Guida per l'utente

AWS Crittografia dei pagamenti



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

AWS Crittografia dei pagamenti: Guida per l’utente

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

I marchi e l’immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà dei rispettivi proprietari, che possono o meno essere affiliati, collegati o sponsorizzati da Amazon.

Table of Contents

Cos'è la crittografia AWS dei pagamenti? 1
Concetti 2
Terminologia di settore 4
Tipi di chiavi comuni 4
Altri termini 7
Servizi correlati 13
Ulteriori informazioni 13
Endpoints 14
Endpoint del piano di controllo 14
Endpoint del piano dati 17
Nozioni di base 21
Prerequisiti 21
Fase 1: Creare una chiave 22
Passaggio 2: generare un CVV2 valore utilizzando la chiave 23
Fase 3: Verificare il valore generato nel passaggio 2 23
Fase 4: Eseguire un test negativo 24
Fase 5: (Facoltativo) Pulizia 24
Gestione delle chiavi 26
Creazione di chiavi 26
Creazione di una chiave di derivazione di base TDES 2KEY 27
Creazione di una chiave TDES 2KEY per CVV/CVV2 29
Creazione di una chiave HMAC 30
Creazione di una chiave AES-256 31
Creazione di una chiave di crittografia PIN (PEK) 32
Creazione di una chiave asimmetrica (RSA) 33
Creazione di una chiave PIN Verification Value (PVV) 34
Creazione di una chiave ECC asimmetrica 35
Elenco delle chiavi 36
Abilitazione e disabilitazione delle chiavi 38
Inizia a utilizzare la chiave 38
Interrompi l'uso delle chiavi 40
Replica delle chiavi 42
Vantaggi della replica delle chiavi Multi-Region 42
Come funziona la replica delle chiavi Multi-Region 42

Considerazioni e limitazioni 42
Abilitazione della replica delle chiavi Multi-Region 44
Disattivazione della replica delle chiavi Multi-Region 46
Considerazioni relative alla sicurezza 47
Best practice 47
Prezzi 47
Eliminazione delle chiavi 48
Informazioni sul periodo di attesa 49
Importazione ed esportazione di chiavi 52
Chiavi di importazione 54
Chiavi di esportazione 80
Argomenti avanzati 103
Utilizzo di alias 122
Informazioni sugli alias 123
Utilizzo di alias nelle applicazioni 126
API correlate 127
Ottieni le chiavi 127
Fa associare il pubblico key/certificate a una coppia di chiavi 129
Chiavi di tagging 130
Informazioni sui tag in AWS Crittografia dei pagamenti 130
Visualizzazione dei tag chiave nella console 132
Gestione dei tag chiave con operazioni API 132
Controllo degli accessi ai tag 135
Utilizzo dei tag per controllare l'accesso alle chiavi 139
Comprensione degli attributi chiave 142
Chiavi simmetriche 143
Chiavi asimmetriche 145
Operazioni sui dati 147
Crittografa, decrittografa e ricrittografa i dati 147
Crittografare i dati 148
Decrittare i dati 154
Genera e verifica i dati della carta 158
Genera i dati delle carte 159
Verifica i dati della carta 160
Genera, traduci e verifica i dati PIN 162
Traduci i dati PIN 163

Genera dati PIN ... 165
Verifica i dati del PIN ... 169
Genera e verifica ARQC ... 173
Genera crittogramma di richiesta di autenticazione (ARQC) ... 174
Crittogramma Verify Auth Request (ARQC) ... 176
Genera e verifica MAC ... 180
Genera MAC ... 182
Verifica MAC ... 186
Tipi di chiave per operazioni specifiche sui dati ... 188
GenerateCardData ... 189
VerifyCardData ... 190
GeneratePinData (per VISA/ABA schemi) ... 191
GeneratePinData (per IBM3624) ... 192
VerifyPinData (per VISA/ABA schemi) ... 193
VerifyPinData (per IBM3624) ... 194
Decrittografia dei dati ... 195
Encrypt Data (Crittografa dati) ... 196
Traduci i dati dei pin ... 197
Generate/Verify MAC ... 198
GenerateMacEmvPinChange ... 200
VerifyAuthRequestCryptogram ... 201
Import/Export Chiave ... 201
Tipi di chiavi non utilizzati ... 202
Casi di utilizzo comune ... 203
Emittenti e responsabili del trattamento degli emittenti ... 203
Funzioni generali ... 203
Funzioni specifiche della rete ... 236
Agevolatori di acquisizione e pagamento ... 269
Uso dei tasti dinamici ... 270
Caratteristiche specifiche della regione ... 273
AS2805 ... 273
Scambio di chiavi iniziali (KEK) ... 275
Convalida del KEK ... 276
Creazione e trasmissione di chiavi funzionanti ... 279
Esportazione di chiavi di lavoro ... 281
Pin Translation ... 282

Generazione e convalida Mac 283
Sicurezza 284
Protezione dei dati 285
Protezione del materiale della chiave 286
Crittografia dei dati 286
Crittografia dei dati a riposo 286
Crittografia dei dati in transito 287
Riservatezza del traffico Internet 287
Resilienza 288
Isolamento regionale 288
Multi-tenant design 289
Sicurezza dell'infrastruttura 289
Isolamento degli host fisici 290
Usa Amazon VPC e AWS PrivateLink 290
Considerazioni per AWS Endpoint VPC per la crittografia dei pagamenti 291
Creazione di un endpoint VPC per AWS Crittografia dei pagamenti 292
Connessione a un endpoint VPC 293
Controllo dell'accesso all'endpoint VPC 293
Utilizzo di un endpoint VPC in un'istruzione di policy 297
Registrazione dell'endpoint VPC 301
Protocollo TLS post-quantistico ibrido 303
Informazioni sul protocollo TLS post quantistico 305
Come utilizzarlo 305
Informazioni su PQC 305
Come utilizzarlo 305
Best practice di sicurezza 305
Convalida della conformità 307
Conformità del servizio 307
Conformità PIN 308
Argomenti comuni 308
Ambito di valutazione 310
Operazioni di elaborazione delle transazioni 312
Conformità P2PE 318
Gestione dell'identità e degli accessi 319
Destinatari 319
Autenticazione con identità 320

Account AWS utente root 320
Utenti e gruppi IAM 320
Ruoli IAM 320
Gestione dell'accesso tramite policy 321
Identity-based politiche 321
Resource-based politiche 321
Liste di controllo degli accessi (ACL) 322
Altri tipi di policy 322
Più tipi di policy 323
In che modo AWS La crittografia dei pagamenti funziona con IAM 323
AWS Politiche di crittografia dei pagamenti Identity-based 323
Autorizzazione basata su AWS Tag di crittografia dei pagamenti 325
Identity-based esempi di politiche 325
Best practice delle policy 326
Utilizzo della console 327
Consentire agli utenti di visualizzare le loro autorizzazioni 328
Capacità di accedere a tutti gli aspetti di AWS Crittografia dei pagamenti 329
Possibilità di chiamare le API utilizzando chiavi specifiche 329
Capacità di negare specificamente una risorsa 330
Resource-based politiche 331
Considerazioni 332
Gestione delle politiche basate sulle risorse 333
Resource-based esempi di politiche 334
Multi-party approvazione 336
Panoramica di 336
Operazioni protette 337
Prerequisiti 337
Abilitazione e disabilitazione di MPA 338
Nozioni di base 339
Esempio: importazione di un certificato root con MPA abilitato 339
AWS CloudTrail registrazione degli eventi MPA 340
Verifica dello stato della richiesta e gestione degli errori 342
Risoluzione dei problemi 345
Monitoraggio 346
CloudTrail registri 346
AWS Informazioni sulla crittografia dei pagamenti in CloudTrail 347

Eventi del piano di controllo in CloudTrail 348
Eventi relativi ai dati in CloudTrail 348
Comprensione delle AWS voci dei file di registro di Payment Cryptography Control Plane ... 349
Comprensione delle voci AWS del file di registro del piano dati della crittografia dei pagamenti 353
Dettagli crittografici 356
Obiettivi di progettazione di 357
Fondamenti 358
Primitive di crittografia 358
Entropia e generazione di numeri casuali 359
Operazioni chiave simmetriche 359
Operazioni chiave asimmetriche 359
Archiviazione delle chiavi 360
Importazione di chiavi tramite chiavi simmetriche 360
Importazione di chiavi tramite chiavi asimmetriche 360
Esportazione di chiavi 361
Protocollo DUKPT (Derived Unique Key Per Transaction) 361
Gerarchia delle chiavi 361
Operazioni interne 365
Protezione HSM 365
Gestione generale delle chiavi 368
Gestione delle chiavi dei clienti 372
Sicurezza delle comunicazioni 374
Registrazione di log e monitoraggio 375
Operazioni con i clienti 375
Generazione delle chiavi 376
Importazione delle chiavi 376
Esportazione delle chiavi 377
Eliminazione delle chiavi 377
Rotazione delle chiavi 378
Quote 379
Cronologia dei documenti 381
..... ccclxxxiii

Cos'è la crittografia AWS dei pagamenti?

AWS Payment Cryptography è un AWS servizio gestito che fornisce l'accesso alle funzioni crittografiche e alla gestione delle chiavi utilizzate nell'elaborazione dei pagamenti in conformità agli standard del settore delle carte di pagamento (PCI) senza la necessità di procurarsi istanze HSM di pagamento dedicate. AWS Payment Cryptography offre ai clienti che svolgono funzioni di pagamento come acquirenti, facilitatori di pagamento, reti, switch, processori e banche la possibilità di spostare le proprie operazioni crittografiche di pagamento più vicino alle applicazioni nel cloud e ridurre al minimo le dipendenze da data center ausiliari o strutture di colocation contenenti pagamenti dedicati HSMs.

Il servizio è progettato per soddisfare le regole di settore applicabili, tra cui PCI PIN, PCI P2PE e PCI DSS, e sfrutta l'hardware certificato PCI PTS HSM V3 e FIPS 140-2 Livello 3. È progettato per supportare una bassa latenza e alti livelli di uptime e resilienza. AWS La crittografia dei pagamenti è completamente elastica ed elimina molti dei requisiti operativi delle strutture locali HSMs, come la necessità di fornire hardware, gestire in modo sicuro il materiale chiave e mantenere i backup di emergenza in strutture sicure. AWS Payment Cryptography ti offre anche la possibilità di condividere le chiavi con i tuoi partner elettronicamente, eliminando la necessità di condividere componenti di testo cartaceo in chiaro.

Puoi utilizzare l'API AWS Payment Cryptography Control Plane per creare e gestire le chiavi.

È possibile utilizzare l'API AWS Payment Cryptography Data Plane per utilizzare le chiavi di crittografia per l'elaborazione delle transazioni relative ai pagamenti e le operazioni crittografiche associate.

AWS Payment Cryptography offre importanti funzionalità che puoi utilizzare per gestire le tue chiavi:

- Crea e gestisci chiavi di crittografia di AWS pagamento simmetriche e asimmetriche, incluse le chiavi TDES, AES e RSA, e specifica lo scopo previsto, ad esempio per la generazione di CVV o la derivazione di chiavi DUKPT.
- Archivia automaticamente le chiavi di crittografia dei AWS pagamenti in modo sicuro, protette da moduli di sicurezza hardware (), applicando al contempo la separazione delle chiavi tra i casi d'uso. HSMs
- Crea, elimina, elenca e aggiorna gli alias, che sono «nomi descrittivi» che possono essere utilizzati per accedere o controllare l'accesso alle tue AWS chiavi di crittografia dei pagamenti.
- Etichetta le tue chiavi AWS di crittografia dei pagamenti per l'identificazione, il raggruppamento, l'automazione, il controllo degli accessi e il monitoraggio dei costi.

- Importa ed esporta chiavi simmetriche tra AWS Payment Cryptography e il tuo HSM (o terze parti) utilizzando Key Encryption Keys (KEK) secondo TR-31 (Interoperable Secure Key Exchange Key Block Specification).
- Importa ed esporta chiavi di crittografia a chiave simmetrica (KEK) tra AWS Payment Cryptography e altri sistemi utilizzando coppie di chiavi asimmetriche, quindi utilizzando strumenti elettronici come TR-34 (metodo per la distribuzione di chiavi simmetriche utilizzando tecniche asimmetriche).

È possibile utilizzare le chiavi di crittografia dei pagamenti in operazioni crittografiche, ad esempio: AWS

- Crittografa, decrittografa e ricrittografa i dati con chiavi di crittografia dei pagamenti simmetriche o asimmetriche. AWS
- Traduci in modo sicuro i dati sensibili (come i pin del titolare della carta) tra le chiavi di crittografia senza esporre il testo non crittografato in conformità alle regole PCI PIN.
- Genera o convalida i dati dei titolari di carta come CVV o ARQC. CVV2
- Genera e convalida i pin del titolare della carta.
- Genera o convalida firme MAC.

Concetti

Scopri i termini e i concetti di base utilizzati nella crittografia dei AWS pagamenti e come utilizzarli per proteggere i tuoi dati.

Alias

Un nome intuitivo associato a una chiave di crittografia dei AWS pagamenti. L'alias può essere utilizzato in modo intercambiabile con la chiave ARN in molte operazioni dell'API Payment Cryptography. AWS Gli alias consentono di ruotare o modificare in altro modo le chiavi senza influire sul codice dell'applicazione. Il nome alias è una stringa di massimo 256 caratteri. Identifica in modo univoco una chiave di crittografia dei AWS pagamenti associata all'interno di un account e di una regione. In AWS Payment Cryptography, i nomi alias iniziano sempre con. `alias/`

Il formato di un nome alias è il seguente:

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

Per esempio:

```
alias/sampleAlias2
```

ARN della chiave

L'ARN chiave è l'Amazon Resource Name (ARN) di una voce chiave in Payment Cryptography. AWS È un identificatore unico e completamente qualificato per la chiave Payment Cryptography. AWS Un ARN chiave include una regione Account AWS, e un ID generato casualmente. L'ARN non è correlato o derivato dal materiale chiave. Poiché vengono assegnati automaticamente durante le operazioni di creazione o importazione, questi valori non sono idempotenti. L'importazione della stessa chiave più volte comporterà più chiavi ARNs con il proprio ciclo di vita.

Il formato di un ARN della chiave è il seguente:

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

Di seguito è riportato un esempio di codice ARN:

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

Identificatore chiave

Un identificatore di chiave è un riferimento a una chiave e uno (o più) di essi sono input tipici delle operazioni di crittografia dei AWS pagamenti. Gli identificatori di chiave validi possono essere un Key Arn un Key Alias.

AWS Chiavi di crittografia dei pagamenti

AWS Le chiavi (chiavi) di crittografia dei pagamenti vengono utilizzate per tutte le funzioni crittografiche. Le chiavi vengono generate direttamente dall'utente utilizzando il comando create key o aggiunte al sistema chiamando key import. L'origine di una chiave può essere determinata esaminando l'attributo KeyOrigin. AWS La crittografia dei pagamenti supporta anche chiavi derivate o intermedie utilizzate durante le operazioni crittografiche come quelle utilizzate da DUKPT.

Queste chiavi hanno attributi immutabili e mutabili definiti al momento della creazione. Gli attributi, come algoritmo, lunghezza e utilizzo, vengono definiti al momento della creazione e non possono essere modificati. Altri, come la data di validità o la data di scadenza, possono essere modificati. Consulta il riferimento all'API AWS Payment Cryptography per un elenco completo degli attributi delle chiavi di crittografia dei AWS pagamenti.

AWS Le chiavi di crittografia dei pagamenti hanno tipi di chiave, definiti principalmente da ANSI X9 TR 31, che ne limitano l'uso allo scopo previsto, come specificato nel requisito 19 del PCI PIN v3.1.

Gli attributi sono associati alle chiavi mediante blocchi chiave quando vengono archiviati, condivisi con altri account o esportati come specificato nel Requisito 18-3 del PCI PIN v3.1.

Le chiavi vengono identificate nella piattaforma AWS Payment Cryptography utilizzando un valore univoco noto come chiave Amazon Resource Name (ARN).

> **Note**
>
> ARNLa chiave viene generata quando una chiave viene inizialmente creata o importata nel servizio AWS Payment Cryptography. Pertanto, se si aggiunge lo stesso materiale chiave più volte utilizzando la funzionalità di importazione delle chiavi, lo stesso materiale chiave verrà posizionato in più chiavi ARNS ma ognuna con un ciclo di vita della chiave diverso.

Terminologia di settore

Argomenti

- Tipi di chiavi comuni
- Altri termini

Tipi di chiavi comuni

AWS Chiave di crittografia dei pagamenti

Una chiave AWS di crittografia dei pagamenti esiste in un'unica chiave. Regione AWSÈ costituita da metadati chiave e materiale archiviati nel AWS Payment Cryptography Service. Una chiave può essere importata da una fonte esterna come blocco TR-31 chiave o generata dal AWS Payment Cryptography Service.

CHIEDERE

Una chiave di lavoro dell'acquirer (AWK) è una chiave tipicamente utilizzata per lo scambio di dati tra un acquirer/acquirer processore e una rete (come Visa o Mastercard). Storicamente AWK utilizza 3DES per la crittografia e veniva rappresentato come TR31_P0_PIN_ENCRYPTION_KEY.

BDK

Una chiave di derivazione di base (BDK) è una chiave di lavoro utilizzata per derivare chiavi successive ed è comunemente usata come parte del processo PCI PIN e PCI P2PE DUKPT. È indicata come TR31_B0_BASE_DERIVATION_KEY.

CMK

Una card master key (CMK) è una o più chiavi specifiche per una carta, tipicamente derivate da una chiave master dell'emittente, da PAN e da PSN e sono in genere chiavi 3DES. Queste chiavi vengono memorizzate sul chip EMV durante la personalizzazione. Esempi di CMK includono le chiavi AC, SMI e SMC.

CMK-AC

Una chiave di crittografia dell'applicazione (AC) viene utilizzata come parte delle transazioni EMV per generare il crittogramma della transazione ed è un tipo di chiave master della carta.

CMK-SMI

Una chiave SMI (Secure Messaging Integrity) viene utilizzata come parte di EMV per verificare l'integrità dei payload inviati alla scheda tramite MAC, ad esempio gli script di aggiornamento dei pin. È un tipo di chiave principale della carta.

CMK-SMC

Una chiave SMC (Secure Messaging Reservatezza) viene utilizzata come parte di EMV per crittografare i dati inviati alla scheda, come gli aggiornamenti dei pin. È un tipo di chiave master della carta.

CVK

Una chiave di verifica della carta (CVK) è una chiave utilizzata per generare CVV, CVV2 e valori simili utilizzando un algoritmo definito e per convalidare un input. È indicata come TR31_C0_CARD_VERIFICATION_KEY.

IMK

Una chiave master dell'emittente (IMK) è una chiave master utilizzata come parte della personalizzazione delle chip card EMV. In genere ci saranno 3 IMK, uno ciascuno per le chiavi AC (crittogramma), SMI (chiave master dello script per integrity/signature) e SMC (chiave master dello script per). confidentiality/encryption

cinematica inversa

Una chiave iniziale (IK) è la prima chiave utilizzata nel processo DUKPT e deriva dalla Base Derivation Key (BDK). Nessuna transazione viene elaborata su questa chiave, ma viene utilizzata per derivare chiavi future che verranno utilizzate per le transazioni. Il metodo di derivazione per la creazione di un IK è stato definito in. X9.24-1:2017 Quando viene utilizzato un TDES BDK, è lo standard applicabile e IK X9.24-1:2009 viene sostituito da Initial Pin Encryption Key (IPEK).

IPEK

Una chiave di crittografia PIN iniziale (IPEK) è la chiave iniziale utilizzata nel processo DUKPT e deriva dalla Base Derivation Key (BDK). Nessuna transazione viene elaborata su questa chiave, ma viene utilizzata per derivare chiavi future che verranno utilizzate per le transazioni. IPEK è un termine improprio in quanto questa chiave può essere utilizzata anche per derivare la crittografia dei dati e le chiavi mac. Il metodo di derivazione per la creazione di un IPEK è stato definito in. X9.24-1:2009 Quando viene utilizzato un AES BDK, X9.24-1:2017 è lo standard applicabile e IPEK viene sostituito da Initial Key (IK).

IWK

Una chiave di lavoro dell'emittente (IWK) è una chiave tipicamente utilizzata per lo scambio di dati tra un issuer/issuer processore e una rete (come Visa o Mastercard). Storicamente IWK utilizza 3DES per la crittografia ed è rappresentato come TR31_P0_PIN_ENCRYPTION_KEY.

KBPK

Una chiave di protezione dei blocchi di chiavi (KBPK) è un tipo di chiave simmetrica utilizzata per proteggere i blocchi di chiavi e quindi altre chiavi. wrap/encrypt Un KBPK è simile a un KEK ma un KEK protegge direttamente il materiale chiave, mentre in TR-31 schemi simili, il KBPK protegge solo indirettamente la chiave funzionante. Quando viene utilizzata TR-31, TR31_K1_KEY_BLOCK_PROTECTION_KEY è il tipo di chiave corretto, sebbene TR31_K0_KEY_ENCRYPTION_KEY sia supportata in modo intercambiabile per scopi storici.

KEK

Una chiave di crittografia (KEK) è una chiave utilizzata per crittografare altre chiavi per la trasmissione o l'archiviazione. Le chiavi destinate a proteggere altre chiavi in genere hanno un valore di KeyUsage TR31_K0_KEY_ENCRYPTION_KEY secondo lo standard. TR-31

PEK

Una chiave di crittografia PIN (PEK) è un tipo di chiave di lavoro utilizzata per crittografare i PIN per l'archiviazione o la trasmissione tra due parti. IWK e AWK sono due esempi di

usi specifici delle chiavi di crittografia dei pin. Queste chiavi sono rappresentate come TR31_P0_PIN_ENCRYPTION_KEY.

PGK

PGK (Pin Generation Key) è un altro nome per una chiave di verifica del PIN. In realtà non viene utilizzato per generare pin (che per impostazione predefinita sono numeri crittograficamente casuali) ma viene invece utilizzato per generare valori di verifica come PVV.

PRK

La chiave Primary Region è la fonte di replica autorevole per una determinata chiave di crittografia di pagamento per la quale è stata abilitata la replica. PRK è un riferimento a un ruolo chiave di Payment Cryptography di origine in una configurazione di replica delle chiavi. Multi-Region Quando la replica è abilitata su una chiave di crittografia dei pagamenti, viene chiamata PRK per quella specifica configurazione di replica delle chiavi.

PVK

Una chiave di verifica PIN (PVK) è un tipo di chiave funzionante utilizzata per generare valori di verifica del PIN come PVV. I due tipi più comuni sono TR31_V1_IBM3624_PIN_VERIFICATION_KEY utilizzato per generare valori di offset IBM3624 e TR31_V2_VISA_PIN_VERIFICATION_KEY utilizzato per i valori di verifica. Visa/ABA Questa può anche essere nota come Pin Generation Key.

RRK

Le Replica Region Key sono il materiale chiave e i metadati replicati copiati in modo sicuro dal PRK a una replica configurata. Regione AWS Un RRK è una replica in sola lettura di una chiave di crittografia dei pagamenti. RRK è un riferimento per il ruolo svolto da una chiave specifica in una Multi-Region configurazione di replica delle chiavi. Qualsiasi modifica chiave dei metadati, incluse le impostazioni di replica, deve essere applicata al PRK.

Altri termini

ARQC

Authorization Request Cryptogram (ARQC) è un crittogramma generato al momento della transazione da una chip card standard EMV (o un'implementazione contactless equivalente). In genere, un ARQC viene generato da una chip card e inoltrato a un emittente o al suo agente per la verifica al momento della transazione.

CVV

Il valore di verifica della carta è un valore segreto statico tradizionalmente incorporato su una banda magnetica e utilizzato per convalidare l'autenticità di una transazione. L'algoritmo viene utilizzato anche per altri scopi come iCVV, CAVV, CVV2. Potrebbe non essere incorporato in questo modo per altri casi d'uso.

CVV2

Il valore 2 di verifica della carta è un valore segreto statico che veniva tradizionalmente stampato sul fronte (o sul retro) di una carta di pagamento e viene utilizzato per verificare l'autenticità per i pagamenti non presenti sulla carta (ad esempio al telefono o online). Utilizza lo stesso algoritmo del CVV ma il codice di servizio è impostato su 000.

iCVV

iCVV è un CVV2-like valore ma è incorporato nei dati equivalenti a track2 su una scheda EMV (Chip). Questo valore viene calcolato utilizzando un codice di servizio 999 ed è diverso da quello utilizzato CVV1/CVV2 per evitare che le informazioni rubate vengano utilizzate per creare nuove credenziali di pagamento di tipo diverso. Ad esempio, se sono stati ottenuti dati di transazione con chip, non è possibile utilizzare questi dati per generare una banda magnetica (CVV1) o per acquisti online (CVV2).

Utilizza una chiave ???

DISCARICA

Derived Unique Key Per Transaction (DUKPT) è uno standard di gestione delle chiavi tipicamente utilizzato per definire l'uso di chiavi di crittografia monouso su dispositivi fisici. POS/POI Storicamente, DUKPT utilizza 3DES per la crittografia. Lo standard di settore per DUKPT è definito in ANSI. X9.24-3-2017

ECC

ECC (Elliptic Curve Cryptography) è un sistema di crittografia a chiave pubblica che utilizza la matematica delle curve ellittiche per creare chiavi di crittografia. ECC offre lo stesso livello di sicurezza dei metodi tradizionali come RSA ma con chiavi di lunghezza molto inferiore, garantendo una sicurezza equivalente in modo più efficiente. Ciò è particolarmente importante nei casi d'uso in cui RSA non è una soluzione pratica (lunghezza della chiave RSA > 4096 bit). AWS La crittografia dei pagamenti supporta le curve definite dal NIST per l'uso nelle operazioni ECDH.

ECDH

L'ECDH (Elliptic Curve Diffie-Hellman) è un protocollo di accordo chiave che consente a due parti di stabilire un segreto condiviso (come un KEK o un PEK). Nell'ECDH, le Parti A e B dispongono ciascuna delle proprie coppie di chiavi pubbliche-private e si scambiano chiavi pubbliche (sotto forma di certificati per la crittografia dei AWS pagamenti) nonché metadati di derivazione delle chiavi (metodo di derivazione, tipo di hash e informazioni condivise). Entrambe le parti moltiplicano la propria chiave privata per la chiave pubblica dell'altra e, grazie alle proprietà della curva ellittica, entrambe le parti sono in grado di derivare (generare) la chiave risultante.

EMV

EMV (originariamente Europay, Mastercard, Visa) è un organismo tecnico che collabora con le parti interessate ai pagamenti per creare standard e tecnologie di pagamento interoperabili. Un esempio di standard riguarda chip/contactless le carte e i terminali di pagamento con cui interagiscono, inclusa la crittografia utilizzata. La derivazione delle chiavi EMV si riferisce ai metodi di generazione di chiavi univoche per ogni carta di pagamento sulla base di un set iniziale di chiavi, ad esempio `IMK`

HSM (HSM)

Un Hardware Security Module (HSM) è un dispositivo fisico che protegge le operazioni crittografiche (ad esempio, crittografia, decrittografia e firme digitali) nonché le chiavi sottostanti utilizzate per tali operazioni.

KCAAS

Un Key Custodian As A Service (KCAAS) fornisce una varietà di servizi relativi alla gestione delle chiavi. Per quanto riguarda le chiavi di pagamento, in genere possono convertire i componenti chiave cartacei in moduli elettronici supportati dalla crittografia dei AWS pagamenti o convertire le chiavi protette elettronicamente in componenti cartacei che potrebbero essere richiesti da determinati fornitori. Possono inoltre fornire servizi di deposito a garanzia di chiavi la cui perdita comprometterebbe le operazioni in corso. I fornitori KCAAS sono in grado di aiutare i clienti a scaricare l'onere operativo della gestione del materiale chiave al di fuori di un servizio sicuro come AWS Payment Cryptography in modo conforme agli standard PCI DSS, PCI PIN e PCI P2PE. AWS Payment Cryptography offre Scambio di chiavi fisiche una funzionalità KCAAS integrata per convertire i componenti chiave cartacei in formato elettronico.

