 buku yang mencakup berbagai topik mulai dari format pesan hingga standar enkripsi.

Bagian 6 memberikan panduan tentang manajemen kunci termasuk host-to-host (node-to-node) komunikasi dan persyaratan kriptografi yang relevan sementara aspek lain tercakup dalam bagian lain. Semua kriptografi dalam standar ini saat ini didasarkan pada TDES.

> **Note**
>
> AS2805 saat ini tersedia di Wilayah ap-southeast-2. Ini akan diluncurkan ke Wilayah tambahan dalam waktu dekat.

AS2805 memiliki sejumlah perbedaan dibandingkan dengan implementasi lain, yang dirangkum di bawah ini.

Perlindungan Kunci

: Mengandalkan varian kunci alih-alih blok kunci seperti di TR-31/X9.143. AWS Kriptografi Pembayaran menyimpan semua kunci sebagai blok kunci secara internal tetapi mengizinkan impor, ekspor, dan perhitungan menggunakan AS28 05 varian yang ditentukan.

Kunci Searah

: AS2805 mengamanatkan penggunaan kunci searah. Jika kedua node perlu menghasilkan kode otentikasi pesan (MAC), mereka menggunakan dua kunci.

Blok Pin

: AS2805 mendefinisikan teknik derivasi kunci untuk kunci enkripsi pin unik per transaksi. Ini dapat digunakan sebagai pengganti DUKPT. Skema AS28 05 bergantung pada data transaksi (trace number dan jumlah transaksi) dibandingkan dengan penggunaan counter transaksi DUKPT.

Validasi Pertukaran Kunci

Mendefinisikan proses untuk memvalidasi KEK sebelum mulai bertukar kunci kerja seperti tombol pin. Dalam skema lain, KEK jarang dipertukarkan dan divalidasi menggunakan KCV.

AS2805 menggunakan konsep varian kunci daripada blok kunci untuk memastikan kunci hanya digunakan untuk tujuan yang dimaksudkan (dan tunggal). Berikut ini adalah bagaimana Kriptografi AWS Pembayaran memetakan antara varian dan pemblokiran tombol saat mengimpor, mengekspor, atau melakukan fungsi kriptografi lainnya dengan kunci.

AS2805 Jenis Kunci	AWS Jenis Kunci Kriptografi Pembayaran
TERMINAL_MAJOR_KEY_VARIANT_00	TR31_K0_KEY_ENCRYPTION_KEY
PIN_ENCRYPTION_KEY_VARIANT_28	TR31_P0_PIN_ENCRYPTION_KEY
MESSAGE_AUTHENTICATION_KEY_ VARIANT_24	TR31_M0_ISO_16609_MAC_KEY
DATA_ENCRYPTION_KEY_VARIANT_22	TR31_D0_SYMMETRIC_DATA_ENCR YPTION_KEY
VARIANT_MASK_82, VARIANT_MASK_82C0	Pilihan tersedia sebagai bagian dari proses validasi KEK. Jenis kunci ini bersifat sementara dan tidak disimpan oleh layanan.

Diberikan dua node, node1 dan node2, contoh berikut adalah dari perspektif node1. AWS Kriptografi Pembayaran mendukung APIs dari kedua sisi proses.

Topik

Bursa Initial Key (KEK)

Pada AS28 05, masing-masing pihak memiliki KEK sendiri. KEK (s) mengacu pada tombol sisi pengiriman yang akan digunakan setiap kali sisi pengirim perlu protect/wrap kunci dan mengirimkannya ke node2. KEK (r) adalah kunci yang dibuat oleh sisi berlawanan (node2).

> **Note**
>
> Istilah-istilah ini relatif - satu sisi membuat kunci (sisi pengirim) dan sisi lain menerimanya. Jadi diberikan KEY1, itu disebut pada node1 sebagai KEK (s) dan pada node2 sebagai KEK (r).

KEK untuk AS28 05 selalu tipe kunci = TR31 _K0_KEY_ENCRYPTION_KEY karena digunakan untuk melindungi kriptogram dan bukan blok kunci. Ini memetakan ke TERMINAL_MAJOR_KEY_VARIANT_00 seperti yang didefinisikan dalam 05 6.1 AS28

Langkah:

1. Buat kunci

   Buat kunci menggunakan CreateKeyapi. Anda akan membuat kunci tipe TR31 _K0_KEY_ENCRYPTION_KEY

2. Tentukan metode untuk bertukar kunci dengan node2

   Tentukan cara menukar KEK dengan pihak konter. Untuk AS28 05, metode yang paling umum dan interoperable adalah RSA Wrap.

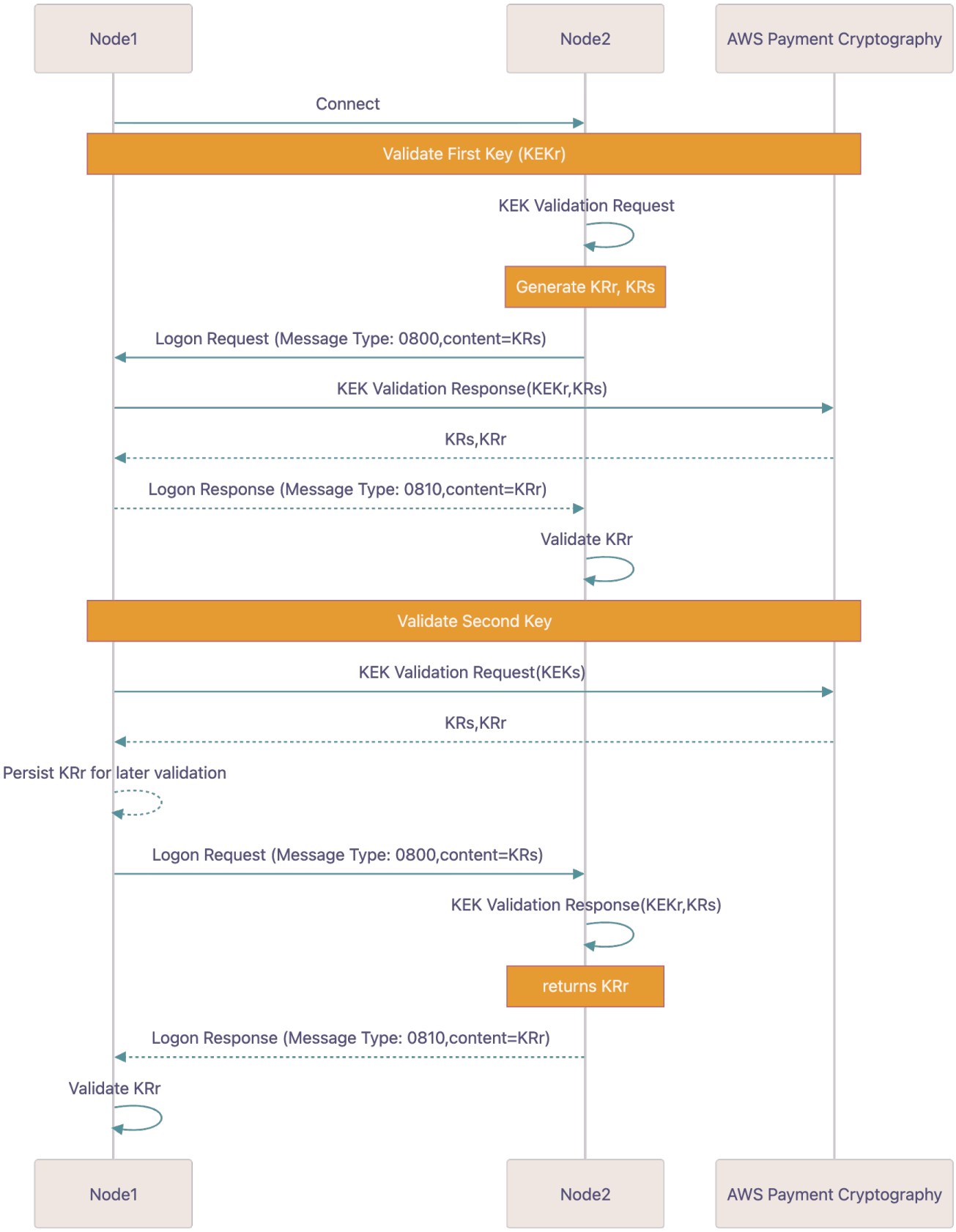
3. Ekspor KEKs

   Berdasarkan pilihan Anda di atas, Anda akan menerima sertifikat kunci publik dari node2. Anda akan menjalankan ekspor menggunakan sertifikat itu untuk melindungi kunci (atau memperoleh kunci jika menggunakan ECDH).

4. Impor KEKr

   Berdasarkan pilihan Anda di atas, Anda akan mengirim sertifikat kunci publik ke node2. Anda akan menjalankan import menggunakan sertifikat itu untuk memuat node 2 KEKr ke dalam layanan.

Validasi KEK



Ketika layanan Anda (node1) terhubung ke node2, masing-masing pihak akan memastikan bahwa mereka menggunakan KEK yang sama untuk operasi selanjutnya menggunakan proses yang disebut Validasi KEK.

1. Langkah-langkah untuk memvalidasi kunci pertama

1.1 Menerima KRs

Node2 akan menghasilkan KRs dan mengirimkannya kepada Anda sebagai bagian dari proses logon. Mereka dapat menggunakan Kriptografi AWS Pembayaran untuk menghasilkan nilai ini atau solusi lain.

1.2 Menghasilkan Respon Validasi KEK

Node Anda akan menghasilkan respons Validasi KEK dengan input sebagai KEK (r) dan yang KRs disediakan pada langkah 1.

Example

```
cat >> generate-kek-validation-response.json
{
    "KekValidationType": {
        "KekValidationResponse": {
            "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
        }
    },
    "RandomKeySendVariantMask": "VARIANT_MASK_82",
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza"
}
```

```
$ aws payment-cryptography-data generate-as2805-kek-validation --cli-input-json
 file://generate-kek-validation-response.json
```

```
{
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
 "KeyCheckValue": "0A3674",
 "RandomKeyReceive": "A4B7E249C40C98178C1B856DB7FB76EB",
 "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
}
```

1.3 Pengembalian dihitung KRr

Kembalikan yang dihitung KRr ke node2. Node itu akan membandingkannya dengan nilai yang dihitung dari langkah 1.

2. Langkah-langkah untuk memvalidasi kunci kedua

2.1 Menghasilkan KRr dan KRs

Node Anda akan menghasilkan nilai acak dan salinan terbalik (terbalik) dari nilai ini menggunakan Kriptografi AWS Pembayaran. Layanan akan menampilkan kedua nilai ini yang dibungkus oleh KEK (s). Ini dikenal sebagai KR (s) dan KR (r).

Example

```
cat >> generate-kek-validation-request.json
{
    "KekValidationType": {
        "KekValidationRequest": {
            "DeriveKeyAlgorithm": "TDES_2KEY"
        }
    },
"RandomKeySendVariantMask": "VARIANT_MASK_82",
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
}
```

```
$ aws payment-cryptography-data generate-as2805-kek-validation --cli-input-json
 file://generate-kek-validation-request.json
```

```
{
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv",
 "KeyCheckValue": "DC1081",
 "RandomKeyReceive": "A4B7E249C40C98178C1B856DB7FB76EB",
 "RandomKeySend": "9217DC67B8763BABCFDF3DADFCD0F84A"
}
```

2.2 Kirim KRs ke node2

Kirim KRs ke node2. Simpan KRr untuk validasi nanti.

2.3 Node2 menghasilkan respons validasi KEK

Node2 menggunakan KEKr dan KRs, menghasilkan KRr dan mengirimkannya kembali ke layanan Anda.

2.4 Validasi tanggapan

Bandingkan KRr dari langkah 1 dan nilai yang dikembalikan dari langkah 3. Jika cocok, lanjutkan.

Pembuatan dan transmisi kunci kerja

Tombol kerja khas yang digunakan dalam AS28 05 mencakup dua set kunci:

Kunci antara node seperti: kunci pin zona (ZPK), kunci enkripsi zona (ZEK) dan kunci otentikasi zona (ZAK).

Kunci antara terminal dan node seperti: kunci utama terminal (TMK) dan kunci pin terminal (TPK) jika tidak menggunakan DUKPT.

> **Note**
>
> Kami merekomendasikan meminimalkan kunci per kunci terminal dan memanfaatkan teknik seperti TR-34 dan DUKPT bila memungkinkan yang menggunakan jumlah kunci yang lebih kecil.

Example

Dalam contoh ini, kami telah menggunakan tag opsional untuk melacak tujuan dan penggunaan kunci ini. Tag tidak digunakan sebagai bagian dari fungsi kriptografi sistem tetapi dapat digunakan untuk kategorisasi, pelacakan keuangan dan dapat digunakan untuk menerapkan kebijakan IAM.

```
cat >> create-zone-pin-key.json
{
    "KeyAttributes": {
        "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
        "KeyClass": "SYMMETRIC_KEY",
        "KeyAlgorithm": "TDES_2KEY",
        "KeyModesOfUse": {
            "Encrypt": true,
            "Decrypt": true,
```

```
            "Wrap": true,
            "Unwrap": true,
            "Generate": false,
            "Sign": false,
            "Verify": false,
            "DeriveKey": false,
            "NoRestrictions": false
        }
    },
    "KeyCheckValueAlgorithm": "ANSI_X9_24",
    "Exportable": true,
    "Enabled": true,
    "Tags": [
        {
            "Key": "AS2805_KEYTYPE",
            "Value": "ZONE_PIN_KEY_VARIANT28"
        }
    ]
}
```

```
$ aws payment-cryptography-data create-key --cli-input-json file://create-zone-pin-key.json --region ap-southeast-2
```

```
{
 "Key": {
 "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/alsuwfxug3pgy6xh",
 "KeyAttributes": {
 "KeyUsage": "TR31_P0_PIN_ENCRYPTION_KEY",
 "KeyClass": "SYMMETRIC_KEY",
 "KeyAlgorithm": "TDES_2KEY",
 "KeyModesOfUse": {
 "Encrypt": true,
 "Decrypt": true,
 "Wrap": true,
 "Unwrap": true,
 "Generate": false,
 "Sign": false,
 "Verify": false,
 "DeriveKey": false,
 "NoRestrictions": false
 }
 },
 "KeyCheckValue": "9A325B",
```

```
 "KeyCheckValueAlgorithm": "ANSI_X9_24",
 "Enabled": true,
 "Exportable": true,
 "KeyState": "CREATE_COMPLETE",
 "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
 "CreateTimestamp": "2025-12-17T09:05:27.586000-08:00",
 "UsageStartTimestamp": "2025-12-17T09:05:27.570000-08:00"
 }
}
```

Mengekspor kunci kerja

Untuk menjaga kompatibilitas dengan pihak lain, AWS Payment Cryptography mendukung AS28 05 teknik pembungkus kunci simetris yang menggunakan varian kunci alih-alih keyblocks seperti TR-31. Jika beberapa kunci dibagi antar pihak, masing-masing harus diekspor satu per satu. Jika data dikirim dua arah, mungkin ada dua kunci antara pihak-pihak dari jenis yang sama seperti ZAK (s) dan ZAK (r) yang digunakan oleh masing-masing pihak untuk menghasilkan kode otentikasi pesan.

Parameter tambahan untuk mengimpor dan mengekspor dalam format ini ditentukan pada perintah.

```
cat >> export-zone-pin-key.json
{
    "ExportKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
alsuwfxug3pgy6xh",
    "KeyMaterial": {
        "As2805KeyCryptogram": {
            "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/rhfm6tenpxapkmrv",
            "As2805KeyVariant: "PIN_ENCRYPTION_KEY_VARIANT_28"
        }
    }
}
```

```
$ aws payment-cryptography-data export-key --cli-input-json file://export-zone-pin-
key.json --region ap-southeast-2
```

```
{
    "WrappedKey": {
        "KeyCheckValue": "DC1081",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyMaterial": "HDC10AEF038E695DDD72AF08DC1BB422D",
```

```
        "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM",
        "WrappingKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
    }
}
```

Terjemahan Pin

AS2805 menjelaskan mode derivasi kunci khusus sesi di bagian 6.4. Ini melayani tujuan yang sama seperti DUKPT dan salah satu algoritma dapat digunakan sebagai DUKPT tercakup dalam bagian 6.7. Dalam skema ini, kunci pin sesi (dikenal sebagai KPE) berasal dari Terminal Pin Key menggunakan SystemTraceAuditNumber (STAN) dan TransactionAmount sebagai data derivasi.

Translate pin adalah fungsi umum yang dapat menerjemahkan to/from berbagai format. Dalam contoh ini, kami menerjemahkan pin dari KPE ke kunci enkripsi pin (PEK) seperti saat mengirim pin ke jaringan pembayaran.

```
cat >> translate-pin-as2805.json
{
    "EncryptedPinBlock": "B3B34B43BAB5F81A",
    "IncomingKeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "IncomingTranslationAttributes": {
        "IsoFormat0": {
            "PrimaryAccountNumber": "9999179999900013"
        }
    },
      "IncomingAs2805Attributes": {
        "SystemTraceAuditNumber": "000348",
        "TransactionAmount": "000000000328"
    },
    "OutgoingKeyIdentifier": "",
    "OutgoingTranslationAttributes": {
        "IsoFormat0": {
            "PrimaryAccountNumber": "9999179999900013"
        }
    }
}
```

```
$ aws payment-cryptography-data translate-pin-data --cli-input-json file://translate-
pin-as2805.json  --region ap-southeast-2
```

```
{
    "WrappedKey": {
        "KeyCheckValue": "DC1081",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "KeyMaterial": "HDC10AEF038E695DDD72AF08DC1BB422D",
        "WrappedKeyMaterialFormat": "KEY_CRYPTOGRAM",
        "WrappingKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
rhfm6tenpxapkmrv"
    }
}
```

Generasi dan Validasi Mac

Menghasilkan dan memverifikasi perintah MAC mendukung berbagai MACs termasuk HMAC, CMAC, EMV MAC, dll. Untuk AS28 05, ada variasi tambahan yang didefinisikan dalam AS28 05.4.1. Biasanya di AS28 05, pesan masuk diverifikasi menggunakan MAC ini dan pesan keluar termasuk MAC juga.

```
cat verify-mac.json
{
    "KeyIdentifier": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "Mac": "86304058",
    "MessageData": "73D8BA54D3852951DAEA41",
    "VerificationAttributes": {
        "Algorithm": "AS2805_4_1"
    }
}
```

```
$ aws payment-cryptography-data verify-mac --cli-input-json file://verify-mac.json --
region ap-southeast-2
```

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

Keamanan dalam Kriptografi AWS Pembayaran

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud —AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku untuk Kriptografi AWS Pembayaran, lihat AWS Services in Scope by Compliance Program .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Topik ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Kriptografi AWS Pembayaran. Ini menunjukkan kepada Anda cara mengkonfigurasi Kriptografi AWS Pembayaran untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya Kriptografi AWS Pembayaran Anda.

Topik

Perlindungan data dalam Kriptografi AWS Pembayaran

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data dalam Kriptografi AWS Pembayaran. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensyal dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan Kriptografi AWS Pembayaran atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan

untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

AWS Payment Cryptography menyimpan dan melindungi kunci enkripsi pembayaran Anda agar sangat tersedia sekaligus memberi Anda kontrol akses yang kuat dan fleksibel.

Topik

Melindungi bahan utama

Secara default, AWS Payment Cryptography melindungi materi kunci kriptografi untuk kunci pembayaran yang dikelola oleh layanan. Selain itu, AWS Payment Cryptography menawarkan opsi untuk mengimpor materi utama yang dibuat di luar layanan. Untuk detail teknis tentang kunci pembayaran dan materi utama, lihat Detail Kriptografi Kriptografi Pembayaran AWS.

Enkripsi data

Data dalam AWS Payment Cryptography terdiri dari kunci AWS Payment Cryptography, materi kunci enkripsi yang mereka wakili, dan atribut penggunaannya. Materi utama ada dalam teks biasa hanya dalam modul keamanan perangkat keras AWS Payment Cryptography (HSMs) dan hanya saat digunakan. Jika tidak, bahan dan atribut utama dienkripsi dan disimpan dalam penyimpanan persisten yang tahan lama.

Materi utama yang dihasilkan atau dimuat oleh AWS Payment Cryptography untuk kunci pembayaran tidak pernah meninggalkan batas Kriptografi Pembayaran AWS tidak terenkripsi. HSMs Ini dapat diekspor dienkripsi oleh operasi AWS Payment Cryptography API.

Enkripsi saat diam

AWS Payment Cryptography menghasilkan materi utama untuk kunci pembayaran di PCI PTS yang terdaftar di HSM. HSMs Saat tidak digunakan, bahan kunci dienkripsi oleh kunci HSM dan ditulis ke

penyimpanan yang tahan lama dan persisten. Materi utama untuk kunci Kriptografi Pembayaran dan kunci enkripsi yang melindungi materi kunci tidak pernah meninggalkan HSMs dalam bentuk teks biasa.

Enkripsi dan pengelolaan materi kunci untuk kunci Kriptografi Pembayaran ditangani sepenuhnya oleh layanan.

Untuk detail selengkapnya, lihat AWS Key Management Service Cryptographic Details.

Enkripsi saat bergerak

Materi kunci yang AWS dihasilkan atau dimuat oleh Kriptografi Pembayaran untuk kunci pembayaran tidak pernah diekspor atau ditransmisikan dalam operasi API Kriptografi AWS Pembayaran di cleartext. AWS Kriptografi Pembayaran menggunakan pengidentifikasi kunci untuk mewakili kunci dalam operasi API.

Namun, beberapa operasi API mengekspor kunci yang dienkripsi oleh kunci pertukaran kunci bersama atau asimetris sebelumnya. Selain itu, pelanggan dapat menggunakan operasi API untuk mengimpor materi kunci terenkripsi untuk kunci pembayaran.

Semua panggilan API Kriptografi AWS Pembayaran harus ditandatangani dan ditransmisikan menggunakan Transport Layer Security (TLS). AWS Kriptografi Pembayaran membutuhkan versi TLS dan cipher suite yang didefinisikan oleh PCI sebagai “kriptografi kuat”. Semua titik akhir layanan mendukung TLS 1.2-1.3 dan TLS pasca-kuantum hibrida.

Untuk detail selengkapnya, lihat AWS Key Management Service Cryptographic Details.

Privasi lalu lintas antarjaringan

AWS Kriptografi Pembayaran mendukung AWS Management Console dan serangkaian operasi API yang memungkinkan Anda membuat dan mengelola kunci pembayaran dan menggunakannya dalam operasi kriptografi.

AWS Kriptografi Pembayaran mendukung dua opsi konektivitas jaringan dari jaringan pribadi Anda ke AWS.

- Koneksi IPSec VPN melalui internet.
- AWS Direct Connect, yang menghubungkan jaringan internal Anda ke lokasi AWS Direct Connect melalui kabel serat optik Ethernet standar.

Semua panggilan API Kriptografi Pembayaran harus ditandatangani dan ditransmisikan menggunakan Transport Layer Security (TLS). Panggilan juga memerlukan suite penyandian modern yang mendukung kerahasiaan penerusan sempurna. Lalu lintas ke modul keamanan perangkat keras (HSMs) yang menyimpan materi kunci untuk kunci pembayaran hanya diizinkan dari host AWS Payment Cryptography API yang diketahui melalui jaringan internal AWS.

Untuk terhubung langsung ke AWS Payment Cryptography dari virtual private cloud (VPC) Anda tanpa mengirimkan lalu lintas melalui internet publik, gunakan titik akhir VPC, yang didukung oleh AWS. PrivateLink Untuk informasi selengkapnya, lihat Menghubungkan ke Kriptografi Pembayaran AWS melalui titik akhir VPC.

AWS Payment Cryptography juga mendukung opsi pertukaran kunci pasca-kuantum hybrid untuk protokol enkripsi jaringan Transport Layer Security (TLS). Anda dapat menggunakan opsi ini dengan TLS saat Anda terhubung ke titik akhir AWS Payment Cryptography API.

Ketahanan dalam AWS Kriptografi Pembayaran

AWS Infrastruktur global dibangun di sekitar AWS Wilayah dan Availability Zone. Wilayah memberikan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik, yang terkoneksi melalui jaringan latensi rendah, throughput tinggi, dan sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

Isolasi regional

AWS Payment Cryptography adalah layanan Regional yang tersedia di beberapa wilayah.

Desain Kriptografi Pembayaran AWS yang terisolasi secara regional memastikan bahwa masalah ketersediaan di satu Wilayah AWS tidak dapat memengaruhi operasi Kriptografi Pembayaran AWS di Wilayah lain mana pun. AWS Payment Cryptography dirancang untuk memastikan nol waktu henti yang direncanakan, dengan semua pembaruan perangkat lunak dan operasi penskalaan dilakukan dengan mulus dan tanpa terasa.

AWS Payment Cryptography Service Level Agreement (SLA) mencakup komitmen layanan sebesar 99,99% untuk semua Kriptografi Pembayaran. APIs Untuk memenuhi komitmen ini, AWS Payment

Cryptography memastikan bahwa semua data dan informasi otorisasi yang diperlukan untuk menjalankan permintaan API tersedia di semua host regional yang menerima permintaan tersebut.

Infrastruktur Kriptografi Pembayaran AWS direplikasi di setidaknya tiga Availability Zone (AZs) di setiap Wilayah. Untuk memastikan bahwa beberapa kegagalan host tidak memengaruhi kinerja Kriptografi Pembayaran AWS, Kriptografi Pembayaran AWS dirancang untuk melayani lalu lintas pelanggan dari salah satu AZs di Wilayah.

Perubahan yang Anda buat pada properti atau izin kunci pembayaran direplikasi ke semua host di Wilayah untuk memastikan bahwa permintaan berikutnya dapat diproses dengan benar oleh host mana pun di Wilayah. Permintaan untuk operasi kriptografi menggunakan kunci pembayaran Anda diteruskan ke armada modul keamanan perangkat keras AWS Payment Cryptography (HSMs), yang mana pun dapat melakukan operasi dengan kunci pembayaran.

Desain multi-penyewa

Desain multi-tenant AWS Payment Cryptography memungkinkannya memenuhi ketersediaan SLA, dan mempertahankan tingkat permintaan yang tinggi, sekaligus melindungi kerahasiaan kunci dan data Anda.

Beberapa mekanisme penegakan integritas digunakan untuk memastikan bahwa kunci pembayaran yang Anda tentukan untuk operasi kriptografi selalu yang digunakan.

Materi kunci plaintext untuk kunci Kriptografi Pembayaran Anda dilindungi secara ekstensif. Materi utama dienkripsi di HSM segera setelah dibuat, dan bahan kunci terenkripsi segera dipindahkan ke penyimpanan yang aman. Kunci terenkripsi diambil dan didekripsi dalam HSM tepat pada waktunya untuk digunakan. Kunci plaintext tetap dalam memori HSM hanya untuk waktu yang dibutuhkan untuk menyelesaikan operasi kriptografi. Materi kunci Plaintext tidak pernah meninggalkan HSMs; itu tidak pernah ditulis ke penyimpanan persisten.

Untuk informasi selengkapnya tentang mekanisme yang digunakan AWS Payment Cryptography untuk mengamankan kunci Anda, lihat AWS Payment Cryptography Cryptography Details.

Keamanan infrastruktur di AWS Payment Cryptography

Sebagai layanan terkelola, AWS Payment Cryptography dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam whitepaper Amazon Web Services: Tinjauan Proses Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses AWS Payment Cryptography melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS)

1.2 atau versi yang lebih baru. Klien juga harus mendukung cipher suite dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem-sistem modern seperti Java 7 dan versi yang lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang dikaitkan dengan pengguna utama IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara untuk menandatangani permintaan.

Isolasi host fisik

Keamanan infrastruktur fisik yang digunakan AWS Payment Cryptography tunduk pada kontrol yang dijelaskan di bagian Keamanan Fisik dan Lingkungan Amazon Web Services: Tinjauan Proses Keamanan. Anda dapat menemukan lebih banyak detail dalam laporan kepatuhan dan temuan audit pihak ketiga yang tercantum di bagian sebelumnya.

AWS Payment Cryptography didukung oleh modul keamanan perangkat keras khusus yang terdaftar di commercial-off-the-shelf PCI PTS HSM (). HSMs Materi utama untuk kunci Kriptografi Pembayaran AWS disimpan hanya dalam memori volatil pada HSMs, dan hanya saat kunci Kriptografi Pembayaran sedang digunakan. HSMs berada di rak yang dikendalikan akses dalam pusat data Amazon yang memberlakukan kontrol ganda untuk akses fisik apa pun. Untuk informasi terperinci tentang pengoperasian Kriptografi Pembayaran AWS HSMs, lihat Detail Kriptografi Kriptografi Pembayaran AWS.

Menghubungkan ke Kriptografi AWS Pembayaran melalui titik akhir VPC

Anda dapat terhubung langsung ke Kriptografi AWS Pembayaran melalui titik akhir antarmuka pribadi di cloud pribadi virtual (VPC) Anda. Saat Anda menggunakan titik akhir VPC antarmuka, komunikasi antara VPC dan Kriptografi AWS Pembayaran dilakukan sepenuhnya di dalam jaringan. AWS

AWS Kriptografi Pembayaran mendukung titik akhir Amazon Virtual Private Cloud (Amazon VPC) yang didukung oleh. AWS PrivateLink Setiap titik akhir VPC diwakili oleh satu atau lebih Elastic Network Interfaces (ENIs) dengan alamat IP pribadi di subnet VPC Anda.

Titik akhir VPC antarmuka menghubungkan VPC Anda langsung ke Kriptografi AWS Pembayaran tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi. AWS Direct Connect Instans

di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi AWS dengan Kriptografi Pembayaran.

Wilayah

AWS Kriptografi Pembayaran mendukung kebijakan titik akhir VPC dan titik akhir VPC Wilayah AWS di mana Kriptografi Pembayaran didukung.AWS

Topik

- Pertimbangan untuk titik akhir AWS VPC Kriptografi Pembayaran
- Membuat titik akhir VPC untuk Kriptografi Pembayaran AWS
- Menghubungkan ke titik akhir AWS VPC Kriptografi Pembayaran
- Mengontrol akses ke VPC endpoint
- Menggunakan VPC endpoint dalam pernyataan kebijakan
- Mencatat VPC endpoint Anda

Pertimbangan untuk titik akhir AWS VPC Kriptografi Pembayaran

> **Note**
>
> Meskipun titik akhir VPC memungkinkan Anda untuk terhubung ke layanan hanya dalam satu zona ketersediaan (AZ), kami merekomendasikan untuk menghubungkan ke tiga zona ketersediaan untuk tujuan ketersediaan dan redundansi yang tinggi.

Sebelum Anda menyiapkan titik akhir VPC antarmuka untuk Kriptografi AWS Pembayaran, tinjau topik properti dan batasan titik akhir Antarmuka di Panduan.AWS PrivateLink

AWS Dukungan Kriptografi Pembayaran untuk titik akhir VPC mencakup yang berikut ini.

- Anda dapat menggunakan titik akhir VPC Anda untuk memanggil semua operasi pesawat Kontrol Kriptografi AWS Pembayaran dan operasi pesawat Data Kriptografi AWS Pembayaran dari VPC.
- Anda dapat membuat titik akhir VPC antarmuka yang terhubung ke titik akhir wilayah Kriptografi AWS Pembayaran.
- AWS Kriptografi Pembayaran terdiri dari bidang kontrol dan bidang data. Anda dapat memilih untuk mengatur satu atau kedua sub-layanan AWS PrivateLink tetapi masing-masing dikonfigurasi secara terpisah.

- Anda dapat menggunakan AWS CloudTrail log untuk mengaudit penggunaan kunci Kriptografi AWS Pembayaran melalui titik akhir VPC. Lihat perinciannya di Mencatat VPC endpoint Anda.

Membuat titik akhir VPC untuk Kriptografi Pembayaran AWS

Anda dapat membuat titik akhir VPC untuk Kriptografi AWS Pembayaran dengan menggunakan konsol VPC Amazon atau API VPC Amazon. Untuk informasi selengkapnya, lihat Membuat titik akhir antarmuka di AWS PrivateLink Panduan.

- Untuk membuat titik akhir VPC untuk Kriptografi AWS Pembayaran, gunakan nama layanan berikut:

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

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

Misalnya, di Wilayah AS Barat (Oregon) (`us-west-2`), nama layanannya adalah:

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

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

Untuk mempermudah penggunaan titik akhir VPC, Anda dapat mengaktifkan nama DNS pribadi untuk titik akhir VPC Anda. Jika Anda memilih opsi Aktifkan Nama DNS, nama host DNS Kriptografi AWS Pembayaran standar akan diselesaikan ke titik akhir VPC Anda. Misalnya, `https://controlplane.payment-cryptography.us-west-2.amazonaws.com` akan menyelesaikan ke titik akhir VPC yang terhubung ke nama layanan. `com.amazonaws.us-west-2.payment-cryptography.controlplane`

Opsi ini mempermudah untuk menggunakan VPC endpoint. Itu AWS SDKs dan AWS CLI gunakan standar AWS Payment Cryptography DNS hostname secara default, sehingga Anda tidak perlu menentukan URL endpoint VPC dalam aplikasi dan perintah.

Untuk informasi selengkapnya, lihat Mengakses layanan melalui titik akhir antarmuka di Panduan.AWS PrivateLink

Menghubungkan ke titik akhir AWS VPC Kriptografi Pembayaran

Anda dapat terhubung ke Kriptografi AWS Pembayaran melalui titik akhir VPC dengan menggunakan SDK, AWS atau. AWS CLI AWS Tools for PowerShell Untuk menentukan VPC endpoint, gunakan nama DNS-nya.

Misalnya, perintah kunci-daftar ini menggunakan parameter `endpoint-url` untuk menentukan VPC endpoint. Untuk menggunakan perintah seperti ini, ganti contoh ID VPC endpoint dengan yang ada di akun Anda.

```
$ aws payment-cryptography list-keys --endpoint-url https://
vpce-1234abcdf5678c90a-09p7654s-us-east-1a.ec2.us-east-1.vpce.amazonaws.com
```

Jika Anda mengaktifkan nama host privat ketika Anda membuat VPC endpoint Anda, Anda tidak perlu menentukan URL VPC endpoint di perintah CLI atau konfigurasi aplikasi. Nama host DNS Kriptografi AWS Pembayaran standar diselesaikan ke titik akhir VPC Anda. SDKs Gunakan AWS CLI dan gunakan nama host ini secara default, sehingga Anda dapat mulai menggunakan titik akhir VPC untuk terhubung ke AWS titik akhir regional Kriptografi Pembayaran tanpa mengubah apa pun dalam skrip dan aplikasi Anda.

Untuk menggunakan nama host pribadi, `enableDnsSupport` atribut `enableDnsHostnames` dan VPC Anda harus disetel ke. `true` Untuk mengatur atribut ini, gunakan ModifyVpcAttributeoperasi. Untuk detailnya, lihat Melihat dan memperbarui atribut DNS untuk VPC Anda di Panduan Pengguna Amazon VPC.

Mengontrol akses ke VPC endpoint

Untuk mengontrol akses ke titik akhir VPC Anda untuk Kriptografi AWS Pembayaran, lampirkan kebijakan titik akhir VPC ke titik akhir VPC Anda. Kebijakan endpoint menentukan apakah prinsipal dapat menggunakan titik akhir VPC untuk memanggil operasi Kriptografi Pembayaran dengan sumber daya Kriptografi AWS Pembayaran tertentu. AWS

Anda dapat membuat kebijakan VPC endpoint ketika Anda membuat titik akhir Anda, dan Anda dapat mengubah kebijakan VPC endpoint setiap saat. Gunakan konsol manajemen VPC, atau operasi atau. CreateVpcEndpointModifyVpcEndpoint Anda juga dapat membuat dan mengubah kebijakan titik akhir VPC dengan menggunakan templat. AWS CloudFormation Untuk bantuan menggunakan konsol manajemen VPC, lihat Membuat titik akhir antarmuka dan Memodifikasi titik akhir antarmuka dalam Panduan.AWS PrivateLink

Untuk mendapatkan bantuan mengenai cara menulis dan memformat dokumen kebijakan JSON, lihat Referensi Kebijakan IAM JSON dalam Panduan Pengguna IAM.

Topik

- Tentang kebijakan VPC endpoint
- Kebijakan VPC endpoint default
- Membuat kebijakan VPC endpoint
- Melihat kebijakan VPC endpoint

Tentang kebijakan VPC endpoint

Agar permintaan Kriptografi AWS Pembayaran yang menggunakan titik akhir VPC berhasil, prinsipal memerlukan izin dari dua sumber:

- Kebijakan berbasis identitas harus memberikan izin utama untuk memanggil operasi pada sumber daya (kunci Kriptografi AWS Pembayaran atau alias).
- Kebijakan VPC endpoint harus memberikan prinsipal izin untuk menggunakan titik akhir untuk membuat permintaan.

Misalnya, kebijakan kunci mungkin memberikan izin utama untuk memanggil Dekripsi pada kunci Kriptografi AWS Pembayaran tertentu. Namun, kebijakan titik akhir VPC mungkin tidak mengizinkan prinsipal tersebut untuk memanggil `Decrypt` kunci Kriptografi AWS Pembayaran tersebut dengan menggunakan titik akhir.

Atau kebijakan titik akhir VPC mungkin memungkinkan prinsipal untuk menggunakan titik akhir untuk memanggil StopKeyUsagekunci Kriptografi Pembayaran tertentu AWS . Tetapi jika prinsipal tidak memiliki izin tersebut dari kebijakan IAM, permintaan gagal.

Kebijakan VPC endpoint default

Setiap VPC endpoint memiliki kebijakan VPC endpoint, tetapi Anda tidak diharuskan untuk menentukan kebijakan. Jika Anda tidak menentukan kebijakan, kebijakan titik akhir default memungkinkan semua operasi oleh semua prinsipal di semua sumber daya pada titik akhir.

Namun, untuk sumber daya Kriptografi AWS Pembayaran, kepala sekolah juga harus memiliki izin untuk memanggil operasi dari kebijakan IAM. Oleh karena itu, dalam praktik, kebijakan default

mengatakan bahwa jika prinsipal memiliki izin untuk memanggil operasi pada sumber daya, mereka juga dapat memanggilnya dengan menggunakan titik akhir.

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

Untuk mengizinkan prinsipal menggunakan titik akhir VPC hanya untuk sebagian dari operasi yang diizinkan, buat atau perbarui kebijakan titik akhir VPC.

Membuat kebijakan VPC endpoint

Kebijakan VPC endpoint menentukan apakah prinsipal memiliki izin untuk menggunakan VPC endpoint untuk melakukan operasi pada sumber daya. Untuk sumber daya Kriptografi AWS Pembayaran, kepala sekolah juga harus memiliki izin untuk melakukan operasi dari kebijakan IAM.

Setiap pernyataan kebijakan VPC endpoint memerlukan unsur-unsur berikut:

- Prinsip-prinsip yang dapat melakukan tindakan
- Tindakan yang dapat dilakukan
- Sumber daya yang dapat digunakan untuk mengambil tindakan

Pernyataan kebijakan tidak menentukan VPC endpoint. Sebaliknya, berlaku untuk VPC endpoint di mana kebijakan tersebut terpasang. Untuk informasi selengkapnya, lihat Mengendalikan akses ke layanan dengan titik akhir VPC dalam Panduan Pengguna Amazon VPC.

Berikut ini adalah contoh kebijakan titik akhir VPC untuk AWS Kriptografi Pembayaran. Saat dilampirkan ke titik akhir VPC, kebijakan ini memungkinkan `ExampleUser` untuk menggunakan titik akhir VPC untuk memanggil operasi yang ditentukan pada kunci Kriptografi Pembayaran yang ditentukan. AWS Sebelum menggunakan kebijakan seperti ini, ganti contoh prinsipal dan pengidentifikasi kunci dengan nilai yang valid dari akun Anda.

```
{
```

```
   "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 mencatat semua operasi yang menggunakan titik akhir VPC. Namun, CloudTrail log Anda tidak menyertakan operasi yang diminta oleh kepala sekolah di akun lain atau operasi untuk kunci Kriptografi AWS Pembayaran di akun lain.

Dengan demikian, Anda mungkin ingin membuat kebijakan titik akhir VPC yang mencegah prinsipal di akun eksternal menggunakan titik akhir VPC untuk memanggil operasi Kriptografi AWS Pembayaran apa pun pada kunci apa pun di akun lokal.

Contoh berikut menggunakan aws: PrincipalAccount global condition key untuk menolak akses ke semua prinsipal untuk semua operasi pada semua kunci Kriptografi AWS Pembayaran kecuali prinsipal ada di akun lokal. Sebelum menggunakan kebijakan seperti ini, ganti ID akun contoh dengan yang valid.

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

```
        }
      }
    }
  ]
}
```

Melihat kebijakan VPC endpoint

Untuk melihat kebijakan titik akhir VPC untuk titik akhir, gunakan konsol manajemen VPC atau operasi. DescribeVpcEndpoints

AWS CLI Perintah berikut mendapatkan kebijakan untuk titik akhir dengan ID titik akhir VPC yang ditentukan.

Sebelum menggunakan perintah ini, ganti ID titik akhir contoh dengan yang valid dari akun Anda.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpce-1234abcdf5678c90a`].[PolicyDocument]'
--output text
```

Menggunakan VPC endpoint dalam pernyataan kebijakan

Anda dapat mengontrol akses ke sumber daya dan operasi Kriptografi AWS Pembayaran ketika permintaan berasal dari VPC atau menggunakan titik akhir VPC. Untuk melakukannya, gunakan salah satu kebijakan IAM

- Gunakan kunci kondisi `aws:sourceVpce` untuk memberikan atau membatasi akses berdasarkan VPC endpoint.
- Gunakan kunci kondisi `aws:sourceVpc` untuk memberikan atau membatasi akses berdasarkan VPC yang menjadi host endpoint privat.

> **Note**
>
> Kunci `aws:sourceIP` kondisi tidak efektif ketika permintaan berasal dari titik akhir VPC Amazon. Untuk membatasi permintaan ke VPC endpoint, gunakan kunci kondisi `aws:sourceVpce` atau `aws:sourceVpc`. Untuk informasi selengkapnya, lihat Identitas dan manajemen akses untuk titik akhir VPC dan layanan titik akhir VPC di Panduan.AWS PrivateLink

Anda dapat menggunakan kunci kondisi global ini untuk mengontrol akses ke kunci Kriptografi AWS Pembayaran, alias, dan operasi seperti CreateKeyitu tidak bergantung pada sumber daya tertentu.