KCV

Key Check Value (KCV) si riferisce a una varietà di metodi di checksum utilizzati principalmente per confrontare le chiavi tra loro senza avere accesso al materiale chiave effettivo. I KCV sono

stati utilizzati anche per la convalida dell'integrità (specialmente durante lo scambio di chiavi), sebbene questo ruolo sia ora incluso come parte di formati di blocchi chiave come. TR-31 Per le chiavi TDES, il KCV viene calcolato crittografando 8 byte, ciascuno con valore zero, con la chiave da controllare e conservando i 3 byte di ordine più alto del risultato crittografato. Per le chiavi AES, il KCV viene calcolato utilizzando un algoritmo CMAC in cui i dati di input sono pari a 16 byte pari a zero e conservano i 3 byte di ordine più alto del risultato crittografato. Per le chiavi HMAC, il KCV viene calcolato utilizzando l'hash selezionato al momento della creazione della chiave su un messaggio di lunghezza zero, utilizzando i 3 byte più a sinistra.

KDH

Un Key Distribution Host (KDH) è un dispositivo o un sistema che invia chiavi in un processo di scambio di chiavi come. TR-34 Quando si inviano chiavi da AWS Payment Cryptography, viene considerato KDH.

KIF

Un Key Injection Facility (KIF) è una struttura sicura utilizzata per inizializzare i terminali di pagamento, incluso il loro caricamento con chiavi di crittografia.

KRD

Un dispositivo di ricezione delle chiavi (KRD) è un dispositivo che riceve chiavi in un processo di scambio di chiavi come. TR-34 Quando si inviano chiavi a AWS Payment Cryptography, questa viene considerata KRD.

KSN

Un Key Serial Number (KSN) è un valore utilizzato come input per DUKPT per creare chiavi di crittografia encryption/decryption univoche per transazione. Il KSN è in genere costituito da un identificatore BDK, un ID terminale semi-univoco e un contatore di transazioni che incrementa ogni transizione elaborata su un determinato terminale di pagamento. Per TDES X9.24, il KSN a 10 byte è in genere composto da 24 bit per l'ID del set di chiavi, 19 bit per l'ID del terminale e 21 bit per il contatore delle transazioni, sebbene il confine tra Key Set ID e ID terminale non abbia alcun impatto sulla funzione della crittografia dei AWS pagamenti. Per AES, il KSN a 12 byte è in genere composto da 32 bit per l'ID BDK, 32 bit per l'identificatore di derivazione (ID) e 32 bit per il contatore delle transazioni.

mPOC

mPOC (Mobile Point of Sale on Commercial hardware) è uno standard PCI che soddisfa i requisiti di sicurezza per le soluzioni che consentono ai commercianti di accettare PIN dei titolari di carta o

pagamenti senza contatto utilizzando uno smartphone o altri dispositivi mobili commerciali pronti all'uso (COTS).

PAN

Un numero di conto primario (PAN) è un identificatore univoco per un account, ad esempio una carta di credito o di debito. In genere, la lunghezza è di 13-19 cifre. Le prime 6-8 cifre identificano la rete e la banca emittente.

Blocco PIN

Un blocco di dati contenente un PIN durante l'elaborazione o la trasmissione e altri elementi di dati. I formati dei blocchi PIN standardizzano il contenuto del blocco PIN e il modo in cui può essere elaborato per recuperare il PIN. La maggior parte dei blocchi PIN è composta dal PIN e dalla lunghezza del PIN e spesso contiene parte o tutto il PAN. AWS Payment Cryptography supporta i formati ISO 9564-1 0, 1, 3 e 4. Il formato 4 è richiesto per le chiavi AES. Durante la verifica o la traduzione dei PIN, è necessario specificare il blocco PIN dei dati in entrata o in uscita.

POI

Point of Interaction (POI), spesso utilizzato anche in forma anonima con Point of Sale (POS), è il dispositivo hardware con cui il titolare della carta interagisce per presentare le proprie credenziali di pagamento. Un esempio di POI è il terminale fisico in una sede commerciale. Per l'elenco dei terminali POI PCI PTS certificati, consulta il sito Web PCI.

PSN

Il PAN Sequence Number (PSN) è un valore numerico utilizzato per differenziare più carte emesse con lo stesso PAN.

Chiavi pubbliche

Quando si utilizzano cifrari asimmetrici (RSA, ECC), la chiave pubblica è il componente pubblico di una coppia di chiavi pubblica-privata. La chiave pubblica può essere condivisa e distribuita alle entità che devono crittografare i dati per il proprietario della coppia di chiavi pubblica-privata. Per le operazioni di firma digitale, la chiave pubblica viene utilizzata per verificare la firma.

Chiave privata

Quando si utilizzano cifrari asimmetrici (RSA, ECC), la chiave privata è il componente privato di una coppia di chiavi pubblica-privata. La chiave privata viene utilizzata per decrittografare i dati o creare firme digitali. Analogamente alle chiavi simmetriche di crittografia dei AWS

pagamenti, le chiavi private vengono create in modo sicuro dagli HSM. Vengono decrittografate solo nella memoria volatile dell'HSM e solo per il tempo necessario all'elaborazione della richiesta crittografica.

PVV

Un valore di verifica del PIN (PVV) è un tipo di output crittografico che può essere utilizzato per verificare un pin senza memorizzare il pin effettivo. Sebbene sia un termine generico, nel contesto della crittografia dei AWS pagamenti, PVV si riferisce al metodo Visa o ABA PVV. Questo PVV è un numero a quattro cifre i cui input sono il numero della carta, il numero di sequenza pan, il pan stesso e una chiave di verifica del PIN. Durante la fase di convalida, AWS Payment Cryptography ricrea internamente il PVV utilizzando i dati della transazione e lo confronta nuovamente con il valore memorizzato dal cliente Payment Cryptography. AWS In questo modo, è concettualmente simile a un hash crittografico o MAC.

RSA Wrap/Unwrap

RSA wrap utilizza una chiave asimmetrica per avvolgere una chiave simmetrica (ad esempio una chiave TDES) per la trasmissione a un altro sistema. Solo il sistema con la chiave privata corrispondente può decrittografare il payload e caricare la chiave simmetrica. Al contrario, RSA unwrap decripterà in modo sicuro una chiave crittografata utilizzando RSA e quindi caricherà la chiave nella crittografia di pagamento. AWS RSA wrap è un metodo di scambio di chiavi di basso livello che non trasmette chiavi in formato di blocco di chiavi e non utilizza la firma del payload da parte della parte mittente. È necessario prendere in considerazione controlli alternativi per accertare la provvidenza e che gli attributi chiave non siano mutati.

TR-34 utilizza anche RSA internamente, ma è un formato separato e non è interoperabile.

Codice di servizio

Un codice di servizio è un valore a tre cifre codificato sulla banda magnetica e sul chip di una carta di pagamento che definisce gli attributi della carta, ad esempio dove la carta può essere utilizzata e se è necessaria un'autorizzazione aggiuntiva. Il codice di servizio viene utilizzato come input per il calcolo dei valori di verifica. I diversi valori del codice di servizio producono diversi tipi di valori di verifica, ad esempio CVV (utilizzando il codice di servizio della banda magnetica della scheda), iCVV (codice di servizio 999 per le schede con chip) e CVV2 (codice di servizio 000).

TR-31

TR-31 (definito formalmente come ANSI X9 TR 31) è un formato di blocco chiave definito dall'American National Standards Institute (ANSI) per supportare la definizione degli attributi chiave nella stessa struttura di dati dei dati chiave stessi. Il formato del blocco TR-31 chiave

definisce un insieme di attributi chiave collegati alla chiave in modo che siano tenuti insieme. AWS La crittografia dei pagamenti utilizza termini TR-31 standardizzati laddove possibile per garantire la corretta separazione delle chiavi e lo scopo delle chiavi. TR-31 è stata sostituita dall'ANSI. X9.143-2022

TR-34

TR-34 è un'implementazione di ANSI X9.24-2 che descriveva un protocollo per distribuire in modo sicuro chiavi simmetriche (come 3DES e AES) utilizzando tecniche asimmetriche (come RSA). AWS La crittografia dei pagamenti utilizza TR-34 metodi per consentire l'importazione e l'esportazione sicure delle chiavi.

X9.143

X9.143 è un formato di blocco chiave definito dall'American National Standards Institute (ANSI) per supportare la protezione di una chiave e degli attributi chiave nella stessa struttura di dati. Il formato del blocco chiave definisce un insieme di attributi chiave collegati alla chiave in modo che siano tenuti insieme. AWS La crittografia dei pagamenti utilizza termini X9.143 standardizzati laddove possibile per garantire la corretta separazione delle chiavi e lo scopo delle chiavi. X9.143 sostituisce la TR-31proposta precedente, sebbene nella maggior parte dei casi siano compatibili con le versioni precedenti e future e i termini siano spesso usati in modo intercambiabile.

Servizi correlati

AWS Key Management Service

AWS Key Management Service (AWS KMS) è un servizio gestito che semplifica la creazione e il controllo delle chiavi crittografiche utilizzate per proteggere i dati. AWS KMS utilizza moduli di sicurezza hardware (HSMs) per proteggere e convalidare le chiavi KMS. AWS

AWS CloudHSM

AWS CloudHSM fornisce ai clienti istanze HSM dedicate per uso generico nel cloud. AWS AWS CloudHSM può fornire una varietà di funzioni crittografiche come la creazione di chiavi, la firma dei dati o la crittografia e la decrittografia dei dati.

Ulteriori informazioni

- Per ulteriori informazioni sui termini e i concetti utilizzati nella crittografia dei pagamenti, consulta AWS Payment Cryptography Concepts.AWS

- Per informazioni sull'API AWS Payment Cryptography Control Plane, consulta AWS Payment Cryptography Control Plane API Reference.
- Per informazioni sull'API AWS Payment Cryptography Data Plane, consulta AWS Payment Cryptography Data Plane API Reference.
- Per informazioni tecniche dettagliate su come AWS Payment Cryptography utilizza la crittografia e protegge le chiavi di crittografia dei AWS pagamenti, consulta Dettagli crittografici.

Endpoint per AWS Payment Cryptography

Per connetterti a livello di codice AWS Payment Cryptography, usi un endpoint, l'URL del punto di ingresso del servizio. Gli AWS SDK e gli strumenti della riga di comando utilizzano automaticamente l'endpoint predefinito per il servizio in Regione AWS base al contesto regionale di una richiesta, quindi in genere non è necessario impostare in modo esplicito questi valori. Se necessario, puoi specificare un endpoint diverso per le tue richieste API.

Endpoint del piano di controllo

Nome della regione	Regione	Endpoint	Protocollo	
US East (Ohio)	us-east-2	controlplane.payment-cryptography.us-east-2.amazonaws.com controlplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
US East (N. Virginia)	us-east-1	controlplane.payment-cryptography.us-east-1.amazonaws.com controlplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
Stati Uniti occidentali (Oregon)	us-west-2	controlplane.payment-cryptography.us-west-2.amazonaws.com	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
		controlplane.payment-cryptography.us-west-2.api.aws		
Africa (Città del Capo)	af-south-1	controlplane.payment-cryptography.af-south-1.amazonaws.com controlplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Asia Pacific (Hyderabad)	ap-south-2	controlplane.payment-cryptography.ap-south-2.amazonaws.com controlplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS	
Asia Pacifico (Mumbai)	ap-south-1	controlplane.payment-cryptography.ap-south-1.amazonaws.com controlplane.payment-cryptography.ap-south-1.api.aws	HTTPS HTTPS	
Asia Pacifico (Osaka-Locale)	ap-northeast-3	controlplane.payment-cryptography.ap-northeast-3.amazonaws.com controlplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS	
Asia Pacifico (Singapore)	ap-southeast-1	controlplane.payment-cryptography.ap-southeast-1.amazonaws.com controlplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Asia Pacifico (Sydney)	ap-southeast-2	controlplane.payment-cryptography.ap-southeast-2.amazonaws.com controlplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS	
Asia Pacifico (Tokyo)	ap-northeast-1	controlplane.payment-cryptography.ap-northeast-1.amazonaws.com controlplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS	
Canada (Centrale)	ca-central-1	controlplane.payment-cryptography.ca-central-1.amazonaws.com controlplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS	
Europa (Francoforte)	eu-central-1	controlplane.payment-cryptography.eu-central-1.amazonaws.com controlplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS	
Europa (Irlanda)	eu-west-1	controlplane.payment-cryptography.eu-west-1.amazonaws.com controlplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS	
Europa (Londra)	eu-west-2	controlplane.payment-cryptography.eu-west-2.amazonaws.com controlplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Europa (Parigi)	eu-west-3	controlplane.payment-cryptography.eu-west-3.amazonaws.com controlplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS	
Sud America (São Paulo)	sa-east-1	controlplane.payment-cryptography.sa-east-1.amazonaws.com controlplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS	

Endpoint del piano dati

Nome della regione	Regione	Endpoint	Protocollo	
US East (Ohio)	us-east-2	dataplane.payment-cryptography.us-east-2.amazonaws.com dataplane.payment-cryptography.us-east-2.api.aws	HTTPS HTTPS	
US East (N. Virginia)	us-east-1	dataplane.payment-cryptography.us-east-1.amazonaws.com dataplane.payment-cryptography.us-east-1.api.aws	HTTPS HTTPS	
Stati Uniti occidentali (Oregon)	us-west-2	dataplane.payment-cryptography.us-west-2.amazonaws.com	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
		dataplane.payment-cryptography.us-west-2.api.aws		
Africa (Città del Capo)	af-south-1	dataplane.payment-cryptography.af-south-1.amazonaws.com dataplane.payment-cryptography.af-south-1.api.aws	HTTPS HTTPS	
Asia Pacific (Hyderabad)	ap-south-2	dataplane.payment-cryptography.ap-south-2.amazonaws.com dataplane.payment-cryptography.ap-south-2.api.aws	HTTPS HTTPS	
Asia Pacifico (Mumbai)	ap-south-1	dataplane.payment-cryptography.ap-south-1.amazonaws.com dataplane.payment-cryptography.ap-south-1.api.aws	HTTPS HTTPS	
Asia Pacifico (Osaka-Locale)	ap-northeast-3	dataplane.payment-cryptography.ap-northeast-3.amazonaws.com dataplane.payment-cryptography.ap-northeast-3.api.aws	HTTPS HTTPS	
Asia Pacifico (Singapore)	ap-southeast-1	dataplane.payment-cryptography.ap-southeast-1.amazonaws.com dataplane.payment-cryptography.ap-southeast-1.api.aws	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Asia Pacifico (Sydney)	ap-southeast-2	dataplane.payment-cryptography.ap-southeast-2.amazonaws.com dataplane.payment-cryptography.ap-southeast-2.api.aws	HTTPS HTTPS	
Asia Pacifico (Tokyo)	ap-northeast-1	dataplane.payment-cryptography.ap-northeast-1.amazonaws.com dataplane.payment-cryptography.ap-northeast-1.api.aws	HTTPS HTTPS	
Canada (Centrale)	ca-central-1	dataplane.payment-cryptography.ca-central-1.amazonaws.com dataplane.payment-cryptography.ca-central-1.api.aws	HTTPS HTTPS	
Europa (Francoforte)	eu-central-1	dataplane.payment-cryptography.eu-central-1.amazonaws.com dataplane.payment-cryptography.eu-central-1.api.aws	HTTPS HTTPS	
Europa (Irlanda)	eu-west-1	dataplane.payment-cryptography.eu-west-1.amazonaws.com dataplane.payment-cryptography.eu-west-1.api.aws	HTTPS HTTPS	
Europa (Londra)	eu-west-2	dataplane.payment-cryptography.eu-west-2.amazonaws.com dataplane.payment-cryptography.eu-west-2.api.aws	HTTPS HTTPS	

Nome della regione	Regione	Endpoint	Protocollo	
Europa (Parigi)	eu-west-3	dataplane.payment-cryptography.eu-west-3.amazonaws.com dataplane.payment-cryptography.eu-west-3.api.aws	HTTPS HTTPS	
Sud America (São Paulo)	sa-east-1	dataplane.payment-cryptography.sa-east-1.amazonaws.com dataplane.payment-cryptography.sa-east-1.api.aws	HTTPS HTTPS	

Guida introduttiva alla crittografia AWS dei pagamenti

Per iniziare con la crittografia dei AWS pagamenti, dovrai prima creare delle chiavi e poi utilizzarle in varie operazioni crittografiche. Il tutorial seguente fornisce un semplice esempio di generazione di una chiave da utilizzare per i generating/verifying CVV2 valori. Per provare altri esempi ed esplorare i modelli di distribuzione all'interno di AWS, prova il seguente Workshop sulla crittografia dei AWS pagamenti o esplora il nostro progetto di esempio disponibile su GitHub

Questo tutorial ti guida nella creazione di una singola chiave e nell'esecuzione di operazioni crittografiche utilizzando la chiave. Successivamente, elimini la chiave se non la desideri più, il che completa il ciclo di vita della chiave.

> **Warning**
>
> Gli esempi contenuti in questa guida per l'utente possono utilizzare valori di esempio. Si consiglia vivamente di non utilizzare valori di esempio in un ambiente di produzione, come i numeri di serie delle chiavi.

Argomenti

- Prerequisiti
- Fase 1: Creare una chiave
- Passaggio 2: generare un CVV2 valore utilizzando la chiave
- Fase 3: Verificare il valore generato nel passaggio 2
- Fase 4: Eseguire un test negativo
- Fase 5: (Facoltativo) Pulizia

Prerequisiti

Prima di iniziare, assicurati di quanto segue:

- Hai il permesso di accedere al servizio. Per ulteriori informazioni, consulta le politiche IAM.
- Hai AWS CLIinstallato il. Puoi anche utilizzare AWS SDKso accedere AWS APIsalla crittografia dei AWS pagamenti, ma le istruzioni di questo tutorial utilizzano la AWS CLI.

Fase 1: Creare una chiave

Il primo passo è creare una chiave. Per questo tutorial, crei una chiave CVK 3DES (2KEY TDES) a doppia lunghezza per generare e verificare i valori CVV/. CVV2

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

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (KCV).

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
tqv5yij6wtxx64pi",
        "KeyAttributes": {
            "KeyUsage": "TR31_C0_CARD_VERIFICATION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_2KEY",
            "KeyModesOfUse": {
                "Encrypt": false,
                "Decrypt": false,
                "Wrap": false,
                "Unwrap": false,
                "Generate": true,
                "Sign": false,
                "Verify": true,
                "DeriveKey": false,
                "NoRestrictions": false
            }
        },
        "KeyCheckValue": "CADDA1",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
        "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
        "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
    }
}
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/tqv5yij6wtxx64pi. Ne hai bisogno nel passaggio successivo.

Passaggio 2: generare un CVV2 valore utilizzando la chiave

In questo passaggio, si genera un CVV2 valore per una determinata PAN data di scadenza utilizzando la chiave del passaggio 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"
}
```

Prendi nota del numero `cardDataValue` 144, in questo caso a 3 cifre. Ne hai bisogno nella fase successiva.

Fase 3: Verificare il valore generato nel passaggio 2

In questo esempio, convalidi il CVV2 risultato del passaggio 2 utilizzando la chiave creata nel passaggio 1.

Eseguite il comando seguente per convalidare il. 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"
}
```

Il servizio restituisce una risposta HTTP di 200 per indicare che ha convalidato il. CVV2

Fase 4: Eseguire un test negativo

In questo passaggio, si crea un test negativo in cui non CVV2 è corretto e non viene convalidato. Si tenta di convalidare un errore CVV2 utilizzando la chiave creata nel passaggio 1. Si tratta di un'operazione prevista, ad esempio se il titolare della carta ha inserito un codice errato CVV2 al momento del pagamento.

```
$  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.
```

Il servizio restituisce una risposta HTTP di 400 con il messaggio «Verifica dei dati di convalida della carta non riuscita» e un motivo di INVALID_VALIDATION_DATA.

Fase 5: (Facoltativo) Pulizia

Ora puoi eliminare la chiave che hai creato nel passaggio 1. Per ridurre al minimo le modifiche irrecuperabili, il periodo di eliminazione della chiave predefinito è di sette giorni.

```
$ 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"
    }
}
```

Prendi nota di due campi nell'output. Per impostazione predefinita, `deletePendingTimestamp` è impostato su sette giorni nel futuro. KeyState è impostato su. `DELETE_PENDING` È possibile annullare questa eliminazione in qualsiasi momento prima dell'orario di eliminazione pianificato `restore-key`chiamando.

Gestione delle chiavi

Per iniziare con la crittografia dei AWS pagamenti, crea una chiave di crittografia dei AWS pagamenti.

Questa sezione spiega come creare e gestire vari tipi di chiavi di crittografia dei AWS pagamenti durante il loro ciclo di vita. Imparerai come creare, visualizzare e modificare le chiavi, nonché come etichettare le chiavi, creare alias di chiave e abilitare o disabilitare le chiavi.

Una chiave AWS di crittografia dei pagamenti è una risorsa regionale. Se intendi utilizzare una determinata chiave in più chiavi Regioni AWS, puoi abilitare la replica delle Multi-Region chiavi che copia in modo sicuro il materiale e i metadati della chiave specificati all'interno della Regioni AWS stessa partizione e account. AWS La chiave di origine nella replica delle chiavi è nota come Multi-Region chiave regionale primaria (PRK) e rimane la fonte autorevole per tutte le attività di gestione delle chiavi. La chiave replicata è nota come chiave della regione di replica (RRK) e si tratta di una replica di sola lettura del PRK. È consigliabile utilizzare Multi-Region le chiavi insieme alle chiavi per raggiungere gli obiettivi di progettazione relativi a disponibilità, disaster recovery e bassa latenza.

Argomenti

- Creazione di chiavi
- Elenco delle chiavi
- Abilitazione e disabilitazione delle chiavi
- Replica AWS Chiavi di crittografia dei pagamenti
- Eliminazione delle chiavi
- Importazione ed esportazione di chiavi
- Utilizzo di alias
- Ottieni le chiavi
- Chiavi di tagging
- Comprendere gli attributi chiave per AWS Chiave di crittografia dei pagamenti

Creazione di chiavi

È possibile creare chiavi AWS di crittografia dei pagamenti utilizzando l'operazione CreateKey API. Quando crei una chiave, specifichi attributi come l'algoritmo della chiave, l'utilizzo della chiave, le

operazioni consentite e se è esportabile. Non puoi modificare queste proprietà dopo aver creato la chiave AWS di crittografia dei pagamenti.

> **Note**
>
> Se Multi-Region la replica delle chiavi è abilitata Account AWS e crei una chiave di crittografia di pagamento, questa chiave diventerà automaticamente una chiave regionale primaria (PRK). PRK viene replicato anche se non si specifica il parametro nel `--replication-regions` comando. CreateKey Per ulteriori informazioni, consulta Come funziona la replica delle chiavi Multi-Region.

Esempi

- Creazione di una chiave di derivazione di base TDES 2KEY
- Creazione di una chiave TDES 2KEY per CVV/CVV2
- Creazione di una chiave HMAC
- Creazione di una chiave AES-256
- Creazione di una chiave di crittografia PIN (PEK)
- Creazione di una chiave asimmetrica (RSA)
- Creazione di una chiave PIN Verification Value (PVV)
- Creazione di una chiave ECC asimmetrica

Creazione di una chiave di derivazione di base TDES 2KEY

> **Note**
>
> Sebbene 3KEY TDES sia supportato per KeyUsage = TR31_B0_BASE_DERIVATION_KEY, nessuna operazione attualmente supporta questo tipo di chiave. Lo standard ANSI (DUKPT) definisce il supporto solo per i tipi di chiave TDES_2KEY X9.24 e AES. ???

Example

Questo comando crea una chiave di derivazione TDES 2KEY per DUKPT che verrà replicata dalle regioni degli Stati Uniti occidentali (Oregon) a quelle degli Stati Uniti orientali (Ohio). La risposta

include i parametri della richiesta, un Amazon Resource Name (ARN) per le chiamate successive e un Key Check Value (KCV).

```
$ aws payment-cryptography create-key --exportable --key-attributes \
     "KeyUsage=TR31_B0_BASE_DERIVATION_KEY, \
     KeyClass=SYMMETRIC_KEY,KeyAlgorithm=TDES_2KEY, \
     KeyModesOfUse={DeriveKey=true}" \
     --replication-regions us-east-2 --region us-west-2
```

Output di esempio:

```
{
    "Key": {
        "CreateTimestamp": "2022-10-26T16:04:11.642000-07:00",
        "Enabled": true,
        "Exportable": true,
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/5rplquuwozodpwsp",
        "KeyAttributes": {
            "KeyAlgorithm": "TDES_2KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyModesOfUse": {
                "Decrypt": false,
                "DeriveKey": true,
                "Encrypt": false,
                "Generate": false,
                "NoRestrictions": false,
                "Sign": false,
                "Unwrap": false,
                "Verify": false,
                "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"
}
```

Creazione di una chiave TDES 2KEY per CVV/CVV2

Example

Questo comando crea una chiave TDES 2KEY per la generazione e la verifica dei valori. CVV/CVV2 La risposta include i parametri della richiesta, un Amazon Resource Name (ARN) per le chiamate successive e un Key Check Value (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}'
```

Output di esempio:

```
{
    "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"
    }
}
```

Creazione di una chiave HMAC

Example

Le chiavi HMAC vengono utilizzate per generare o verificare i codici di autenticazione dei messaggi hash (HMAC). Con le chiavi HMAC, il tipo di hash viene assegnato al momento della creazione della chiave (come HMAC_SHA224 e HMAC_SHA512) e non può essere modificato.

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

Output di esempio:

```
{
 "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"
 }
}
```

Creazione di una chiave AES-256

Example

Questo comando crea una chiave AES-256 simmetrica per la crittografia e la decrittografia dei dati. Le chiavi AES forniscono una crittografia avanzata per i dati sensibili e sono comunemente utilizzate nell'elaborazione dei pagamenti per crittografare i dati dei titolari di carte e altre informazioni sensibili, tuttavia TDES è più comunemente usato per casi d'uso degli emittenti come EMV.

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

Output di esempio:

```
{
    "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"
    }
}
```

Creazione di una chiave di crittografia PIN (PEK)

Example

Questo comando crea una chiave TDES 3KEY per la crittografia dei valori PIN, sebbene le chiavi pin possano anche essere AES a seconda delle esigenze di interoperabilità. È possibile utilizzare questa chiave per archiviare in modo sicuro i PIN o decrittografare i PIN durante la verifica, ad esempio in una transazione. La risposta include i parametri della richiesta, un ARN per le chiamate successive e un 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}'
```

Output di esempio:

```
{
    "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"
    }
}
```

Creazione di una chiave asimmetrica (RSA)

Example

Questo comando genera una nuova coppia di chiavi RSA asimmetrica a 2048 bit. Crea una nuova chiave privata e la chiave pubblica corrispondente. Puoi recuperare la chiave pubblica utilizzando l'PublicCertificateAPI get.

```
$ 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}'
```

Output di esempio:

```
{
    "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"
    }
}
```

Creazione di una chiave PIN Verification Value (PVV)

Example

Questo comando crea una chiave TDES 3KEY per la generazione di valori PVV. È possibile utilizzare questa chiave per generare un PVV che può essere confrontato con un PVV calcolato successivamente. La risposta include i parametri della richiesta, un ARN per le chiamate successive e un 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}'
```

Output di esempio:

```
{
    "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"
    }
}
```

Creazione di una chiave ECC asimmetrica

Esempio

Questo comando genera una coppia di chiavi ECC per stabilire un accordo chiave ECDH (Elliptic Curve Diffie-Hellman) tra due parti. Con ECDH, ciascuna parte genera la propria coppia di chiavi ECC con lo scopo chiave K3 e la modalità d'uso X e si scambia le chiavi pubbliche. Entrambe le parti utilizzano quindi la propria chiave privata e la chiave pubblica ricevuta per stabilire una chiave derivata condivisa.

Per mantenere il principio delle chiavi crittografiche monouso nei pagamenti, consigliamo di non riutilizzare le coppie di chiavi ECC per molteplici scopi, come la derivazione e la firma delle chiavi 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}'
```

Output di esempio:

```
{
    "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"
    }
}
```

Elenco delle chiavi

Usa l'ListKeysoperazione per ottenere un elenco di chiavi accessibili nel tuo account e nella tua regione.