Misalnya, kebijakan kunci sampel berikut memungkinkan pengguna untuk melakukan operasi kriptografi tertentu dengan kunci Kriptografi AWS Pembayaran hanya ketika permintaan menggunakan titik akhir VPC yang ditentukan, memblokir akses baik dari Internet dan AWS PrivateLink koneksi (jika pengaturan). Ketika pengguna membuat permintaan ke Kriptografi AWS Pembayaran, ID titik akhir VPC dalam permintaan dibandingkan `aws:sourceVpce` dengan nilai kunci kondisi dalam kebijakan. Jika tidak cocok, permintaan ditolak.

Untuk menggunakan kebijakan seperti ini, ganti Akun AWS ID placeholder dan titik akhir VPC IDs dengan nilai yang valid untuk akun Anda.

JSON

```
{
    "Id": "example-key-1",
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "EnableIAMPolicies",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "payment-cryptography:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RestrictUsageToMyVPCEndpoint",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "payment-cryptography:EncryptData",
                "payment-cryptography:DecryptData"
            ],
            "Resource": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
*",
```

```
            "Condition": {
                "StringNotEquals": {
                    "aws:sourceVpce": "vpce-1234abcdf5678c90a"
                }
            }
        }

    ]
}
```

Anda juga dapat menggunakan tombol `aws:sourceVpc` kondisi untuk membatasi akses ke kunci Kriptografi AWS Pembayaran Anda berdasarkan VPC tempat titik akhir VPC berada.

Kebijakan kunci sampel berikut memungkinkan perintah yang mengelola kunci Kriptografi AWS Pembayaran hanya ketika mereka berasal`vpc-12345678`. Selain itu, ini memungkinkan perintah yang menggunakan kunci Kriptografi AWS Pembayaran untuk operasi kriptografi hanya ketika mereka berasal. `vpc-2b2b2b2b` Anda mungkin menggunakan kebijakan seperti ini jika aplikasi berjalan dalam satu VPC, tetapi Anda menggunakan VPC terisolasi kedua untuk fungsi manajemen.

Untuk menggunakan kebijakan seperti ini, ganti Akun AWS ID placeholder dan titik akhir VPC IDs dengan nilai yang valid untuk akun Anda.

JSON

```
{
    "Id": "example-key-2",
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminActionsFromVPC12345678",
            "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": "AllowKeyUsageFromVPC2b2b2b2b",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "payment-cryptography:Encrypt*",
                "payment-cryptography:Decrypt*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:sourceVpc": "vpc-2b2b2b2b"
                }
            }
        },
        {
            "Sid": "AllowListReadActionsFromEverywhere",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "payment-cryptography:List*",
                "payment-cryptography:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Mencatat VPC endpoint Anda

AWS CloudTrail mencatat semua operasi yang menggunakan titik akhir VPC. Ketika permintaan ke Kriptografi AWS Pembayaran menggunakan titik akhir VPC, ID titik akhir VPC muncul di entri log yang mencatat permintaan AWS CloudTrail tersebut. Anda dapat menggunakan ID titik akhir untuk mengaudit penggunaan titik akhir VPC Kriptografi AWS Pembayaran Anda.

Untuk melindungi VPC Anda, permintaan yang ditolak oleh kebijakan titik akhir VPC, tetapi sebaliknya diizinkan, tidak dicatat. AWS CloudTrail

Misalnya, entri log contoh ini mencatat GenerateMacpermintaan yang menggunakan titik akhir VPC. Bidang `vpcEndpointId` muncul di akhir entri log.


```
{
      "eventVersion": "1.08",
      "userIdentity": {
          "principalId": "TESTXECZ5U9M4LGF2N6Y5:i-98761b8890c09a34a",
          "arn": "arn:aws:sts::111122223333:assumed-role/samplerole/
i-98761b8890c09a34a",
          "accountId": "111122223333",
          "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
          "sessionContext": {
              "sessionIssuer": {
                  "type": "Role",
                  "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                  "arn": "arn:aws:iam::111122223333:role/samplerole",
                  "accountId": "111122223333",
                  "userName": "samplerole"
              },
              "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,
```

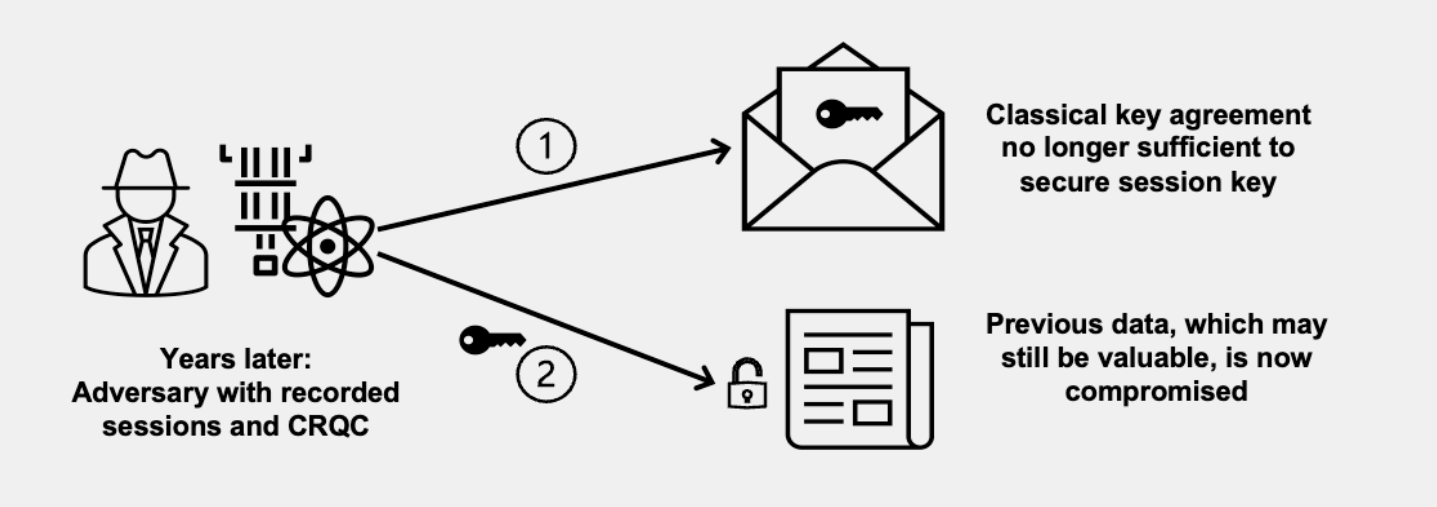
```
            "exportable": true,
            "keyState": "CREATE_COMPLETE",
            "keyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "createTimestamp": "May 27, 2024, 7:49:54 PM",
            "usageStartTimestamp": "May 27, 2024, 7:49:54 PM"
        }
    },
    "requestID": "f3020b3c-4e86-47f5-808f-14c7a4a99161",
    "eventID": "b87c3d30-f3ab-4131-87e8-bc54cfef9d29",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "vpcEndpointId": "vpce-1234abcdf5678c90a",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "vpce-1234abcdf5678c90a-
oo28vrvr.controlplane.payment-cryptography.us-east-1.vpce.amazonaws.com"
    }
}
```


Menggunakan TLS pasca-kuantum hibrida

AWS Kriptografi Pembayaran dan banyak layanan lainnya mendukung opsi pertukaran kunci pasca-kuantum hibrida untuk protokol enkripsi jaringan Transport Layer Security (TLS). Anda dapat menggunakan opsi TLS ini saat Anda terhubung ke titik akhir API atau saat menggunakan AWS. SDKs Fitur pertukaran kunci pasca-kuantum hibrida opsional ini setidaknya seaman enkripsi TLS yang kami gunakan saat ini dan cenderung memberikan manfaat keamanan jangka panjang tambahan.

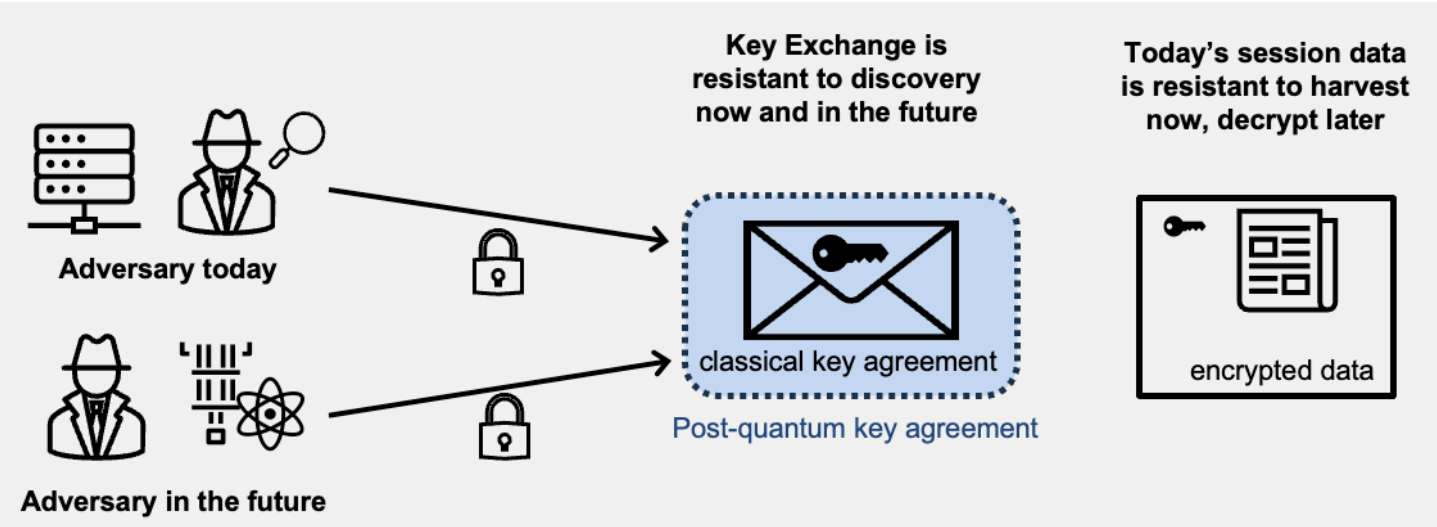
Data yang Anda kirim ke layanan yang diaktifkan dilindungi saat transit oleh enkripsi yang disediakan oleh koneksi Transport Layer Security (TLS). Suite cipher klasik berdasarkan RSA dan ECC yang didukung AWS Payment Cryptography untuk sesi TLS membuat serangan brute force pada mekanisme pertukaran kunci tidak layak dengan teknologi saat ini. Namun, jika komputer kuantum skala besar atau kriptografi yang relevan (CRQC) menjadi praktis di masa depan, mekanisme pertukaran kunci TLS yang ada akan rentan terhadap serangan ini. Ada kemungkinan bahwa musuh dapat mulai memanen data terenkripsi sekarang dengan harapan mereka dapat mendekripsi di masa depan (panen sekarang, dekripsi nanti). Jika Anda mengembangkan aplikasi yang mengandalkan

kerahasiaan jangka panjang dari data yang melewati koneksi TLS, Anda harus mempertimbangkan rencana untuk bermigrasi ke kriptografi pasca-kuantum sebelum komputer kuantum skala besar tersedia untuk digunakan. AWS sedang bekerja untuk mempersiapkan masa depan ini, dan kami ingin Anda juga dipersiapkan dengan baik.



Untuk melindungi data yang dienkripsi hari ini terhadap potensi serangan future, AWS berpartisipasi dengan komunitas kriptografi dalam pengembangan algoritma tahan kuantum atau pasca-kuantum. AWS telah menerapkan suite cipher pertukaran kunci pasca-kuantum hibrida yang menggabungkan elemen klasik dan pasca-kuantum untuk memastikan bahwa koneksi TLS Anda setidaknya sekuat dengan suite cipher klasik.

Suite sandi hibrida ini tersedia untuk digunakan pada beban kerja produksi Anda saat menggunakan AWS versi terbaru. SDKs Untuk informasi lebih lanjut tentang bagaimana perilaku enable/disable ini, silakan lihat ???



Tentang pertukaran kunci pasca-kuantum hibrida di TLS

Algoritma yang AWS digunakan adalah hibrida yang menggabungkan Elliptic Curve Diffie-Hellman (ECDH), algoritma pertukaran kunci klasik yang digunakan saat ini di TLS, dengan Module-Lattice-Based Key-Encapsulation Mechanism (), enkripsi kunci publik dan algoritma pembentukan kunci yang National Institute for Standards and Technology (NISTML-KEM) telah ditetapkan sebagai algoritma kesepakatan kunci pasca-kuantum standar pertama. Hibrida ini menggunakan masing-masing algoritma secara independen untuk menghasilkan kunci. Selanjutnya menggabungkan dua kunci kriptografi.

Pelajari lebih lanjut tentang PQC

Untuk informasi tentang proyek kriptografi pasca-kuantum di National Institute for Standards and Technology (NIST), lihat Kriptografi Pasca Kuantum.

Untuk informasi tentang standardisasi kriptografi pasca-kuantum NIST, lihat Standardisasi Kriptografi Pasca-Kuantum.

Mengaktifkan TLS pasca-kuantum hibrida

AWS SDKs dan alat memiliki kemampuan dan konfigurasi kriptografi yang berbeda antar bahasa dan runtime. Ada tiga cara AWS SDK atau alat saat ini menyediakan dukungan PQ TLS:

Topik

SDKs dengan PQ TLS diaktifkan secara default

> **Note**
>
> Mulai 6-Nov-2025, AWS SDK dan pustaka CRT yang mendasarinya untuk macOS dan Windows menggunakan pustaka sistem untuk TLS, sehingga kemampuan PQ TLS pada platform tersebut umumnya ditentukan oleh dukungan tingkat sistem.

AWS SDK for Go

AWS SDK for Go menggunakan implementasi TLS Golang sendiri yang disediakan oleh pustaka standarnya. Golang mendukung dan lebih memilih PQ TLS pada v1.24, sehingga pengguna AWS SDK for Go dapat mengaktifkan PQ TLS hanya dengan memutakhirkan Golang ke v1.24

AWS SDK untuk JavaScript (browser)

AWS SDK for JavaScript (browser) menggunakan tumpukan TLS browser, sehingga SDK akan menegosiasikan PQ TLS jika runtime browser mendukung dan memilihnya. Firefox meluncurkan dukungan untuk PQ TLS di v132.0. Chrome mengumumkan dukungan untuk PQ TLS di v131. Edge mendukung opt-in PQ TLS di v120 untuk desktop dan 140 untuk Android.

AWS SDK for Node.js

Pada Node.js v22.20 (LTS) dan v24.9.0, Node.js secara statis menautkan dan membundel OpenSSL 3.5. Ini berarti bahwa PQ TLS diaktifkan dan disukai secara default untuk versi tersebut dan versi berikutnya.

AWS SDK untuk Kotlin

Kotlin SDK mendukung dan lebih memilih PQ TLS di Linux pada v1.5.78. Karena AWS SDK untuk klien berbasis CRT Kotlin bergantung pada pustaka sistem untuk TLS di macOS dan Windows, dukungan untuk PQ TLS akan bergantung pada pustaka sistem yang mendasarinya.

AWS SDK untuk Rust

AWS SDK for Rust mendistribusikan paket berbeda (dikenal sebagai “peti” di ekosistem Rust) untuk setiap klien layanan. Ini semua dikelola dalam GitHub repositori terkonsolidasi, tetapi setiap klien layanan mengikuti versi dan irama rilisnya sendiri. SDK terkonsolidasi merilis preferensi PQ TLS pada 8/29/25, sehingga setiap versi klien layanan individu yang dirilis setelah tanggal tersebut akan mendukung dan lebih memilih PQ TLS secara default.

Anda dapat menentukan versi minimum yang mendukung PQ TLS untuk klien layanan tertentu dengan menavigasi ke URL versi crates.io yang relevan (misalnya, Kriptografi AWS Pembayaran ada di sini) dan menemukan versi pertama yang diterbitkan setelah 29-Agustus-25. Setiap versi klien layanan yang diterbitkan setelah 29-Agustus-25 akan mengaktifkan PQ TLS dan disukai secara default.

Dukungan PQ TLS ikut serta

AWS SDK for C++

Secara default, C++ SDK menggunakan klien platform-native seperti libcurl dan. WinHttp Libcurl umumnya bergantung pada sistem OpenSSL untuk TLS, jadi PQ TLS hanya diaktifkan secara default jika sistem OpenSSL ≥ v3.5. Anda dapat mengganti default ini di C ++ SDK v1.11.673 atau yang lebih baru, dan memilih untuk mendukung dan mengaktifkan PQ TLS AwsCrtHttpClient secara default.

Catatan tentang Membangun untuk Opt-In PQ TLS Anda dapat mengambil dependensi CRT SDK dengan skrip ini. Membangun SDK dari sumber dijelaskan di sini dan di sini, tetapi perhatikan bahwa Anda mungkin memerlukan beberapa CMake tanda tambahan:

```
-DUSE_CRT_HTTP_CLIENT=ON \
-DUSE_TLS_V1_2=OFF \
-DUSE_TLS_V1_3=ON \
-DUSE_OPENSSL=OFF \
```

AWS SDK for Java

Mulai v2, AWS SDK for Java menyediakan Klien HTTP AWS Common Runtime (AWS CRT) yang dapat dikonfigurasi untuk melakukan PQ TLS. Mulai v2.35.11, AwsCrtHttpClient mengaktifkan dan lebih memilih PQ TLS secara default di mana pun digunakan.

SDKs yang mengandalkan Sistem OpenSSL

Beberapa AWS SDKs dan alat bergantung pada libcrypto/libssl pustaka sistem untuk TLS. Perpustakaan sistem yang paling sering digunakan adalah OpenSSL. OpenSSL mengaktifkan dukungan PQ TLS di versi 3.5, jadi cara termudah untuk mengkonfigurasi SDKs ini dan alat untuk PQ TLS adalah dengan menggunakannya pada distribusi sistem operasi yang setidaknya memiliki OpenSSL 3.5 diinstal.

Anda juga dapat mengonfigurasi wadah Docker untuk menggunakan OpenSSL 3.5 untuk mengaktifkan PQ TLS pada sistem apa pun yang mendukung Docker. Lihat TLS pasca-kuantum dengan Python untuk contoh pengaturan ini untuk Python.

AWS CLI

Dukungan PQ TLS dengan penginstal AWS CLI akan segera hadir. Untuk mengaktifkan segera, Anda dapat menggunakan penginstal alternatif untuk AWS CLI, yang bervariasi menurut sistem operasi, dan dapat mengaktifkan PQ TLS.

Untuk macOS, instal AWS CLI melalui Homebrew dan pastikan OpenSSL yang dijual di Homebrew Anda ditingkatkan ke versi 3.5+. Anda dapat melakukan ini dengan “brew install openssl @3 .6" dan validasi dengan “brew list | grep openssl”.

Untuk Ubuntu atau Debian Linux: pastikan distribusi Linux yang Anda gunakan memiliki OpenSSL 3.5+ diinstal sebagai sistem OpenSSL. Kemudian, instal AWS CLI menggunakan apt atau PyPI. Dengan prasyarat ini, AWS CLI yang dijual oleh apt atau PyPI akan dikonfigurasi untuk menegosiasikan PQ-TLS. Untuk step-by-step petunjuk untuk memvalidasi instalasi, lihat repositori github dan posting blog yang menyertainya.

AWS SDK for PHP

AWS SDK for PHP bergantung pada sistem libssl/libcrypto. Untuk menggunakan PQ TLS, gunakan SDK ini pada distribusi sistem operasi yang memiliki setidaknya OpenSSL 3.5 diinstal.

AWS SDK untuk Python (Boto3)

AWS SDK for Python (Boto3) mengandalkan libssl/libcrypto sistem. Untuk menggunakan PQ TLS, gunakan SDK ini pada distribusi sistem operasi yang memiliki setidaknya OpenSSL 3.5 diinstal.

AWS SDK for Ruby

AWS SDK for Ruby bergantung pada sistem libssl/libcrypto. Untuk menggunakan PQ TLS, gunakan SDK ini pada distribusi sistem operasi yang memiliki setidaknya OpenSSL 3.5 diinstal.

AWS SDKs dan alat tidak berencana untuk mendukung PQ TLS

Saat ini tidak ada rencana untuk mendukung bahasa SDKs dan alat berikut:

- AWS SDK for .NET
- AWS SDK untuk Swift
- Alat AWS untuk Windows PowerShell

Praktik terbaik keamanan untuk Kriptografi AWS Pembayaran

AWS Kriptografi Pembayaran mendukung banyak fitur keamanan yang built-in atau yang dapat Anda terapkan secara opsional untuk meningkatkan perlindungan kunci enkripsi Anda dan memastikan bahwa mereka digunakan untuk tujuan yang dimaksudkan, termasuk kebijakan IAM, serangkaian kunci kondisi kebijakan yang ekstensif untuk menyempurnakan kebijakan utama Anda dan kebijakan IAM dan penegakan aturan PIN PCI bawaan mengenai blok kunci.

> **Important**
>
> Pedoman umum yang diberikan tidak mewakili solusi keamanan yang lengkap. Karena tidak semua praktik terbaik sesuai untuk semua situasi, ini tidak dimaksudkan untuk menjadi preskriptif.

- Penggunaan Utama dan Mode Penggunaan: Kriptografi AWS Pembayaran mengikuti dan memberlakukan pembatasan penggunaan utama dan mode penggunaan seperti yang dijelaskan dalam ANSI X9 TR 31-2018 Spesifikasi Blok Kunci Pertukaran Kunci Aman yang Dapat Dioperasikan dan konsisten dengan Persyaratan Keamanan PIN PCI 18-3. Ini membatasi kemampuan untuk menggunakan satu kunci untuk berbagai tujuan dan secara kriptografis mengikat metadata kunci (seperti operasi yang diizinkan) ke materi kunci itu sendiri. AWS Kriptografi Pembayaran secara otomatis memberlakukan pembatasan ini seperti kunci enkripsi kunci (TR31_K0_KEY_ENCRYPTION_KEY) juga tidak dapat digunakan untuk dekripsi data. Lihat Memahami atribut kunci untuk kunci Kriptografi AWS Pembayaran untuk detail selengkapnya.
- Batasi pembagian materi kunci simetris: Hanya bagikan materi kunci simetris (seperti Kunci Enkripsi Pin atau Kunci Enkripsi Kunci) dengan paling banyak satu entitas lainnya. Jika ada kebutuhan untuk mentransmisikan materi sensitif ke lebih banyak entitas atau mitra, buat kunci tambahan. AWS Kriptografi Pembayaran tidak pernah mengekspos materi kunci simetris atau materi kunci pribadi asimetris secara jelas.
- Gunakan alias atau tag untuk mengaitkan kunci dengan kasus penggunaan atau mitra tertentu: Alias dapat digunakan untuk dengan mudah menunjukkan kasus penggunaan yang terkait dengan kunci seperti alias/BIN_12345_CVK untuk menunjukkan kunci verifikasi kartu yang terkait dengan BIN 12345. Untuk memberikan lebih banyak fleksibilitas, pertimbangkan untuk membuat tag seperti bin = 12345, use_case=acquiring, country=us, partner=foo. Alias dan tag juga dapat digunakan untuk membatasi akses seperti menegakkan kontrol akses antara mengeluarkan dan memperoleh kasus penggunaan.

- Praktekkan akses yang paling tidak istimewa: IAM dapat digunakan untuk membatasi akses produksi ke sistem daripada individu, seperti melarang pengguna individu membuat kunci atau menjalankan operasi kriptografi. IAM juga dapat digunakan untuk membatasi akses ke perintah dan kunci yang mungkin tidak berlaku untuk kasus penggunaan Anda, seperti membatasi kemampuan untuk menghasilkan atau memvalidasi pin untuk pengakuisisi. Cara lain untuk menggunakan akses yang paling tidak memiliki hak istimewa adalah dengan membatasi operasi sensitif (seperti impor kunci) ke akun layanan tertentu. Lihat AWS Contoh kebijakan berbasis identitas Kriptografi Pembayaran sebagai contoh.

Lihat juga

- Manajemen identitas dan akses untuk Kriptografi AWS Pembayaran
- Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM

Validasi kepatuhan untuk Kriptografi AWS Pembayaran

Seperti AWS layanan lainnya, pelanggan memerlukan pemahaman yang jelas tentang model tanggung jawab bersama untuk keamanan dan kepatuhan. Sebagai layanan yang secara khusus mendukung pembayaran, kepatuhan terhadap standar PCI yang berlaku sangat penting untuk dipahami bagi pelanggan Kriptografi AWS Pembayaran. AWS Penilaian PCI DSS dan PCI 3DS mencakup Kriptografi Pembayaran. AWS Mungkin ada referensi ke layanan dalam Panduan Tanggung Jawab Bersama, tersedia dari AWS Artifact, untuk laporan ini. Penilaian PCI PIN Security and Point-to-Point Encryption (P2PE) khusus untuk Kriptografi Pembayaran. AWS

Bagian ini memberikan informasi tentang status dan ruang lingkup kepatuhan layanan dan informasi yang akan membantu dalam merencanakan PCI PIN Security dan PCI P2PE penilaian aplikasi Anda.

Topik

Kepatuhan layanan

Auditor pihak ketiga menilai keamanan dan kepatuhan Kriptografi AWS Pembayaran sebagai bagian dari beberapa program AWS kepatuhan. Ini termasuk SOC, PCI, dan lainnya.

AWS Kriptografi Pembayaran telah dinilai untuk beberapa standar PCI selain PCI DSS dan PCI 3DS. Ini termasuk PCI PIN Security (PCI PIN) dan PCI Point-to-Point (P2PE) Enkripsi. Silakan lihat AWS Artifact untuk pengesahan dan panduan kepatuhan yang tersedia.

Untuk daftar AWS layanan dalam lingkup program kepatuhan tertentu, lihat AWS Services in Scope by Compliance Program . Untuk informasi umum, lihat Program Kepatuhan AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Kriptografi AWS Pembayaran ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan kepatuhan. AWS
- AWS Sumber Daya AWS —Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang —AWS Config; menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub CSPM AWS Layanan ini memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Perencanaan Kepatuhan PIN

Panduan ini menjelaskan dokumentasi dan bukti bahwa Anda perlu mempersiapkan penilaian PIN PCI dari aplikasi pemrosesan PIN Anda yang menggunakan Kriptografi AWS Pembayaran.

Seperti standar kepatuhan Layanan AWS lainnya, Anda bertanggung jawab untuk menggunakan layanan dengan aman, mengonfigurasi kontrol akses, dan menggunakan parameter keamanan sesuai dengan persyaratan PIN PCI. Panduan ini akan membahas konfigurasi tersebut bila sesuai untuk memenuhi persyaratan.

Topik

Topik Umum

Migrasi aplikasi dari menghubungkan ke HSM ke layanan terkelola seperti Kriptografi AWS Pembayaran memunculkan masalah dan konsep umum bagi pelanggan dan penilai mereka. Bagian ini memberikan informasi untuk memperjelas bagaimana penggunaan layanan yang aman mengatasi situasi ini.

Topik

Tanggung Jawab Bersama

Pelanggan yang telah memikul tanggung jawab keamanan dan kepatuhan penuh untuk aplikasi akan merestrukturisasi kepatuhan mereka untuk memanfaatkan manajemen kunci AWS Payment Cryptography, kontrol keamanan dan kemampuan HSM yang dikelola (“layanan”). Ini akan mengalihkan beberapa persyaratan sepenuhnya ke AWS, sebagaimana dibuktikan oleh penilaian pihak ketiga AWS Payment Cryptography. Beberapa persyaratan akan dibagi antara aplikasi pelanggan dan layanan. Aplikasi bertanggung jawab untuk:

- Memberikan informasi yang akurat untuk layanan
- Menggunakan kontrol keamanan sesuai dengan rekomendasi layanan dan persyaratan keamanan PIN PCI
- Menerapkan kontrol keamanan yang diperlukan menggunakan alat yang disediakan oleh layanan

Pelanggan dan penilai mereka akan menggunakan tanggung jawab bersama dan panduan implementasi yang diterbitkan dengan pengesahan kepatuhan AWS Artifact untuk menerapkan kontrol dan pemantauan kontrol kemudian merencanakan dan menyelesaikan penilaian.

Konfigurasi HSM minimum

Standar Keamanan Data PCI, standar dasar untuk standar PCI lainnya, mengharuskan semua sistem dikonfigurasi dengan fungsionalitas minimum yang diperlukan untuk fungsinya. PCI PIN, P2PE, dan standar solusi lainnya menerapkan persyaratan ini HSMs dalam solusi. HSMs hanya harus mengaktifkan fungsi yang diperlukan untuk solusi.

AWS Layanan harus diperlakukan sebagai sistem dan dikonfigurasi untuk fungsionalitas minimum yang diperlukan. Standar Keamanan Data Industri Kartu Pembayaran (PCI DSS) v4.0 di AWS merekomendasikan penggunaan IAM untuk mengonfigurasi fungsionalitas minimum untuk setiap layanan AWS yang digunakan oleh solusi. Ini juga berlaku untuk Kriptografi AWS Pembayaran. Kebijakan IAM memungkinkan izin halus untuk membatasi fungsi kriptografi hanya untuk komponen aplikasi yang mengandalkannya.

Pertukaran kunci antara pelanggan dan APC

PIN PIN Persyaratan keamanan 8-4 dan 15-2 memerlukan kunci publik untuk pertukaran dan pemuatan kunci diautentikasi dan dilindungi integritas. Untuk pemuatan kunci jarak jauh POI, yang dijelaskan secara fungsional dalam ANSI/ASC X9 TR-34 dan diatur oleh PCI PIN Annex A, kunci publik paling sering disampaikan dalam sertifikat yang ditandatangani oleh otoritas sertifikat yang sesuai dengan Lampiran A2. Untuk pertukaran antar organisasi, kunci publik menggunakan mekanisme lain untuk keaslian dan integritas.

Semua interaksi antara pelanggan dan AWS dilakukan melalui AWS APIs, yang saling mengautentikasi setiap panggilan API dan memastikan integritas panggilan dan respons menggunakan TLS. Otentikasi aplikasi pelanggan dikelola oleh AWS Identity and Access Management dengan mekanisme seperti Token Keamanan dan SigV4. Titik akhir AWS API diautentikasi oleh pelanggan menggunakan otentikasi server TLS, yang dibangun ke AWS. SDKs Kemudian TLS menjamin kerahasiaan dan integritas semua data yang diteruskan antara pelanggan dan setiap AWS API.

APC APIs GetParametersForImport dan ImportKey menerapkan transfer kunci dari pelanggan ke layanan. Meskipun Otoritas Sertifikat (CA) yang GetParametersForImport disediakan oleh tidak sesuai dengan Lampiran A2, itu aman dan unik untuk akun. Meskipun CA ini tidak dapat diandalkan untuk memenuhi persyaratan 8-4 dan 15-2, CA ini memberikan verifikasi integritas kunci impor. Anda juga dapat menggunakan CA Anda sendiri dengan memanfaatkan GetCertificateSigningRequest API.

Mekanisme yang menyediakan otentikasi kunci publik dan jaminan integritas adalah:

- Otentikasi disediakan oleh autentikasi AWS API
- Integritas kunci disediakan oleh fitur MAC dari sertifikat yang disediakan oleh GetParametersForImport, bahkan jika informasi identitas dalam sertifikat tidak dipercaya. Integritas kunci juga dijamin oleh MAC yang digunakan oleh TLS yang melindungi sesi antara pelanggan dan AWS

Sertifikat dan blok kunci yang disediakan oleh APC sesuai dengan Lampiran A1, yang menetapkan persyaratan untuk sertifikat dan perlindungan kunci dengan metode asimetris.

Lingkup Penilaian

Langkah pertama dalam merencanakan penilaian apa pun adalah mendokumentasikan ruang lingkup. Untuk PIN PCI, ruang lingkupnya adalah sistem dan proses yang melindungi PINs, termasuk

perlindungan kunci kriptografi dan perangkat yang melindunginya - terminal pembayaran, juga disebut points-of-interaction (POI), HSMs, dan perangkat kriptografi aman lainnya (SCD).

Kami tidak akan menangani persyaratan di mana Anda mempertahankan tanggung jawab penuh karena area alamat ini di luar ruang lingkup layanan. Misalnya, konfigurasi dan penyediaan terminal pembayaran. Lihat Panduan Tanggung Jawab Bersama Kriptografi AWS Pembayaran untuk PIN PCI, tersedia di AWS Artifact

Topik

Tanggung Jawab Bersama

AWS Kriptografi Pembayaran adalah Organisasi Enkripsi dan Dukungan (ESO) dan Layanan Pihak Ketiga yang memperoleh PIN (TPS), sebagaimana didefinisikan oleh Program Keamanan PIN Visa dan terdaftar di Visa Global Service Provider Registry, di bawah “Amazon Web Services, LLC”. Ini berarti bahwa layanan ini diizinkan oleh Visa untuk digunakan oleh VisaNet Prosesor Pihak Ketiga (VNP) yang memperoleh PIN, Prosesor Klien VisaNet yang memperoleh PIN yang bertindak sebagai Penyedia Layanan, dan penyedia TPS dan ESO lainnya tanpa memerlukan penilaian lebih lanjut oleh penilai PIN pelanggan (PCI Qualified PIN Assesors atau PCI QPA).

Merek kartu lain atau penyedia jaringan pembayaran dapat mengandalkan Program Keamanan PIN Visa atau memiliki program mereka sendiri. Hubungi AWS Dukungan untuk pertanyaan tentang kepatuhan layanan untuk program jaringan pembayaran lainnya.

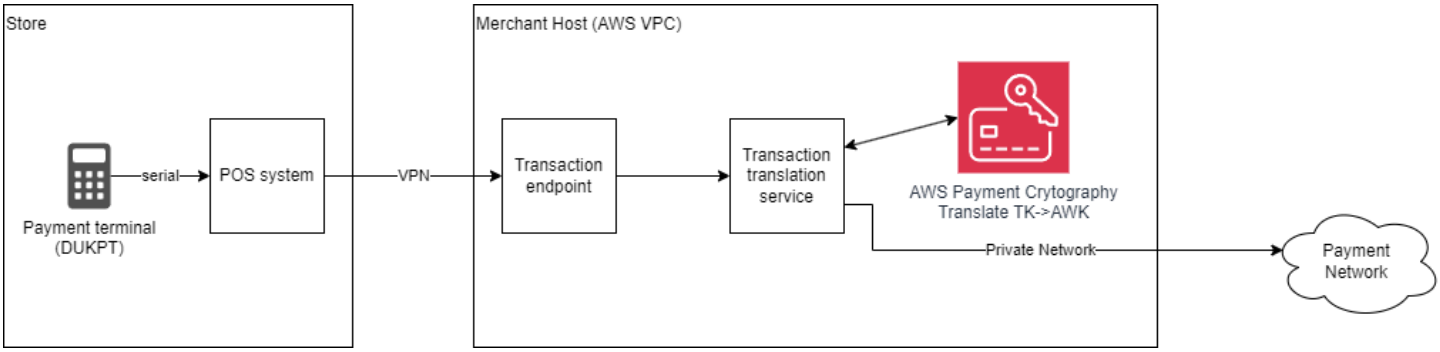
AWS menyediakan pengesahan kepatuhan Keamanan PIN PCI (AOC) dan Panduan Tanggung Jawab Bersama untuk AWS Kriptografi Pembayaran di. AWS Artifact Penggunaan penyedia layanan dalam pemrosesan PIN sudah umum selama bertahun-tahun, namun Standar Keamanan PIN PCI, hingga versi 3.1, tidak membahas manajemen penyedia layanan pihak ketiga. Program Keamanan PIN Visa juga tidak. QPA pelanggan telah mengikuti model yang ditetapkan dengan PCI DSS AOC dan Panduan Tanggung Jawab Bersama yang mengacu pada AWS'kepatuhan sebagai keberhasilan pengujian untuk persyaratan yang berlaku.

Diagram Jaringan Tingkat Tinggi

Template Pelaporan PIN PCI mensyaratkan, “Untuk entitas yang terlibat dalam pemrosesan transaksi berbasis PIN, sediakan skema jaringan yang menjelaskan aliran transaksi berbasis PIN dengan penggunaan tipe kunci terkait. Selain itu, KIFs dan entitas yang terlibat dalam distribusi kunci jarak jauh menggunakan teknik asimetris harus menyediakan aliran material kunci”

AWS Kriptografi Pembayaran telah melaporkan struktur layanan internal untuk penilaian PIN PCI kami. Diagram Anda akan menggambarkan panggilan layanan APIs untuk pemrosesan PIN.

Contoh diagram jaringan tingkat tinggi untuk aplikasi PIN menggunakan Kriptografi AWS Pembayaran:



Tabel Kunci

Laporan tersebut mensyaratkan bahwa semua kunci yang melindungi PINs, secara langsung atau tidak langsung, dicantumkan. Kunci apa pun yang ada dalam layanan dapat dicantumkan dengan ListKeysAPI.

Pastikan untuk memberikan daftar kunci untuk semua wilayah dan akun yang memiliki kunci untuk aplikasi Anda.

Referensi Dokumen

Dokumentasi dan rekomendasi vendor untuk penggunaan Kriptografi AWS Pembayaran yang aman ada di Panduan Pengguna dan Referensi API. Ini terkait, sebagaimana mestinya, dalam panduan ini.

Operasi Pemrosesan Transaksi

Persyaratan PIN PCI diatur dalam Tujuan Pengendalian. Setiap Objektif Kontrol mengelompokkan persyaratan untuk mengamankan aspek keamanan untuk PINs.

Topik

Tujuan Kontrol 1: PINs digunakan dalam transaksi yang diatur oleh persyaratan ini diproses menggunakan peralatan dan prosedur yang memastikan mereka tetap aman.

Persyaratan 1: HSMs digunakan oleh Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI kami. Untuk pelanggan yang menggunakan layanan ini, Persyaratan 1-3 dan 1-4 adalah “In Place” relatif terhadap HSM yang dikelola oleh layanan. Temuan untuk HSM akan menyatakan bahwa pengujian telah dibuktikan oleh QPA. AWS Pengesahan Kepatuhan PIN tersedia untuk direferensikan. AWS Artifact SCD lainnya, seperti POI, dalam solusi Anda perlu diinventarisasi dan direferensikan.

Persyaratan 2: Dokumentasi prosedur Anda harus menentukan bagaimana pemegang kartu PINs dilindungi sehubungan dengan membocorkan kepada personel Anda, protokol terjemahan PIN yang diterapkan, dan perlindungan selama pemrosesan on-line dan off-line. Selain itu, dokumentasi Anda harus berisi ringkasan metode manajemen kunci kriptografi yang digunakan dalam setiap zona.

Persyaratan 3: POI harus dikonfigurasi untuk enkripsi dan transmisi PIN yang aman. AWS Kriptografi Pembayaran hanya mendukung terjemahan blok PIN yang ditentukan dalam Persyaratan 3-3.

Persyaratan 4: Aplikasi tidak boleh menyimpan blok PIN. Blok PIN, bahkan dienkripsi, tidak boleh disimpan dalam jurnal transaksi atau log. Layanan tidak menyimpan blok PIN dan penilaian PIN memverifikasi bahwa mereka tidak ada dalam log.

Perhatikan bahwa standar Keamanan PIN PCI berlaku untuk memperoleh “manajemen, pemrosesan, dan transmisi data nomor identifikasi pribadi (PIN) yang aman selama pemrosesan transaksi kartu pembayaran online dan offline di terminal ATMs dan point-of-sale (POS)”, sebagaimana dinyatakan dalam standar. Namun, standar ini sering digunakan untuk menilai manajemen kunci kriptografi untuk pembayaran di luar ruang lingkup yang dimaksudkan. Ini mungkin termasuk kasus penggunaan penerbit di mana PINs disimpan. Pengecualian untuk persyaratan untuk kasus-kasus ini harus disepakati dengan audiens yang dituju untuk penilaian.

Tujuan Kontrol 2: Kunci kriptografi yang digunakan untuk PIN encryption/decryption dan manajemen kunci terkait dibuat menggunakan proses yang memastikan bahwa tidak mungkin untuk memprediksi kunci apa pun atau menentukan bahwa kunci tertentu lebih mungkin daripada kunci lainnya.

Persyaratan 5: Pembuatan kunci dengan Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI kami. Ini dapat ditentukan dalam tabel kunci “Dihasilkan oleh” kolom.

Persyaratan 6: Kontrol keamanan untuk kunci yang disimpan dalam Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Sertakan deskripsi kontrol keamanan yang berkaitan dengan pembuatan kunci dalam aplikasi Anda dan dengan penyedia layanan lainnya.

Persyaratan 7: Anda harus memiliki dokumentasi kebijakan pembuatan kunci yang harus menentukan bagaimana kunci dihasilkan dan semua pihak yang terkena dampak harus mengetahui prosedur/kebijakan ini. Prosedur untuk pembuatan kunci menggunakan APC API harus mencakup penggunaan peran dengan izin pembuatan kunci dan persetujuan untuk menjalankan skrip atau kode lain yang membuat kunci. AWS CloudTrail log berisi semua CreateKeyperistiwa dengan tanggal dan waktu, ARN kunci, dan id pengguna. Nomor seri dan log HSM untuk akses ke media fisik dinilai sebagai bagian dari penilaian PIN layanan.

Tujuan Kontrol 3: Kunci disampaikan atau ditransmisikan dengan cara yang aman.

Persyaratan 8: Alat angkut kunci dengan Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI kami. Anda perlu mendokumentasikan mekanisme perlindungan kunci untuk transfer sebelum impor ke dan setelah ekspor dari Kriptografi AWS Pembayaran. Layanan ini menyediakan nilai pemeriksaan kunci untuk semua kunci untuk memvalidasi pengiriman yang benar.

Persyaratan 8-4 mensyaratkan bahwa kunci publik disampaikan dengan cara yang melindungi integritas dan keasliannya. Pengangkutan antara aplikasi Anda dan AWS dikendalikan oleh otentikasi aplikasi ke AWS, menggunakan AWS Identity and Access Management metode, AWS'otentikasi titik akhir API ke aplikasi melalui sertifikat server TLS. Selain itu,