Example

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

Output di esempio:

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

Abilitazione e disabilitazione delle chiavi

Puoi disattivare e riattivare le chiavi AWS di crittografia dei pagamenti. Quando si crea una chiave, questa è abilitata per impostazione predefinita. Se disabiliti una chiave, non può essere utilizzata in alcuna operazione di crittografia finché non la riattivi. Start/stop i comandi di utilizzo hanno effetto immediato, pertanto è consigliabile verificarne l'utilizzo prima di apportare tali modifiche. Puoi anche impostare una modifica (avvio o interruzione dell'utilizzo) in modo che abbia effetto in futuro utilizzando il `timestamp` parametro opzionale.

Poiché è temporanea e può essere annullata facilmente, la disattivazione di una chiave crittografica AWS di pagamento è un'alternativa più sicura all'eliminazione di una chiave di crittografia di AWS pagamento, un'azione distruttiva e irreversibile. Se stai pensando di eliminare una chiave crittografica di AWS pagamento, disattivala prima e assicurati di non dover utilizzare la chiave per crittografare o decrittografare i dati in futuro.

Argomenti

- Inizia a utilizzare la chiave
- Interrompi l'uso delle chiavi

Inizia a utilizzare la chiave

L'utilizzo della chiave deve essere abilitato per poter utilizzare una chiave per le operazioni crittografiche. Se una chiave non è abilitata, è possibile utilizzare questa operazione per renderla utilizzabile. Il campo `UsageStartTimestamp` rappresenterà quando la chiave became/will diventa attiva. Ciò avverrà in passato per un token abilitato e in futuro se in attesa di attivazione.

Example

In questo esempio, viene richiesta l'attivazione di una chiave per l'utilizzo della chiave. La risposta include le informazioni chiave e il flag enable è stato impostato su true. Ciò si rifletterà anche nell'oggetto di risposta 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"
  }
}
```

Interrompi l'uso delle chiavi

Se non intendi più utilizzare una chiave, puoi interromperne l'utilizzo per impedire ulteriori operazioni crittografiche. Questa operazione non è permanente, quindi puoi annullarla avviando l'utilizzo della chiave. Puoi anche impostare una chiave da disattivare in futuro. Il campo `UsageStopTimestamp` rappresenterà il momento in cui la chiave viene became/will disattivata.

Example

In questo esempio, viene richiesto di interrompere l'utilizzo della chiave in futuro. Dopo l'esecuzione, questa chiave non può essere utilizzata per operazioni crittografiche a meno che non venga riattivata tramite l'utilizzo della chiave di avvio. La risposta include le informazioni sulla chiave e il flag di abilitazione è stato impostato su false. Ciò si rifletterà anche nell'oggetto di risposta 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"
  }
}
```

Replica AWS Chiavi di crittografia dei pagamenti

AWS La crittografia dei pagamenti supporta Multi-Region la replica delle chiavi, consentendo di distribuire in modo sicuro il materiale e i metadati chiave da una determinata chiave di crittografia di AWS pagamento a una o più Regioni AWS chiavi all'interno della stessa partizione e account. AWS

La chiave sorgente è nota come chiave della regione primaria (PRK) e rimane la fonte autorevole per tutte le attività di gestione delle chiavi, mentre sia la chiave PRK che la chiave Replica Region (RRK) possono essere utilizzate per le rispettive operazioni crittografiche. Regioni AWS

Vantaggi della replica delle chiavi Multi-Region

Di seguito vengono illustrati alcuni vantaggi della replica delle Multi-Region chiavi.

- Configurazione più semplice per applicazioni ad alta disponibilità: AWS Payment Cryptography gestisce la distribuzione delle chiavi per consentirti di utilizzare una chiave in più chiavi Regioni AWS senza dover creare copie disaccoppiate di una determinata chiave.
- Chiavi ad alta disponibilità e bassa latenza: con Multi-Region la replica delle chiavi, puoi accedere alle tue chiavi in modo multiplo, Regioni AWS rendendole altamente disponibili e con conseguente latenza inferiore.
- Durata del materiale chiave: le chiavi Replica Region sono repliche di chiavi complete e possono essere utilizzate indipendentemente dalla loro chiave regionale primaria nelle operazioni crittografiche. Un RRK fornisce una replica duratura in caso di perdita catastrofica dei dati di una PRK.

Come funziona la replica delle chiavi Multi-Region

Quando Multi-Region la replica delle chiavi è abilitata, il servizio AWS Payment Cryptography utilizza meccanismi di distribuzione sicuri delle chiavi per copiare il materiale e i metadati delle chiavi nella replica specificata. Regioni AWS Le modifiche ai metadati chiave di una regione primaria, come gli attributi della chiave, lo stato e l'abilitazione, vengono replicate automaticamente nelle chiavi della regione di replica.

Considerazioni e limitazioni

Di seguito sono riportate alcune limitazioni e considerazioni Multi-Region chiave sulla replica.

- Le chiavi di crittografia dei pagamenti devono essere impostate come esportabili per utilizzare la replica delle chiavi. Multi-Region
- È necessario abilitare questa funzionalità per una Regione AWS o per specifiche chiavi di crittografia dei pagamenti.
  - Se questa funzionalità è abilitata per un Regione AWS, tutte le chiavi AWS di crittografia dei pagamenti create dopo l'attivazione verranno replicate nel formato specificato. Regione AWS Le chiavi create in questa regione diventeranno le chiavi della regione primaria. Le chiavi esistenti in questa regione non verranno replicate automaticamente. È possibile abilitare Multi-Region la replica delle chiavi esistenti all'interno e Regione AWS a livello di chiave.
  - Ciascuna Regione AWS può avere impostazioni di replica Multi-Region delle chiavi uniche.
  - Le impostazioni di Multi-Region replica di una chiave hanno la precedenza sull'impostazione di replica della Regione AWS Multi-Region chiave.
- Una chiave della regione di replica non può essere configurata per la replica su un'altra. Regioni AWS
- Multi-Region la replica delle chiavi è disponibile per chiavi di crittografia di pagamento simmetriche come Triple Data Encryption Standard (3DES), Advanced Encryption Standard (AES) e Hash-based Message Authentication Code (HMAC), nonché per chiavi di crittografia di pagamento asimmetriche come RSA e Elliptic Curve Cryptography (ECC).
- Le chiavi Replica Region sono chiavi di sola lettura. Tutte le modifiche alla chiave della regione primaria verranno applicate alle chiavi della regione di replica.
- Le modifiche alle chiavi della regione primaria sono infine coerenti con le chiavi della regione di replica.
- Le chiavi di crittografia dei pagamenti possono essere replicate solo con la stessa AWS partizione e lo stesso account.
- Le chiavi della regione di replica vengono conteggiate ai fini del limite di crittografia dei AWS pagamenti Account AWS del tuo livello.
- La chiave della regione primaria e la chiave della regione di replica utilizzano lo stesso identificatore di chiave che consente di fare riferimento a entrambe le chiavi tramite lo stesso ARN nelle politiche IAM.
- È necessario disporre `CreateKey` delle autorizzazioni nella replica affinché la replica abbia Regione AWS esito positivo.

Abilitazione della replica delle chiavi Multi-Region

È possibile abilitare la replica delle chiavi per le Multi-Region chiavi di crittografia dei AWS pagamenti in due modi.

1. Regione AWS: Multi-Region la replica delle chiavi viene applicata a tutte le nuove chiavi create in tale Regione AWS chiave quando abilitata. Questo metodo fornisce una replica coerente per tutte le chiavi.
2. Chiavi AWS di crittografia di pagamento specifiche: è possibile gestire Multi-Region la replica delle chiavi per singole chiavi, garantendo un livello di controllo più granulare.

Una volta abilitata la replica delle Multi-Region chiavi, le tue chiavi di crittografia dei pagamenti verranno replicate nella posizione specificata. Regioni AWS

> **Important**
>
> Multi-Region la replica delle chiavi non può essere messa in pausa. Le chiavi vengono replicate automaticamente nel modo specificato una volta Regioni AWS abilitata la replica. Multi-Region la replica delle chiavi può essere disabilitata per una chiave specifica Regione AWS o per una chiave di crittografia di pagamento. È necessario rimuovere la regione Regione AWS di replica dalla chiave della regione primaria per eliminare la chiave della regione di replica.
> In alternativa, puoi chiamare il comando `StopKeyUsage` API o stop-key-usage CLI sul tuo PRK per interrompere l'utilizzo sia del PRK che di tutti gli RRK associati. Non sarai in grado di utilizzare queste chiavi nelle operazioni crittografiche. L'uso del comando `StopKeyUsage` API o stop-key-usage CLI non interromperà la replica continua delle Multi-Region chiavi abilitata per il tuo PRK.

È possibile verificare Multi-Region le impostazioni di replica delle chiavi di AWS Payment Cryptography in una determinata area Regione AWS chiamando il comando API o CLI`GetDefaultKeyReplicationRegions`. get-default-key-replication-regions Le chiavi nel punto in Regione AWS cui chiami questa azione o comando API diventeranno il tuo PRK. ???

Utilizza le seguenti procedure per abilitare la replica Multi-Region delle chiavi.

For Regione AWS

- Utilizzate il comando seguente per abilitare Multi-Region la replica delle chiavi per una Regione AWS chiave specificata. In questo esempio, Multi-Region la replica delle chiavi è abilitata negli Stati Uniti orientali (Ohio) e negli Stati Uniti occidentali (Oregon). Per utilizzare questo comando, sostituisci il comando *italicized placeholder text* in the example con le tue informazioni.

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

> **Note**
>
> L'attivazione della replica delle Multi-Region chiavi per un non Regione AWS modificherà la configurazione di replica di alcuna chiave di crittografia di AWS pagamento esistente. È possibile abilitare questa funzionalità per le chiavi esistenti a livello di chiave. Solo le chiavi create dopo l' Multi-Region abilitazione della replica delle chiavi Regione AWS utilizzeranno le impostazioni di replica della regione.

For specific AWS Payment Cryptography keys

- Utilizza il seguente comando per abilitare la replica Multi-Region delle chiavi per specifiche chiavi di crittografia dei pagamenti. In questo esempio, la replica Multi-Region delle chiavi è abilitata negli Stati Uniti orientali (Ohio). Per utilizzare questo comando, sostituisci il comando *italicized placeholder text* nell'esempio con le tue informazioni.

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

In alternativa, puoi creare una nuova chiave di crittografia dei pagamenti con questa funzionalità abilitata includendo la replica Regioni AWS nella richiesta di creazione della chiave.

> **Note**
>
> Le impostazioni di replica delle chiavi hanno la precedenza sull'impostazione di replica. Regione AWS

Disattivazione della replica delle chiavi Multi-Region

Se desideri disabilitare la replica delle Multi-Region chiavi, puoi chiamare i comandi disable-default-key-replication o remove-key-replication-regions CLI, a seconda di come è abilitata la replica delle Multi-Region chiavi. Dovrai specificare l'ARN della chiave e disabilitare la replica della Regione AWS chiave. Multi-Region

Considerazioni

Le eliminazioni delle chiavi della regione di replica sono alla fine coerenti.

È possibile verificare Multi-Region le impostazioni di replica delle chiavi di AWS Payment Cryptography in una determinata area Regione AWS richiamando il comando API o CLI`GetDefaultKeyReplicationRegions`. get-default-key-replication-regions

Utilizza le seguenti procedure per disabilitare Multi-Region la replica delle chiavi.

For Regione AWS

- Utilizzate il comando seguente per disabilitare Multi-Region la replica delle chiavi per una Regione AWS chiave specificata. In questo esempio, Multi-Region la replica delle chiavi è disabilitata negli Stati Uniti orientali (Ohio). Per utilizzare questo comando, sostituisci il comando *`italicized placeholder text`* nell'esempio con le tue informazioni.

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

For specific AWS Payment Cryptography keys

- Usa il comando seguente per disabilitare Multi-Region la replica delle chiavi per una chiave di crittografia di pagamento specifica. In questo esempio, la replica Multi-Region delle chiavi è disabilitata negli Stati Uniti orientali (Ohio). Per utilizzare questo comando, sostituisci il comando *`italicized placeholder text`* nell'esempio con le tue informazioni.

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

Considerazioni relative alla sicurezza

Di seguito sono riportate le considerazioni di sicurezza relative all'utilizzo della replica delle Multi-Region chiavi di crittografia dei pagamenti. Per ulteriori informazioni, consulta Le migliori pratiche di sicurezza per AWS Crittografia dei pagamenti.

- Limita la condivisione dei materiali chiave.
- Segui il principio delle autorizzazioni con privilegi minimi quando crei le politiche IAM.
- Non è possibile apportare modifiche alla chiave Replica Region poiché è una chiave di sola lettura.

Best practice

Di seguito sono riportate alcune best practice per l'utilizzo della replica delle chiavi con le Multi-Region chiavi di AWS Payment Cryptography.

- Assicurati che l'applicazione continui a funzionare anche se la replica della Multi-Region chiave su quella specificata non Regione AWS è immediata. Se hai bisogno di sapere quando Multi-Region la replica delle chiavi è completa, puoi monitorarla con l'azione GetKey API. È possibile monitorare gli eventi di replica chiave con. AWS CloudTrail
- Testa e implementa processi di distribuzione automatizzati in caso di failover da una regione all'altra Regione AWS .

Prezzi

Ti verranno addebitati i costi delle chiavi Replica Region create con AWS Payment Cryptography. Queste chiavi vengono addebitate per. Regione AWS Per le informazioni più recenti sui prezzi di Payment Cryptography, consulta la pagina dei prezzi AWS di Payment Cryptography.

Eliminazione delle chiavi

L'eliminazione di una chiave crittografica AWS di pagamento elimina il materiale chiave e tutti i metadati associati alla chiave ed è irreversibile a meno che una copia della chiave non sia disponibile al di fuori di Payment Cryptography. AWS Dopo l'eliminazione di una chiave, non è più possibile decrittografare i dati crittografati con tale chiave, il che significa che i dati potrebbero diventare irrecuperabili. È necessario eliminare una chiave solo quando si è certi di non doverla più utilizzare e che nessun'altra parte la utilizza. Se non sei sicuro, valuta la possibilità di interrompere l'utilizzo della chiave anziché eliminarla. Puoi riattivare una chiave disattivata se hai bisogno di riutilizzarla in un secondo momento, ma non puoi recuperare una chiave di crittografia di AWS pagamento eliminata a meno che tu non sia in grado di reimportarla da un'altra fonte.

Prima di eliminare una chiave, assicurati di non averne più bisogno. AWS Payment Cryptography non memorizza i risultati di operazioni crittografiche come CVV2 e non è in grado di determinare se è necessaria una chiave per qualsiasi materiale crittografico persistente.

AWS Payment Cryptography non elimina mai le chiavi appartenenti agli AWS account attivi a meno che l'utente non ne pianifichi esplicitamente l'eliminazione e scada il periodo di attesa obbligatorio.

Tuttavia, potresti scegliere di eliminare una chiave di crittografia dei AWS pagamenti per uno o più dei seguenti motivi:

- Per completare il ciclo di vita di una chiave che non ti serve più
- Per evitare il sovraccarico di gestione associato al mantenimento delle chiavi crittografiche di pagamento inutilizzate AWS

> **Note**
>
> Se chiudi o elimini le tue Account AWS, la tua chiave di crittografia dei AWS pagamenti diventa inaccessibile. Non è necessario programmare l'eliminazione della chiave di crittografia dei AWS pagamenti separatamente dalla chiusura dell'account.

AWS Payment Cryptography registra una voce nel AWS CloudTrail registro quando si pianifica l'eliminazione della chiave di crittografia di AWS pagamento e quando la chiave di crittografia di AWS pagamento viene effettivamente eliminata.

Quando si utilizza Multi-Region la replica delle chiavi, si elimina una chiave di crittografia di pagamento che è una chiave della regione primaria (PRK), anche le chiavi della regione di replica (RRK) verranno eliminate automaticamente. Un RRK non può essere eliminato come un PRK. Se desideri eliminare un RRK, dovrai modificare le regioni di replica per il tuo PRK.

Informazioni sul periodo di attesa

Poiché l'eliminazione di una chiave è irreversibile, AWS Payment Cryptography richiede di impostare un periodo di attesa compreso tra 3 e 180 giorni. Il periodo di attesa predefinito è di sette giorni.

Tuttavia, il periodo di attesa effettivo potrebbe essere fino a 24 ore più lungo di quello pianificato. Per ottenere la data e l'ora effettive in cui la chiave AWS di crittografia dei pagamenti verrà eliminata, utilizza le GetKey operazioni. Assicurati di segnare il fuso orario.

Durante il periodo di attesa, lo stato della chiave AWS di crittografia di pagamento e lo stato della chiave sono In attesa di eliminazione.

> **Note**
>
> Una chiave AWS di crittografia di pagamento in attesa di eliminazione non può essere utilizzata in alcuna operazione crittografica. ???

Al termine del periodo di attesa, AWS Payment Cryptography elimina la chiave di crittografia di pagamento, i relativi alias e tutti i relativi metadati di AWS Payment Cryptography. AWS

Utilizza il periodo di attesa per assicurarti di non aver bisogno della chiave di crittografia di AWS pagamento ora o in futuro. Se ritieni di aver bisogno della chiave durante il periodo di attesa, puoi annullare l'eliminazione della chiave prima della fine del periodo di attesa. Al termine del periodo di attesa, non è possibile annullare l'eliminazione della chiave e il servizio elimina la chiave.

Example

In questo esempio, viene richiesta l'eliminazione di una chiave. Oltre alle informazioni di base sulla chiave, due campi pertinenti indicano che lo stato della chiave è stato modificato in DELETE_PENDING e delete PendingTimestamp indica quando è attualmente pianificata l'eliminazione della chiave.

```
$ 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

In questo esempio, un'eliminazione in sospeso viene annullata. Una volta completata con successo, una chiave non verrà più eliminata in base alla pianificazione precedente. La risposta contiene le informazioni chiave di base; inoltre, sono stati modificati due campi pertinenti: `KeyState` e`deletePendingTimestamp`. `KeyState`viene restituito al valore CREATE_COMPLETE, mentre `DeletePendingTimestamp` viene rimosso.

```
$ 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"
    }
}
```

Importazione ed esportazione di chiavi

È possibile importare le chiavi AWS di crittografia dei pagamenti da altre soluzioni ed esportarle in altre soluzioni, come gli HSM. Molti clienti scambiano le chiavi con i fornitori di servizi utilizzando funzionalità di importazione ed esportazione. Abbiamo progettato la crittografia dei AWS pagamenti per utilizzare un approccio elettronico moderno alla gestione delle chiavi che consente di mantenere la conformità e i controlli. Consigliamo di utilizzare lo scambio di chiavi elettroniche basato su standard anziché componenti chiave cartacei. Se è necessario continuare a elaborare i componenti chiave cartacei fino a quando tutti i partner non supporteranno lo scambio di chiavi elettroniche, è possibile utilizzare. Scambio di chiavi fisiche

Punti di forza minimi delle chiavi ed effetto sulle funzioni di importazione ed esportazione

Il PCI richiede specifiche caratteristiche minime delle chiavi per le operazioni crittografiche, l'archiviazione e la trasmissione delle chiavi. Questi requisiti possono cambiare quando gli standard PCI vengono rivisti. Le regole specificano che le chiavi di archiviazione utilizzate per l'archiviazione o il trasporto devono essere sicure almeno quanto la chiave da proteggere. Applichiamo questo requisito automaticamente durante l'esportazione e impediamo che le chiavi vengano protette da chiavi più deboli, come mostrato nella tabella seguente.

La tabella seguente mostra le combinazioni supportate di chiavi di protezione, chiavi da proteggere e metodi di protezione.

	Chiave di avvolgimento											
Chiave per proteggere	TDES	CHIAVE TDES	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Note
CHIAVE TDES_2	TR-3	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDH	ECDH	ECDH	
CHIAVE TDES_3	✗ Non supp o	TR-3	TR-3	TR-3	TR-3	TR-3 RSA	TR-3 RSA	TR-3 RSA	ECDH	ECDH	ECDH	

	Chiave di avvolgimento											
Chiave per proteggere	TDES	CHIAVE TDES	AES_	AES_	AES_	RSA_	RSA_	RSA_	ECC_	ECC_	ECC_	Note
AES_128	✗ Non supporto	✗ Non supporto	TR-3	TR-3	TR-3	✗ Non supporto	TR-3 RSA	TR-3 RSA	ECDH	ECDH	ECDH	
AES_192	✗ Non supporto	✗ Non supporto	✗ Non supporto	TR-3	TR-3	✗ Non supporto	✗ Non supporto	✗ Non supporto	✗ Non supporto	ECDH	ECDH	
AES_256	✗ Non supporto	✗ Non supporto	✗ Non supporto	✗ Non supporto	TR-3	✗ Non supporto	✗ Non supporto	✗ Non supporto	✗ Non supporto	✗ Non supporto	ECDH	

Per ulteriori informazioni, vedere l'Appendice D - Dimensioni e punti di forza minimi ed equivalenti delle chiavi per algoritmi approvati negli standard PCI HSM.

Key Encryption Key (KEK) Exchange

Si consiglia di utilizzare lo standard ANSI X9.24 TR-34. Questo tipo di chiave iniziale può essere chiamato Key Encryption Key (KEK), Zone Master Key (ZMK) o Zone Control Master Key (ZCMK). Se i tuoi sistemi o partner non supportano TR-34 ancora, puoi usare RSA. Wrap/Unwrap Se le tue esigenze includono lo scambio di AES-256 chiavi, puoi utilizzare ECDH.

> **Note**
>
> Per importare le tue chiavi di test o sincronizzare le chiavi con i tuoi HSM esistenti, consulta il codice di esempio di AWS Payment Cryptography su. GitHub

Working Key (WK) Exchange

Utilizziamo gli standard del settore (ANSI X9.24 TR 31-2018 e X9.143) per lo scambio di chiavi di lavoro. Ciò richiede che tu abbia già scambiato un KEK utilizzando RSA Wrap TR-34, ECDH o schemi simili. Questo approccio soddisfa il requisito PCI PIN per associare crittograficamente il materiale chiave al tipo e all'utilizzo in ogni momento. Le chiavi di lavoro includono le chiavi di lavoro dell'acquirente, le chiavi di lavoro dell'emittente, BDK e IPEK.

Argomenti

- Chiavi di importazione
- Chiavi di esportazione
- Argomenti avanzati

Chiavi di importazione

> **Important**
>
> Gli esempi richiedono la versione più recente di AWS CLI V2. Prima di iniziare, assicurati di aver effettuato l'aggiornamento alla versione più recente. https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

Indice

- Introduzione all'importazione delle chiavi
- Importazione di chiavi simmetriche
  - Importazione di chiavi utilizzando tecniche asimmetriche () TR-34
  - Importa le chiavi utilizzando tecniche asimmetriche (ECDH)
  - Importa le chiavi utilizzando tecniche asimmetriche (RSA Unwrap)
  - Importa chiavi simmetriche utilizzando una chiave di scambio di chiavi prestabilita () TR-31
- Importazione di chiavi pubbliche asimmetriche (RSA, ECC)
  - Importazione di chiavi pubbliche RSA
  - Importazione di chiavi pubbliche ECC

Introduzione all'importazione delle chiavi

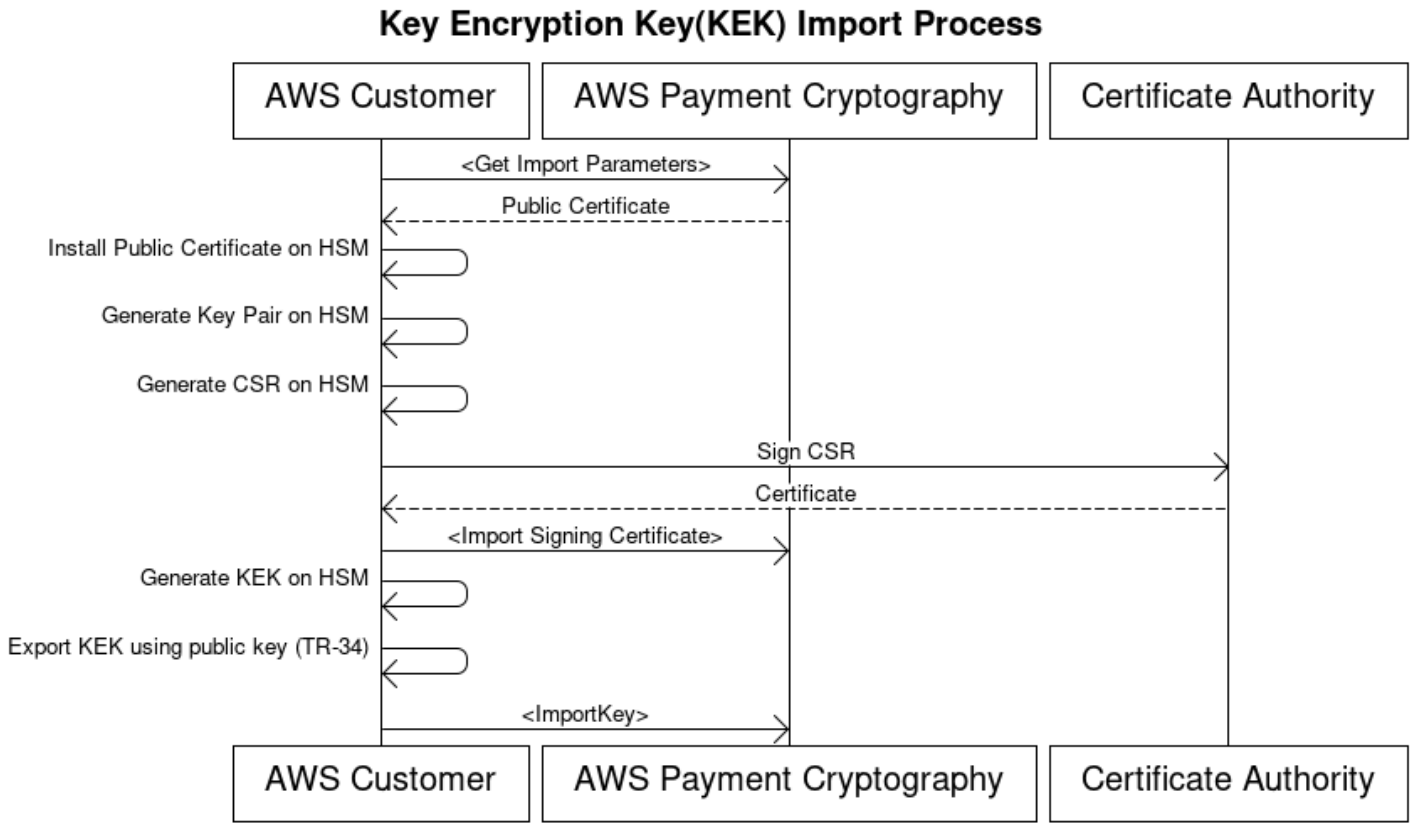
> **Note**
>
> Quando si importano chiavi utilizzando X9.143 TR-31 o blocchi di TR-34 chiavi, AWS Payment Cryptography in genere conserva (ma non utilizza) le intestazioni opzionali. L'intestazione HM (HMAC hash type) viene utilizzata durante le operazioni crittografiche. L'intestazione KP (KCV o chiave di wrapping) è specifica del processo di importazione e non viene conservata.

Quando si scambiano chiavi con una controparte, in genere si scambia prima una chiave di scambio di chiavi (KEK). Questa chiave verrà quindi utilizzata per proteggere le chiavi successive. Utilizzando formati elettronici, il KEK può essere scambiato utilizzando tecniche asimmetriche come ECDH o RSA TR-34 wrap. Le chiavi successive verranno scambiate utilizzando uno scambio di chiavi simmetrico come. TR-31 Questo KEK avrà una lunga durata e potrebbe essere aggiornato solo ogni pochi anni in base alla politica e al periodo crittografico definito.

Se vengono scambiate solo una o due chiavi, puoi anche scegliere di utilizzare tecniche asimmetriche per scambiare direttamente quella chiave, come un BDK. AWS La crittografia dei pagamenti supporta entrambi i metodi di scambio di chiavi.

Importazione di chiavi simmetriche

Importazione di chiavi utilizzando tecniche asimmetriche () TR-34



TR-34 utilizza la crittografia asimmetrica RSA per crittografare e firmare chiavi simmetriche per lo scambio. Ciò garantisce sia la riservatezza (crittografia) che l'integrità (firma) della chiave racchiusa.

Per importare le tue chiavi, dai un'occhiata al progetto di esempio AWS di Payment Cryptography su. GitHub Per istruzioni su come ottenere le import/export chiavi da altre piattaforme, il codice di esempio è disponibile su GitHub o consulta la guida per l'utente di tali piattaforme.

1. Chiamate il comando Initialize Import

   Chiama `get-parameters-for-import` per inizializzare il processo di importazione. Questa API genera una coppia di chiavi per l'importazione delle chiavi, firma la chiave e restituisce il certificato e la radice del certificato. Crittografa la chiave da esportare utilizzando questa chiave. Nella TR-34 terminologia, questo è noto come certificato KRD. Questi certificati sono codificati in base64, di breve durata e destinati esclusivamente a questo scopo. Salva il valore. `ImportToken`

```
$ 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. Installa il certificato pubblico sul sistema di origine delle chiavi

   Con la maggior parte degli HSM, è necessario installare, caricare o considerare attendibile il certificato pubblico generato nel passaggio 1 per esportare le chiavi utilizzandolo. Questo potrebbe includere l'intera catena di certificati o solo il certificato principale del passaggio 1, a seconda dell'HSM.