kunci publik yang diekspor dari atau diimpor ke Kriptografi AWS Pembayaran memiliki sertifikat yang ditandatangani oleh sementara, khusus pelanggan CAs (Lihat,, dan). GetPublicKeyCertificateGetParametersForImportGetParametersForExport Ini CAs tidak dapat digunakan sebagai satu-satunya metode otentikasi, karena tidak sesuai dengan PCI PIN Security Annex A2. Namun, sertifikat masih memberikan jaminan integritas untuk kunci publik dengan IAM menyediakan otentikasi.

Saat bertukar kunci publik dengan mitra bisnis Anda menggunakan metode asimetris, Anda harus menyediakan otentikasi bisnis melalui saluran komunikasi, menggunakan situs web pertukaran file yang aman, misalnya.

Persyaratan 9: Layanan tidak menggunakan atau secara langsung mendukung komponen kunci teks yang jelas.

Persyaratan 10: Layanan memberlakukan kekuatan kunci relatif untuk melindungi kunci untuk pengangkutan. Anda bertanggung jawab atas pengiriman kunci sebelum impor ke dan setelah ekspor dari Kriptografi AWS Pembayaran dan menggunakan parameter API dan TR-31 yang akurat untuk impor, ekspor, dan pembuatan kunci. Anda harus mendokumentasikan prosedur untuk menggambarkan mekanisme pengangkutan utama dan daftar kunci kriptografi yang digunakan untuk pengangkutan.

Persyaratan 11: Dokumentasi prosedur Anda harus menentukan bagaimana kunci disampaikan. Prosedur untuk pengangkutan kunci menggunakan AWS Payment Cryptography API harus mencakup penggunaan peran dengan izin impor dan ekspor kunci dan persetujuan untuk menjalankan skrip atau kode lain yang membuat kunci. AWS CloudTrail log berisi semua ImportKeydan ExportKeyperistiwa.

Tujuan Kontrol 4: Pemuatan kunci ke HSMs dan perangkat penerimaan PIN POI ditangani dengan cara yang aman.

Persyaratan 12: Anda bertanggung jawab untuk memuat kunci dari komponen atau saham. Manajemen kunci utama HSM dinilai sebagai bagian dari penilaian PIN layanan. AWS Kriptografi Pembayaran tidak memuat kunci dari saham atau komponen individu. Lihat Detail kriptografi bagiannya.

Persyaratan 13 dan 14: Anda harus menjelaskan perlindungan kunci untuk transfer sebelum impor ke dan setelah ekspor dari layanan.

Persyaratan 15: Kriptografi AWS Pembayaran memberikan nilai cek kunci untuk semua kunci dalam layanan dan jaminan integritas untuk kunci publik. Aplikasi Anda bertanggung jawab untuk

menggunakan pemeriksaan ini untuk memvalidasi kunci setelah mengimpor atau mengekspor dari layanan. Anda harus mendokumentasikan prosedur untuk memastikan bahwa mekanisme validasi ada.

Persyaratan 15-2 mensyaratkan bahwa kunci publik dimuat dengan cara yang melindungi integritas dan keasliannya. ImportKey, bersama dengan GetParametersForImport, menyediakan validasi sertifikat penandatanganan yang disediakan. Jika sertifikat yang diberikan ditandatangani sendiri, maka otentikasi harus disediakan oleh mekanisme terpisah, misalnya pertukaran file aman.

Persyaratan 16: Dokumentasi prosedur Anda harus menentukan bagaimana kunci dimuat ke layanan. Prosedur untuk impor kunci menggunakan API harus mencakup penggunaan peran dengan izin impor kunci dan persetujuan untuk menjalankan skrip atau kode lain yang memuat kunci. AWS CloudTrail log berisi semua ImportKeyperistiwa. Anda harus menyertakan mekanisme logging dalam dokumentasi. Layanan ini menyediakan nilai pemeriksaan kunci untuk semua kunci untuk memvalidasi pemuatan kunci yang benar.

Tujuan Kontrol 5: Kunci digunakan dengan cara yang mencegah atau mendeteksi penggunaannya yang tidak sah.

Persyaratan 17: Layanan ini menyediakan mekanisme, seperti tag dan alias, untuk kunci yang memungkinkan pelacakan hubungan berbagi kunci. Selain itu, nilai pemeriksaan kunci harus disimpan secara terpisah untuk menunjukkan bahwa nilai kunci yang diketahui atau default tidak digunakan saat kunci dibagikan.

Persyaratan 18: Layanan ini menyediakan pemeriksaan integritas utama, melalui GetKeydan ListKeys, dan peristiwa manajemen utama, melalui AWS CloudTrail, yang dapat digunakan untuk mendeteksi substitusi yang tidak sah atau memantau sinkronisasi kunci antar pihak. Layanan menyimpan kunci secara eksklusif di blok kunci. Anda bertanggung jawab atas penyimpanan kunci dan penggunaan sebelum mengimpor ke dan setelah ekspor dari Kriptografi AWS Pembayaran.

Anda harus memiliki prosedur untuk penyelidikan segera jika terjadi perbedaan selama pemrosesan transaksi berbasis PIN atau peristiwa manajemen kunci yang tidak terduga.

Persyaratan 19: Layanan menggunakan kunci secara eksklusif di blok kunci, penegakan KeyUsage KeyModeOfUse, dan atribut kunci lainnya untuk semua operasi. Ini termasuk pembatasan operasi kunci pribadi. Anda harus menggunakan kunci publik untuk satu tujuan e:g enkripsi atau verifikasi tanda tangan digital tetapi tidak keduanya. Anda harus menggunakan akun terpisah untuk produksi dan test/development sistem.

Persyaratan 20: Anda tetap bertanggung jawab atas persyaratan ini.

Tujuan Kontrol 6: Kunci dikelola dengan cara yang aman.

Persyaratan 21: Penyimpanan kunci dan penggunaan dengan Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Untuk persyaratan penyimpanan terkait komponen utama, Anda bertanggung jawab untuk menyimpannya seperti yang digambarkan di bawah 21-2 dan 21-3. Anda perlu menjelaskan mekanisme perlindungan utama dalam dokumentasi kebijakan Anda sebelum mengimpor ke dan setelah ekspor dari layanan.

Persyaratan 22: Prosedur kompromi utama untuk Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Anda perlu menjelaskan prosedur deteksi dan respons kompromi utama, termasuk pemantauan dan respons terhadap pemberitahuan dari AWS.

Persyaratan 23: Kriptografi AWS Pembayaran tidak mendukung varian atau metode perhitungan kunci reversibel lainnya. Kunci atau kunci utama APC yang dienkripsi oleh mereka tidak pernah tersedia untuk pelanggan. Penggunaan perhitungan kunci reversibel dinilai sebagai bagian dari penilaian PIN PCI layanan.

Persyaratan 24: Praktik penghancuran untuk rahasia internal dan kunci pribadi Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Anda perlu menjelaskan prosedur penghancuran kunci untuk kunci sebelum mengimpor ke dan setelah ekspor dari APC. Persyaratan penghancuran terkait komponen utama (24-2.2 dan 24-2.3) tetap menjadi tanggung jawab Anda.

Persyaratan 25: Akses ke kunci rahasia dan pribadi dalam Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Anda harus memiliki proses dan dokumentasi untuk kontrol akses kunci sebelum mengimpor ke dan setelah ekspor dari Kriptografi AWS Pembayaran.

Persyaratan 26: Anda perlu menjelaskan pencatatan untuk setiap akses ke kunci, komponen utama, atau materi terkait yang digunakan di luar layanan. Log untuk semua aktivitas manajemen utama yang dilakukan aplikasi Anda dengan layanan tersedia melalui AWS CloudTrail.

Persyaratan 27: Anda perlu menjelaskan prosedur cadangan untuk kunci, komponen utama, atau materi terkait yang digunakan di luar layanan.

Persyaratan 28: Prosedur untuk semua administrasi kunci yang menggunakan API harus mencakup penggunaan peran dengan izin administrasi kunci dan persetujuan untuk menjalankan skrip atau kode lain yang mengelola kunci. AWS CloudTrail log berisi semua acara administrasi utama

Tujuan Kontrol 7: Peralatan yang digunakan untuk memproses PINs dan kunci dikelola dengan cara yang aman.

Persyaratan 29: Persyaratan Anda untuk perlindungan fisik dan logis HSMs dipenuhi dengan menggunakan Kriptografi AWS Pembayaran.

Persyaratan 30: Aplikasi Anda akan bertanggung jawab atas semua perlindungan fisik dan logis dari persyaratan perangkat POI.

Persyaratan 31: Perlindungan perangkat kriptografi aman (SCD) yang digunakan oleh Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Anda perlu menunjukkan perlindungan dari yang lain yang SCDs digunakan oleh aplikasi Anda.

Persyaratan 32: Penggunaan yang SCDs digunakan oleh Kriptografi AWS Pembayaran dinilai sebagai bagian dari penilaian PIN PCI layanan. Anda perlu menunjukkan kontrol akses dan perlindungan dari yang lain yang SCDs digunakan oleh aplikasi Anda.

Persyaratan 33: Anda perlu menjelaskan perlindungan peralatan pemrosesan PIN apa pun di bawah kendali Anda.

Menggunakan Komponen Dekripsi Kriptografi AWS Pembayaran dalam solusi P2PE

Solusi PCI P2PE dapat menggunakan Komponen Dekripsi Kriptografi AWS Pembayaran. Hal ini didokumentasikan dalam Point-to-Point Enkripsi PCI: Persyaratan Keamanan dan Prosedur Pengujian, Bagian Solusi P2PE dan Penggunaan Penyedia Komponen and/or P2PE Pihak Ketiga: “Penyedia solusi (atau pedagang sebagai penyedia solusi) dapat melakukan outsourcing fungsi P2PE tertentu ke penyedia komponen P2PE yang terdaftar di PCI dan melaporkan penggunaan komponen P2PE yang terdaftar di PCI dalam Laporan P2PE tentang Validasi (P-ROV) mereka”, yang tersedia di situs web PCI.

Seperti layanan AWS lainnya dan standar kepatuhan, Anda bertanggung jawab untuk menggunakan layanan dengan aman, mengonfigurasi kontrol akses, dan menggunakan parameter keamanan sesuai dengan persyaratan PCI P2PE. Panduan Pengguna Komponen Dekripsi P2PE Kriptografi Pembayaran AWS, yang tersedia di AWS Artifact, memiliki petunjuk terperinci untuk mengintegrasikan Kriptografi AWS Pembayaran dengan Solusi PCI P2PE Anda dan laporan komponen dekripsi tahunan, yang diperlukan untuk pelaporan kepatuhan.

Manajemen identitas dan akses untuk Kriptografi AWS Pembayaran

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan AWS sumber daya Kriptografi Pembayaran. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda berdasarkan peran Anda:

- Pengguna layanan - minta izin dari administrator Anda jika Anda tidak dapat mengakses fitur (lihat Pemecahan Masalah Identitas dan AWS akses Kriptografi Pembayaran)
- Administrator layanan - tentukan akses pengguna dan mengirimkan permintaan izin (lihat Bagaimana Kriptografi AWS Pembayaran bekerja dengan IAM)
- Administrator IAM - tulis kebijakan untuk mengelola akses (lihat AWS Contoh kebijakan berbasis identitas Kriptografi Pembayaran)

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensyal identitas Anda. Anda harus diautentikasi sebagai Pengguna root akun AWS, pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk sebagai identitas federasi menggunakan kredensyal dari sumber identitas seperti AWS IAM Identity Center (Pusat Identitas IAM), autentikasi masuk tunggal, atau kredensyal. Google/ Facebook Untuk informasi selengkapnya tentang cara masuk, lihat Cara masuk ke Akun AWS Anda dalam Panduan Pengguna AWS Sign-In .

Untuk akses terprogram, AWS sediakan SDK dan CLI untuk menandatangani permintaan secara kriptografis. Untuk informasi selengkapnya, lihat AWS Signature Version 4 untuk permintaan API dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang disebut pengguna Akun AWS root yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Untuk tugas yang memerlukan kredensial pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Pengguna dan grup IAM

Pengguna IAM adalah identitas dengan izin khusus untuk satu orang atau aplikasi. Sebaiknya gunakan kredensial sementara alih-alih pengguna IAM dengan kredensial jangka panjang. Untuk informasi selengkapnya, lihat Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensyal sementara di Panduan Pengguna IAM.

Grup IAM menentukan kumpulan pengguna IAM dan mempermudah pengelolaan izin untuk pengguna dalam jumlah besar. Untuk mempelajari selengkapnya, lihat Kasus penggunaan untuk pengguna IAM dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dengan izin khusus yang menyediakan kredensial sementara. Anda dapat mengambil peran dengan beralih dari pengguna ke peran IAM (konsol) atau dengan

memanggil operasi AWS CLI atau AWS API. Untuk informasi selengkapnya, lihat Metode untuk mengambil peran dalam Panduan Pengguna IAM.

Peran IAM berguna untuk akses pengguna terfederasi, izin pengguna IAM sementara, akses lintas akun, akses lintas layanan, dan aplikasi yang berjalan di Amazon EC2. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan menentukan izin saat dikaitkan dengan identitas atau sumber daya. AWS mengevaluasi kebijakan ini ketika kepala sekolah membuat permintaan. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Menggunakan kebijakan, administrator menentukan siapa yang memiliki akses ke apa dengan mendefinisikan principal mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Administrator IAM membuat kebijakan IAM dan menambahkannya ke peran, yang kemudian dapat diambil oleh pengguna. Kebijakan IAM mendefinisikan izin terlepas dari metode yang Anda gunakan untuk melakukan operasinya.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang Anda lampirkan ke identitas (pengguna, grup, atau peran). Kebijakan ini mengontrol tindakan apa yang bisa dilakukan oleh identitas tersebut, terhadap sumber daya yang mana, dan dalam kondisi apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Tentukan izin IAM kustom dengan kebijakan yang dikelola pelanggan dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat berupa kebijakan inline (disematkan langsung ke dalam satu identitas) atau kebijakan terkelola (kebijakan mandiri yang dilampirkan pada banyak identitas). Untuk mempelajari cara memilih antara kebijakan terkelola dan kebijakan inline, lihat Pilih antara kebijakan terkelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contohnya termasuk kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3.

Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Anda harus menentukan principal dalam kebijakan berbasis sumber daya.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung. ACLs Untuk mempelajari selengkapnya ACLs, lihat Ringkasan daftar kontrol akses (ACL) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang dapat menetapkan izin maksimum yang diberikan oleh jenis kebijakan yang lebih umum:

- Batasan izin – Menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM. Untuk informasi selengkapnya, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCPs) — Tentukan izin maksimum untuk organisasi atau unit organisasi di AWS Organizations. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam Panduan Pengguna AWS Organizations .
- Kebijakan kontrol sumber daya (RCPs) — Tetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda. Untuk informasi selengkapnya, lihat Kebijakan kontrol sumber daya (RCPs) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan lanjutan yang diteruskan sebagai parameter saat membuat sesi sementara untuk peran atau pengguna terfederasi. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana Kriptografi AWS Pembayaran bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Kriptografi AWS Pembayaran, Anda harus memahami fitur IAM apa yang tersedia untuk digunakan dengan AWS Kriptografi Pembayaran. Untuk mendapatkan pandangan tingkat tinggi tentang bagaimana Kriptografi AWS Pembayaran dan AWS layanan lainnya bekerja dengan IAM, lihat AWS Layanan yang Bekerja dengan IAM di Panduan Pengguna IAM.

Topik

- AWS Kebijakan berbasis identitas Kriptografi Pembayaran
- Otorisasi berdasarkan tag Kriptografi AWS Pembayaran

AWS Kebijakan berbasis identitas Kriptografi Pembayaran

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak serta kondisi di mana tindakan diizinkan atau ditolak. AWS Kriptografi Pembayaran mendukung tindakan, sumber daya, dan kunci kondisi tertentu. Untuk mempelajari semua elemen yang Anda gunakan dalam kebijakan JSON, lihat Referensi Elemen Kebijakan JSON IAM dalam Panduan Pengguna IAM.

Tindakan

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Tindakan kebijakan dalam Kriptografi AWS Pembayaran menggunakan awalan berikut sebelum tindakan:. `payment-cryptography:` Misalnya, untuk memberikan izin kepada seseorang untuk

menjalankan operasi `VerifyCardData` API Kriptografi AWS Pembayaran, Anda menyertakan `payment-cryptography:VerifyCardData` tindakan tersebut dalam kebijakan mereka. Pernyataan kebijakan harus memuat elemen `Action` atau `NotAction`. AWS Kriptografi Pembayaran mendefinisikan serangkaian tindakannya sendiri yang menggambarkan tugas yang dapat Anda lakukan dengan layanan ini.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma seperti berikut:

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

Anda dapat menentukan beberapa tindakan menggunakan wildcard (*). Misalnya, untuk menentukan semua tindakan yang dimulai dengan kata `List` (seperti `ListKeys` dan`ListAliases`), sertakan tindakan berikut:

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

Untuk melihat daftar tindakan Kriptografi AWS Pembayaran, lihat Tindakan yang Ditentukan oleh Kriptografi AWS Pembayaran di Panduan Pengguna IAM.

Sumber daya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

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

Sumber daya kunci kriptografi pembayaran memiliki ARN berikut:

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

Untuk informasi selengkapnya tentang format ARNs, lihat Amazon Resource Names (ARNs) dan Ruang Nama AWS Layanan.

Misalnya, untuk menentukan instans `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` dalam pernyataan Anda, gunakan ARN berikut:

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

Untuk menentukan semua kunci milik akun tertentu, gunakan wildcard (*):

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

Beberapa tindakan Kriptografi AWS Pembayaran, seperti untuk membuat kunci, tidak dapat dilakukan pada sumber daya tertentu. Dalam kasus tersebut, Anda harus menggunakan wildcard (*).

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

Untuk menentukan beberapa sumber daya dalam satu pernyataan, gunakan koma seperti yang ditunjukkan di bawah ini:

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

Contoh

Untuk melihat contoh kebijakan berbasis identitas Kriptografi AWS Pembayaran, lihat. AWS Contoh kebijakan berbasis identitas Kriptografi Pembayaran

Otorisasi berdasarkan tag Kriptografi AWS Pembayaran

Anda dapat melampirkan tag ke sumber daya Kriptografi AWS Pembayaran atau meneruskan tag dalam permintaan ke Kriptografi AWS Pembayaran. Untuk mengendalikan akses berdasarkan

tanda, berikan informasi tentang tanda di elemen kondisi dari kebijakan menggunakan kunci kondisi `payment-cryptography:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

AWS Contoh kebijakan berbasis identitas Kriptografi Pembayaran

Secara default, pengguna dan peran IAM tidak memiliki izin untuk membuat atau memodifikasi sumber daya Kriptografi AWS Pembayaran. Mereka juga tidak dapat melakukan tugas menggunakan Konsol Manajemen AWS, AWS CLI, atau AWS API. Administrator IAM harus membuat kebijakan IAM yang memberikan izin kepada pengguna dan peran untuk melakukan operasi API tertentu pada sumber daya yang diperlukan. Administrator kemudian harus melampirkan kebijakan tersebut ke pengguna IAM atau grup yang memerlukan izin tersebut.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat Kebijakan pada Tab JSON dalam Panduan Pengguna IAM.

Topik

- Praktik terbaik kebijakan
- Menggunakan konsol Kriptografi AWS Pembayaran
- Izinkan para pengguna untuk melihat izin mereka sendiri
- Kemampuan untuk mengakses semua aspek Kriptografi AWS Pembayaran
- Kemampuan untuk memanggil APIs menggunakan kunci yang ditentukan
- Kemampuan untuk secara khusus menolak sumber daya

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Kriptografi AWS Pembayaran di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan

yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.

- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Menggunakan konsol Kriptografi AWS Pembayaran

Untuk mengakses konsol Kriptografi AWS Pembayaran, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya

Kriptografi AWS Pembayaran di akun Anda AWS . Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tersebut tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna IAM atau peran) dengan kebijakan tersebut.

Untuk memastikan bahwa entitas tersebut masih dapat menggunakan konsol Kriptografi AWS Pembayaran, lampirkan juga kebijakan AWS terkelola berikut ke entitas. Untuk informasi selengkapnya, lihat Menambahkan Izin ke Pengguna dalam Panduan Pengguna IAM.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai alternatif, hanya izinkan akses ke tindakan yang cocok dengan operasi API yang sedang Anda coba lakukan.

Izinkan para pengguna untuk melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI 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": "*"
        }
    ]
}
```

Kemampuan untuk mengakses semua aspek Kriptografi AWS Pembayaran

> **Warning**
>
> Contoh ini memberikan izin luas dan tidak disarankan. Pertimbangkan model akses yang paling tidak priviledged sebagai gantinya.

Dalam contoh ini, Anda ingin memberikan pengguna IAM di AWS akun Anda akses ke semua kunci Kriptografi AWS Pembayaran Anda dan kemampuan untuk memanggil semua API Kriptografi AWS Pembayaran termasuk keduanya `ControlPlane` dan operasi. `DataPlane`

JSON

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

Kemampuan untuk memanggil APIs menggunakan kunci yang ditentukan

Dalam contoh ini, Anda ingin memberikan pengguna IAM di AWS akun Anda akses ke salah satu kunci Kriptografi AWS Pembayaran Anda, `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` dan kemudian menggunakan sumber daya ini dalam dua APIs, `GenerateCardValidationData` dan. `VerifyCardValidationData` Sebaliknya, pengguna IAM tidak akan memiliki akses untuk menggunakan kunci ini pada operasi lain seperti atau `DeleteKey` `ExportKey`

Sumber daya dapat berupa kunci, diawali dengan `key` atau alias, diawali dengan. `alias`

JSON

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

Kemampuan untuk secara khusus menolak sumber daya

> **Warning**
>
> Pertimbangkan dengan cermat implikasi pemberian akses wildcard. Pertimbangkan model hak istimewa yang paling tidak.

Dalam contoh ini, Anda ingin mengizinkan pengguna IAM di AWS akun Anda mengakses salah satu kunci Kriptografi AWS Pembayaran Anda tetapi ingin menolak izin ke satu kunci tertentu. Pengguna akan memiliki akses ke `VerifyCardData` dan `GenerateCardData` dengan semua kunci dengan pengecualian yang ditentukan dalam pernyataan penolakan.

JSON

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

Pemecahan Masalah Identitas dan AWS akses Kriptografi Pembayaran

Topik akan ditambahkan ke bagian ini karena masalah terkait IAM yang khusus untuk Kriptografi AWS Pembayaran diidentifikasi. Untuk konten pemecahan masalah umum tentang topik IAM, lihat bagian pemecahan masalah pada Panduan Pengguna IAM.

Pemantauan Kriptografi AWS Pembayaran

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Kriptografi AWS Pembayaran dan solusi AWS Anda yang lain. AWS menyediakan alat pemantauan berikut untuk menonton Kriptografi AWS Pembayaran, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang ditentukan. Misalnya, Anda dapat CloudWatch melacak penggunaan tertentu APIs atau memberi tahu Anda jika Anda mendekati kuota Kriptografi AWS Pembayaran Anda. Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.
- Amazon CloudWatch Logs memungkinkan Anda memantau, menyimpan, dan mengakses file log Anda dari EC2 instans Amazon CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log dalam penyimpanan yang sangat durabel. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon CloudWatch Logs.
- AWS CloudTrailmenangkap panggilan API dan peristiwa terkait yang dibuat oleh atau atas nama AWS akun Anda dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang dipanggil AWS, titik akhir yang dipanggil, sumber daya (kunci) yang digunakan, alamat IP sumber dari mana panggilan dilakukan, dan kapan panggilan terjadi. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS CloudTrail.

Topik

- Logging panggilan API Kriptografi AWS Pembayaran menggunakan AWS CloudTrail

Logging panggilan API Kriptografi AWS Pembayaran menggunakan AWS CloudTrail

AWS Kriptografi Pembayaran terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan dalam Kriptografi AWS Pembayaran. CloudTrail menangkap semua panggilan API untuk Kriptografi AWS Pembayaran sebagai peristiwa. Panggilan yang direkam mencakup panggilan dari konsol dan panggilan kode ke

operasi API ini. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon S3, termasuk acara untuk Kriptografi AWS Pembayaran. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa manajemen (Control Plane) terbaru di CloudTrail konsol dalam riwayat Acara. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk Kriptografi AWS Pembayaran, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Topik

- AWS Informasi Kriptografi Pembayaran di CloudTrail
- Mengontrol peristiwa pesawat di CloudTrail
- Peristiwa data di CloudTrail
- Memahami Kriptografi AWS Pembayaran Kontrol Pesawat entri file log
- Memahami Kriptografi AWS Pembayaran Entri file log pesawat data

AWS Informasi Kriptografi Pembayaran di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi dalam Kriptografi AWS Pembayaran, aktivitas tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS . Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan peristiwa yang sedang berlangsung di AWS akun Anda, termasuk acara untuk Kriptografi AWS Pembayaran, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol tersebut, jejak diterapkan ke semua Wilayah AWS . Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran umum untuk membuat jejak
- CloudTrail layanan dan integrasi yang didukung
- Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail

- Menerima file CloudTrail log dari beberapa Wilayah
- Menerima file CloudTrail log dari beberapa akun

Setiap entri peristiwa atau log berisi informasi tentang entitas yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut ini:

- Apakah permintaan itu dibuat dengan kredenal pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna gabungan.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat Elemen userIdentity CloudTrail .

Mengontrol peristiwa pesawat di CloudTrail

CloudTrail mencatat operasi Kriptografi AWS Pembayaran, seperti CreateKey, ImportKey,, DeleteKey, ListKeys, TagResource, dan semua operasi pesawat kontrol lainnya.

Peristiwa data di CloudTrail

Peristiwa data memberikan informasi tentang operasi sumber daya yang dilakukan pada atau di sumber daya, seperti mengenkripsi muatan atau menerjemahkan pin. Peristiwa data adalah aktivitas volume tinggi yang CloudTrail tidak masuk secara default. Anda dapat mengaktifkan log tindakan API peristiwa data untuk peristiwa bidang data Kriptografi AWS Pembayaran dengan menggunakan CloudTrail APIs atau konsol. Untuk informasi selengkapnya, lihat Mencatat peristiwa data dalam AWS CloudTrail Panduan Pengguna.

Dengan CloudTrail, Anda harus menggunakan penyeleksi acara lanjutan untuk memutuskan aktivitas API Kriptografi AWS Pembayaran mana yang dicatat dan dicatat. Untuk mencatat peristiwa bidang data Kriptografi AWS Pembayaran, Anda harus menyertakan jenis sumber daya `AWS Payment Cryptography key` dan`AWS Payment Cryptography alias`. Setelah ini diatur, Anda dapat memperbaiki preferensi logging Anda lebih lanjut dengan memilih peristiwa data tertentu untuk direkam, seperti menggunakan `eventName` filter untuk melacak `EncryptData` peristiwa. Untuk informasi selengkapnya, lihat `AdvancedEventSelector` di dalam Referensi API AWS CloudTrail .

> **Note**
>
> Untuk berlangganan peristiwa data Kriptografi AWS Pembayaran, Anda harus menggunakan pemilih acara tingkat lanjut. Kami merekomendasikan berlangganan acara kunci dan alias untuk memastikan bahwa Anda menerima semua acara.

AWS Peristiwa data Kriptografi Pembayaran:

- `DecryptData`
- `EncryptData`
- `GenerateCardValidationData`
- `GenerateMac`
- `GeneratePinData`
- `ReEncryptData`
- `TranslatePinData`
- `VerifyAuthRequestCryptogram`
- `VerifyCardValidationData`
- `VerifyMac`
- `VerifyPinData`

Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya, silakan lihat Harga AWS CloudTrail .

Memahami Kriptografi AWS Pembayaran Kontrol Pesawat entri file log

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan tindakan Kriptografi AWS `CreateKey` Pembayaran.

```
      {
    CloudTrailEvent: {
      tlsDetails= {
        TlsDetails: {
          cipherSuite=TLS_AES_128_GCM_SHA256,
          tlsVersion=TLSv1.3,
          clientProvidedHostHeader=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/
ControlPlane-IntegTest-68211a2a-3e9d-42b7-86ac-c682520e0410,
            invokedBy=null,
            accessKeyId=TESTXECZ5U2ZULLHHMJG,
            type=AssumedRole,
            sessionContext={
              SessionContext: {
                sessionIssuer={
                  SessionIssuer: {arn=arn:aws:iam::111122223333:role/TestAssumeRole-us-
west-2,
                  type=Role,
                  accountId=111122223333,
                  userName=TestAssumeRole-us-west-2,
                  principalId=TESTXECZ5U9M4LGF2N6Y5}
                },
                attributes={
                  SessionContextAttributes: {
                    creationDate=Sun May 21 18:58:31 UTC 2023,
                    mfaAuthenticated=false
                  }
```

```
                },
                webIdFederationData=null
              }
            },
          username=null,
          principalId=TESTXECZ5U9M4LGF2N6Y5: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={
    }
  }
}
```

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan Kriptografi AWS Pembayaran yang memungkinkan replikasi kunci Multi-Region.

```
{
    "eventVersion": "1.11",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "eventTime": "2025-08-15T17:50:41Z",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventName": "SynchronizeMultiRegionKey",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "payment-cryptography.amazonaws.com",
    "userAgent": "payment-cryptography.amazonaws.com",
    "requestParameters": null,
    "responseElements": null,
```

```
    "eventID": "55c0fcbc-5b2e-4bd2-a976-99305be6e6fc",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "keyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/key-id",
        "replicationRegion": "us-east-2"
    },
    "eventCategory": "Management"
}
```

Memahami Kriptografi AWS Pembayaran Entri file log pesawat data

Peristiwa bidang data secara opsional dapat dikonfigurasi dan berfungsi mirip dengan log bidang kontrol tetapi biasanya volumenya jauh lebih tinggi. Mengingat sifat sensitif dari beberapa input dan output untuk operasi pesawat data Kriptografi AWS Pembayaran, Anda mungkin menemukan bidang tertentu dengan pesan "*** Data Sensitif Diedit ***". Ini tidak dapat dikonfigurasi dan dimaksudkan untuk mencegah data sensitif muncul di log atau jejak.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan tindakan Kriptografi AWS `EncryptData` Pembayaran.

```
      {
        "Records": [
            {
                "eventVersion": "1.09",
                "userIdentity": {
                    "type": "AssumedRole",
                    "principalId": "TESTXECZ5U2ZULLHHMJG:DataPlane-User",
                    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/DataPlane-
User",
                    "accountId": "111122223333",
                    "accessKeyId": "TESTXECZ5U2ZULLHHMJG",
                    "userName": "",
                    "sessionContext": {
                        "sessionIssuer": {
                            "type": "Role",
                            "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                            "arn": "arn:aws:iam::111122223333:role/Admin",
                            "accountId": "111122223333",
```

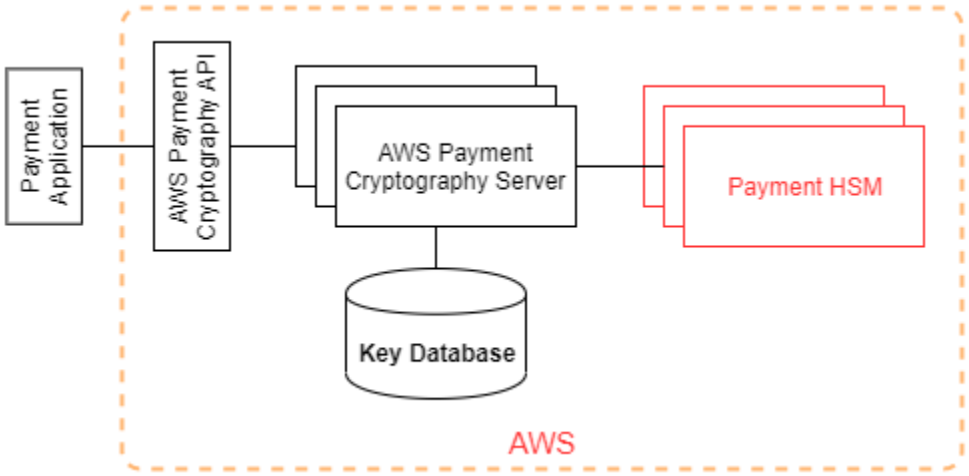
```
                        "userName": "Admin"
                    },
                    "attributes": {
                        "creationDate": "2024-07-09T14:23:05Z",
                        "mfaAuthenticated": "false"
                    }
                }
            },
            "eventTime": "2024-07-09T14:24:02Z",
            "eventSource": "payment-cryptography.amazonaws.com",
            "eventName": "GenerateCardValidationData",
            "awsRegion": "us-east-2",
            "sourceIPAddress": "192.158.1.38",
            "userAgent": "aws-cli/2.17.6 md/awscrt#0.20.11 ua/2.0 os/macos#23.4.0
 md/arch#x86_64 lang/python#3.11.8 md/pyimpl#CPython cfg/retry-mode#standard md/
installer#exe md/prompt#off md/command#payment-cryptography-data.generate-card-
validation-data",
            "requestParameters": {
                "key_identifier": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp",
                "primary_account_number": "*** Sensitive Data Redacted ***",
                "generation_attributes": {
                    "CardVerificationValue2": {
                        "card_expiry_date": "*** Sensitive Data Redacted ***"
                    }
                }
            },
            "responseElements": null,
            "requestID": "f2a99da8-91e2-47a9-b9d2-1706e733991e",
            "eventID": "e4eb3785-ac6a-4589-97a1-babdd3d4dd95",
            "readOnly": true,
            "resources": [
                {
                    "accountId": "111122223333",
                    "type": "AWS::PaymentCryptography::Key",
                    "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp"
                }
            ],
            "eventType": "AwsApiCall",
            "managementEvent": false,
            "recipientAccountId": "111122223333",
            "eventCategory": "Data",
            "tlsDetails": {
```

```
                    "tlsVersion": "TLSv1.3",
                    "cipherSuite": "TLS_AES_128_GCM_SHA256",
                    "clientProvidedHostHeader": "dataplane.payment-cryptography.us-
east-2.amazonaws.com"
                }
            }
        ]
    }
```

Detail kriptografi

AWS Kriptografi Pembayaran menyediakan antarmuka web untuk menghasilkan dan mengelola kunci kriptografi untuk transaksi pembayaran. AWS Kriptografi Pembayaran menawarkan layanan manajemen kunci standar dan kriptografi transaksi pembayaran dan alat yang dapat Anda gunakan untuk manajemen dan audit terpusat. Dokumentasi ini memberikan penjelasan rinci tentang operasi kriptografi yang dapat Anda gunakan dalam Kriptografi AWS Pembayaran untuk membantu Anda dalam mengevaluasi fitur yang ditawarkan oleh layanan.

AWS Kriptografi Pembayaran berisi beberapa antarmuka (termasuk RESTful API, melalui AWS CLI, AWS SDK, dan) untuk meminta operasi kriptografi armada Konsol Manajemen AWS terdistribusi modul keamanan perangkat keras yang divalidasi HSM PCI PTS.



AWS Kriptografi Pembayaran adalah layanan berjenjang yang terdiri dari host Kriptografi AWS Pembayaran yang menghadap ke web dan tingkat. HSMs Pengelompokan host berjenjang ini membentuk tumpukan Kriptografi AWS Pembayaran. Semua permintaan untuk Kriptografi AWS Pembayaran harus dilakukan melalui protokol Transport Layer Security (TLS) dan berakhir pada host Kriptografi AWS Pembayaran. Host layanan hanya mengizinkan TLS dengan cipher suite yang memberikan kerahasiaan maju yang sempurna. Layanan mengautentikasi dan mengotorisasi permintaan Anda menggunakan kredensyal dan mekanisme kebijakan IAM yang sama yang tersedia untuk semua operasi API lainnya. AWS

AWS Server Kriptografi Pembayaran terhubung ke HSM yang mendasarinya melalui jaringan pribadi, non-virtual. Koneksi antara komponen layanan dan HSM diamankan dengan TLS bersama (MTL) untuk otentikasi dan enkripsi.

Topik

Tujuan desain

AWS Kriptografi Pembayaran dirancang untuk memenuhi persyaratan berikut:

- Terpercaya — Penggunaan kunci dilindungi oleh kebijakan kontrol akses yang Anda tentukan dan kelola. Tidak ada mekanisme untuk mengekspor kunci Kriptografi AWS Pembayaran teks biasa. Kerahasiaan kunci kriptografi Anda sangat penting. Beberapa karyawan Amazon dengan akses khusus peran ke kontrol akses berbasis kuorum diperlukan untuk melakukan tindakan administratif pada. HSMs Tidak ada karyawan Amazon yang memiliki akses ke kunci utama (atau master) atau cadangan HSM. Kunci utama tidak dapat disinkronkan dengan HSMs yang bukan bagian dari wilayah Kriptografi AWS Pembayaran. Semua kunci lainnya dilindungi oleh kunci utama HSM. Oleh karena itu, kunci Kriptografi AWS Pembayaran pelanggan tidak dapat digunakan di luar layanan Kriptografi AWS Pembayaran yang beroperasi di dalam akun pelanggan.
- Latensi rendah dan throughput tinggi — Kriptografi AWS Pembayaran menyediakan operasi kriptografi pada tingkat latensi dan throughput yang sesuai untuk mengelola kunci kriptografi pembayaran dan memproses transaksi pembayaran.
- Daya tahan — Daya tahan kunci kriptografi dirancang agar sama dengan layanan dengan daya tahan tertinggi di AWS. Kunci kriptografi tunggal dapat dibagikan dengan terminal pembayaran, kartu chip EMV, atau perangkat kriptografi aman lainnya (SCD) yang digunakan selama bertahun-tahun.
- Wilayah Independen — AWS menyediakan wilayah independen bagi pelanggan yang perlu membatasi akses data di berbagai wilayah atau perlu mematuhi persyaratan residensi data. Penggunaan kunci dapat diisolasi dalam Wilayah AWS.
- Sumber angka acak yang aman — Karena kriptografi yang kuat bergantung pada generasi angka acak yang benar-benar tidak dapat diprediksi, Kriptografi AWS Pembayaran menyediakan sumber angka acak berkualitas tinggi dan tervalidasi. Semua generasi kunci untuk Kriptografi AWS Pembayaran menggunakan HSM yang terdaftar di PCI PTS HSM, beroperasi dalam mode PCI.
- Audit — Kriptografi AWS Pembayaran mencatat penggunaan dan pengelolaan kunci kriptografi dalam CloudTrail log dan log layanan yang tersedia melalui Amazon. CloudWatch Anda dapat menggunakan CloudTrail log untuk memeriksa penggunaan kunci kriptografi Anda, termasuk

penggunaan kunci oleh akun yang telah Anda bagikan kunci dengan. AWS Kriptografi Pembayaran diaudit oleh penilai pihak ketiga terhadap PCI, merek kartu, dan standar keamanan pembayaran regional yang berlaku. Panduan pengesahan dan Tanggung Jawab Bersama tersedia di Artifact AWS.

- Elastis — Kriptografi AWS Pembayaran berskala dan sesuai dengan permintaan Anda. Alih-alih memprediksi dan memesan kapasitas HSM, Kriptografi Pembayaran menyediakan kriptografi AWS pembayaran sesuai permintaan. AWS Kriptografi Pembayaran bertanggung jawab untuk menjaga keamanan dan kepatuhan HSM untuk menyediakan kapasitas yang cukup untuk memenuhi permintaan puncak pelanggan.

Fondasi

Topik dalam Bab ini menjelaskan primitif kriptografi Kriptografi AWS Pembayaran dan di mana mereka digunakan. Mereka juga memperkenalkan elemen dasar layanan.

Topik

Primitif kriptografi

AWS Kriptografi Pembayaran menggunakan algoritma kriptografi standar yang dapat parameter sehingga aplikasi dapat mengimplementasikan algoritma yang diperlukan untuk kasus penggunaannya. Himpunan algoritma kriptografi ditentukan oleh standar PCI, ANSI X9 EMVco, dan ISO. Semua kriptografi dilakukan oleh PCI PTS HSM yang terdaftar standar berjalan dalam mode HSMs PCI.

Entropi dan pembangkitan bilangan acak

AWS Pembangkitan kunci Kriptografi Pembayaran dilakukan pada Kriptografi AWS HSMs Pembayaran. HSMs Mengimplementasikan generator angka acak yang memenuhi persyaratan PCI PTS HSM untuk semua jenis dan parameter kunci yang didukung.

Operasi kunci simetris

Algoritma kunci simetris dan kekuatan kunci yang ditentukan dalam ANSI X9 TR 31, ANSI X9.24, dan PCI PIN Annex C didukung:

- Fungsi hash — Algoritma dari SHA2 dan SHA3 keluarga dengan ukuran output lebih besar dari 2551. Kecuali untuk kompatibilitas mundur dengan terminal pra-PCI PTS POI v3.
- Enkripsi dan dekripsi — AES dengan ukuran kunci lebih besar dari atau sama dengan 128 bit, atau TDEA dengan ukuran kunci lebih besar dari atau sama dengan 112 bit (2 kunci atau 3 kunci).
- Kode Otentikasi Pesan (MACs) CMAC atau GMAC dengan AES, serta HMAC dengan fungsi hash yang disetujui dan ukuran kunci lebih besar dari atau sama dengan 128.

AWS Kriptografi Pembayaran menggunakan AES 256 untuk kunci utama HSM, kunci perlindungan data, dan kunci sesi TLS.

Catatan: Beberapa fungsi yang terdaftar digunakan secara internal untuk mendukung protokol standar dan struktur data. Lihat dokumentasi API untuk algoritme yang didukung oleh tindakan tertentu.

Operasi kunci asimetris

Algoritma kunci asimetris dan kekuatan kunci yang ditentukan dalam ANSI X9 TR 31, ANSI X9.24, dan PCI PIN Annex C didukung:

- Skema pendirian kunci yang disetujui - seperti yang dijelaskan dalam NIST SP8 00-56A (). ECC/ FCC2-based key agreement), NIST SP800-56B (IFC-based key agreement), and NIST SP800-38F (AES-based key encryption/wrapping

AWS Host Payment Cryptography hanya mengizinkan koneksi ke layanan menggunakan TLS dengan cipher suite yang memberikan kerahasiaan penerusan yang sempurna.