3. Genera una coppia di chiavi sul sistema di origine e fornisci la catena di certificati a AWS Crittografia dei pagamenti

   Per garantire l'integrità del payload trasmesso, la parte mittente (Key Distribution Host o KDH) lo firma. Genera una chiave pubblica per questo scopo e crea un certificato a chiave pubblica (X509) da restituire a Payment Cryptography. AWS

   Quando trasferisci le chiavi da un HSM, crea una coppia di chiavi su quell'HSM. L'HSM, una terza parte o un servizio simile AWS Private CA può generare il certificato.

   Carica il certificato principale in AWS Payment Cryptography utilizzando il `importKey` comando con KeyMaterialType of `RootCertificatePublicKey` e KeyUsageType of. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

   Per i certificati intermedi, usa il `importKey` comando con KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` Ripetere questa procedura per più certificati intermedi. Utilizzate `KeyArn` l'ultimo certificato importato della catena come input per i successivi comandi di importazione.

> **Note**
>
> Non importate il certificato Leaf. Forniscilo direttamente durante il comando di importazione.

4. Chiave di esportazione dal sistema di origine

   Molti HSM e sistemi correlati supportano l'esportazione delle chiavi utilizzando la TR-34 norma. Specificate la chiave pubblica della fase 1 come certificato KRD (crittografia) e la chiave della fase 3 come certificato KDH (firma). Per importare in AWS Payment Cryptography, specifica il formato TR-34.2012 non CMS a due passaggi, che può anche essere indicato come formato Diebold. TR-34

5. Chiama la chiave di importazione

   Chiama l'API ImportKey con un KeyMaterialType . `TR34_KEY_BLOCK` Usa il KeyArn dell'ultima CA importata nel passaggio 3 per`certificate-authority-public-key-identifier`, il materiale chiave racchiuso nel passaggio 4 per `key-material` e il certificato leaf del passaggio 3 per. `signing-key-certificate` Includi il token di importazione del passaggio 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.
 \
    }'
```

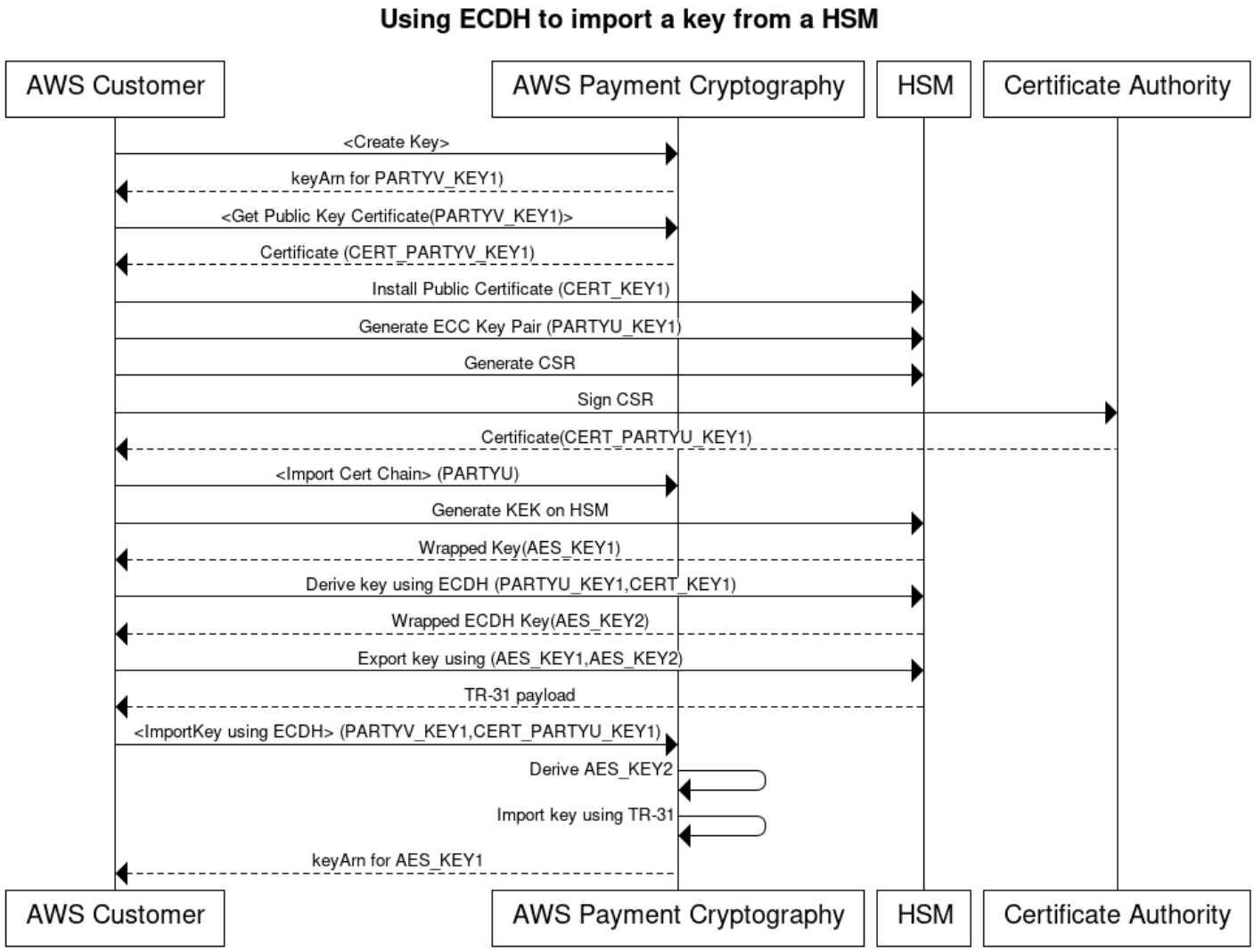
```
{
  "Key": {
    "CreateTimestamp": "2023-06-13T16:52:52.859000-04:00",
    "Enabled": true,
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "KeyAttributes": {
```

```
      "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. Usa la chiave importata per le operazioni crittografiche o l'importazione successiva

   Se la chiave importata KeyUsage era TR31_K0_KEY_ENCRYPTION_KEY, è possibile utilizzare questa chiave per le successive importazioni di chiavi utilizzando. TR-31 Per altri tipi di chiave (come TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), è possibile utilizzare la chiave direttamente per le operazioni crittografiche.

Importa le chiavi utilizzando tecniche asimmetriche (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) utilizza la crittografia asimmetrica ECC per stabilire una chiave condivisa tra due parti senza richiedere chiavi prescambiate. Le chiavi ECDH sono effimere, quindi Payment Cryptography non le memorizza. AWS In questo processo, viene derivata una singola volta KBPK/KEK utilizzando l'ECDH. Tale chiave derivata viene immediatamente utilizzata per racchiudere la chiave effettiva che si desidera trasferire, che potrebbe essere un'altra chiave KBPK, una chiave IPEK o altri tipi di chiave.

Durante l'importazione, il sistema di invio è comunemente noto come Party U (Initiator) e AWS Payment Cryptography è nota come Party V (Responder).

> **Note**
>
> Sebbene l'ECDH possa essere utilizzato per scambiare qualsiasi tipo di chiave simmetrica, è l'unico approccio in grado di trasferire le chiavi in modo sicuro. AES-256

1. Genera una coppia di chiavi ECC

   Chiama `create-key` per creare una coppia di chiavi ECC per questo processo. Questa API genera una coppia di chiavi per le importazioni o le esportazioni di chiavi. Al momento della creazione, specifica il tipo di chiavi che possono essere derivate utilizzando questa chiave ECC. Quando si utilizza ECDH per scambiare (avvolgere) altre chiavi, utilizzare un valore di. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

   > **Note**
   >
   > Sebbene l'ECDH di basso livello generi una chiave derivata che può essere utilizzata per qualsiasi scopo, AWS Payment Cryptography limita il riutilizzo accidentale di una chiave per più scopi consentendo di utilizzare una chiave solo per un singolo tipo di chiave derivata.

```
$ 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. Ottieni il certificato a chiave pubblica

   Chiama `get-public-key-certificate` per ricevere la chiave pubblica sotto forma di X.509 certificato firmato dalla CA del tuo account, specifico per la crittografia dei AWS pagamenti in una regione specifica.

   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. Installa il certificato pubblico sul sistema della controparte (Parte U)

   Con molti HSM, è necessario installare, caricare o considerare attendibile il certificato pubblico generato nel passaggio 1 per esportare le chiavi utilizzandolo. Questo potrebbe includere l'intera catena di certificati o solo il certificato principale del passaggio 1, a seconda dell'HSM. Consulta la documentazione HSM per ulteriori informazioni.

4. Genera una coppia di chiavi ECC sul sistema di origine e fornisci una catena di certificati a AWS Crittografia dei pagamenti

   In ECDH, ciascuna parte genera una coppia di chiavi e concorda una chiave comune. AWS Affinché Payment Cryptography possa derivare la chiave, è necessaria la chiave pubblica della controparte in formato chiave pubblica. X.509

   Quando trasferisci le chiavi da un HSM, crea una coppia di chiavi su quell'HSM. Per gli HSM che supportano i blocchi di chiavi, l'intestazione della chiave sarà simile a. `D0144K3EX00E0000` Quando si crea il certificato, in genere si genera un CSR sull'HSM e poi l'HSM, una terza parte o un servizio in grado di generare il AWS Private CA certificato.

   Carica il certificato principale in AWS Payment Cryptography utilizzando il `importKey` comando con KeyMaterialType of e of. `RootCertificatePublicKey` KeyUsageType `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

   Per i certificati intermedi, usa il `importKey` comando con KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` Ripetere questa procedura per più certificati intermedi. Utilizzate `KeyArn` l'ultimo certificato importato della catena come input per i successivi comandi di importazione.

   > **Note**
   >
   > Non importate il certificato Leaf. Forniscilo direttamente durante il comando di importazione.

5. Ricava una chiave monouso utilizzando ECDH su Party U HSM

   Molti HSM e sistemi correlati supportano la creazione di chiavi tramite ECDH. Specifica la chiave pubblica della fase 1 come chiave pubblica e la chiave della fase 3 come chiave privata. Per le opzioni consentite, come i metodi di derivazione, consulta la guida API.

   > **Note**
   >
   > I parametri di derivazione, come il tipo di hash, devono corrispondere esattamente su entrambi i lati. Altrimenti, genererai una chiave diversa.

6. Chiave di esportazione dal sistema di origine

Infine, esporta la chiave che desideri trasportare in AWS Payment Cryptography utilizzando TR-31 i comandi standard. Specifica la chiave derivata da ECDH come KBPK. La chiave da esportare può essere qualsiasi chiave TDES o AES soggetta a combinazioni TR-31 valide, a condizione che la chiave di composizione sia almeno altrettanto forte della chiave da esportare.

7. Chiama la chiave di importazione

   Chiama l'`import-key`API con KeyMaterialType un`DiffieHellmanTr31KeyBlock`. Usa il KeyArn dell'ultima CA importata nel passaggio 3 per`certificate-authority-public-key-identifier`, il materiale chiave racchiuso nel passaggio 4 per `key-material` e il certificato leaf del passaggio 3 per. `public-key-certificate` Includi la chiave privata ARN del passaggio 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. Usa la chiave importata per le operazioni crittografiche o l'importazione successiva

   Se la chiave importata KeyUsage era TR31_K0_KEY_ENCRYPTION_KEY, è possibile utilizzare questa chiave per le successive importazioni di chiavi utilizzando. TR-31 Per altri tipi di chiave (come TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY), è possibile utilizzare la chiave direttamente per le operazioni crittografiche.

Importa le chiavi utilizzando tecniche asimmetriche (RSA Unwrap)

Panoramica: AWS Payment Cryptography supporta RSA wrap/unwrap per lo scambio di chiavi quando non è possibile. TR-34 Ad esempio TR-34, questa tecnica utilizza la crittografia asimmetrica RSA per crittografare le chiavi simmetriche per lo scambio. Tuttavia, a differenza di questo metodo TR-34, non è il mittente a firmare il payload. Inoltre, questa tecnica RSA wrap non mantiene l'integrità dei metadati chiave durante il trasferimento perché non include blocchi chiave.

> **Note**
>
> È possibile utilizzare RSA wrap per importare o esportare TDES e chiavi. AES-128

1. Chiama il comando Initialize Import

   Chiama get-parameters-for-import per inizializzare il processo di importazione con un `KeyMaterialType` of. `KEY_CRYPTOGRAM` Usare `RSA_2048` per `WrappingKeyAlgorithm` lo scambio di chiavi TDES. Usare `RSA_3072` o per `RSA_4096` scambiare TDES o chiavi. AES-128 Questa API genera una coppia di chiavi per l'importazione delle chiavi, firma la chiave utilizzando una radice del certificato e restituisce sia il certificato che la radice del certificato. Crittografa la chiave da esportare utilizzando questa chiave. Questi certificati sono di breve durata e sono destinati esclusivamente a questo scopo.

   ```
   $ 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. Installa il certificato pubblico sul sistema di origine chiave

   Con molti HSM, è necessario installare, caricare o considerare attendibile il certificato pubblico (and/orla radice) generato nel passaggio 1 per esportare le chiavi utilizzandolo.

3. Esporta la chiave dal sistema di origine

   Molti HSM e sistemi correlati supportano l'esportazione delle chiavi tramite RSA wrap. Specificate la chiave pubblica del passaggio 1 come certificato di crittografia (). `WrappingKeyCertificate` Se hai bisogno della catena di fiducia, usa quella del `WrappingKeyCertificateChain` passaggio 1. Quando esportate la chiave dal vostro HSM, specificate il formato RSA, con Padding Mode = PKCS #1 v2.2 OAEP (con SHA 256 o SHA 512).

4. Esegui una chiamata a chiave di importazione

   Chiama l'import-keyAPI con un `KeyMaterialType` off. `KeyMaterial` È necessario il codice `ImportToken` del passaggio 1 e il `key-material` (materiale chiave confezionato) del

passaggio 3. Fornisci i parametri chiave (ad esempio Key Usage) perché RSA wrap non utilizza blocchi chiave.

```
$ 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
```

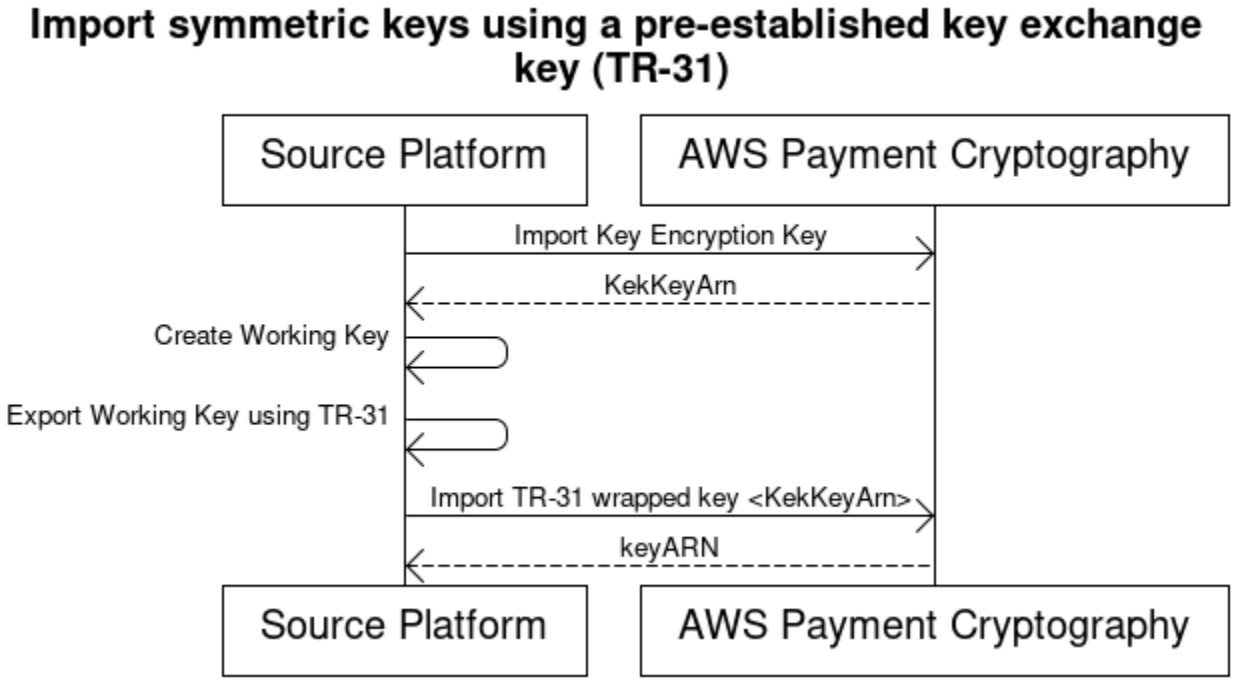
```
{
  "Key": {
    "KeyOrigin": "EXTERNAL",
    "Exportable": true,
    "KeyCheckValue": "DA1ACF",
    "UsageStartTimestamp": 1697643478.92,
```

```
    "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. Utilizza la chiave importata per le operazioni crittografiche o l'importazione successiva

   Se la chiave importata `KeyUsage` era `TR31_K0_KEY_ENCRYPTION_KEY` o`TR31_K1_KEY_BLOCK_PROTECTION_KEY`, è possibile utilizzare questa chiave per le successive importazioni di chiavi utilizzando TR-31. Se il tipo di chiave era di qualsiasi altro tipo (ad esempio`TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY`), è possibile utilizzare la chiave direttamente per le operazioni crittografiche.

Importa chiavi simmetriche utilizzando una chiave di scambio di chiavi prestabilita () TR-31



Quando si scambiano più chiavi o si supporta la rotazione delle chiavi, in genere i partner si scambiano prima una chiave di crittografia a chiave iniziale (KEK). È possibile scambiare KEK con AWS Payment Cryptography, utilizzando tecniche come o. TR-34 Scambio di chiavi fisiche

Dopo aver stabilito un KEK, puoi utilizzarlo per trasportare le chiavi successive (inclusi altri KEK). AWS La crittografia dei pagamenti supporta questo scambio di chiavi tramite ANSI TR-31, ampiamente utilizzato e supportato dai fornitori di HSM.

1. Import Key Encryption Key (KEK)

   Assicurati di aver già importato il tuo KEK e di avere KeyArn (o KeyAlias) disponibile.

2. Crea la chiave sulla piattaforma di origine

   Se la chiave non esiste, creala sulla piattaforma di origine. In alternativa, puoi creare la chiave su AWS Payment Cryptography e utilizzare il export comando.

3. Esporta la chiave dalla piattaforma di origine

   Durante l'esportazione, specifica il formato di esportazione come TR-31. La piattaforma di origine chiederà la chiave da esportare e la chiave di crittografia da utilizzare.

4. Importa in AWS Crittografia dei pagamenti

Quando chiami il import-key comando, usa il keyArn (o alias) della tua chiave di crittografia per. `WrappingKeyIdentifier` Usa l'output della piattaforma sorgente per. `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"
  }
}
```

Importazione di chiavi pubbliche asimmetriche (RSA, ECC)

Tutti i certificati importati devono avere una validità almeno pari a quella del certificato di emissione (predecessore) nella catena. Ciò significa che una CA RSA_2048 può essere utilizzata solo per proteggere un certificato leaf RSA_2048 e un certificato ECC deve essere protetto da un altro certificato ECC di livello equivalente. Un certificato ECC P384 può essere rilasciato solo da una CA P384 o P521. Tutti i certificati non devono essere scaduti al momento dell'importazione.

Importazione di chiavi pubbliche RSA

AWS La crittografia dei pagamenti supporta l'importazione di chiavi RSA pubbliche come certificati. X.509 Per importare un certificato, importate prima il relativo certificato principale. Tutti i certificati non devono essere scaduti al momento dell'importazione. Il certificato deve essere in formato PEM e codificato in base 64.

1. Importa il certificato principale in AWS Crittografia dei pagamenti

   Usa il seguente comando per importare il certificato principale:

Example

2. Importa il certificato a chiave pubblica in AWS Crittografia dei pagamenti

   Ora puoi importare una chiave pubblica. Poiché TR-34 ECDH si basa sul passaggio del certificato leaf in fase di esecuzione, questa opzione viene utilizzata solo quando si crittografano i dati utilizzando una chiave pubblica di un altro sistema. KeyUsage sarà impostato su 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"
  }
}
```

Importazione di chiavi pubbliche ECC

AWS La crittografia dei pagamenti supporta l'importazione di chiavi ECC pubbliche come certificati. X.509 Per importare un certificato, importate innanzitutto il relativo certificato CA principale e tutti i certificati intermedi. Tutti i certificati non devono essere scaduti al momento dell'importazione. Il certificato deve essere in formato PEM e codificato in base 64.

1. Importa il certificato radice ECC in AWS Crittografia dei pagamenti

   Usa il seguente comando per importare il certificato principale:

Example

2. Importa certificato intermedio in AWS Crittografia dei pagamenti

   Usa il seguente comando per importare un certificato intermedio:

Example

3. Importa il certificato a chiave pubblica (Leaf) in AWS Crittografia dei pagamenti

   Sebbene sia possibile importare un certificato LEAF ECC, attualmente nella crittografia dei AWS pagamenti non sono definite altre funzioni oltre all'archiviazione. Questo perché quando si utilizzano le funzioni ECDH, il certificato leaf viene passato in fase di esecuzione.

Chiavi di esportazione

Indice

- Esporta chiavi simmetriche
  - Esportazione delle chiavi utilizzando tecniche asimmetriche () TR-34
  - Esporta le chiavi utilizzando tecniche asimmetriche (ECDH)
  - Esporta le chiavi utilizzando tecniche asimmetriche (RSA Wrap)
  - Esporta le chiavi simmetriche utilizzando una chiave di scambio di chiavi prestabilita () TR-31
- Esporta le chiavi iniziali DUKPT () IPEK/IK
- Specifica le intestazioni dei blocchi chiave per l'esportazione
  - Intestazioni comuni
- Esporta chiavi asimmetriche (RSA)

Esporta chiavi simmetriche

> **Important**
>
> Assicurati di avere la versione più recente di AWS CLI prima di iniziare. Per eseguire l'aggiornamento, vedi Installazione di AWS CLI.

Esportazione delle chiavi utilizzando tecniche asimmetriche () TR-34

TR-34 utilizza la crittografia asimmetrica RSA per crittografare e firmare chiavi simmetriche per lo scambio. La crittografia protegge la riservatezza, mentre la firma garantisce l'integrità. Quando si esportano le chiavi, AWS Payment Cryptography funge da host di distribuzione delle chiavi (KDH) e il sistema di destinazione diventa il dispositivo di ricezione delle chiavi (KRD).

> **Note**
>
> Se il tuo HSM supporta TR-34 l'esportazione ma non l' TR-34 importazione, ti consigliamo di stabilire prima un KEK condiviso tra il tuo HSM e Payment Cryptography utilizzando. AWS TR-34 È quindi possibile utilizzarlo TR-31 per trasferire le chiavi rimanenti.

1. Inizializza il processo di esportazione

   Esegui get-parameters-for-export per generare una coppia di chiavi per le esportazioni di chiavi. Utilizziamo questa coppia di chiavi per firmare il TR-34 payload. Nella TR-34 terminologia, questo è il certificato di firma KDH. I certificati sono di breve durata e validi solo per la durata specificata in. `ParametersValidUntilTimestamp`

   > **Note**
   >
   > Tutti i certificati sono in codifica 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. Importa il AWS Certificato di crittografia dei pagamenti sul tuo sistema di ricezione

   Importa la catena di certificati dalla fase 1 al tuo sistema di ricezione.

3. Configura i certificati del tuo sistema di ricezione

Per proteggere il payload trasmesso, il mittente (KDH) lo crittografa. Il sistema ricevente (in genere il tuo HSM o l'HSM del tuo partner) deve generare una chiave pubblica e creare un certificato con chiave pubblica. X.509 È possibile AWS Private CA utilizzarlo per generare certificati, ma è possibile utilizzare qualsiasi autorità di certificazione.

Una volta ottenuto il certificato, importate il certificato principale in AWS Payment Cryptography utilizzando il ImportKey comando. Imposta `KeyMaterialType` su `RootCertificatePublicKey` e `KeyUsageType` su `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`.

Usiamo `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` come `KeyUsageType` perché questa è la chiave principale che firma il certificato foglia. Non è necessario importare i certificati leaf in AWS Payment Cryptography: puoi trasmetterli in linea.

> **Note**
>
> Se in precedenza hai importato il certificato principale, salta questo passaggio. Per i certificati intermedi, usa. `TrustedCertificatePublicKey`

4. Esporta la tua chiave

   Chiama l'ExportKeyAPI con `KeyMaterialType` set to`TR34_KEY_BLOCK`. È necessario fornire:

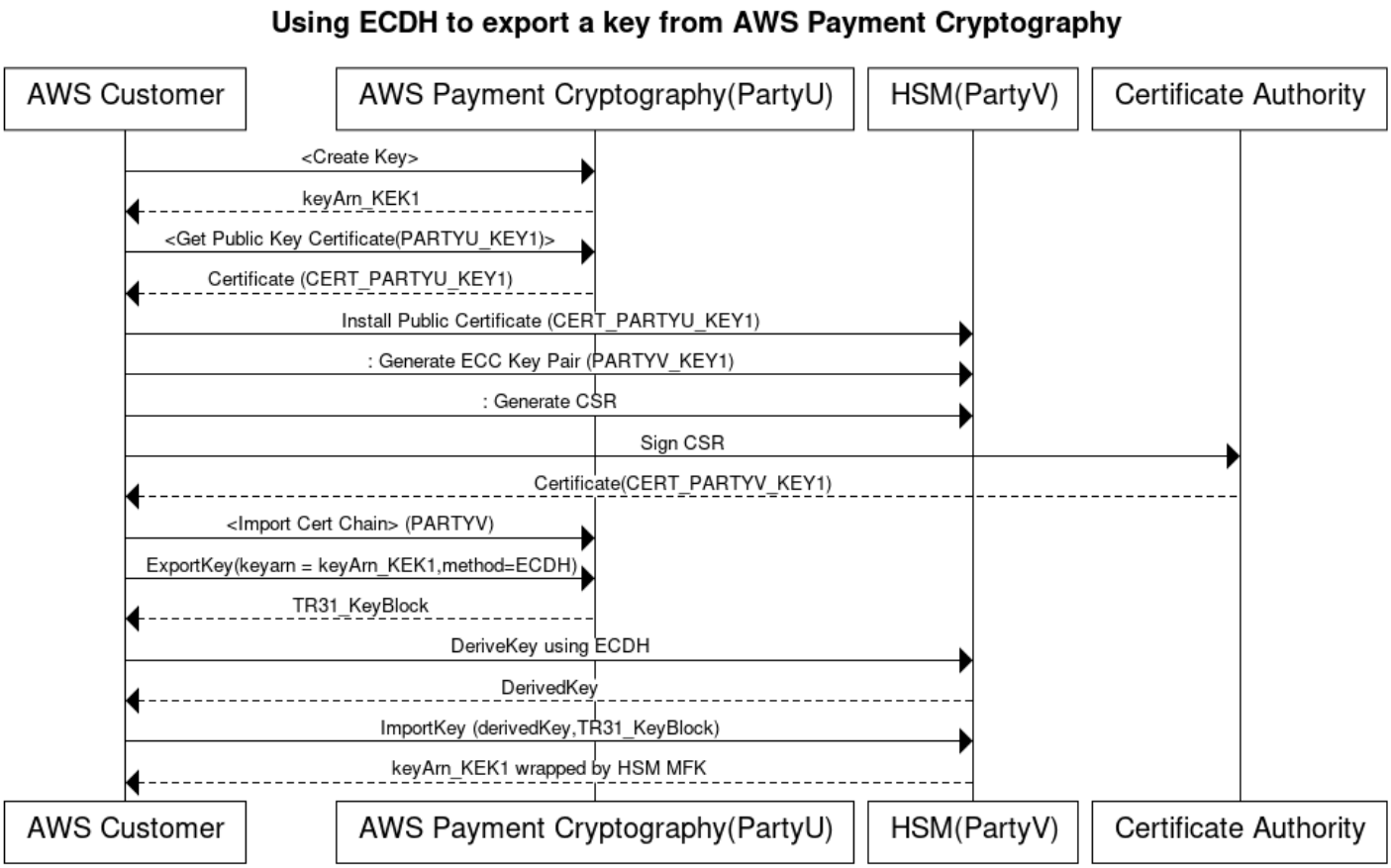
   - Il KeyArn della CA principale del passaggio 3 è `CertificateAuthorityPublicKeyIdentifier`
   - Il certificato principale della fase 3 era `WrappingKeyCertificate`
   - Il keyArn (o alias) della chiave che desideri esportare come `--export-key-identifier`
   - Il token di esportazione del passaggio 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"
  }
}
```

Esporta le chiavi utilizzando tecniche asimmetriche (ECDH)



Elliptic Curve Diffie-Hellman (ECDH) utilizza la crittografia asimmetrica ECC per stabilire una chiave condivisa tra due parti senza richiedere chiavi prescambiate. Le chiavi ECDH sono effimere, quindi Payment Cryptography non le memorizza. AWS In questo processo, viene derivata una singola volta KBPK/KEK utilizzando l'ECDH. Tale chiave derivata viene immediatamente utilizzata per racchiudere la chiave che si desidera trasferire, che potrebbe essere un'altra chiave KBPK, BDK, IPEK o altri tipi di chiave.

Durante l'esportazione, la crittografia dei AWS pagamenti viene denominata Parte U (Iniziatore) e il sistema ricevente è noto come Parte V (Risponditore).

> **Note**
>
> L'ECDH può essere utilizzato per scambiare qualsiasi tipo di chiave simmetrica, ma è l'unico approccio che può essere utilizzato per trasferire le AES-256 chiavi se non è già stato stabilito un KEK.

1. Genera una coppia di chiavi ECC

   Chiama `create-key` per creare una coppia di chiavi ECC per questo processo. Questa API genera una coppia di chiavi per le importazioni o le esportazioni di chiavi. Al momento della creazione, specifica il tipo di chiavi che possono essere derivate utilizzando questa chiave ECC. Quando si utilizza ECDH per scambiare (avvolgere) altre chiavi, utilizzare un valore di. `TR31_K1_KEY_BLOCK_PROTECTION_KEY`

   > **Note**
   >
   > Sebbene l'ECDH di basso livello generi una chiave derivata che può essere utilizzata per qualsiasi scopo, AWS Payment Cryptography limita il riutilizzo accidentale di una chiave per più scopi consentendo di utilizzare una chiave solo per un singolo tipo di chiave derivata.

```
$ 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. Ottieni il certificato a chiave pubblica

   Chiama `get-public-key-certificate` per ricevere la chiave pubblica sotto forma di X.509 certificato firmato dalla CA del tuo account, specifico per la crittografia dei AWS pagamenti in una regione specifica.

   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. Installa il certificato pubblico sul sistema della controparte (Parte V)

   Con molti HSM, è necessario installare, caricare o considerare attendibile il certificato pubblico generato nel passaggio 1 per stabilire le chiavi. Ciò potrebbe includere l'intera catena di certificati o solo il certificato principale, a seconda dell'HSM. Consultate la documentazione HSM per istruzioni specifiche.

4. Genera una coppia di chiavi ECC sul sistema sorgente e fornisci una catena di certificati a AWS Crittografia dei pagamenti

   In ECDH, ciascuna parte genera una coppia di chiavi e concorda una chiave comune. AWS Affinché Payment Cryptography possa derivare la chiave, è necessaria la chiave pubblica della controparte in formato chiave pubblica. X.509

Quando trasferisci le chiavi da un HSM, crea una coppia di chiavi su quell'HSM. Per gli HSM che supportano i blocchi di chiavi, l'intestazione della chiave sarà simile a. `D0144K3EX00E0000` Quando si crea il certificato, in genere si genera un CSR sull'HSM, quindi l'HSM, una terza parte o un servizio in grado di generare il AWS Private CA certificato.

Carica il certificato principale in AWS Payment Cryptography utilizzando il `importKey` comando con KeyMaterialType of e of. `RootCertificatePublicKey` KeyUsageType `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE`

Per i certificati intermedi, usa il `importKey` comando con KeyMaterialType of `TrustedCertificatePublicKey` e KeyUsageType of. `TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE` Ripetere questa procedura per più certificati intermedi. Utilizzate `KeyArn` l'ultimo certificato importato della catena come input per i successivi comandi di esportazione.

> **Note**
>
> Non importare il certificato foglia. Forniscilo direttamente durante il comando di esportazione.

5. Deriva la chiave e la chiave di esportazione da AWS Crittografia dei pagamenti

Durante l'esportazione, il servizio ricava una chiave utilizzando ECDH e quindi la utilizza immediatamente come KBPK per racchiudere la chiave da esportare utilizzando. TR-31 La chiave da esportare può essere qualsiasi chiave TDES o AES soggetta a combinazioni TR-31 valide, purché la chiave di wrapping sia forte almeno quanto la chiave da esportare.

```
$ 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. Ricava una chiave una tantum usando ECDH su Party V HSM

   Molti HSM e sistemi correlati supportano la creazione di chiavi tramite ECDH. Specifica la chiave pubblica della fase 1 come chiave pubblica e la chiave della fase 3 come chiave privata. Per le opzioni consentite, come i metodi di derivazione, consulta la guida API.

   > **Note**
   >
   > I parametri di derivazione, come il tipo di hash, devono corrispondere esattamente su entrambi i lati. Altrimenti, genererai una chiave diversa.

7. Importa la chiave nel sistema di destinazione

   Infine, importa la chiave da AWS Payment Cryptography utilizzando TR-31 i comandi standard. Specifica la chiave derivata da ECDH come KBPK e usa il blocco di TR-31 chiavi precedentemente esportato da Payment Cryptography. AWS

Esporta le chiavi utilizzando tecniche asimmetriche (RSA Wrap)

Quando TR-34 non è disponibile, è possibile utilizzare RSA wrap/unwrap per lo scambio di chiavi. Ad esempio TR-34, questo metodo utilizza la crittografia asimmetrica RSA per crittografare le chiavi simmetriche. Tuttavia, RSA wrap non include:

- Firma del payload da parte del mittente
- Blocchi chiave che mantengono l'integrità dei metadati chiave durante il trasporto

> **Note**
>
> È possibile utilizzare RSA wrap per esportare TDES e chiavi. AES-128

1. Crea una chiave e un certificato RSA sul tuo sistema ricevente

   Crea o identifica una chiave RSA per ricevere la chiave integrata. Richiediamo che le chiavi siano in formato X.509 certificato. Assicurati che il certificato sia firmato da un certificato radice che puoi importare in AWS Payment Cryptography.
2. Importa il certificato pubblico principale in AWS Crittografia dei pagamenti

   Usalo import-key con l'`--key-material`opzione per importare il certificato

```
$ 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. Esporta la tua chiave

   Chiedi a AWS Payment Cryptography di esportare la tua chiave utilizzando il tuo certificato leaf. Devi specificare:

   - L'ARN per il certificato principale importato nel passaggio 2
   - Il certificato principale per l'esportazione
   - La chiave simmetrica per l'esportazione

   L'output è una versione binaria (crittografata) con codifica esadecimale della chiave simmetrica.

Example Esempio: esportazione di una chiave

```
$ 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. Importa la chiave nel tuo sistema di ricezione

   Molti HSM e sistemi correlati supportano l'importazione di chiavi tramite RSA unwrap (inclusa AWS la crittografia dei pagamenti). Durante l'importazione, specificare:

   - La chiave pubblica del passaggio 1 come certificato di crittografia
   - Il formato è RSA
   - Modalità di riempimento come PKCS #1 v2.2 OAEP (con SHA 256)

> **Note**
>
> Emettiamo la chiave racchiusa in formato HexBinary. Potrebbe essere necessario convertire il formato se il sistema richiede una rappresentazione binaria diversa, ad esempio base64.

Esporta le chiavi simmetriche utilizzando una chiave di scambio di chiavi prestabilita () TR-31

Quando si scambiano più chiavi o si supporta la rotazione delle chiavi, in genere i partner si scambiano prima una chiave di crittografia a chiave iniziale (KEK). È possibile scambiare KEK con AWS Payment Cryptography, utilizzando tecniche come o. TR-34 Scambio di chiavi fisiche Dopo aver creato un KEK, puoi utilizzarlo per trasportare le chiavi successive, inclusi altri KEK. Supportiamo questo scambio di chiavi utilizzando ANSI TR-31, ampiamente supportato dai fornitori di HSM.

1. Configura la tua chiave di crittografia delle chiavi (KEK)

   Assicurati di aver già scambiato il tuo KEK e di avere a disposizione il KeyArn (o KeyAlias).

2. Crea la tua chiave su AWS Crittografia dei pagamenti

   Crea la tua chiave se non esiste già. In alternativa, puoi creare la chiave sull'altro sistema e utilizzare il comando import.

3. Esporta la tua chiave da AWS Crittografia dei pagamenti

   Quando esporti in TR-31 formato, specifica la chiave che desideri esportare e la chiave di avvolgimento da utilizzare.

Example Esempio: esportazione di una chiave utilizzando il blocco di tasti TR31

```
$ 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. Importa la chiave nel tuo sistema

   Usa l'implementazione della chiave di importazione del tuo sistema per importare la chiave.

Esporta le chiavi iniziali DUKPT () IPEK/IK

Quando si utilizza DUKPT, è possibile generare una singola chiave di derivazione di base (BDK) per una flotta di terminali. I terminali non hanno accesso diretto al BDK. Ogni terminale riceve invece una chiave terminale iniziale univoca, nota come IPEK o Initial Key (IK). Ogni IPEK è derivato dal BDK utilizzando un Key Serial Number (KSN) univoco.

La struttura KSN varia in base al tipo di crittografia:

- Per TDES: il KSN a 10 byte include:
  - 24 bit per l'ID del set di chiavi
  - 19 bit per l'ID del terminale
  - 21 bit per il contatore delle transazioni
- Per AES: il KSN a 12 byte include:
  - 32 bit per l'ID BDK

- 32 bit per l'identificatore di derivazione (ID)
- 32 bit per il contatore delle transazioni

Forniamo un meccanismo per generare ed esportare queste chiavi iniziali. È possibile esportare le chiavi generate utilizzando i metodi TR-31 TR-34,, o RSA wrap. Tieni presente che le chiavi IPEK non sono permanenti e non possono essere utilizzate per operazioni successive sulla crittografia dei pagamenti. AWS

Non applichiamo la divisione tra le prime due parti del KSN. Se desideri memorizzare l'identificatore di derivazione nel BDK, puoi utilizzare i tag. AWS

> **Note**
>
> La parte contatrice del KSN (32 bit per AES DUKPT) non viene utilizzata per la derivazione. IPEK/IK Ad esempio, gli input di 12345678901234560001 e 12345678901234569999 genereranno lo stesso IPEK.

```
$ 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"
}
}
```

Specifica le intestazioni dei blocchi chiave per l'esportazione

È possibile modificare o aggiungere informazioni sui blocchi chiave durante l'esportazione in formato TR-31 ASC o in formato. TR-34 La tabella seguente descrive il formato del blocco TR-31 chiave e quali elementi è possibile modificare durante l'esportazione.

Attributo del blocco chiave	Scopo	È possibile modificare durante l'esportazione?	Note
ID versione	Definisce il metodo utilizzato per proteggere il materiale chiave. Lo standard include: • Versione A e C (variante chiave - obsoleta) • Versione B (derivazione tramite TDES) • Versione D (derivazione della chiave tramite AES)	No	Utilizziamo la versione B per le chiavi di wrapping TDES e la versione D per le chiavi di wrapping AES. Supportiamo le versioni A e C solo per le operazioni di importazione.
Lunghezza del blocco chiave	Specifica la lunghezza del messaggio rimanente	No	Questo valore viene calcolato automaticamente. La lunghezza potrebbe apparire errata prima di decrittografare il payload perché potremmo aggiungere il keypadding come richiesto dalle specifiche.

Attributo del blocco chiave	Scopo	È possibile modificare durante l'esportazione?	Note
Utilizzo delle chiavi	Definisce gli scopi consentiti per la chiave, ad esempio: • C0 (verifica della carta) • B0 (chiave di derivazione di base)	No	
Algoritmo	Specifica l'algoritmo della chiave sottostante. Supportiamo: • (MAREE) • (HMAC) • (A ES)	No	Esportiamo questo valore così com'è.
Utilizzo delle chiavi	Definisce le operazioni consentite, ad esempio: • Genera e verifica (C) • Encrypt/Decrypt/Wrap/Unwrap (B)	Sì*	
Versione chiave	Indica il numero di versione della chiave replacement/rotation. Il valore predefinito è 00 se non specificato.	Sì - Può aggiungere	

Attributo del blocco chiave	Scopo	È possibile modificare durante l'esportazione?	Note
Esportabilità chiave	Controlla se la chiave può essere esportata: • N - Nessuna esportabilità • E - Esportazione in base a X9.24 (blocchi chiave) • S - Esportazione in formati a blocchi chiave o non a blocchi di chiavi	Sì*	
Blocchi chiave opzionali	Sì, può essere aggiunto	I blocchi di chiavi opzionali sono name/value coppie associate crittograficamente alla chiave. Ad esempio, KeySet ID per le chiavi DUKPT. Calcoliamo automaticamente il numero di blocchi, la lunghezza di ogni blocco e il padding block (PB) in base all'input della coppia. name/value	

*Quando si modificano i valori, il nuovo valore deve essere più restrittivo rispetto al valore corrente in Payment Cryptography. AWS Ad esempio:

- Se la modalità di utilizzo della chiave corrente è Generate=True, Verify=True, puoi cambiarla in Generate=True, Verify=False
- Se la chiave è già impostata come non esportabile, non puoi cambiarla in esportabile

Quando esporti le chiavi, applichiamo automaticamente i valori correnti della chiave da esportare. Tuttavia, potresti voler modificare o aggiungere quei valori prima di inviarli al sistema ricevente. Ecco alcuni scenari comuni:

- Quando esportate una chiave su un terminale di pagamento, impostate l'esportabilità su `Not Exportable` perché i terminali in genere importano solo chiavi e non dovrebbero esportarle.
- Quando devi trasmettere i metadati della chiave associati al sistema ricevente, utilizza intestazioni TR-31 opzionali per associare crittograficamente i metadati alla chiave invece di creare un payload personalizzato.
- Imposta la versione della chiave utilizzando il campo per tracciare la rotazione della chiave. `KeyVersion`

TR-31/X9.143 definisce intestazioni comuni, ma puoi utilizzarne altre purché soddisfino i parametri di AWS Payment Cryptography e il tuo sistema di ricezione le accetti. Per ulteriori informazioni sulle intestazioni dei blocchi chiave durante l'esportazione, consulta Key Block Headers nella Guida API.

Ecco un esempio di esportazione di una chiave BDK (ad esempio, in un KIF) con queste specifiche:

- Versione chiave: 02
- KeyExportability: NON_EXPORTABLE
- KeySetID: 00ABCDEFAB (00 indica la chiave TDES, ABCDEFABCD è la chiave iniziale)

Poiché non specifichiamo le modalità di utilizzo delle chiavi, questa chiave eredita la modalità di utilizzo da arn:aws:payment-cryptography:us-east- 2:111122223333: (= true). key/5rplquuwozodpwsp DeriveKey

> **Note**
>
> Anche se imposti l'esportabilità su Non esportabile in questo esempio, il KIF può comunque: ???
>
> - Derivare chiavi come quelle utilizzate in DUKPT IPEK/IK

- Esporta queste chiavi derivate per installarle sui dispositivi

Ciò è specificamente consentito dagli standard.

```
$ 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"
    }
}
```

Intestazioni comuni

X9.143 definisce alcune intestazioni per i casi d'uso più comuni. Ad eccezione dell'intestazione HM (HMAC Hash), AWS Payment Cryptography non analizza né utilizza queste intestazioni.

Header Name (Nome intestazione)	Scopo	Validazione tipica	Note
BI	Identificatore chiave di derivazione di base per DUKPT	2 caratteri esadecimali (00 per TDES, 11 per AES) quindi 10 caratteri esadecimali per TDES KSI o 8 caratteri esadecimali per l'ID BDK (AES DUKPT).	Contiene il (ID BDK, per AES DUKPT) o il Key Set Identifier (KSI, per TDES DUKPT). Può essere utilizzato per lo scambio dell'ID BDK o del KSI, ma non è necessario scambiare gli altri dati contenuti nei blocchi IK e KS. In genere, BI viene utilizzato per la trasmissione a un KIF mentre IK o KS vengono utilizzati per l'iniezione nel terminale stesso.
HM	Specifica il tipo di hash per le operazioni 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	Il servizio compila automaticamente questo campo durante l'esportazione e lo analizza durante l'importazione. I tipi di hash non supportati dal servizio, come SHAKE128, possono essere importati ma potrebbero non essere utilizzabili per le funzioni crittografiche.

Header Name (Nome intestazione)	Scopo	Validazione tipica	Note
		• 41 — FRULLATO 256	
cinematica inversa	Numero di serie della chiave iniziale per AES DUKPT	16 caratteri esadecimali	Questo valore viene utilizzato per istanziare e l'uso della chiave DUKPT iniziale sul dispositivo ricevente e identifica la chiave iniziale derivata da un BDK. Questo campo contiene in genere i dati di derivazione ma non un contatore. Usa KS per TDES DUKPT.
È	Numero di serie chiave iniziale per TDES DUKPT	20 caratteri esadecimali	Questo valore viene utilizzato per istanziare e l'uso della chiave DUKPT iniziale sul dispositivo ricevente e identifica la chiave iniziale derivata da un BDK. Questo campo contiene in genere i dati di derivazio ne + un valore del contatore azzerato. Usa IK per AES DUKPT.

Header Name (Nome intestazione)	Scopo	Validazione tipica	Note
KP	Valore di controllo chiave del KBPK ???	2 caratteri esadecimali rappresentano il metodo KCV (00 per il metodo e 01 per il X9.24 metodo CMAC). Seguito dal valore KCV che in genere è composto da 6 caratteri esadecimali. Ad esempio 010FA329 rappresenta il KCV di 0FA329 calcolato utilizzando il metodo 01 (CMAC).	Calcolato secondo l' X9.24-1-2017 allegato A dell'ANSI; il KCV viene utilizzato per la convalida delle chiavi e non come meccanismo di integrità.
PB	Blocco di imbottitura	caratteri ASCII stampabili casuali	Il servizio compila automaticamente questo campo durante l'esportazione per garantire che le intestazioni opzionali siano multipli della lunghezza del blocco di crittografia

Esporta chiavi asimmetriche (RSA)

Per esportare una chiave pubblica sotto forma di certificato, usa il comando. get-public-key-certificate Questo comando restituisce:

- Il certificato
- Il certificato principale

Entrambi i certificati sono in codifica base64.

> **Note**
>
> Questa operazione non è idempotente: le chiamate successive potrebbero generare certificati diversi anche quando si utilizza la stessa chiave sottostante.

Example

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

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

Argomenti avanzati

Questa sezione descrive scenari e configurazioni avanzati di scambio di chiavi.

Argomenti

- BYOCA (Bring Your Own Certificate Authority)
- Scambio di chiavi fisiche

BYOCA (Bring Your Own Certificate Authority)

Per impostazione predefinita, quando è necessario un certificato a chiave pubblica per le chiavi asimmetriche (RSA, ECC) create all'interno del servizio, questi certificati vengono emessi da una AWS Payment Cryptography e da un'autorità di certificazione (CA) univoca per l'account. Questo ha lo scopo di semplificarne l'uso X.509 senza l'onere di identificare o configurare una CA o gestire le richieste di firma dei certificati (CSR).

AWS La crittografia dei pagamenti offre anche la possibilità di utilizzare la propria CA quando è richiesta per motivi di policy o conformità.

Argomenti

- Panoramica di
- Flusso di lavoro di esportazione BYOCA
- Flusso di lavoro di importazione BYOCA
- Note aggiuntive

Panoramica di

Con BYOCA, puoi utilizzare la tua CA per i trasferimenti di chiavi TR-34 import/export, RSA Unwrap ed ECDH. Usa BYOCA quando hai bisogno di una catena di certificati coerente in tutta l'organizzazione o quando i partner richiedono certificati CA specifici. Gli esempi seguenti mostrano il flusso di lavoro BYOCA per l'esportazione e l'importazione TR-34 delle chiavi. TR-34

Esistono tre differenze principali rispetto al flusso standard: TR-34

1. La chiave RSA viene creata con CreateKey. Prima `GetParametersForExport` o l'hai `GetParametersForImport` creata per te.
2. L'GetCertificateSigningRequestAPI crea un CSR. La tua CA esterna può quindi firmarlo.
3. Le ImportKey API ExportKey e accettano un certificato al momento della chiamata. Il token è ora opzionale.

⚠ Considerazioni importanti

- Questi esempi utilizzano RSA-2048 le chiavi e racchiudono una TDES-2KEY chiave. Durante l'esportazione AES-128, assicurati che tutte le chiavi siano RSA-3072 o RSA-4096.
- L'errore più comune è che la chiave rappresentata da `SigningKeyIdentifier` e `SigningKeyCertificate` non corrisponde.

Flusso di lavoro di esportazione BYOCA

I passaggi seguenti illustrano il flusso di lavoro BYOCA completo per l'esportazione. TR-34

Fasi

- Fase 1: Creare una chiave RSA
- Fase 2: Generazione della richiesta di firma del certificato
- Fase 3: Rivedi la CSR (opzionale)
- Fase 4: Firma il CSR con un'autorità di certificazione
- Fase 5: Importazione del certificato CA
- Fase 6: Ottieni il certificato di crittografia KRD
- Passaggio 7: Esportazione della chiave con BYOCA

Fase 1: Creare una chiave RSA

Innanzitutto, crea una coppia di chiavi RSA che alla fine sarà il certificato di firma KDH. Puoi aggiungere tag per identificare lo scopo della chiave.

Example Crea una chiave RSA per la firma

```
$ 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"
    }
}
```

Prendi nota di ciò che ti `KeyArn` servirà nel passaggio successivo.

Fase 2: Generazione della richiesta di firma del certificato

Genera una richiesta di firma del certificato (CSR) da firmare dalla tua CA esterna utilizzando l'GetCertificateSigningRequestAPI. L'output è un file PEM con codifica base64. Se decodificate i contenuti in base64 e li salvate, avrete un CSR valido in formato PEM.

Example Genera CSR

```
$ aws payment-cryptography 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..."
}
```

Il `CertificateSigningRequest` campo contiene l'intero file PEM, codificato in base64. Il file PEM include i marcatori e. `-----BEGIN CERTIFICATE REQUEST-----` `-----END CERTIFICATE REQUEST-----` Invia questo valore alla tua CA per la firma.

Fase 3: Rivedi la CSR (opzionale)

Facoltativamente, puoi utilizzare OpenSSL per esaminare i contenuti della CSR e assicurarti che siano validi e conformi alle aspettative.

Example Rivedi la CSR con OpenSSL

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

Fase 4: Firma il CSR con un'autorità di certificazione

Dopo aver generato il CSR, è necessario farlo firmare da un'autorità di certificazione (CA). Negli ambienti di produzione, in genere si utilizza l' CA privata AWS infrastruttura CA consolidata della propria organizzazione. A scopo di test, puoi usare OpenSSL per creare un certificato autofirmato.

Utilizzo CA privata AWS

Per firmare il CSR utilizzando CA privata AWS, decodifica prima il CSR con codifica base64 e salvalo in un file, quindi utilizza l'API. IssueCertificate

Example Firma la CSR con 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"
}
```

Quindi recupera il certificato firmato:

Example Recupera il certificato firmato

```
$ 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-----"
```

```
}
```

Salva il contenuto del certificato per utilizzarlo nella fase di esportazione. Dovrai codificarlo in base 64 quando lo fornisci all'API. `ExportKey`

Usare OpenSSL per i test

A scopo di test, è possibile utilizzare OpenSSL per creare una CA autofirmata e firmare il CSR. Innanzitutto, crea una chiave privata CA e un certificato autofirmato:

Example Crea una CA di test con 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"
```

Quindi decodifica il CSR del passaggio precedente e firmalo con la tua CA di prova:

Example Firma la CSR con 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
```

Il certificato firmato è ora disponibile. `signed-cert.pem` Dovrai codificare questo certificato in base 64 quando lo fornisci all'API: `ExportKey`

Example Base64: codifica il certificato firmato

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

Fase 5: Importazione del certificato CA

Qualsiasi CA utilizzata deve prima essere considerata attendibile per evitare l'uso di certificati arbitrari. Importa il certificato principale della tua CA esterna utilizzando l'ImportKeyAPI. Se utilizzi una CA intermedia, richiama `import-key` nuovamente ma specifica `TrustedPublicKey` invece `RootCertificatePublicKey` e specifica l'ARN CA radice.

Example Importa il certificato CA principale

```
$ 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"
    }
}
```

Prendi nota delle CA `KeyArn` da utilizzare nella fase di esportazione.

Fase 6: Ottieni il certificato di crittografia KRD

In questo esempio, stiamo reimportando in AWS Payment Cryptography, quindi chiamiamo il servizio per ricevere un certificato a chiave pubblica KRD utilizzando l'API. GetParametersForImport In uno scenario reale, questo sarebbe fornito dall'altro sistema, come un HSM, un bancomat, un terminale di pagamento o un sistema di gestione dei terminali di pagamento.

Example Ottieni i parametri per l'importazione

```
$ aws payment-cryptography 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"
}
```

Passaggio 7: Esportazione della chiave con BYOCA

Infine, esporta la chiave utilizzando TR-34 il tuo CA-signed certificato utilizzando l'ExportKeyAPI. Fornisci il certificato di firma firmato dalla tua CA esterna.

Example TR-34 Esporta con BYOCA

```
$ aws payment-cryptography 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"
    }
}
```

Il blocco chiave esportato può ora essere importato dal sistema ricevente utilizzando il processo di importazione standard TR-34 .

Flusso di lavoro di importazione BYOCA

I passaggi seguenti illustrano il flusso di lavoro BYOCA per l'importazione delle chiavi. TR-34 Usa questo flusso di lavoro per ricevere una chiave di AWS Payment Cryptography con il tuo certificato di crittografia CA-signed KRD (Key Receiving Device). Con l'importazione BYOCA, controlli la catena di fiducia utilizzando una CA che già gestisci.

Fasi

- Fase 1: Creare una chiave RSA
- Fase 2: Generazione della richiesta di firma del certificato
- Fase 3: Rivedi la CSR (opzionale)
- Fase 4: Firma il CSR con un'autorità di certificazione
- Fase 5: Importazione del certificato CA
- Passaggio 6: Ottieni il certificato di firma KDH dal sistema di invio
- Fase 7: Importazione della chiave con BYOCA

Fase 1: Creare una chiave RSA

Crea una coppia di chiavi RSA da utilizzare come chiave di crittografia KRD. Questa chiave decomprime e decrittografa la chiave in ingresso durante l'importazione. TR-34 La chiave non è esportabile perché rimane all'interno del servizio come chiave di ricezione.