Catatan: Beberapa fungsi yang terdaftar digunakan secara internal untuk mendukung protokol standar dan struktur data. Lihat dokumentasi API untuk algoritme yang didukung oleh tindakan tertentu.

Penyimpanan kunci

AWS Kunci Kriptografi Pembayaran dilindungi oleh kunci utama HSM AES 256 dan disimpan dalam blok kunci ANSI X9 TR 31 dalam database terenkripsi. Basis data direplikasi ke database dalam memori pada server Kriptografi AWS Pembayaran.

Menurut PCI PIN Security Normative Annex C, kunci AES 256 sama kuatnya dengan atau lebih kuat dari:

- TDEA 3-kunci
- RSA 15360 bit
- ECC 512 bit
- DSA, DH, dan MQV 15360/512

Impor kunci menggunakan tombol simetris

AWS Kriptografi Pembayaran mendukung impor kriptogram dan blok kunci dengan kunci simetris atau publik dengan kunci enkripsi kunci simetris (KEK) yang kuat atau lebih kuat dari kunci yang dilindungi untuk impor.

Impor kunci menggunakan tombol asimetris

AWS Kriptografi Pembayaran mendukung impor kriptogram dan blok kunci dengan kunci simetris atau publik yang dilindungi oleh kunci enkripsi kunci pribadi (KEK) yang sekuat atau lebih kuat dari kunci yang dilindungi untuk impor. Kunci publik yang disediakan untuk dekripsi harus memiliki keaslian dan integritasnya yang dijamin oleh sertifikat dari otoritas yang dipercaya oleh pelanggan.

KEK Publik yang disediakan oleh Kriptografi AWS Pembayaran memiliki otentikasi dan perlindungan integritas otoritas sertifikat (CA) dengan kepatuhan yang terbukti terhadap Keamanan PIN PCI dan Lampiran PCI P2PE A.

Ekspor kunci

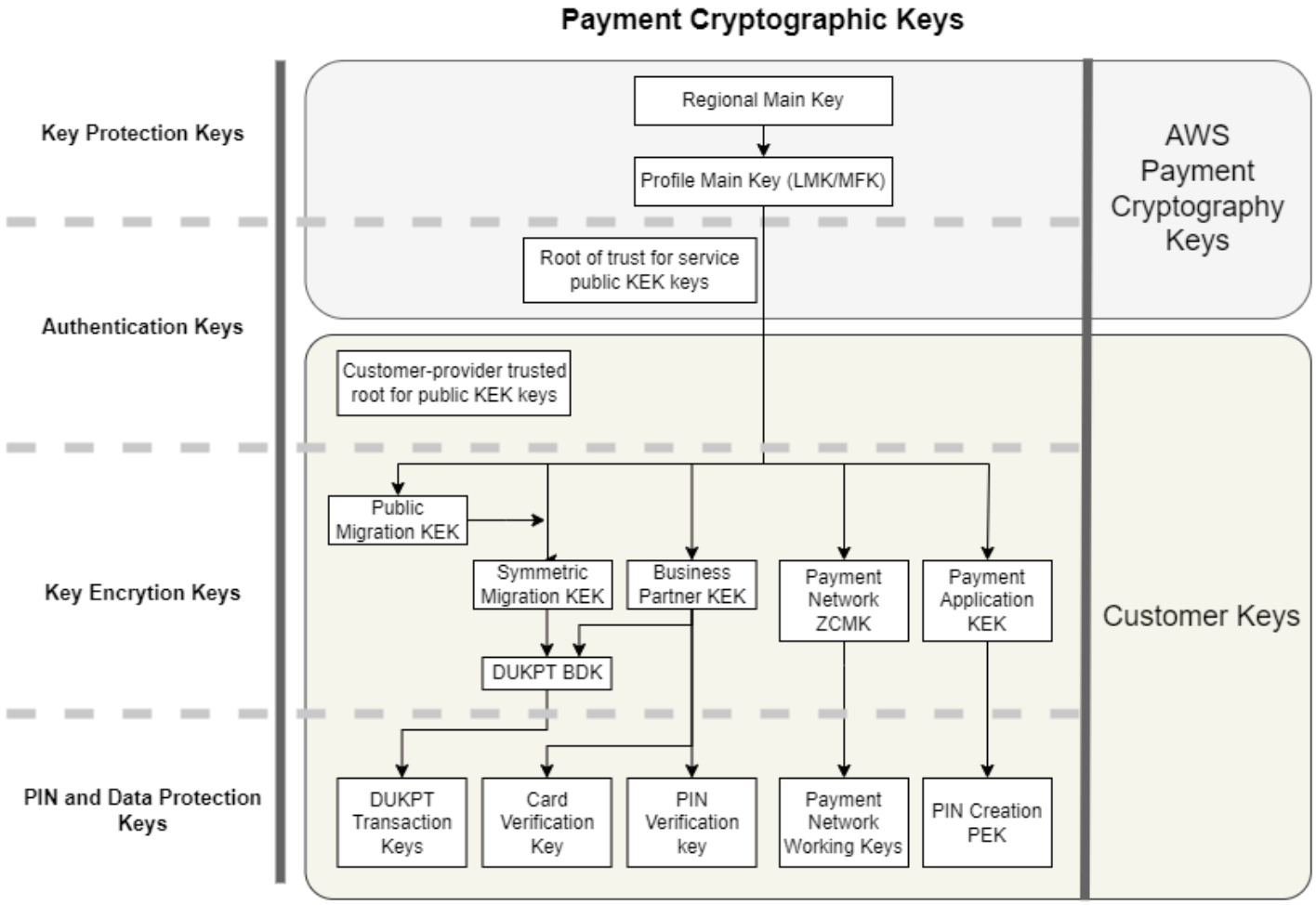
Kunci dapat diekspor dan dilindungi oleh kunci dengan yang sesuai KeyUsage dan yang sekuat atau lebih kuat dari kunci yang akan diekspor.

Protokol Kunci Per Transaksi Unik Berasal (DUKPT)

AWS Kriptografi Pembayaran mendukung dengan kunci derivasi dasar TDEA dan AES (BDK) seperti yang dijelaskan oleh ANSI X9.24-3.

Hirarki kunci

Hirarki kunci Kriptografi AWS Pembayaran memastikan bahwa kunci selalu dilindungi oleh kunci sekuat atau lebih kuat dari kunci yang mereka lindungi.



AWS Kunci Kriptografi Pembayaran digunakan untuk perlindungan kunci dalam layanan:

Key	Deskripsi
Kunci Utama Regional	Melindungi gambar HSM virtual, atau profil, yang digunakan untuk pemrosesan kriptografi. Kunci ini hanya ada di HSM dan backup aman.
Profil Kunci Utama	Kunci perlindungan kunci pelanggan tingkat atas, secara tradisional disebut Local Master Key (LMK) atau Master File Key (MFK) untuk kunci pelanggan. Kunci ini hanya ada di HSM dan backup aman. Profil mendefinisikan konfigurasi HSM yang berbeda seperti yang dipersyaratkan oleh standar keamanan untuk kasus penggunaan pembayaran.
Akar kepercayaan untuk kunci kunci enkripsi kunci publik (KEK) Kriptografi AWS Pembayaran	Kunci publik root tepercaya dan sertifikat untuk mengautentikasi dan memvalidasi kunci publik yang disediakan oleh Kriptografi AWS Pembayaran untuk impor dan ekspor kunci menggunakan kunci asimetris.

Kunci pelanggan dikelompokkan berdasarkan kunci yang digunakan untuk melindungi kunci dan kunci lain yang melindungi data terkait pembayaran. Ini adalah contoh kunci pelanggan dari kedua jenis:

Key	Deskripsi
Root tepercaya yang disediakan pelanggan untuk kunci KEK publik	Kunci publik dan sertifikat yang diberikan oleh Anda sebagai akar kepercayaan untuk mengautentikasi dan memvalidasi kunci publik yang Anda berikan untuk impor dan ekspor kunci menggunakan kunci asimetris.
Kunci Enkripsi Kunci (KEK)	KEK digunakan semata-mata untuk mengenkripsi kunci lain untuk pertukaran antara toko kunci eksternal dan Kriptografi AWS

Key	Deskripsi
	Pembayaran, mitra bisnis, jaringan pembayaran, atau aplikasi lain dalam organisasi Anda.
Kunci turunan Unik Per Transaksi (DUKPT) kunci derivasi dasar (BDK)	BDKs digunakan untuk membuat kunci unik untuk setiap terminal pembayaran dan menerjemahkan transaksi dari beberapa terminal ke bank pengakuisisi tunggal, atau pengakuisisi, kunci kerja. Praktik terbaik, yang diperlukan oleh Point-to-Point Enkripsi PCI (P2PE), adalah bahwa berbeda BDKs digunakan untuk model terminal yang berbeda, layanan injeksi atau inisialisasi kunci, atau segmentasi lain untuk membatasi dampak kompromi BDK.
Kunci master kontrol zona jaringan pembayaran (ZCMK)	ZCMK, juga disebut sebagai kunci zona atau kunci master zona, disediakan oleh jaringan pembayaran untuk membuat kunci kerja awal.
Kunci transaksi DUKPT	Terminal pembayaran yang dikonfigurasi untuk DUKPT memperoleh kunci unik untuk terminal dan transaksi. HSM yang menerima transaksi dapat menentukan kunci dari pengenal terminal dan nomor urut transaksi.
Kunci persiapan data kartu	Kunci master penerbit EMV, kunci kartu EMV dan nilai verifikasi, dan kunci perlindungan file data personalisasi kartu digunakan untuk membuat data untuk masing-masing kartu untuk digunakan oleh penyedia personalisasi kartu. Kunci dan data validasi kriptografi ini juga digunakan oleh bank penerbit, atau penerbit, untuk mengautentikasi data kartu sebagai bagian dari otorisasi transaksi.

Key	Deskripsi
Kunci persiapan data kartu	Kunci master penerbit EMV, kunci kartu EMV dan nilai verifikasi, dan kunci perlindungan file data personalisasi kartu digunakan untuk membuat data untuk masing-masing kartu untuk digunakan oleh penyedia personalisasi kartu. Kunci dan data validasi kriptografi ini juga digunakan oleh bank penerbit, atau penerbit, untuk mengautentikasi data kartu sebagai bagian dari otorisasi transaksi.
Kunci kerja jaringan pembayaran	Sering disebut sebagai kunci kerja penerbit atau kunci kerja pengakuisisi, ini adalah kunci yang mengenkripsi transaksi yang dikirim ke atau diterima dari jaringan pembayaran. Kunci-kunci ini sering diputar oleh jaringan, sering setiap hari atau setiap jam. Ini adalah kunci enkripsi PIN (PEK) untuk PIN/Debit transaksi.
Kunci enkripsi Nomor Identifikasi Pribadi (PIN) (PEK)	Aplikasi yang membuat atau mendekripsi blok PIN menggunakan PEK untuk mencegah penyimpanan atau transmisi PIN teks yang jelas.

Operasi internal

Topik ini menjelaskan persyaratan internal yang diterapkan oleh layanan untuk mengamankan kunci pelanggan dan operasi kriptografi untuk kriptografi pembayaran yang didistribusikan secara global dan terukur dan layanan manajemen kunci.

Topik

- Pencatatan log dan pemantauan

Perlindungan HSM

Spesifikasi dan siklus hidup HSM

AWS Kriptografi Pembayaran menggunakan armada yang tersedia secara komersial. HSMs FIPS 140-2 Level 3 divalidasi dan juga menggunakan versi firmware dan kebijakan keamanan yang tercantum pada daftar Perangkat PCI PCI PTS yang disetujui Dewan Standar Keamanan PCI sebagai sesuai dengan PCI HSM v3. HSMs Standar PCI PTS HSM mencakup persyaratan tambahan untuk pembuatan, pengiriman, penyebaran, manajemen, dan penghancuran perangkat keras HSM yang penting untuk keamanan dan kepatuhan pembayaran tetapi tidak ditangani oleh FIPS 140.

Asesor pihak ketiga memverifikasi model pembuatan HSM, firmware, konfigurasi, manajemen fisik siklus hidup, kontrol perubahan, kontrol akses operator, manajemen kunci utama, dan semua persyaratan PIN dan P2PE PCI yang terkait dengan dan operasi HSM. HSMs

Semua HSMs dioperasikan dalam Mode PCI dan dikonfigurasi dengan kebijakan keamanan PCI PTS HSM. Hanya fungsi yang diperlukan untuk mendukung kasus penggunaan Kriptografi AWS Pembayaran yang diaktifkan. AWS Kriptografi Pembayaran tidak menyediakan pencetakan, tampilan, atau pengembalian teks PINs yang jelas.

Keamanan fisik perangkat HSM

Hanya HSMs yang memiliki kunci perangkat yang ditandatangani oleh otoritas sertifikat Kriptografi AWS Pembayaran (CA) oleh produsen sebelum pengiriman dapat digunakan oleh layanan. Kriptografi AWS Pembayaran adalah sub-CA dari CA pabrikan yang merupakan akar kepercayaan untuk produsen HSM dan sertifikat perangkat. CA pabrikan telah membuktikan kepatuhan dengan PCI PIN Security Annex A dan PCI P2PE Annex A. Pabrikan memverifikasi bahwa semua HSM dengan kunci perangkat yang ditandatangani oleh Payment Cryptography CA dikirim ke penerima yang ditunjuk AWS. AWS

Seperti yang dipersyaratkan oleh PCI PIN Security, pabrikan memasok daftar nomor seri melalui saluran komunikasi yang berbeda dari pengiriman HSM. Nomor seri ini diperiksa pada setiap langkah dalam proses instalasi HSM ke pusat data AWS. Akhirnya, operator Kriptografi AWS Pembayaran memvalidasi daftar HSM yang diinstal terhadap daftar pabrikan sebelum menambahkan nomor seri ke daftar HSM yang diizinkan untuk menerima AWS kunci Kriptografi Pembayaran.

HSMs berada dalam penyimpanan aman atau di bawah kendali ganda setiap saat, yang meliputi:

- Pengiriman dari pabrikan ke fasilitas perakitan rak AWS.
- Selama perakitan rak.
- Pengiriman dari fasilitas perakitan rak ke pusat data.
- Tanda terima dan pemasangan ke ruang pemrosesan aman pusat data. Rak HSM memberlakukan kontrol ganda dengan kunci yang dikontrol akses kartu, sensor pintu alarm, dan kamera.
- Selama operasi.
- Selama dekomisioning dan penghancuran.

Lengkap chain-of-custody, dengan akuntabilitas individu, dipertahankan dan dipantau untuk setiap HSM.

Inisialisasi HSM

HSM hanya diinisialisasi sebagai bagian dari armada Kriptografi AWS Pembayaran setelah identitas dan integritasnya divalidasi oleh nomor seri, kunci perangkat yang diinstal pabrikan, dan checksum firmware. Setelah keaslian dan integritas HSM divalidasi, itu dikonfigurasi, termasuk mengaktifkan Mode PCI. Kemudian kunci utama wilayah Kriptografi AWS Pembayaran dan kunci utama profil ditetapkan dan HSM tersedia untuk layanan.

Layanan dan perbaikan HSM

HSM memiliki komponen yang dapat diservis yang tidak memerlukan pelanggaran batas kriptografi perangkat. Komponen-komponen ini termasuk kipas pendingin, catu daya, dan baterai. Jika HSM atau perangkat lain dalam rak HSM membutuhkan servis, kontrol ganda dipertahankan selama seluruh periode rak terbuka.

Penonaktifan HSM

Penonaktifan terjadi karena end-of-life atau kegagalan HSM. HSM secara logis dizeroisasi sebelum dikeluarkan dari raknya, jika berfungsi, kemudian dihancurkan di dalam ruang pemrosesan yang aman dari pusat data AWS. Mereka tidak pernah dikembalikan ke pabrikan untuk diperbaiki, digunakan untuk tujuan lain, atau dikeluarkan dari ruang pemrosesan yang aman sebelum kehancuran.

Pembaruan firmware HSM

Pembaruan firmware HSM diterapkan bila diperlukan untuk menjaga keselarasan dengan versi terdaftar PCI PTS HSM dan FIPS 140-2 (atau FIPS 140-3), jika pembaruan terkait keamanan, atau

ditentukan bahwa pelanggan dapat memperoleh manfaat dari fitur dalam versi baru. AWS Kriptografi Pembayaran HSMs menjalankan off-the-shelf firmware, cocok dengan versi PCI PTS yang terdaftar di HSM. Versi firmware baru divalidasi untuk integritas dengan versi firmware bersertifikat PCI atau FIPS kemudian diuji fungsionalitasnya sebelum diluncurkan ke semua. HSMs

Akses operator

Operator dapat memiliki akses non-konsol ke HSM untuk pemecahan masalah dalam kasus yang jarang terjadi bahwa informasi yang dikumpulkan dari HSM selama operasi normal tidak cukup untuk mengidentifikasi masalah atau merencanakan perubahan. Langkah-langkah berikut dijalankan:

- Kegiatan pemecahan masalah dikembangkan dan disetujui dan sesi non-konsol dijadwalkan.
- HSM dihapus dari layanan pemrosesan pelanggan.
- Tombol utama dihapus, di bawah kendali ganda.
- Operator diizinkan akses non-konsol ke HSM untuk melakukan aktivitas pemecahan masalah yang disetujui, di bawah kendali ganda.
  - Setelah penghentian sesi non-konsol, proses penyediaan awal dilakukan pada HSM, mengembalikan firmware dan konfigurasi standar, kemudian menyinkronkan kunci utama, sebelum mengembalikan HSM untuk melayani pelanggan.
  - Catatan sesi dicatat dalam pelacakan perubahan.
  - Informasi yang diperoleh dari sesi digunakan untuk merencanakan perubahan masa depan.

Semua catatan akses non-konsol ditinjau untuk kepatuhan proses dan potensi perubahan pada pemantauan HSM, proses non-console-access manajemen, atau pelatihan operator.

Manajemen kunci umum

Semua HSM di suatu wilayah disinkronkan ke Kunci Utama Wilayah. Kunci Utama Wilayah melindungi setidaknya satu Kunci Utama Profil. Kunci Utama Profil melindungi kunci pelanggan.

Semua kunci utama dihasilkan oleh HSM dan didistribusikan dengan distribusi kunci simetris menggunakan teknik asimetris, selaras dengan ANSI X9 TR 34 dan PCI PIN Lampiran A.

Generasi

Kunci utama AES 256 bit dihasilkan pada salah satu HSM yang disediakan untuk armada layanan HSM, menggunakan generator nomor acak PCI PTS HSM.

Sinkronisasi kunci utama wilayah

Kunci utama wilayah HSM disinkronkan oleh layanan di seluruh armada regional dengan mekanisme yang ditentukan oleh ANSI X9 TR-34, yang meliputi:

- Otentikasi bersama menggunakan kunci dan sertifikat host distribusi kunci (KDH) dan perangkat penerima kunci (KRD) untuk memberikan otentikasi dan integritas untuk kunci publik.
- Sertifikat ditandatangani oleh otoritas sertifikat (CA) yang memenuhi persyaratan PCI PIN Annex A2, kecuali untuk algoritma asimetris dan kekuatan kunci yang sesuai untuk melindungi kunci AES 256 bit.
- Identifikasi dan perlindungan kunci untuk kunci simetris terdistribusi konsisten dengan ANSI X9 TR-34 dan PCI PIN Annex A1, kecuali untuk algoritma asimetris dan kekuatan kunci yang sesuai untuk melindungi kunci AES 256 bit.

Kunci utama wilayah dibuat untuk HSMs yang telah diautentikasi dan disediakan untuk suatu wilayah dengan:

- Kunci utama dihasilkan pada HSM di wilayah tersebut. HSM itu ditetapkan sebagai host distribusi utama.
- Semua yang disediakan HSMs di wilayah tersebut menghasilkan token otentikasi KRD, yang berisi kunci publik HSM dan informasi otentikasi yang tidak dapat diputar ulang.
- Token KRD ditambahkan ke daftar izin KDH setelah KDH memvalidasi identitas dan izin HSM untuk menerima kunci.
- KDH menghasilkan token kunci utama yang dapat diautentikasi untuk setiap HSM. Token berisi informasi otentikasi KDH dan kunci utama terenkripsi yang hanya dapat dimuat pada HSM yang telah dibuat untuknya.
- Setiap HSM dikirimkan token kunci utama yang dibuat untuk itu. Setelah memvalidasi informasi otentikasi HSM sendiri dan informasi otentikasi KDH, kunci utama didekripsi oleh kunci pribadi KRD dan dimuat ke kunci utama.

Jika satu HSM harus disinkronkan ulang dengan suatu wilayah:

- Ini divalidasi ulang dan disediakan dengan firmware dan konfigurasi.
- Jika baru di wilayah ini:
  - HSM menghasilkan token otentikasi KRD.

- KDH menambahkan token ke daftar izinnya.
- KDH menghasilkan token kunci utama untuk HSM.
- HSM memuat kunci utama.
- HSM tersedia untuk layanan ini.

Ini memastikan bahwa:

- Hanya HSM yang divalidasi untuk pemrosesan Kriptografi AWS Pembayaran dalam suatu wilayah yang dapat menerima kunci utama wilayah tersebut.
- Hanya kunci master dari Kriptografi AWS Pembayaran HSM yang dapat didistribusikan ke HSM di armada.