Example Crea una chiave di crittografia KRD

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

```
    --key-attributes
 KeyAlgorithm=RSA_2048,KeyUsage=TR31_K2_TR34_ASYMMETRIC_KEY,KeyClass=ASYMMETRIC_KEY_PAIR,KeyMod
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
bm7t4qv8hzk24jce",
        "KeyAttributes": {
            "KeyUsage": "TR31_K2_TR34_ASYMMETRIC_KEY",
            "KeyClass": "ASYMMETRIC_KEY_PAIR",
            "KeyAlgorithm": "RSA_2048",
            "KeyModesOfUse": {
                "Unwrap": true,
                "Decrypt": true
            }
        },
        "KeyCheckValue": "7FA29C1E",
        "KeyCheckValueAlgorithm": "SHA_1",
        "Enabled": true,
        "Exportable": false,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY"
    }
}
```

Annotare il valore `KeyArn`. Questo valore servirà nella fase successiva.

Fase 2: Generazione della richiesta di firma del certificato

Genera una richiesta di firma del certificato (CSR) per la chiave di crittografia KRD utilizzando l'GetCertificateSigningRequestAPI. L'output è un file PEM con codifica base64. Se decodificate i contenuti in base64 e li salvate, avrete un CSR valido in formato PEM.

Example Genera CSR per KRD Key

```
$ aws payment-cryptography get-certificate-signing-request \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
bm7t4qv8hzk24jce \
    --signing-algorithm SHA512 \
    --certificate-subject '{
        "CommonName": "MyImportKRDCertAWSUSEAST",
        "Organization": "Amazon",
        "OrganizationUnit": "PaymentCryptography",
```

```
        "Country": "US",
        "StateOrProvince": "Virginia",
        "City": "Arlington"
    }'
```

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

Il `CertificateSigningRequest` campo contiene l'intero file PEM, codificato in base64. Il file PEM include i marcatori e. `-----BEGIN CERTIFICATE REQUEST-----` `-----END CERTIFICATE REQUEST-----` Invia questo valore alla tua CA per la firma.

Fase 3: Rivedi la CSR (opzionale)

Facoltativamente, puoi utilizzare OpenSSL per esaminare i contenuti della CSR e assicurarti che siano validi e conformi alle aspettative.

Example Rivedi la CSR con OpenSSL

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

Fase 4: Firma il CSR con un'autorità di certificazione

Dopo aver generato il CSR, fatelo firmare da un'autorità di certificazione (CA). Negli ambienti di produzione, in genere si utilizza l' CA privata AWS infrastruttura CA consolidata della propria organizzazione. A scopo di test, puoi usare OpenSSL per creare un certificato autofirmato.

Utilizzo CA privata AWS

Per firmare il CSR utilizzando CA privata AWS, decodifica prima il CSR con codifica base64 e salvalo in un file, quindi utilizza l'API. IssueCertificate

Example Firma la CSR con AWS Private CA

```
$ echo "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0tLS0..." | base64 -d > krd-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://krd-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/fedcba0987654321"
}
```

Quindi recupera il certificato firmato:

Example Recupera il certificato firmato

```
$ 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/fedcba0987654321
```

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

Salva il contenuto del certificato per utilizzarlo nella fase di importazione. Base64-encode il certificato prima di fornirlo all'`ImportKey`API.

Usare OpenSSL per i test

A scopo di test, è possibile utilizzare OpenSSL per creare una CA autofirmata e firmare il CSR. Innanzitutto, crea una chiave privata CA e un certificato autofirmato:

Example Crea una CA di test con 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"
```

Quindi decodifica il CSR del passaggio precedente e firmalo con la tua CA di prova:

Example Firma la CSR con OpenSSL

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

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

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

Il certificato firmato è ora disponibile. `krd-signed-cert.pem` Base64-encode questo certificato prima di fornirlo all'`ImportKey`API:

Example Base64: codifica il certificato firmato

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

Fase 5: Importazione del certificato CA

Prima di utilizzare una CA, devi prima fidarti di essa. Importa il certificato principale della tua CA esterna utilizzando l'ImportKeyAPI. Se utilizzi una CA intermedia, richiama `import-key` nuovamente ma specifica `TrustedPublicKey` invece `RootCertificatePublicKey` l'ARN CA radice.

Example Importa il certificato CA principale

```
$ 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/
nqr3c5v2wp6yl8az",
        "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"
    }
}
```

Annota le CA `KeyArn` da utilizzare nella fase di importazione.

Passaggio 6: Ottieni il certificato di firma KDH dal sistema di invio

In uno scambio di TR-34 chiavi, la parte mittente (Key Distribution Host o KDH) fornisce il certificato con chiave pubblica per la firma. Questo certificato viene utilizzato per verificare la firma sul blocco di chiavi avvolto durante l'importazione, assicurando che il materiale chiave provenga da un mittente affidabile. In fase di produzione, questo certificato proviene in genere dal sistema di invio, ad esempio un HSM, un terminale di pagamento o un sistema di gestione delle chiavi, attraverso un proprio processo di distribuzione dei certificati.

In questo esempio, utilizzi la crittografia dei AWS pagamenti sia come mittente che come destinatario per mantenere i passaggi brevi. Chiama GetParametersForExport per ottenere il certificato di firma KDH che normalmente viene fornito dal sistema di invio.

Example Ottieni i parametri per l'esportazione

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

```
{
    "ExportToken": "export-token-k8sfhe2w9rjy4bx6",
    "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "SigningKeyCertificateChain": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
    "SigningKeyAlgorithm": "RSA_2048",
    "ParametersValidUntilTimestamp": "2025-11-01T18:45:31.271000-07:00"
}
```

Annota la lettera `ExportToken` e `SigningKeyCertificate` da utilizzare nella fase di importazione.

Fase 7: Importazione della chiave con BYOCA

Infine, importa la chiave utilizzando TR-34 il tuo CA-signed certificato utilizzando l'`ImportKey`API. Fornisci il certificato di crittografia KRD firmato dalla tua CA esterna.

Il `RandomNonce` parametro è obbligatorio quando si utilizza TR-34 2-pass. Il servizio non genera un nonce per te: è un valore esadecimale casuale che generi e deve essere lo stesso sia sul lato di invio che su quello di ricezione. Quando si utilizza 1-pass TR-34 (che si basa su un timestamp anziché su un nonce), il timestamp non deve essere futuro. Il servizio non convalida alcuna particolare freschezza (ad esempio, entro un certo numero di ore), ma puoi imporre controlli di freschezza da parte tua, se necessario.

Example TR-34 Importa con BYOCA

```
$ aws payment-cryptography import-key --key-material='{
    "Tr34KeyBlock": {
        "CertificateAuthorityPublicKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/nqr3c5v2wp6yl8az",
        "KeyBlockFormat": "X9_TR34_2012",
        "RandomNonce": "4997FBB4587D571F",
        "SigningKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
        "WrappingKeyCertificate": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t...",
        "WrappingKeyIdentifier": "arn:aws:payment-cryptography:us-
east-1:111122223333:key/bm7t4qv8hzk24jce",
        "WrappedKeyBlock": "D0112B0AX00E0000B82679114F470F540165EDFBF407..."
    }
}'
```

```
{
```

```
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
gwz7u3rfxd4o52sb",
        "KeyAttributes": {
            "KeyUsage": "TR31_K0_KEY_ENCRYPTION_KEY",
            "KeyClass": "SYMMETRIC_KEY",
            "KeyAlgorithm": "TDES_2KEY",
            "KeyModesOfUse": {
                "Encrypt": true,
                "Decrypt": true
            }
        },
        "KeyCheckValue": "3DCA31",
        "KeyCheckValueAlgorithm": "ANSI_X9_24",
        "Enabled": true,
        "Exportable": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL"
    }
}
```

La chiave è ora importata in AWS Payment Cryptography e pronta per l'uso. Il `KeyOrigin` campo `EXTERNAL` indica che la chiave proviene dall'esterno del servizio.

Note aggiuntive

- Questi esempi sono mostrati utilizzando l'interfaccia a riga di comando di AWS. La stessa funzionalità è disponibile in tutti gli SDK AWS, inclusi Java, Python, Go e Rust.
- Se stai eseguendo il test con una CA autofirmata, puoi usare OpenSSL per creare una CA di test e firmare la CSR. In produzione, utilizza l'infrastruttura CA consolidata della tua organizzazione.

Scambio di chiavi fisiche

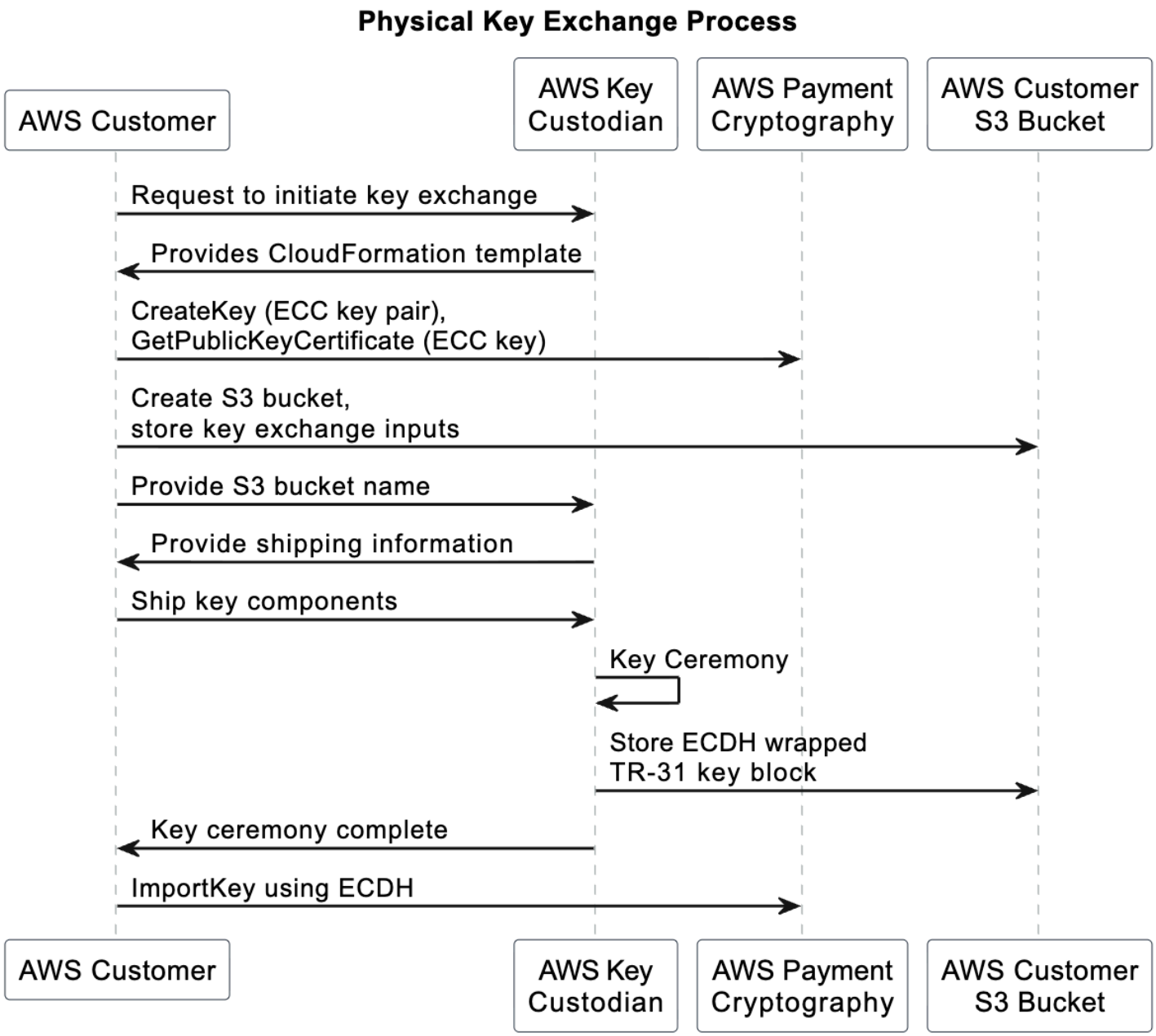
È possibile utilizzare Physical Key Exchange per convertire in modo sicuro i componenti delle chiavi crittografiche cartacee in formato elettronico quando i partner o i fornitori non supportano lo scambio di chiavi elettroniche. Custodi AWS delle chiavi qualificati eseguono cerimonie chiave in strutture AWS-operated sicure certificate PCI PIN e P2PE, convertendo i componenti chiave cartacei in formato elettronico utilizzando un HSM offline. Il servizio utilizza lo scambio di ECDH-based chiavi per fornire un blocco di chiavi, che viene importato ECDH-wrapped TR-31 direttamente nel tuo account Payment Cryptography. AWS

> **Note**
>
> Raccomandiamo di utilizzare soluzioni basate su standard laddove possibileImportazione ed esportazione di chiavi. Utilizzate Physical Key Exchange solo quando i vostri partner o fornitori non supportano metodi di scambio di chiavi elettroniche come ANSI X9.24 TR-34, RSA o ECDH. wrap/unwrap ???

Come funziona Physical Key Exchange

Per avviare lo scambio di chiavi cartacee, un CloudFormation modello guida l'utente nella configurazione dei prerequisiti, inclusa la creazione di una coppia di chiavi ECC e di un bucket S3 nel proprio account. Tu o il tuo partner spedite quindi i componenti cartacei delle chiavi alla struttura AWS sicura, dove custodi delle AWS chiavi addestrati eseguono la cerimonia delle chiavi utilizzando un HSM offline. L'output è un blocco ECDH-wrapped TR-31 chiave caricato nel bucket S3, che importi nel tuo account utilizzando il metodo. Importa le chiavi utilizzando tecniche asimmetriche (ECDH) Physical Key Exchange supporta l'importazione di chiavi KEK (utilizzo della chiave K1) o BDK (utilizzo della chiave B0) negli algoritmi di chiave TDES e AES.

Il diagramma seguente mostra il processo completo di Physical Key Exchange.



1. Avvio: invii un ticket di assistenza o collabori con il tuo account manager per inviare una richiesta.
2. Configurazione del cliente: AWS Payment Cryptography fornisce un CloudFormation modello per completare i seguenti passaggi preliminari:
   - Crea una coppia di chiavi ECC P521 nel tuo account AWS Payment Cryptography e recupera il certificato con chiave pubblica.
   - Crea un bucket Amazon S3 con una policy che conceda al servizio Payment Cryptography l'accesso principale. AWS read/write

- Memorizza il certificato pubblico ECC e la CA root di firma nel bucket Amazon S3.
- Fornisci gli attributi chiave: utilizzo della chiave, modalità di utilizzo chiave e numero di componenti chiave cartacei da inviare.

3. Condividi il nome del bucket S3: il cliente condivide il nome del bucket S3 creato dallo CloudFormation stack, dove vengono archiviati il certificato a chiave pubblica, la catena di certificati e gli attributi chiave per consentire a AWS Payment Cryptography di avviare lo scambio di chiavi.
4. Coordinamento delle spedizioni: AWS Payment Cryptography fornisce i dettagli di spedizione per la struttura sicura. US-based Tu o il tuo partner spedite i componenti chiave cartacei ai custodi AWS chiave.
5. Ricezione dei componenti: i custodi AWS chiave ricevono ogni componente cartaceo e inviano una conferma separata per ogni componente.
6. Cerimonia delle chiavi: i custodi AWS chiave eseguono la cerimonia delle chiavi utilizzando un HSM offline. Il blocco di TR-31 chiavi risultante, racchiuso con una ECDH-derived AES-256 chiave, il certificato pubblico ECC dell'HSM offline e il relativo certificato di firma vengono caricati nel bucket Amazon S3.
7. Completamento: AWS Payment Cryptography invia una conferma che la cerimonia delle chiavi è completa. Puoi quindi importare il blocco di TR-31 chiavi in formato ECDH nel tuo account AWS Payment Cryptography utilizzando il metodo. Importa le chiavi utilizzando tecniche asimmetriche (ECDH)
8. Fatturazione: una volta completata con successo la cerimonia delle chiavi, ti verrà addebitato il costo della chiave scambiata.

Conformità e sicurezza

Physical Key Exchange opera in strutture AWS sicure progettate per soddisfare i requisiti di sicurezza fisica e logica PCI P2PE PCI. Sono disponibili i seguenti controlli:

Doppio controllo e separazione dei compiti

AWS i custodi chiave sono assegnati da diversi team con strutture di rendicontazione separate. Sono in atto processi per garantire che le fasi chiave delle cerimonie si svolgano sotto duplice controllo.

HSM offline

Le cerimonie chiave vengono eseguite utilizzando moduli di sicurezza HSM-listed hardware certificati PCI PTS che funzionano offline senza connettività di rete. La tua chiave non esiste mai in chiaro al di fuori del confine HSM.

Consegna della chiave crittografica

Il materiale chiave viene trasferito dall'HSM offline all'account AWS Payment Cryptography mediante lo scambio di ECDH-based chiavi, garantendo una protezione crittografica end-to-end.

Audit e conformità

AWS dispone di processi per soddisfare i requisiti di conformità applicabili che vengono valutati periodicamente per le attestazioni PCI PIN e P2PE. Consulta il pacchetto di conformità in AWS Artifact per i report a cui fai riferimento nelle tue valutazioni PCI.

Utilizzo di alias

Un alias è un nome descrittivo per una chiave di crittografia dei pagamenti. AWS Ad esempio, un alias consente di fare riferimento a una chiave come `alias/test-key` anziché a. `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h`

È possibile utilizzare un alias per identificare una chiave nella maggior parte delle operazioni di gestione delle chiavi (piano di controllo) e nelle operazioni crittografiche (piano dati).

Puoi anche consentire e negare l'accesso alla chiave di crittografia di AWS pagamento in base ai relativi alias senza modificare le politiche o gestire le sovvenzioni. Questa funzionalità fa parte del supporto del servizio per il controllo degli accessi basato sugli attributi (ABAC).

Gli alias semplificano anche il riutilizzo dello stesso codice in diverse Regioni AWS. Crea alias con lo stesso nome in più regioni e associa ogni alias a una chiave di crittografia dei AWS pagamenti nella relativa regione. Quando il codice viene eseguito in ciascuna regione, l'alias si riferisce alla chiave crittografica di AWS pagamento associata in quella regione.

Puoi creare un alias per una chiave AWS di crittografia dei pagamenti utilizzando l'API. `CreateAlias`

L'API AWS Payment Cryptography fornisce il pieno controllo degli alias in ogni account e regione. L'API include operazioni per creare un alias (CreateAlias), visualizzare i nomi degli alias e il keyARN

collegato (list-alias), modificare la chiave di crittografia di AWS pagamento associata a un alias (update-alias) ed eliminare un alias (delete-alias).

Argomenti

- Informazioni sugli alias
- Utilizzo di alias nelle applicazioni
- API correlate

Informazioni sugli alias

Scopri come funzionano gli alias in Payment Cryptography. AWS

Un alias è una risorsa indipendente AWS

Un alias non è una proprietà di una chiave di crittografia di AWS pagamento. Le azioni intraprese sull'alias non influiscono sulla chiave associata. Puoi creare un alias per una chiave AWS di crittografia dei pagamenti e quindi aggiornarlo in modo che sia associato a una chiave di crittografia di pagamento diversa AWS . Puoi anche eliminare l'alias senza alcun effetto sulla chiave di crittografia di pagamento associata. AWS Se si elimina una chiave AWS di crittografia dei pagamenti, tutti gli alias associati a tale chiave non saranno assegnati.

Se specifichi un alias come risorsa in una policy IAM, la policy si riferisce all'alias, non alla chiave di crittografia di pagamento associata. AWS

Ogni alias ha un nome descrittivo

Quando si crea un alias, si specifica il nome alias preceduto da. `alias/` Ad esempio `alias/test_1234`

Ogni alias è associato a una chiave AWS di crittografia dei pagamenti alla volta

L'alias e la relativa chiave AWS di crittografia dei pagamenti devono trovarsi nello stesso account e nella stessa regione.

Una chiave AWS di crittografia dei pagamenti può essere associata a più di un alias contemporaneamente, ma ogni alias può essere mappato a una sola chiave

Ad esempio, questo `list-aliases` risultato mostra che l'`alias/sampleAlias1`alias è associato esattamente a una chiave di crittografia di AWS pagamento di destinazione, rappresentata dalla proprietà. `KeyArn`

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

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

È possibile associare più alias alla stessa chiave di crittografia di pagamento AWS

Ad esempio, è possibile associare gli `alias/sampleAlias2` alias `alias/sampleAlias1;` and alla stessa chiave.

```
$ 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"
            }
        ]
    }
```

Un alias deve essere univoco per un determinato account e regione

Ad esempio, è possibile avere un solo alias `alias/sampleAlias1` in ogni account e regione. Gli alias fanno distinzione tra maiuscole e minuscole, ma si consiglia di non utilizzare alias che differiscono solo nelle lettere maiuscole in quanto possono essere soggetti a errori. Non è possibile modificare un nome alias. Tuttavia, puoi eliminare l'alias e creare un nuovo alias con il nome desiderato.

Puoi creare un alias con lo stesso nome in diverse regioni

Ad esempio, puoi avere un alias `alias/sampleAlias2` negli Stati Uniti orientali (Virginia settentrionale) e un `alias/sampleAlias2` alias negli Stati Uniti occidentali (Oregon). Ogni alias verrebbe associato a una chiave di crittografia dei AWS pagamenti nella relativa regione. Se il tuo codice fa riferimento a un nome alias come `alias/finance-key`, puoi eseguirlo in più regioni. In ogni regione utilizza una diversa alias/sampleAlias2. Per informazioni dettagliate, vedi Utilizzo di alias nelle applicazioni.

È possibile modificare la chiave AWS di crittografia dei pagamenti associata a un alias

È possibile utilizzare l'`UpdateAlias`operazione per associare un alias a una chiave di crittografia di AWS pagamento diversa. Ad esempio, se l'`alias/sampleAlias2`alias è associato alla chiave `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` AWS Payment Cryptography, è possibile aggiornarlo in modo che sia associato alla chiave. `arn:aws:payment-cryptography:us-east-2:111122223333:key/tqv5yij6wtxx64pi`

> **Warning**
>
> AWS La crittografia dei pagamenti non convalida che la vecchia e la nuova chiave abbiano tutti gli stessi attributi, ad esempio l'utilizzo delle chiavi. L'aggiornamento con un tipo di chiave diverso può causare problemi nell'applicazione.

> **Warning**
>
> Gli aggiornamenti degli alias sono soggetti a un'eventuale coerenza. Dopo aver aggiornato un alias, c'è un breve periodo durante il quale l'alias potrebbe trasformarsi nella vecchia chiave o nella nuova chiave. Non aggiornate gli alias se questo comportamento potrebbe interrompere i processi.

Alcune chiavi non hanno alias

Un alias è una funzionalità opzionale e non tutte le chiavi avranno alias a meno che non si scelga di gestire l'ambiente in questo modo. Le chiavi possono essere associate agli alias utilizzando il comando. `create-alias` Inoltre, è possibile utilizzare l'operazione update-alias per modificare la chiave AWS Payment Cryptography associata a un alias e l'operazione delete-alias per eliminare un alias. Di conseguenza, alcune chiavi di crittografia dei AWS pagamenti potrebbero avere diversi alias e altre no.

Mappatura di una chiave a un alias

È possibile mappare una chiave (rappresentata da un ARN) a uno o più alias utilizzando il comando. `create-alias` Questo comando non è idempotente: per aggiornare un alias, usa il comando 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"
    }
}
```

Utilizzo di alias nelle applicazioni

È possibile utilizzare un alias per rappresentare una chiave di crittografia dei pagamenti nel codice dell'applicazione. AWS Il `key-identifier` parametro nelle operazioni sui dati AWS di Payment Cryptography e in altre operazioni come List Keys accetta un nome alias o un 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}
```

Quando utilizzi un alias ARN, ricorda che l'alias mappato a una chiave di crittografia di AWS pagamento è definito nell'account che possiede la chiave di crittografia di AWS pagamento e potrebbe differire in ciascuna regione.

Uno degli usi più efficaci degli alias è nelle applicazioni eseguite in più Regioni AWS.

Puoi creare una versione diversa dell'applicazione in ogni regione o utilizzare un dizionario, una configurazione o un'istruzione switch per selezionare la chiave di crittografia di AWS pagamento corretta per ciascuna regione. Ma potrebbe essere più semplice creare un alias con lo stesso nome alias in ogni regione. Tieni presente che il nome alias rispetta la distinzione tra maiuscole e minuscole.

API correlate

Tag

I tag sono coppie di chiavi e valori che fungono da metadati per l'organizzazione delle chiavi di crittografia dei pagamenti. AWS Possono essere utilizzati per identificare in modo flessibile le chiavi o raggruppare una o più chiavi.

Ottieni le chiavi

Una chiave crittografica di AWS pagamento rappresenta una singola unità di materiale crittografico e può essere utilizzata solo per le operazioni crittografiche relative a questo servizio. L' GetKeys API accetta un input e restituisce KeyIdentifier i metadati chiave, inclusi attributi, stato e timestamp, ma non restituisce il materiale effettivo della chiave crittografica.

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"
  }
}
```

Fa associare il pubblico key/certificate a una coppia di chiavi

Get Public Key/Certificate restituisce la chiave pubblica indicata da`KeyArn`. Può essere la porzione di chiave pubblica di una coppia di chiavi generata su AWS Payment Cryptography o una chiave pubblica precedentemente importata. Il caso d'uso più comune è fornire la chiave pubblica a un servizio esterno che crittograferà i dati. Tali dati possono quindi essere passati a un'applicazione che sfrutta AWS Payment Cryptography e i dati possono essere decrittografati utilizzando la chiave privata protetta all'interno di Payment Cryptography. AWS

Il servizio restituisce le chiavi pubbliche come certificato pubblico. Il risultato dell'API contiene la CA e il certificato a chiave pubblica. Entrambi gli elementi di dati sono codificati in base64.

> **Note**
>
> Il certificato pubblico restituito è destinato ad essere di breve durata e non è destinato a essere idempotente. Potresti ricevere un certificato diverso per ogni chiamata API, anche se la chiave pubblica stessa è invariata.

Example

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

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

Chiavi di tagging

In AWS Payment Cryptography, puoi aggiungere tag a una chiave di AWS Payment Cryptography quando crei una chiave e taggare o detaggare le chiavi esistenti a meno che non siano in attesa di eliminazione. I tag sono opzionali, ma possono essere molto utili.

Per informazioni generali sui tag, incluse le migliori pratiche, le strategie di tagging e il formato e la sintassi dei tag, consulta Risorse sulla codifica nel. AWS Riferimenti generali di Amazon Web Services

Argomenti

- Informazioni sui tag in AWS Crittografia dei pagamenti
- Visualizzazione dei tag chiave nella console
- Gestione dei tag chiave con operazioni API
- Controllo degli accessi ai tag
- Utilizzo dei tag per controllare l'accesso alle chiavi

Informazioni sui tag in AWS Crittografia dei pagamenti

Un tag è un'etichetta di metadati opzionale che puoi assegnare (o AWS puoi assegnare) a una risorsa. AWS Ogni tag è costituito da una chiave di tag e da un valore di tag, entrambe le stringhe fanno distinzione tra maiuscole e minuscole. Il valore di tag può essere una stringa vuota (null). Ogni tag di una risorsa deve avere una chiave tag diversa, ma puoi aggiungere lo stesso tag a più AWS risorse. Ogni risorsa può avere fino a 50 tag creati dall'utente.

Non includere informazioni riservate o sensibili nella chiave o nel valore del tag. I tag sono accessibili a molti Servizi AWS, inclusa la fatturazione.

In AWS Payment Cryptography, puoi aggiungere tag a una chiave quando la crei e taggare o detaggare le chiavi esistenti a meno che non siano in attesa di eliminazione. Non è possibile contrassegnare gli alias. I tag sono opzionali, ma possono essere molto utili.

Ad esempio, puoi aggiungere un `"Project"="Alpha"` tag a tutte le chiavi AWS di crittografia dei pagamenti e ai bucket Amazon S3 utilizzati per il progetto Alpha. Un altro esempio è l'aggiunta di un `"BIN"="20130622"` tag a tutte le chiavi associate a uno specifico numero di identificazione bancaria (BIN).

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

Per informazioni generali sui tag, inclusi il formato e la sintassi, consulta AWS Risorse relative ai tag in. Riferimenti generali di Amazon Web Services

I tag consentono di:

- Identifica e organizza le tue risorse. AWS Molti AWS servizi supportano l'etichettatura, quindi puoi assegnare lo stesso tag a risorse di servizi diversi per indicare che le risorse sono correlate. Ad esempio, puoi assegnare lo stesso tag a una chiave di crittografia dei AWS pagamenti e a un volume o segreto di Amazon Elastic Block Store (Amazon EBS). AWS Secrets Manager Puoi anche utilizzare i tag per identificare le chiavi per l'automazione.
- Tieni traccia AWS dei costi. Quando aggiungi tag alle tue AWS risorse, AWS genera un rapporto di allocazione dei costi con utilizzo e costi aggregati per tag. È possibile utilizzare questa funzionalità per tenere traccia dei costi AWS di crittografia dei pagamenti per un progetto, un'applicazione o un centro di costo.

  Per ulteriori informazioni sull'utilizzo dei tag per l'allocazione dei costi, consulta Uso dei tag per l'allocazione dei costi nella Guida per l'utente di AWS Billing . Per informazioni sulle regole per le chiavi e i valori dei tag, consulta Restrizioni relative ai User-Defined tag nella Guida per l'AWS Billing utente.
- Controlla l'accesso alle tue AWS risorse. Consentire e negare l'accesso alle chiavi in base ai relativi tag fa parte del supporto di AWS Payment Cryptography per il controllo degli accessi basato sugli attributi (ABAC). Per informazioni sul controllo dell'accesso alla crittografia di AWS pagamento in base ai relativi tag, consulta. Autorizzazione basata su AWS Tag di crittografia dei pagamenti Per informazioni più generali sull'uso dei tag per controllare l'accesso alle AWS risorse, consulta Controlling Access to Resources Using AWS Resources Using Resource Tags nella IAM User Guide.

AWS Payment Cryptography scrive una voce nel tuo AWS CloudTrail registro quando usi le ListTagsForResource operazioni TagResource, UntagResource, o.

Visualizzazione dei tag chiave nella console

Per visualizzare i tag nella console, è necessaria l'autorizzazione di etichettatura sulla chiave in base a una policy IAM che include la chiave. Queste autorizzazioni sono necessarie in aggiunta alle autorizzazioni per visualizzare le chiavi nella console.

Gestione dei tag chiave con operazioni API

Puoi utilizzare l'API AWS Payment Cryptography per aggiungere, eliminare ed elencare i tag per le chiavi che gestisci. Questi esempi utilizzano la AWS Command Line Interface (AWS CLI), ma puoi usare anche qualsiasi linguaggio di programmazione supportato. Non puoi Chiavi gestite da AWS etichettare.

Per aggiungere, modificare, visualizzare ed eliminare i tag per una chiave, è necessario disporre delle autorizzazioni richieste. Per informazioni dettagliate, vedi Controllo degli accessi ai tag.

Argomenti

- CreateKey: aggiungi tag a una nuova chiave
- TagResource: aggiungi o modifica i tag per una chiave
- ListResourceTags: Ottieni i tag per una chiave
- UntagResource: Elimina i tag da una chiave

CreateKey: aggiungi tag a una nuova chiave

Puoi aggiungere tag quando crei una chiave. Per specificare i tag, utilizzate il `Tags` parametro dell'CreateKeyoperazione.

Per aggiungere tag durante la creazione di una chiave, il chiamante deve disporre dell'`payment-cryptography:TagResource`autorizzazione prevista da una policy IAM. Come minimo, l'autorizzazione deve coprire tutte le chiavi dell'account e della regione. Per informazioni dettagliate, vedi Controllo degli accessi ai tag.

Il valore del parametro `Tags` di `CreateKey` è una raccolta di coppie di chiave di tag e valore di tag per cui si applica la distinzione tra maiuscole e minuscole. Ogni tag su una chiave deve avere un nome di tag diverso. Il valore di tag può essere una stringa nulla o vuota.

Ad esempio, il AWS CLI comando seguente crea una chiave di crittografia simmetrica con un `Project:Alpha` tag. Quando si specificano più coppie chiave-valore, utilizzare uno spazio per separare ciascuna coppia.

```
$ 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"}]'
```

Quando questo comando ha esito positivo, restituisce un `Key` oggetto con informazioni sulla nuova chiave. Tuttavia, `Key` non include tag. Per ottenere i tag, usa l'ListResourceTagsoperazione.

TagResource: aggiungi o modifica i tag per una chiave

L'TagResourceoperazione aggiunge uno o più tag a una chiave. Non puoi utilizzare questa operazione per aggiungere o modificare tag in un Account AWS diverso.

Per aggiungere un tag, specifica una nuova chiave di tag e un valore di tag. Per modificare un tag, specifica una chiave di tag esistente e un nuovo valore di tag. Ogni tag su una chiave deve avere una chiave tag diversa. Il valore di tag può essere una stringa nulla o vuota.

Ad esempio, il comando seguente aggiunge **`UseCase`** un **`BIN`** tag a una chiave di esempio.

```
$ 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"}]'
```

Quando questo comando ha esito positivo, non restituisce alcun output. Per visualizzare i tag su una chiave, utilizzare l'ListResourceTagsoperazione.

Puoi inoltre usare TagResource per modificare il valore di un tag esistente. Per sostituire un valore di tag, specifica la stessa chiave di tag con un valore diverso. I tag non elencati in un comando di modifica non vengono modificati o rimossi.

Ad esempio, questo comando modifica il valore del tag `Project` da `Alpha` a `Noe`.

Il comando verrà restituito http/200 senza contenuto. Per visualizzare le modifiche, usa `ListTagsForResource`

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

ListResourceTags: Ottieni i tag per una chiave

L'ListResourceTagsoperazione ottiene i tag per una chiave. Il parametro `ResourceArn` (keyArn o keyAlias) è obbligatorio. Non è possibile utilizzare questa operazione per visualizzare i tag sulle chiavi in un altro formato. Account AWS

Ad esempio, il comando seguente ottiene i tag per una chiave di esempio.

```
$ 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: Elimina i tag da una chiave

L'UntagResourceoperazione elimina i tag da una chiave. Per identificare i tag da eliminare, specifica le chiavi dei tag. Non è possibile utilizzare questa operazione per eliminare tag da chiavi diverse Account AWS.

Quando l'operazione `UntagResource` ha esito positivo non restituisce alcun output. Inoltre, se la chiave tag specificata non viene trovata sulla chiave, non genera un'eccezione né restituisce una risposta. Per confermare che l'operazione ha funzionato, usa l'ListResourceTagsoperazione.

Ad esempio, questo comando elimina il **`Purpose`** tag e il relativo valore dalla chiave specificata.

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

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

Controllo degli accessi ai tag

Per aggiungere, visualizzare ed eliminare i tag utilizzando l'API, i principali necessitano delle autorizzazioni di etichettatura nelle politiche IAM.

Puoi anche limitare queste autorizzazioni utilizzando le chiavi di condizione AWS globali per i tag. In AWS Payment Cryptography, queste condizioni possono controllare l'accesso alle operazioni di tagging, come e. TagResource UntagResource

Per esempi di policy e ulteriori informazioni, consulta Controllo dell'accesso in base alle chiavi di tag nella Guida per l'utente di IAM.

Le autorizzazioni per creare e gestire i tag funzionano come descritto di seguito.

crittografia dei pagamenti: TagResource

Consente ai principali di aggiungere o modificare tag. Per aggiungere tag durante la creazione di una chiave, il principale deve disporre dell'autorizzazione prevista da una policy IAM che non sia limitata a chiavi particolari.

crittografia dei pagamenti: ListTagsForResource

Consente ai committenti di visualizzare i tag sulle chiavi.

crittografia dei pagamenti: UntagResource

Consente ai committenti di eliminare i tag dalle chiavi.

Autorizzazioni ad assegnare tag nelle policy

Puoi fornire l'autorizzazione ad assegnare tag in una policy delle chiavi o in una policy IAM. Ad esempio, il seguente criterio chiave di esempio fornisce agli utenti selezionati l'autorizzazione di etichettatura sulla chiave. Fornisce a tutti gli utenti che possono assumere l'esempio dei ruoli di Amministratore o Sviluppatore il permesso di visualizzare i tag.

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": "*"
    }
  ]
}
```

Per concedere ai principali il permesso di taggare più chiavi, puoi utilizzare una policy IAM. Affinché questa policy sia efficace, la policy chiave per ciascuna chiave deve consentire all'account di utilizzare le policy IAM per controllare l'accesso alla chiave.

Ad esempio, la seguente politica IAM consente ai principali di creare chiavi. Inoltre, consente loro di creare e gestire tag su tutte le chiavi dell'account specificato. Questa combinazione consente ai principali di utilizzare il parametro tags dell'CreateKeyoperazione per aggiungere tag a una chiave durante la creazione.

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/*"
    }
  ]
}
```

Limitazione delle autorizzazioni ad assegnare tag

È possibile limitare le autorizzazioni di assegnazione dei tag utilizzando Condizioni della policy. Le seguenti condizioni della policy possono essere applicate alle autorizzazioni `payment-cryptography:TagResource` e `payment-cryptography:UntagResource`. Ad esempio, è possibile utilizzare la condizione `aws:RequestTag/`*`tag-key`* per consentire a un principale di aggiungere solo tag specifici o impedire a un principale di aggiungere tag con chiavi tag particolari.

- leggi: RequestTag
- aws:ResourceTag/tag-key (solo politiche IAM)

- aws: TagKeys

Come best practice quando utilizzi i tag per controllare l'accesso alle chiavi, utilizza la chiave di `aws:TagKeys` condizione `aws:RequestTag/`*`tag-key`* or per determinare quali tag (o tag key) sono consentiti.

Ad esempio, la seguente istruzione della policy IAM è simile a quella precedente. Tuttavia, questa policy consente ai principali di creare tag (`TagResource`) ed eliminare i tag `UntagResource` solo per i tag con chiave di tag `Project`.

Poiché `TagResource` le `UntagResource` richieste possono includere più tag, è necessario specificare un operatore `ForAllValues` or `ForAnyValue` set con la TagKeys condizione aws:. L'operatore `ForAnyValue` richiede che almeno una delle chiavi di tag nella richiesta corrisponda a una delle chiavi di tag nella policy. L'operatore `ForAllValues` richiede che tutte le chiavi di tag nella richiesta corrispondano a una delle chiavi di tag nella policy. L'`ForAllValues`operatore restituisce anche `true` se non ci sono tag nella richiesta, ma TagResource UntagResource fallisce quando non viene specificato alcun tag. Per dettagli sugli operatori del set, consulta Utilizzare più chiavi e valori nella Guida per l'utente di 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"}
        }
    }
  ]
}
```

Utilizzo dei tag per controllare l'accesso alle chiavi

È possibile controllare l'accesso alla crittografia dei AWS pagamenti in base ai tag sulla chiave. Ad esempio, puoi scrivere una policy IAM che consenta ai principali di abilitare e disabilitare solo le chiavi con un tag particolare. Oppure puoi utilizzare una policy IAM per impedire ai principali di utilizzare le chiavi nelle operazioni crittografiche a meno che la chiave non abbia un tag particolare.

Questa funzionalità fa parte del supporto di AWS Payment Cryptography per il controllo degli accessi basato sugli attributi (ABAC). Per informazioni sull'uso dei tag per controllare l'accesso alle AWS risorse, vedi A cosa serve ABAC? AWSe Controllo dell'accesso alle AWS risorse utilizzando i tag delle risorse nella Guida per l'utente IAM.

AWS Payment Cryptography supporta la chiave di contesto di condizione globale aws:ResourceTag/tag-key, che consente di controllare l'accesso alle chiavi in base ai tag sulla chiave. Poiché più chiavi possono avere lo stesso tag, questa funzione consente di applicare l'autorizzazione a un set selezionato di chiavi. È inoltre possibile modificare facilmente le chiavi del set modificando i relativi tag.

In AWS Payment Cryptography, la chiave `aws:ResourceTag/`*`tag-key`* condizionale è supportata solo nelle policy IAM. Non è supportata nelle policy chiave, che si applicano solo a una chiave, o nelle operazioni che non utilizzano una chiave particolare, come le ListAliases operazioni ListKeys or.

Il controllo dell'accesso con i tag offre un modo semplice, scalabile e flessibile per gestire le autorizzazioni. Tuttavia, se non è progettato e gestito correttamente, può consentire o negare l'accesso alle chiavi inavvertitamente. Se utilizzi tag per controllare l'accesso, prendi in considerazione le seguenti procedure.

- Utilizza i tag per rafforzare le best practice di Accesso meno privilegiato. Concedi ai responsabili IAM solo le autorizzazioni di cui hanno bisogno solo sulle chiavi che devono usare o gestire. Ad

esempio, utilizza i tag per etichettare le chiavi utilizzate per un progetto. Quindi concedi al team di progetto il permesso di utilizzare solo le chiavi con il tag di progetto.

- Fai attenzione a dare ai principali le autorizzazioni `payment-cryptography:TagResource` e `payment-cryptography:UntagResource` che consentono di aggiungere, modificare ed eliminare tag. Quando utilizzi i tag per controllare l'accesso alle chiavi, la modifica di un tag può autorizzare i responsabili a utilizzare chiavi che altrimenti non avrebbero il permesso di usare. Può anche negare l'accesso alle chiavi necessarie ad altri presidi per svolgere il proprio lavoro. Gli amministratori chiave che non dispongono dell'autorizzazione per modificare le politiche chiave o creare concessioni possono controllare l'accesso alle chiavi se dispongono dell'autorizzazione per gestire i tag.

  Quando possibile, utilizza una condizione politica, ad esempio `aws:TagKeys` per limitare le `aws:RequestTag/`*`tag-key`* autorizzazioni di etichettatura di un principale a determinati tag o modelli di tag su determinate chiavi.
- Controlla i tuoi committenti Account AWS che attualmente dispongono delle autorizzazioni di etichettatura e rimozione dei tag e modificali, se necessario. Le policy IAM potrebbero consentire autorizzazioni di tag e untag su tutte le chiavi. Ad esempio, la policy gestita dall'amministratore consente ai principali di etichettare, rimuovere i tag ed elencare i tag su tutte le chiavi.
- Prima di impostare una politica che dipenda da un tag, rivedi i tag sulle chiavi del tuo. Account AWS Assicurati che la tua policy si applichi solo ai tag che intendi includere. Utilizza CloudTrail registri e CloudWatch allarmi per avvisarti delle modifiche ai tag che potrebbero influire sull'accesso alle tue chiavi.
- Le condizioni delle policy basate su tag utilizzano la corrispondenza dei modelli; non sono legate a una particolare istanza di un tag. Una policy che utilizza chiavi di condizione basate su tag influisce su tutti i tag nuovi ed esistenti che corrispondono al modello. Se si elimina e si ricrea un tag che corrisponde a una condizione della policy, la condizione si applica al nuovo tag, proprio come quello precedente.

Ad esempio, considerare il seguente esempio di policy IAM. Consente ai principali di eseguire le operazioni di Decrypt solo sulle chiavi del tuo account che si trovano nella regione Stati Uniti orientali (Virginia settentrionale) e dotate di un tag. `"Project"="Alpha"` È possibile collegare questa policy ai ruoli nel progetto Alpha di esempio.

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"
        }
      }
    }
  ]
}
```

Il seguente esempio di politica IAM consente ai principali di utilizzare qualsiasi chiave dell'account per determinate operazioni crittografiche. Ma vieta ai principali di utilizzare queste operazioni crittografiche su chiavi con o senza 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"
        }
      }
    }
  ]
}
```

Comprendere gli attributi chiave per AWS Chiave di crittografia dei pagamenti

Un principio per una corretta gestione delle chiavi è che le chiavi hanno un ambito appropriato e possono essere utilizzate solo per le operazioni consentite. Pertanto, alcune chiavi possono essere create solo con determinate modalità di utilizzo. Ove possibile, ciò si allinea con le modalità di utilizzo disponibili definite da TR-31.

Sebbene la crittografia dei AWS pagamenti impedisca la creazione di chiavi non valide, qui vengono fornite combinazioni valide per comodità.

Chiavi simmetriche

- CHIAVE DI DERIVAZIONE TR31_B0_BASE_DERIVATION_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {= DeriveKey true}, {= true} NoRestrictions
- TR31_C0_CARD_VERIFICATION_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY*, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true} DeriveKey , {= true} NoRestrictions
- TR31_E1_EMV_MKEY_CONFIDENTIALITY
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true DeriveKey }, {= true} NoRestrictions
- TR31_E2_EMV_MKEY_INTEGRITY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true DeriveKey }, {= true} NoRestrictions
- TR31_E4_EMV_MKEY_DYNAMIC_NUMBERS
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true DeriveKey }, {= true} NoRestrictions
- TR31_E5_EMV_MKEY_CARD_PERSONALIZATION

  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true DeriveKey }, {= true} NoRestrictions
- TR31_E6_EMV_MKEY_OTHER
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128*, AES_192*, AES_256*
  - Combinazione consentita di modalità di utilizzo chiave: {= true DeriveKey }, {= true} NoRestrictions
- TR31_K0_KEY_ENCRYPTION_KEY
  - Si consiglia di utilizzare TR31_K1_KEY_BLOCK_PROTECTION_KEY. Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_K1_KEY_BLOCK_PROTECTION_KEY
  - Algoritmi chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {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
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M3_ISO_9797_3_MAC_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M6_ISO_9797_5_CMAC_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_M7_HMAC_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256

- Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions

- TR31_P0_PIN_ENCRYPTION_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}, {= true} NoRestrictions
- TR31_V1_IBM3624_PIN_VERIFICATION_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions
- TR31_V2_VISA_PIN_VERIFICATION_KEY
  - Algoritmi a chiave consentiti: TDES_2KEY, TDES_3KEY, AES_128, AES_192, AES_256
  - Combinazione consentita di modalità di utilizzo chiave: {Generate = true}, {Verify = true}, {Generate = true, Verify= true}, {= true} NoRestrictions

Chiavi asimmetriche

- TR31_D1_KEY_ASIMMETRIC_FOR_DATA_ENCRYPTION
  - Algoritmi a chiave consentiti: RSA_2048, RSA_3072, RSA_4096
  - Combinazione consentita di modalità di utilizzo chiave: {Encrypt = true, Decrypt = true, Wrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true, Unwrap = true}
  - NOTA: {Encrypt = true, Wrap = true} è l'unica opzione valida per l'importazione di una chiave pubblica destinata alla crittografia dei dati o al wrapping di una chiave
- TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE
  - Algoritmi a chiave consentiti: RSA_2048, RSA_3072, RSA_4096
  - Combinazione consentita di modalità di utilizzo chiave: {Sign = true}, {Verify = true}
  - NOTA: {Verify = true} è l'unica opzione valida per l'importazione di una chiave destinata alla firma, ad esempio un certificato radice, un certificato intermedio o un certificato di firma per. TR-34
- TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT
  - Utilizzato per algoritmi di accordo chiave come ECDH

  - Algoritmi chiave consentiti: ECC_NIST_P256, ECC_NIST_P384, ECC_NIST_P521
  - Combinazione consentita di modalità di utilizzo chiave: {= true}. DeriveKey
  - NOTA: DeriveKeyUsage viene utilizzato per specificare il tipo di chiave che verrà derivata da questa chiave di base. Questo è corretto in key creation/import.
- TR31_K2_TR34_ASIMMETRIC_KEY
  - Chiave asimmetrica utilizzata per meccanismi di scambio di chiavi compatibili come X9.24 TR-34
  - Algoritmi chiave consentiti: RSA_2048, RSA_3072, RSA_4096
  - Combinazione consentita di modalità di utilizzo chiave: {= true}. DeriveKey
  - Combinazione consentita di modalità di utilizzo dei tasti: {Encrypt = true, Decrypt = true, Unwrap = true, Unwrap = true}, {Encrypt = true, Wrap = true}, {Decrypt = true}
  - NOTA: {Encrypt = true, Wrap = true} è l'unica opzione valida per l'importazione di una chiave pubblica destinata alla crittografia dei dati o al wrapping di una chiave

* Questa combinazione algorithm/key di tipi non è attualmente supportata da alcuna operazione crittografica

Operazioni sui dati

Dopo aver stabilito una chiave AWS di crittografia dei pagamenti, questa può essere utilizzata per eseguire operazioni crittografiche. Diverse operazioni eseguono diversi tipi di attività, dalla crittografia all'hashing fino agli algoritmi specifici del dominio come la generazione CVV2.

I dati crittografati non possono essere decrittografati senza la chiave di decrittografia corrispondente (la chiave simmetrica o la chiave privata a seconda del tipo di crittografia). Analogamente, gli algoritmi di hashing e quelli specifici del dominio non possono essere verificati senza la chiave simmetrica o la chiave pubblica.

Per informazioni sui tipi di chiave validi per operazioni specifiche, consulta Chiavi valide per operazioni crittografiche

> **Note**
>
> Si consiglia di utilizzare i dati di test in un ambiente non di produzione. L'utilizzo di chiavi e dati di produzione (PAN, BDK ID, ecc.) in un ambiente non di produzione può influire sull'ambito di conformità, ad esempio per PCI DSS e PCI P2PE.

Argomenti

- Crittografa, decrittografa e ricrittografa i dati
- Genera e verifica i dati della carta
- Genera, traduci e verifica i dati PIN
- Genera e verifica ARQC
- Genera e verifica MAC
- Chiavi valide per operazioni crittografiche

Crittografa, decrittografa e ricrittografa i dati

I metodi di crittografia e decrittografia possono essere utilizzati per crittografare o decrittografare i dati utilizzando una varietà di tecniche simmetriche e asimmetriche tra cui TDES, AES e RSA. Questi metodi supportano anche chiavi derivate utilizzando le tecniche DUKPT ed EMV. Nei casi d'uso in

cui si desidera proteggere i dati con una nuova chiave senza esporre i dati sottostanti, è possibile utilizzare anche il ReEncrypt comando.

> **Note**
>
> Quando si utilizzano le encrypt/decrypt funzioni, si presume che tutti gli input siano in HexBinary, ad esempio un valore 1 verrà immesso come 31 (hex) e una t minuscola viene rappresentata come 74 (hex). Tutti gli output sono anche in HexBinary.

Per i dettagli su tutte le opzioni disponibili, consulta la Guida API per Encrypt, Decrypte Re-Encrypt.

Argomenti

- Crittografare i dati
- Decrittare i dati

Crittografare i dati

L'`Encrypt Data`API viene utilizzata per crittografare i dati utilizzando chiavi di crittografia dei dati simmetriche e asimmetriche, nonché chiavi derivate da DUKPT ed EMV. Sono supportati vari algoritmi e varianti, tra cui, e. `TDES` `RSA` `AES`

Gli input principali sono la chiave di crittografia utilizzata per crittografare i dati, i dati in testo semplice in formato HexBinary da crittografare e gli attributi di crittografia come il vettore di inizializzazione e la modalità per i cifrari a blocchi come TDES. I dati in chiaro devono essere espressi in multipli di 8 byte per, 16 byte per `TDES` e della lunghezza della chiave nel caso di. `AES` `RSA` Gli input chiave simmetrici (TDES, AES, DUKPT, EMV) devono essere aggiunti nei casi in cui i dati di input non soddisfino questi requisiti. La tabella seguente mostra la lunghezza massima del testo in chiaro per ogni tipo di chiave e il tipo di padding definito per le chiavi RSA. `EncryptionAttributes`

Tipo di imbottitura	RSA_2048	RSA_3072	RSA_4096
`OAEP SHA1`	428	684	940
`OAEP SHA256`	380	636	892
`OAEP SHA512`	252	508	764

Tipo di imbottitura	RSA_2048	RSA_3072	RSA_4096
PKCS1	488	744	1000
None	488	744	1000

Gli output primari includono i dati crittografati come testo cifrato in formato HexBinary e il valore di checksum per la chiave di crittografia. Per i dettagli su tutte le opzioni disponibili, consulta la Guida API per Encrypt.

Esempi

- Crittografa i dati utilizzando la chiave simmetrica AES
- Crittografa i dati utilizzando la chiave DUKPT
- Crittografa i dati utilizzando una chiave simmetrica derivata da EMV
- Crittografa i dati utilizzando una chiave RSA

Crittografa i dati utilizzando la chiave simmetrica AES

> **Note**
>
> Tutti gli esempi presuppongono che la chiave pertinente esista già. Le chiavi possono essere create utilizzando l'CreateKeyoperazione o importate utilizzando l'ImportKeyoperazione.

Example

In questo esempio, crittograferemo i dati in chiaro utilizzando una chiave simmetrica che è stata creata utilizzando l'CreateKeyOperazione o importata utilizzando l'Operazione. ImportKey Per questa operazione, la chiave deve essere impostata su e KeyModesOfUse impostata su. `Encrypt` KeyUsage `TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY` Per ulteriori opzioni, consulta la sezione Chiavi per le operazioni crittografiche.

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

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

Crittografa i dati utilizzando la chiave DUKPT

Example

In questo esempio, crittograferemo i dati in chiaro utilizzando una chiave DUKPT. AWS Supporti per la crittografia dei pagamenti e le chiavi DUKPT. `TDES AES` Per questa operazione, la chiave deve essere impostata `DeriveKey` e KeyModesOfUse KeyUsage impostata su. `TR31_B0_BASE_DERIVATION_KEY` Per ulteriori opzioni, consulta la sezione Chiavi per le operazioni crittografiche.

```
$ 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"
}
```

Crittografa i dati utilizzando una chiave simmetrica derivata da EMV

Example

In questo esempio, crittograferemo i dati di testo non crittografato utilizzando una chiave simmetrica derivata da EMV che è già stata creata. È possibile utilizzare un comando come questo per inviare dati a una scheda EMV. Per questa operazione, la chiave deve essere KeyModesOfUse impostata su `Derive` e KeyUsage impostata su `TR31_E1_EMV_MKEY_CONFIDENTIALITY` o`TR31_E6_EMV_MKEY_OTHER`. Per maggiori dettagli, consulta Keys for Cryptographic Operations.

```
$ 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"
}
```

Crittografa i dati utilizzando una chiave RSA

Example

In questo esempio, crittograferemo i dati in chiaro utilizzando una chiave pubblica RSA che è stata importata utilizzando l'operazione. ImportKey Per questa operazione, la chiave deve essere impostata su e KeyModesOfUse impostata su. `Encrypt` KeyUsage `TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION` Per ulteriori opzioni, consulta la sezione Chiavi per le operazioni crittografiche.