Rotasi kunci utama wilayah

Kunci utama wilayah diputar pada saat berakhirnya periode kripto, jika terjadi dugaan kompromi kunci, atau setelah perubahan pada layanan yang ditentukan untuk memengaruhi keamanan kunci.

Kunci utama wilayah baru dihasilkan dan didistribusikan seperti penyediaan awal. Kunci utama profil yang disimpan harus diterjemahkan ke kunci utama wilayah baru.

Rotasi kunci utama wilayah tidak memengaruhi pemrosesan pelanggan.

Sinkronisasi kunci utama profil

Kunci utama profil dilindungi oleh kunci utama wilayah. Ini membatasi profil ke wilayah tertentu.

Kunci utama profil disediakan sesuai:

- Kunci utama profil dihasilkan pada HSM yang memiliki kunci utama wilayah yang disinkronkan.
- Kunci utama profil disimpan dan dienkripsi dengan konfigurasi profil dan konteks lainnya.
- Profil ini digunakan untuk fungsi kriptografi pelanggan oleh HSM mana pun di wilayah dengan kunci utama wilayah.

Profil rotasi kunci utama

Kunci utama profil diputar pada saat berakhirnya periode kripto, setelah dugaan kompromi kunci, atau setelah perubahan pada layanan yang ditentukan untuk memengaruhi keamanan kunci.

Langkah-langkah rotasi:

- Kunci utama profil baru dihasilkan dan didistribusikan sebagai kunci utama yang tertunda seperti penyediaan awal.
- Proses latar belakang menerjemahkan materi kunci pelanggan dari kunci utama profil yang ditetapkan ke kunci utama yang tertunda.
- Ketika semua kunci pelanggan telah dienkripsi dengan kunci yang tertunda, kunci yang tertunda dipromosikan ke kunci utama profil.
- Proses latar belakang menghapus materi kunci pelanggan yang dilindungi oleh kunci kedaluwarsa.

Rotasi kunci utama profil tidak memengaruhi pemrosesan pelanggan.

Perlindungan

Kunci hanya bergantung pada hierarki kunci untuk perlindungan. Perlindungan kunci utama sangat penting untuk mencegah kehilangan atau membahayakan semua kunci pelanggan.

Kunci utama wilayah dapat dipulihkan dari cadangan hanya ke HSM yang diautentikasi dan disediakan untuk layanan. Kunci ini hanya dapat disimpan sebagai token kunci utama yang dapat diautentikasi dan dienkripsi dari KDH tertentu untuk HSM tertentu.

Kunci master profil disimpan dengan konfigurasi profil dan informasi konteks yang dienkripsi berdasarkan wilayah.

Kunci pelanggan disimpan dalam blok kunci, dilindungi oleh kunci master profil.

Semua kunci ada secara eksklusif dalam HSM atau disimpan dilindungi oleh kunci lain dengan kekuatan kriptografi yang sama atau lebih kuat.

Daya tahan

Kunci pelanggan untuk kriptografi transaksi dan fungsi bisnis harus tersedia bahkan dalam situasi ekstrem yang biasanya menyebabkan pemadaman. AWS Kriptografi Pembayaran menggunakan model redundansi beberapa tingkat di seluruh zona ketersediaan dan wilayah. AWS Pelanggan yang membutuhkan ketersediaan dan daya tahan yang lebih tinggi untuk operasi kriptografi pembayaran daripada yang disediakan oleh layanan harus menerapkan arsitektur multi-wilayah.

Otentikasi HSM dan token kunci utama disimpan dan dapat digunakan untuk mengembalikan kunci utama atau menyinkronkan dengan kunci utama baru, jika HSM harus diatur ulang. Token diarsipkan dan digunakan hanya di bawah kontrol ganda bila diperlukan.

Akses operator ke kunci utama HSM

Kunci utama hanya ada di HSM yang dikelola oleh layanan dan diamankan di fasilitas AWS yang aman. Kunci utama tidak dapat diekspor dari HSM atau disinkronkan ke HSM yang tidak diinisialisasi oleh produsen untuk digunakan dalam layanan. Operator AWS tidak dapat memperoleh kunci utama dalam bentuk apa pun yang dapat dimuat ke HSM yang tidak dikelola oleh layanan.

Manajemen kunci pelanggan

Pada AWS, kepercayaan pelanggan adalah prioritas utama kami. Anda mempertahankan kontrol penuh atas kunci yang Anda impor atau buat di layanan di bawah akun AWS Anda. Anda tetap bertanggung jawab untuk mengonfigurasi akses ke kunci.

AWS Payment Cryptography adalah penyedia layanan yang menggunakan HSMs dan mengelola kunci atas nama pelanggan, mirip dengan penyedia layanan pembayaran lama. Layanan ini memiliki tanggung jawab penuh untuk keamanan fisik dan logis HSM. Tanggung jawab manajemen utama dibagi antara layanan dan pelanggan karena pelanggan harus memberikan informasi yang akurat tentang kunci yang dibuat oleh atau diimpor ke layanan, yang digunakan layanan untuk menegakkan penggunaan dan manajemen kunci yang benar. Perlindungan pemisahan data AWS digunakan untuk memastikan bahwa kompromi kunci milik satu akun AWS tidak dapat membahayakan kunci milik yang lain.

AWS Kriptografi Pembayaran memiliki tanggung jawab penuh atas kepatuhan fisik HSM dan manajemen kunci untuk kunci yang dikelola oleh layanan. Ini membutuhkan kepemilikan dan pengelolaan kunci utama HSM dan perlindungan kunci pelanggan yang dikelola oleh Kriptografi AWS Pembayaran.

Pemisahan ruang kunci pelanggan

AWS Kriptografi Pembayaran memberlakukan kebijakan utama untuk semua penggunaan kunci, termasuk membatasi prinsipal ke akun yang memiliki kunci, kecuali kunci secara eksplisit dibagikan dengan akun lain.

Akun AWS menyediakan pemisahan lingkungan yang lengkap antara pelanggan atau aplikasi yang analog dengan implementasi non-cloud di pusat data yang berbeda. Setiap akun menyediakan kontrol akses terisolasi, jaringan, sumber daya komputasi, penyimpanan data, kunci kriptografi untuk perlindungan data dan transaksi pembayaran, dan semua sumber daya AWS. Layanan AWS seperti Organizations and Control Tower memungkinkan pengelolaan perusahaan dari akun aplikasi terpisah, analog dengan kandang atau ruangan dalam pusat data perusahaan.

Akses operator ke kunci pelanggan

Kunci pelanggan yang dikelola oleh layanan disimpan dilindungi oleh kunci utama partisi dan hanya dapat digunakan oleh akun pelanggan yang memiliki atau akun yang telah dikonfigurasi secara khusus oleh pemilik untuk berbagi kunci. Operator AWS tidak dapat mengekspor atau melakukan manajemen kunci atau operasi kriptografi dengan kunci pelanggan menggunakan akses manual ke layanan, yang dikelola oleh mekanisme akses operator manual AWS.

Kode layanan yang mengimplementasikan manajemen dan penggunaan kunci pelanggan tunduk pada praktik kode aman AWS sebagaimana dinilai sesuai penilaian AWS PCI DSS.

Pencadangan dan pemulihan

Kunci dan informasi kunci yang disimpan secara internal oleh layanan untuk suatu wilayah dicadangkan ke arsip terenkripsi oleh. AWS Arsip membutuhkan kontrol ganda AWS untuk memulihkan.

Blok kunci

Semua kunci disimpan dan diproses dalam blok kunci format ANSI X9.143.

Kunci dapat diimpor ke layanan dari kriptogram atau format blok kunci lainnya yang didukung oleh ImportKey. Demikian pula, kunci dapat diekspor, jika dapat diekspor, ke format blok kunci lain atau kriptogram yang didukung oleh profil ekspor utama.

Penggunaan kunci

Penggunaan kunci dibatasi untuk yang dikonfigurasi KeyUsage oleh layanan. Layanan akan gagal setiap permintaan dengan penggunaan kunci yang tidak tepat, mode penggunaan, atau algoritma untuk operasi kriptografi yang diminta.

Hubungan pertukaran kunci

PCI PIN Security dan PCI P2PE mengharuskan organisasi yang berbagi kunci yang mengenkripsi PINs atau data kartu, termasuk kunci pertukaran kunci (KEK) yang digunakan untuk berbagi kunci tersebut, tidak berbagi kunci yang sama dengan organisasi lain. Ini adalah praktik terbaik bahwa kunci simetris dibagi antara hanya 2 pihak untuk satu tujuan, termasuk dalam organisasi yang sama. Ini meminimalkan dampak dari dugaan kompromi kunci yang memaksa penggantian kunci yang terkena dampak.

Bahkan kasus bisnis yang memerlukan kunci berbagi antara lebih dari 2 pihak, harus menjaga jumlah pihak ke jumlah minimum.

AWS Kriptografi Pembayaran menyediakan tag kunci yang dapat digunakan untuk melacak dan menegakkan penggunaan kunci dalam persyaratan tersebut.

Misalnya, KEK dan BDK untuk fasilitas injeksi kunci yang berbeda dapat diidentifikasi dengan menetapkan “KIF” = “POSStation” untuk semua kunci yang dibagikan dengan penyedia layanan tersebut. Contoh lain adalah menandai kunci yang dibagikan dengan jaringan pembayaran dengan “Jaringan” = “PayCard”. Penandaan memungkinkan Anda membuat kontrol akses dan membuat laporan audit untuk menegakkan dan mendemonstrasikan praktik manajemen utama Anda.

Penghapusan kunci

DeleteKey menandai kunci dalam database untuk dihapus setelah periode yang dapat dikonfigurasi pelanggan. Setelah periode ini kuncinya dihapus secara permanen. Ini adalah mekanisme keamanan untuk mencegah penghapusan kunci yang tidak disengaja atau berbahaya. Kunci yang ditandai untuk penghapusan tidak tersedia untuk tindakan apa pun kecuali. RestoreKey

Kunci yang dihapus tetap dalam cadangan layanan selama 7 hari setelah penghapusan. Mereka tidak dapat dipulihkan selama periode ini.

Kunci milik akun AWS tertutup ditandai untuk dihapus. Jika akun diaktifkan kembali sebelum periode penghapusan tercapai, kunci apa pun yang ditandai untuk penghapusan dipulihkan, tetapi dinonaktifkan. Mereka harus diaktifkan kembali oleh Anda untuk menggunakannya untuk operasi kriptografi.

Keamanan komunikasi

Eksternal

AWS Titik akhir API Kriptografi Pembayaran memenuhi standar AWS keamanan termasuk TLS pada atau di atas 1.2 dan Signature Versi 4 untuk otentikasi dan integritas permintaan.

Koneksi TLS yang masuk dihentikan pada penyeimbang beban jaringan dan diteruskan ke penangan API melalui koneksi TLS internal.

Internal

Komunikasi internal antara komponen layanan dan antara komponen layanan dan layanan AWS lainnya dilindungi oleh TLS menggunakan kriptografi yang kuat.

HSM berada di jaringan pribadi non-virtual yang hanya dapat dijangkau dari komponen layanan. Semua koneksi antara HSM dan komponen layanan diamankan dengan TLS bersama (MTL), pada atau di atas TLS 1.2. Sertifikat internal untuk TLS dan mTL dikelola oleh Amazon Certificate Manager menggunakan AWS Private Certificate Authority. Internal VPCs dan jaringan HSM dipantau untuk aktivitas tak terduga dan perubahan konfigurasi.

Pencatatan log dan pemantauan

Log layanan internal meliputi:

- CloudTrail log panggilan layanan AWS yang dilakukan oleh layanan
- CloudWatch log dari kedua peristiwa langsung masuk ke CloudWatch log atau peristiwa dari HSM
- File log dari HSM dan sistem layanan
- Arsip log

Semua sumber log memantau dan memfilter informasi sensitif, termasuk tentang kunci. Log ditinjau secara sistematis untuk memastikan bahwa mereka mengandung tidak mengandung informasi pelanggan yang sensitif.

Akses ke log dibatasi untuk individu yang dibutuhkan untuk menyelesaikan peran pekerjaan.

Semua log disimpan selaras dengan kebijakan penyimpanan log AWS.

Operasi pelanggan

AWS Kriptografi Pembayaran memiliki tanggung jawab penuh atas kepatuhan fisik HSM berdasarkan standar PCI. Layanan ini juga menyediakan penyimpanan kunci yang aman dan memastikan bahwa kunci hanya dapat digunakan untuk tujuan yang diizinkan oleh standar PCI dan ditentukan oleh Anda selama pembuatan atau impor. Anda bertanggung jawab untuk mengonfigurasi atribut utama dan akses untuk memanfaatkan kemampuan keamanan dan kepatuhan layanan.

Topik

- Merotasi kunci

Menghasilkan kunci

Saat membuat kunci, Anda menyetel atribut yang digunakan layanan untuk menerapkan penggunaan kunci yang sesuai:

- Algoritma dan panjang kunci
- Penggunaan
- Ketersediaan dan kedaluwarsa

Tag yang digunakan untuk kontrol akses berbasis atribut (ABAC) digunakan untuk membatasi kunci untuk digunakan dengan mitra atau aplikasi tertentu juga harus ditetapkan selama pembuatan. Pastikan untuk menyertakan kebijakan untuk membatasi peran yang diizinkan untuk menghapus atau mengubah tag.

Anda harus memastikan bahwa kebijakan yang menentukan peran yang mungkin menggunakan dan mengelola kunci ditetapkan sebelum pembuatan kunci.

> **Note**
>
> Kebijakan IAM pada CreateKey perintah dapat digunakan untuk menegakkan dan mendemonstrasikan kontrol ganda untuk pembuatan kunci.

Mengimpor kunci

Saat mengimpor kunci, atribut untuk menerapkan penggunaan kunci yang sesuai ditetapkan oleh layanan menggunakan informasi yang terikat secara kriptografis di blok kunci. Mekanisme untuk mengatur konteks kunci fundamental adalah dengan menggunakan blok kunci yang dibuat dengan sumber HSM dan dilindungi oleh KEK bersama atau asimetris. Ini sejalan dengan persyaratan PIN PCI dan mempertahankan penggunaan, algoritme, dan kekuatan kunci dari aplikasi sumber.

Atribut kunci penting, tag, dan kebijakan kontrol akses harus ditetapkan pada impor selain informasi di blok kunci.

Mengimpor kunci menggunakan kriptogram tidak mentransfer atribut kunci dari aplikasi sumber. Anda harus mengatur atribut dengan tepat dengan menggunakan mekanisme ini.

Seringkali kunci dipertukarkan menggunakan komponen teks yang jelas, ditransmisikan oleh penjaga kunci, kemudian dimuat dengan upacara yang menerapkan kontrol ganda di ruang aman. Ini tidak didukung secara langsung oleh Kriptografi AWS Pembayaran. API akan mengekspor kunci publik dengan sertifikat yang dapat diimpor oleh HSM Anda sendiri untuk mengekspor blok kunci yang dapat diimpor oleh layanan. Ini memungkinkan penggunaan HSM Anda sendiri untuk memuat komponen teks yang jelas.

Anda harus menggunakan Nilai cek kunci (KCV) untuk memverifikasi bahwa kunci yang diimpor cocok dengan kunci sumber.

Kebijakan IAM pada ImportKey API dapat digunakan untuk menegakkan dan menunjukkan kontrol ganda untuk impor kunci.

Mengekspor kunci

Berbagi kunci dengan mitra atau aplikasi lokal mungkin memerlukan kunci ekspor. Menggunakan blok kunci untuk ekspor mempertahankan konteks kunci fundamental dengan materi kunci terenkripsi.

Tag kunci dapat digunakan untuk membatasi ekspor kunci ke KEK yang berbagi tag dan nilai yang sama.

AWS Kriptografi Pembayaran tidak menyediakan atau menampilkan komponen kunci teks yang jelas. Ini memerlukan akses langsung oleh penjaga kunci ke PCI PTS HSM atau ISO 13491 perangkat kriptografi aman (SCD) yang diuji untuk tampilan atau pencetakan. Anda dapat membuat KEK asimetris atau KEK simetris dengan SCD Anda untuk melakukan upacara pembuatan komponen kunci teks yang jelas di bawah kendali ganda.

Nilai pemeriksaan kunci (KCV) harus digunakan untuk memverifikasi bahwa diimpor oleh kunci sumber pencocokan HSM tujuan.

Menghapus kunci

Anda dapat menggunakan API kunci hapus untuk menjadwalkan kunci untuk dihapus setelah periode waktu yang Anda konfigurasi. Sebelum itu kunci waktu dapat dipulihkan. Setelah kunci dihapus, kunci akan dihapus secara permanen dari layanan.

Kebijakan IAM pada DeleteKey API dapat digunakan untuk menegakkan dan mendemonstrasikan kontrol ganda untuk penghapusan kunci.

Merotasi kunci

Efek rotasi kunci dapat diimplementasikan menggunakan alias kunci dengan membuat atau mengimpor kunci baru, kemudian memodifikasi alias kunci untuk merujuk ke kunci baru. Kunci lama akan dihapus atau dinonaktifkan, tergantung pada praktik manajemen Anda.

Kuota untuk AWS Payment Cryptography

Akun AWS Anda memiliki kuota default, yang sebelumnya disebut sebagai batas, untuk setiap layanan AWS. Kecuali dinyatakan lain, setiap kuota bersifat spesifik wilayah. Anda dapat meminta peningkatan untuk beberapa kuota dan kuota lainnya tidak dapat ditingkatkan.

Nama	Default	Dapat disesuaikan	Deskripsi
Alias	Setiap Wilayah yang didukung: 2.000	Ya	Jumlah maksimum alias yang dapat Anda miliki di akun ini di Wilayah saat ini.
Tingkat gabungan permintaan bidang kontrol	Setiap Wilayah yang didukung: 5 per detik	Ya	Jumlah maksimum permintaan pesawat kontrol per detik yang dapat Anda buat di akun ini di Wilayah saat ini. Kuota ini berlaku untuk semua operasi pesawat kontrol yang digabungkan.
Tingkat gabungan permintaan bidang data (asimetris)	Setiap Wilayah yang didukung: 20 per detik	Ya	Jumlah maksimum permintaan per detik untuk operasi bidang data dengan kunci asimetris yang dapat Anda buat di akun ini di Wilayah saat ini. Kuota ini berlaku untuk semua operasi pesawat data yang digabungkan.

Nama	Default	Dapat disesuaikan	Deskripsi
Tingkat gabungan permintaan bidang data (simetris)	Setiap Wilayah yang didukung: 500 per detik	Ya	Jumlah maksimum permintaan per detik untuk operasi bidang data dengan kunci simetris yang dapat Anda buat di akun ini di Wilayah saat ini. Kuota ini berlaku untuk semua operasi pesawat data yang digabungkan.
Kunci	Setiap Wilayah yang didukung: 2.000	Ya	Jumlah maksimum kunci yang dapat Anda miliki di akun ini di Wilayah saat ini, tidak termasuk kunci yang dihapus.

Riwayat dokumen untuk Panduan Pengguna Kriptografi AWS Pembayaran

Tabel berikut menjelaskan rilis dokumentasi untuk Kriptografi AWS Pembayaran.

Perubahan	Deskripsi	Tanggal
Fitur Baru - AS28 05	Support untuk algoritme dan alur untuk mendukung AS28 05 dukungan regional	Desember 17, 2025
Fitur Baru - Replikasi kunci Multi-Region	Dengan replikasi kunci Multi-Region, Anda dapat mereplikasi kunci Kriptografi AWS Pembayaran Anda ke beberapa. Wilayah AWS	September 10, 2025
Fitur Baru - ECDH	Dengan rilis ini, ECDH dapat digunakan untuk membuat KEK bersama untuk pertukaran kunci lebih lanjut.	Maret 30, 2025
Panduan Pertukaran Kunci Baru	Panduan baru disediakan untuk pertukaran utama. Informasi tentang perintah JCB umum juga ditambahkan.	Januari 31, 2025
Peluncuran wilayah baru	Menambahkan titik akhir untuk peluncuran kawasan baru di Eropa (Frankfurt), Eropa (Irlandia), Asia Pasifik (Singapura) dan Asia Pasifik (Tokyo)	Juli 31, 2024
CloudTrail untuk Data Plane dan Dynamic Keys	Menambahkan informasi tentang penggunaan CloudTrail untuk operasi	Juli 10, 2024

	bidang data (kriptografi) termasuk contoh. Juga menambahkan informasi tentang penggunaan Dynamic Keys untuk fungsi-fungsi tertentu untuk lebih mendukung kunci penggunaan satu kali atau terbatas yang tidak boleh diimpor ke AWS Kriptografi Pembayaran	
Contoh yang Diperbarui	Menambahkan contoh baru untuk penerbitan kartu	Juli 1, 2024
Rilis fitur	Menambahkan informasi tentang titik akhir VPC (PrivateLink) dan contoh iCVV.	30 Mei 2024
Rilis fitur	Informasi ditambahkan pada fitur baru di sekitar kunci import/export menggunakan RSA dan mengekspor kunci IPEK/IK DUKPT.	Januari 15, 2024
Rilis awal	Rilis awal Panduan Pengguna Kriptografi AWS Pembayaran	8 Juni 2023

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.