Per PKCS #7 o altri schemi di padding non attualmente supportati, richiedi prima di chiamare il servizio e seleziona no padding omettendo l'indicatore di 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"
}
```

Decrittare i dati

L'`Decrypt Data`API viene utilizzata per decrittografare i dati utilizzando chiavi di crittografia dei dati simmetriche e asimmetriche, nonché chiavi derivate DUKPT ed EMV. Sono supportati vari algoritmi e varianti, tra cui, e. `TDES` `RSA` `AES`

Gli input principali sono la chiave di decrittografia utilizzata per decrittografare i dati, i dati di testo cifrato in formato HexBinary da decrittografare e gli attributi di decrittografia come il vettore di inizializzazione, la modalità come i cifrari a blocchi ecc. Gli output principali includono i dati decrittografati come testo semplice in formato HexBinary e il valore di checksum per la chiave di decrittografia. Per i dettagli su tutte le opzioni disponibili, consulta la Guida API per Decrypt.

Esempi

- Decrittografa i dati utilizzando la chiave simmetrica AES
- Decrittografa i dati utilizzando la chiave DUKPT
- Decrittografa i dati utilizzando una chiave simmetrica derivata da EMV
- Decrittografa i dati utilizzando una chiave RSA

Decrittografa i dati utilizzando la chiave simmetrica AES

Example

In questo esempio, decifreremo i dati di testo cifrato utilizzando una chiave simmetrica. Questo esempio mostra una AES chiave ma sono anche supportate. TDES_2KEY TDES_3KEY Per questa operazione, la chiave deve essere KeyModesOfUse impostata Decrypt e KeyUsage impostata suTR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY. Per ulteriori opzioni, consulta la sezione Chiavi per le operazioni crittografiche.

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

Decrittografa i dati utilizzando la chiave DUKPT

> **Note**
>
> L'utilizzo dei dati di decrittografia con DUKPT per le transazioni P2PE può restituire all'applicazione i dati PAN della carta di credito e di altri titolari di carta di credito che dovranno essere presi in considerazione per determinare l'ambito PCI DSS.

Example

In questo esempio, decifreremo i dati di testo cifrato utilizzando una chiave DUKPT che è stata creata utilizzando l'Operazione o importata utilizzando l'Operazione. CreateKeyImportKey Per questa operazione, la chiave deve essere impostata e impostata su. KeyModesOfUse `DeriveKey` KeyUsage `TR31_B0_BASE_DERIVATION_KEY` Per ulteriori opzioni, consulta la sezione Chiavi per le operazioni crittografiche. Quando si utilizzaDUKPT, per l'`TDES`algoritmo, la lunghezza dei dati del testo cifrato deve essere un multiplo di 16 byte. Per l'`AES`algoritmo, la lunghezza dei dati del testo cifrato deve essere un multiplo di 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": "31323334313233343132333431323334"
}
```

Decrittografa i dati utilizzando una chiave simmetrica derivata da EMV

Example

In questo esempio, decifreremo i dati di testo cifrato utilizzando una chiave simmetrica derivata da EMV che è stata creata utilizzando l'operazione o importata utilizzando l'operazione. CreateKeyImportKey Per questa operazione, la chiave deve essere impostata su e impostata su o. KeyModesOfUse `Derive` KeyUsage `TR31_E1_EMV_MKEY_CONFIDENTIALITY` `TR31_E6_EMV_MKEY_OTHER` Per maggiori dettagli, consulta Keys for Cryptographic Operations.

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

Decrittografa i dati utilizzando una chiave RSA

Example

In questo esempio, decifreremo i dati di testo cifrato utilizzando una coppia di chiavi RSA creata utilizzando l'operazione. CreateKey Per questa operazione, la chiave deve essere impostata su enable e KeyModesOfUse impostata su. `Decrypt` KeyUsage `TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION` Per ulteriori opzioni, consulta Keys for Cryptographic Operations.

Per PKCS #7 o altri schemi di padding non attualmente supportati, seleziona no padding omettendo l'indicatore di padding 'Asymmetric= {} 'e rimuovi il padding dopo aver chiamato il servizio.

```
$ 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": "313233343132333431323334313233343"
}
```

Genera e verifica i dati della carta

Genera e verifica i dati delle carte incorpora i dati derivati dai dati delle carte, ad esempio CVV CVV2, CVC e DCVV.

Argomenti

- Genera i dati delle carte
- Verifica i dati della carta

Genera i dati delle carte

L'`Generate Card Data`API viene utilizzata per generare i dati delle carte utilizzando algoritmi come CVV o Dynamic. CVV2 CVV2 Per vedere quali chiavi possono essere utilizzate per questo comando, consulta la sezione Chiavi valide per le operazioni crittografiche.

Molti valori crittografici come CVV, iCVV CVV2, CAVV V7 utilizzano lo stesso algoritmo crittografico ma variano i valori di input. Ad esempio CardVerificationValue1 ha gli input di, numero di carta e data di scadenza. ServiceCode Sebbene CardVerificationValue2 abbia solo due di questi input, ciò è dovuto al fatto che per CVV2/CVC2, ServiceCode è fissato a 000. Analogamente, per iCVV ServiceCode è fissato a 999. Alcuni algoritmi possono riutilizzare i campi esistenti, come CAVV V8, nel qual caso sarà necessario consultare il manuale del provider per i valori di input corretti.

> **Note**
>
> La data di scadenza deve essere inserita nello stesso formato (ad esempio MMYY o YYMM) affinché la generazione e la convalida producano risultati corretti.

Genera CVV2

Example

In questo esempio, genereremo un messaggio CVV2 per un determinato PAN con gli input PAN e la data di scadenza della carta. Ciò presuppone che sia stata generata una chiave di verifica della carta.

```
$ 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"
  }
```

Genera iCVV

Example

In questo esempio, genereremo un iCVV per un determinato PAN con gli input diPAN, un codice di servizio 999 e la data di scadenza della carta. Ciò presuppone che sia stata generata una chiave di verifica della carta.

Per tutti i parametri disponibili, vedi CardVerificationValue1 nella guida di riferimento dell'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"
}
```

Verifica i dati della carta

`Verify Card Data`viene utilizzato per verificare i dati che sono stati creati utilizzando algoritmi di pagamento che si basano su principi di crittografia come. `DISCOVER_DYNAMIC_CARD_VERIFICATION_CODE`

I valori di input vengono in genere forniti come parte di una transazione in entrata a un emittente o a un partner della piattaforma di supporto. Per verificare un crittogramma ARQC (utilizzato per le schede con chip EMV), consulta Verify ARQC.

Per ulteriori informazioni, consulta la guida alle API. VerifyCardValidationData

Se il valore è verificato, l'api restituirà http/200. Se il valore non è verificato, restituirà http/400.

Verifica CVV2

Example

In questo esempio, convalideremo un CVV/ CVV2 per un determinato PAN. In genere CVV2 viene fornito dal titolare della carta o dall'utente durante il momento della transazione per la convalida. Per convalidare i dati immessi, in fase di esecuzione verranno forniti i seguenti valori: Key to Use for validation (CVK)PAN, data di scadenza della carta e immessi. CVV2 Il formato di scadenza della carta deve corrispondere a quello utilizzato nella generazione iniziale del valore.

Per tutti i parametri disponibili, vedi CardVerificationValue2 nella guida di riferimento dell'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"
  }
```

Verifica iCVV

Example

In questo esempio, verificheremo un iCVV per un determinato PAN inserendo i seguenti campi: Key to Use for validation (CVK), un codice di servizio 999PAN, la data di scadenza della carta e l'iCVV fornito dalla transazione per la convalida.

iCVV non è un valore inserito dall'utente (come) ma è incorporato in una scheda EMV. CVV2 Si dovrebbe valutare se debba sempre essere convalidato quando fornito.

Per tutti i parametri disponibili, vedere, CardVerificationValue1 nella guida di riferimento dell'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"
}
```

Genera, traduci e verifica i dati PIN

Le funzioni relative ai dati PIN consentono di generare pin casuali, valori di verifica dei pin (PVV) e convalidare i pin crittografati in entrata rispetto a PVV o offset PIN.

La traduzione dei pin consente di tradurre un pin da una chiave funzionante a un'altra senza esporlo in testo non crittografato, come specificato dal requisito 1 del PIN PCI.

> **Note**
>
> Poiché la generazione e la convalida del PIN sono in genere funzioni dell'emittente e la traduzione del PIN è una tipica funzione dell'acquirente, consigliamo di prendere in

considerazione l'accesso con privilegi meno privilegiati e di impostare le politiche appropriate per il caso d'uso del sistema.

Argomenti

- Traduci i dati PIN
- Genera dati PIN
- Verifica i dati del PIN

Traduci i dati PIN

Le funzioni Translate PIN vengono utilizzate per tradurre i dati PIN crittografati da un set di chiavi all'altro senza che i dati crittografati escano dall'HSM. Viene utilizzato per la crittografia P2PE in cui le chiavi di lavoro devono cambiare ma il sistema di elaborazione non deve o non è autorizzato a decrittografare i dati. Gli input principali sono i dati crittografati, la chiave di crittografia utilizzata per crittografare i dati, i parametri utilizzati per generare i valori di input. L'altro set di input è costituito dai parametri di output richiesti, ad esempio la chiave da utilizzare per crittografare l'output e i parametri utilizzati per creare tale output. Gli output principali sono un nuovo set di dati crittografato e i parametri utilizzati per generarlo.

> **Note**
>
> Per la conformità PCI, i valori in entrata e in uscita PrimaryAccountNumber devono corrispondere. La conversione di un PIN da un PAN a un altro non è consentita.

Argomenti

- PIN da PEK a DUKPT
- PIN da PEK a PEK

PIN da PEK a DUKPT

Example

In questo esempio, tradurremo un PIN da un blocco AES ISO 4 PIN utilizzando la crittografia da DUKPT a PEK TDES utilizzando il blocco PIN ISO 0. Ciò è comune quando un terminale di pagamento crittografa un pin in ISO 4 e quindi può essere ricondotto in TDES per l'elaborazione a valle se la connessione successiva non supporta ancora 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 da PEK a PEK

Example

In questo esempio, traduciamo un PIN crittografato con un PEK (PIN Encryption Key) in un altro PEK. Viene comunemente utilizzato per instradare transazioni tra diversi sistemi o partner che utilizzano chiavi di crittografia diverse, mantenendo al contempo la conformità al PCI PIN mantenendo il PIN crittografato durante l'intero processo. Entrambe le chiavi utilizzano la crittografia TDES 3KEY in questo esempio, ma sono disponibili diverse opzioni, tra cui AES ISO-4 a TDES ISO-0, DUKPT a PEK o AS2805 a PEK.

```
$ 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"
}
```

Il blocco PIN di uscita è ora crittografato con il secondo PEK e può essere trasmesso in modo sicuro al sistema a valle che contiene la chiave corrispondente.

Genera dati PIN

Le funzioni di generazione dei dati PIN vengono utilizzate per generare PIN-related valori, come PVV e i pin block offset utilizzati per convalidare l'immissione dei pin da parte degli utenti durante la transazione o l'autorizzazione. Questa API può anche generare un nuovo pin casuale utilizzando vari algoritmi.

Genera un pin casuale e corrispondente a Visa PVV

Example

In questo esempio, genereremo un nuovo pin (casuale) in cui gli output saranno un pin crittografato `PIN block` (PinData.PinBlock) e un PVV (pinData.Offset). Gli input chiave sonoPAN, thePin Verification Key, the `Pin Encryption Key` e the. `PIN block format`

Questo comando richiede che la chiave sia di tipo`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"
            }
        }
```

Genera un Visa PVV per un pin noto

Example

In questo esempio, genereremo un PVV per un determinato pin (crittografato). Un pin crittografato può essere ricevuto a monte, ad esempio da un terminale di pagamento o dal titolare della carta utilizzando il flusso di pin selezionabile dall'utente. Gli input chiave sonoPAN, thePin Verification Key, the e thePin Encryption Key. `Encrypted Pin Block` `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"
            }
        }
```

Genera l'offset del pin IBM3624 per un pin

IBM 3624 PIN Offset è talvolta chiamato anche metodo IBM. Questo metodo genera un natural/intermediate PIN utilizzando i dati di convalida (in genere il PAN) e una chiave PIN (PVK). I pin naturali sono effettivamente un valore derivato e, essendo deterministici, sono molto efficienti da gestire per l'emittente perché non è necessario archiviare i dati dei pin a livello di titolare della carta. Lo svantaggio più evidente è che questo schema non tiene conto dei pin selezionabili o casuali dal titolare della carta. Per consentire questi tipi di pin, allo schema è stato aggiunto un algoritmo di offset. L'offset rappresenta la differenza tra il pin selezionato dall'utente (o a caso) e la chiave naturale. Il valore dell'offset viene memorizzato dall'emittente o dal processore della carta.

Al momento della transazione, il servizio AWS Payment Cryptography ricalcola internamente il pin naturale e applica l'offset per trovarlo. Quindi lo confronta con il valore fornito dall'autorizzazione alla transazione.

Esistono diverse opzioni per IBM3624:

- `Ibm3624NaturalPin`emetterà il pin naturale e un blocco di pin crittografato
- `Ibm3624PinFromOffset`genererà un blocco di pin crittografato dato un offset
- `Ibm3624RandomPin`genererà un pin casuale e quindi l'offset corrispondente e il blocco di pin crittografato.
- `Ibm3624PinOffset`genera l'offset del pin in base a un pin selezionato dall'utente.

Internamente alla crittografia dei AWS pagamenti, vengono eseguiti i seguenti passaggi:

- Riempi la panoramica fornita a 16 caratteri. Se viene fornito <16, riempi il campo sul lato destro utilizzando il carattere di riempimento fornito.
- Crittografa i dati di convalida utilizzando la chiave di generazione del PIN.
- Decimalizza i dati crittografati utilizzando la tabella di decimalizzazione. Questo associa le cifre esidecimali alle cifre decimali, ad esempio «A» può corrispondere a 9 e 1 può corrispondere a 1.
- Ottieni le prime 4 cifre da una rappresentazione esadecimale dell'output. Questa è la spilla naturale.
- Se è stato selezionato un pin casuale o selezionato dall'utente, il modulo sottrae il pin naturale dal pin del cliente. Il risultato è l'offset del pin.

Esempi

- Esempio: generare un offset di pin IBM3624 per un pin

Esempio: generare un offset di pin IBM3624 per un pin

In questo esempio, genereremo un nuovo pin (casuale) in cui gli output saranno un valore crittografato `PIN block` (PinData.PinBlock) e un valore di `IBM3624` offset (pin). Data.Offset Gli input sono `PAN` dati di convalida (in genere il pan), il carattere di riempimento, the, the e the`Pin Verification Key`. `Pin Encryption Key` `PIN block format`

Questo comando richiede che la chiave di generazione del pin sia del tipo `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` e la chiave di crittografia sia del tipo `TR31_P0_PIN_ENCRYPTION_KEY`

Example

L'esempio seguente mostra la generazione di un pin casuale e l'emissione del blocco di pin crittografato e del valore di offset IBM3624 utilizzando Ibm3624RandomPin

```
$ 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"
             }
         }
```

Verifica i dati del PIN

Le funzioni di verifica dei dati del PIN vengono utilizzate per verificare se un PIN è corretto. Ciò comporta in genere il confronto del valore del pin precedentemente memorizzato con quello inserito dal titolare della carta in un POI. Queste funzioni confrontano due valori senza rivelare il valore sottostante di nessuna delle due fonti.

Convalida il PIN crittografato utilizzando il metodo PVV

Example

In questo esempio, convalideremo un PIN per un determinato PAN. Il PIN viene in genere fornito dal titolare della carta o dall'utente durante il momento della transazione per la convalida e viene confrontato con il valore in archivio (l'input del titolare della carta viene fornito come valore crittografato dal terminale o da un altro provider upstream). Per convalidare questo input, verranno forniti anche i seguenti valori in fase di esecuzione: la chiave utilizzata per crittografare il pin di input (spesso è un`IWK`) PAN e il valore con cui effettuare la verifica (a o). `PVV` `PIN offset`

Se AWS Payment Cryptography è in grado di convalidare il PIN, viene restituito un. http/200 Se il PIN non viene convalidato, restituirà un. 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",

        }
```

Convalida il PIN crittografato utilizzando il metodo PVV - error bad pin

Example

In questo esempio, cercheremo di convalidare un PIN per un determinato PAN ma falliremo perché il PIN non è corretto.

Quando si utilizzano gli SDK, questo appare come {"Message» :"Verifica del blocco PIN non riuscita». , "Reason» :"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.
```

Convalida il PIN crittografato utilizzando il metodo PVV: errori di input errati

Example

In questo esempio, tenteremo di convalidare un PIN per un determinato PAN ma fallirà a causa di input errati e i dati in arrivo non erano un pin valido. Le cause più comuni sono: 1/wrong la chiave utilizzata, 2/input i parametri come il formato pan o pin block sono errati, il 3/pin blocco è danneggiato.

```
$ 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.
```

Convalida un PIN rispetto all'offset dei pin IBM3624 precedentemente memorizzato

In questo esempio, si convalida un PIN fornito dal titolare della carta rispetto all'offset del pin memorizzato nel file presso l'emittente o il processore della carta. Gli input includono la chiave di verifica (PVK), la chiave di crittografia (PEK), il numero di conto principale, il blocco PIN crittografato del terminale di pagamento (o di un altro fornitore a monte, ad esempio una rete di carte) e l'offset del PIN precedentemente memorizzato. Se il PIN corrisponde, l'API restituisce una risposta. HTTP/200

Questo comando richiede che la chiave di verifica del pin sia del tipo `TR31_V1_IBM3624_PIN_VERIFICATION_KEY` e la chiave di crittografia sia del tipo`TR31_P0_PIN_ENCRYPTION_KEY`.

> ⚠ Convalida della lunghezza del PIN
>
> Il `--pin-data-length` parametro opzionale specifica la lunghezza del PIN da verificare (da 4 a 12 cifre). Se non si specifica questo parametro, il servizio convalida l'intera lunghezza

del PIN protetta dal valore di offset. Se desideri convalidare solo una parte del PIN, utilizza il parametro. `--pin-data-length` Ad esempio, se `--pin-data-length` è impostato su 4 ma il PIN effettivo è di 6 cifre, il servizio convalida solo le 4 cifre più a destra. Le 2 cifre rimanenti non vengono controllate. Potresti farlo intenzionalmente se il PIN del titolare della carta è composto da 6 cifre ma il PIN in entrata è composto da sole 4 cifre, ad esempio quando la rete o il commerciante in entrata supporta solo un PIN di 4 cifre.

Example

```
$ 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 \
    --encrypted-pin-block AC17DC148BDA645E \
    --pin-data-length 4 \
    --verification-attributes
 Ibm3624PinOffset="{DecimalizationTable=9876543210654321,PinValidationDataPadCharacter=D,PinVal
```

```
{
    "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"
}
```

Genera e verifica ARQC

Un Authorization Request Cryptogram (ARQC) è un crittogramma generato da una chip card EMV (o un equivalente digitale come un portafoglio mobile) durante una transazione di pagamento. L'ARQC convalida crittograficamente la carta e garantisce l'integrità dei dati della transazione. Incorpora i dati delle carte, i dati dei terminali e i dati specifici della transazione per produrre un valore unico per ogni transazione.

AWS Payment Cryptography supporta due operazioni ARQC. È possibile generare un ARQC per scopi di sviluppo e test e verificare un ARQC per i flussi di autorizzazione delle transazioni di produzione.

Argomenti

- Genera crittogramma di richiesta di autenticazione (ARQC)
- Crittogramma Verify Auth Request (ARQC)

Genera crittogramma di richiesta di autenticazione (ARQC)

L'API di crittografia generate auth request viene utilizzata per generare ARQC. Questa API consente di generare ARQC utilizzando AWS Payment Cryptography per scopi di sviluppo e test.

> **Important**
>
> Questa operazione è destinata esclusivamente a scenari di sviluppo e test. Non è consigliabile utilizzare questa operazione come sostituto della generazione di crittogrammi basata su carte nei flussi di pagamento di produzione.

Per tutte le opzioni disponibili, consulta l'API GenerateAuthRequestCryptogramReference.

I crittogrammi ARQC richiedono in genere i seguenti input (sebbene ciò possa variare in base all'implementazione):

- PAN - Specificato nel campo PrimaryAccountNumber
- Numero di sequenza PAN (PSN): specificato nel campo PanSequenceNumber
- Metodo di derivazione delle chiavi come Common Session Key (CSK) - Specificato nel SessionKeyDerivationAttributes
- Modalità di derivazione della chiave principale (ad esempio EMV Option A): specificata nella MajorKeyDerivationMode
- Dati sulle transazioni: una stringa di dati relativi a transazioni, terminali e carte, ad esempio Importo e Data. Specificato nel TransactionData campo.
- Chiave principale dell'emittente: la chiave master utilizzata per derivare la chiave del crittogramma (AC). Questa chiave protegge le singole transazioni ed è specificata nel campo. KeyIdentifier

Per ulteriori informazioni sulla creazione e il riempimento dei dati delle transazioni, consulta il crittogramma Verify auth request (ARQC).

Argomenti

- Esempi

Esempi

Visa CVN10

Example

L'esempio seguente genera un ARQC utilizzando Visa CVN10.

Se AWS Payment Cryptography genera correttamente l'ARQC, ne viene restituito un con il crittogramma generato. http/200

```
$ aws payment-cryptography-data generate-auth-request-cryptogram \
--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",
  "AuthRequestCryptogram": "D791093C8A921769"
}
```

Visa CVN18 e Visa CVN22

Example

L'esempio seguente genera un ARQC utilizzando Visa CVN18 o CVN22. Le operazioni crittografiche sono le stesse tra CVN18 e CVN22, ma i dati contenuti nei dati delle transazioni variano. Rispetto a CVN10, viene generato un crittogramma completamente diverso anche con gli stessi input.

Se AWS Payment Cryptography genera correttamente l'ARQC, ne viene restituito uno con il crittogramma generato. http/200

```
$ aws payment-cryptography-data generate-auth-request-cryptogram \
--key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
--major-key-derivation-mode EMV_OPTION_A \
--transaction-data
 0000000017000000000000000840008000800008401605170000000093800000B1F22010300000000000
 \
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",
  "AuthRequestCryptogram": "61EDCC708B4C97B4"
}
```

Crittogramma Verify Auth Request (ARQC)

L'API del crittogramma di verifica della richiesta di autenticazione viene utilizzata per verificare l'ARQC. La generazione dell'ARQC viene in genere eseguita su una Chip Card EMV (o un equivalente digitale come un portafoglio mobile) durante il periodo di autorizzazione della transazione, sebbene il servizio possa generare dati per scopi di sviluppo. Un ARQC è unico per ogni transazione e ha lo scopo di mostrare crittograficamente sia la validità della carta che di garantire che i dati della transazione corrispondano esattamente alla transazione corrente (prevista).

AWS La crittografia dei pagamenti offre una varietà di opzioni per la convalida dell'ARQC e la generazione di valori ARPC opzionali, inclusi quelli definiti in EMV 4.4 Book 2 e altri schemi utilizzati

da Visa e Mastercard. Per un elenco completo di tutte le opzioni disponibili, consulta la sezione della Guida API. VerifyCardValidationData

I crittogrammi ARQC richiedono in genere i seguenti input (sebbene ciò possa variare in base all'implementazione):

- PAN - Specificato nel campo PrimaryAccountNumber
- Numero di sequenza PAN (PSN) - specificato nel campo PanSequenceNumber
- Metodo di derivazione delle chiavi come Common Session Key (CSK) - Specificato nel SessionKeyDerivationAttributes
- Modalità di derivazione della chiave principale (ad esempio EMV Option A): specificata nella MajorKeyDerivationMode
- Dati sulle transazioni: una stringa di vari dati relativi a transazioni, terminali e carte, come Importo e Data, specificati nel campo TransactionData
- Chiave principale dell'emittente: la chiave master utilizzata per derivare la chiave crittografica (AC) utilizzata per proteggere le singole transazioni e specificata nel campo KeyIdentifier

Argomenti

- Creazione di dati sulle transazioni
- Riempimento dei dati delle transazioni
- Esempi

Creazione di dati sulle transazioni

Il contenuto esatto (e l'ordine) del campo di dati della transazione varia in base all'implementazione e allo schema di rete, ma i campi minimi consigliati (e la sequenza di concatenazione) sono definiti nel Libro 2 di EMV 4.4, Sezione 8.1.1 - Selezione dei dati. Se i primi tre campi sono importo (17,00), altro importo (0,00) e paese di acquisto, i dati della transazione inizierebbero come segue:

- 000000001700 - importo - 12 posizioni implicavano un decimale a due cifre
- 000000000000 - altro importo - 12 posizioni implicavano un decimale a due cifre
- 0124 - prefisso internazionale a quattro cifre
- Dati di transazione (parziali) in uscita - 00000000170000000000000000124

Riempimento dei dati delle transazioni

I dati delle transazioni devono essere aggiunti prima dell'invio al servizio. La maggior parte degli schemi utilizza il padding ISO 9797 Metodo 2, in cui una stringa esadecimale viene aggiunta dall'esadecimale 80 seguito da 00 fino a quando il campo non è un multiplo della dimensione del blocco di crittografia; 8 byte o 16 caratteri per TDES e 16 byte o 32 caratteri per AES. L'alternativa (metodo 1) non è così comune, ma utilizza solo 00 come caratteri di riempimento.

ISO 9797 Metodo 1: Imbottitura

Senza imbottitura:
0000000000000000000840008000800008401605170000000093800000B03011203 (74 caratteri o 37 byte)

Imbottito: 0000000000000000000840008000084016051700000000093800000B03011203 000000 (80 caratteri o 40 byte)

Imbottitura ISO 9797 Metodo 2

Senza imbottitura:
0000000000000000000840008000800008401605170000000093800000B1F220103000000 (80 caratteri o 40 byte)

Imbottito: 0000000000000008400080000840160517000000009380000 0B1F220103000000 8000000000000000 (88 caratteri o 44 byte)

Esempi

Visa CVN10

Example

In questo esempio, convalideremo un ARQC generato utilizzando Visa CVN10.

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un. http/200 Se quindi ARCQ (Authorization Request Cryptogram) non viene convalidato, restituirà una risposta. 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 e Visa CVN22

Example

In questo esempio, convalideremo un ARQC generato utilizzando Visa CVN18 o CVN22. Le operazioni crittografiche sono le stesse tra CVN18 e CVN22, ma i dati contenuti nei dati delle transazioni variano. Rispetto a CVN10, viene generato un crittogramma completamente diverso anche con gli stessi input.

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un. http/200 Se l'ARCQ non è convalidato, restituirà un. 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"
}
```

Genera e verifica MAC

I codici di autenticazione dei messaggi (MAC) vengono in genere utilizzati per autenticare l'integrità di un messaggio (indipendentemente dal fatto che sia stato modificato). Gli hash crittografici come HMAC (Hash-Based Message Authentication Code) CBC-MAC e CMAC (Cipher-based Message Authentication Code) forniscono un'ulteriore garanzia del mittente del MAC utilizzando la crittografia. HMAC si basa su funzioni hash mentre CMAC si basa su cifrari a blocchi. Il servizio supporta anche gli algoritmi ISO9797 1 e 3, che sono tipi di. CBC-MACs

Tutti gli algoritmi MAC di questo servizio combinano una funzione hash crittografica e una chiave segreta condivisa. Accettano un messaggio e una chiave segreta, ad esempio il materiale chiave contenuto in una chiave, e restituiscono un tag o mac univoco. Se anche solo un carattere del messaggio cambia o se la chiave segreta cambia, il tag risultante è completamente diverso. Richiedendo una chiave segreta, i MAC crittografici garantiscono anche l'autenticità; è impossibile generare un mac identico senza la chiave segreta. I MAC crittografici sono talvolta chiamati firme simmetriche, perché funzionano come le firme digitali, ma utilizzano un'unica chiave sia per la firma che per la verifica.

AWS Payment Cryptography supporta diversi tipi di MAC:

ALGORITMO ISO9797 1

Denotato da ISO9797_ALGORITHM1. `KeyUsage` Se il campo non è un multiplo della dimensione del blocco (8 caratteri bytes/16 esadecimali per TDES, 16 bytes/32 caratteri per AES), Payment Cryptography applica automaticamente il metodo di riempimento ISO9797. AWS Se sono necessari altri metodi di riempimento, è possibile applicarli prima di chiamare il servizio.

ALGORITMO ISO9797 3 (RETAIL MAC)

Denotato da ISO9797_ALGORITHM3. `KeyUsage` Le stesse regole di riempimento si applicano all'algoritmo 1

ALGORITMO ISO9797 5 (CMAC)

Denotato con o TR31_M6_ISO_9797_5_CMAC_KEY `KeyUsage`

HMAC

Denotato con `KeyUsage` TR31_M7_HMAC_KEY, inclusi HMAC_SHA224, HMAC_SHA256, HMAC_SHA384 e HMAC_SHA512

AS2805.4.1 MAC

Denotato da `KeyUsage` TR31_M0_ISO_16609_MAC_KEY. Per ulteriori dettagli su AS2805, vedere ???

DUMP MAC

DUKPT MAC viene in genere utilizzato per confermare la fonte e il payload dei terminali di pagamento dei messaggi. to/from Deriva una chiave utilizzando le tecniche di derivazione DUKPT e quindi esegue il MAC. Le chiavi utilizzate con questa opzione sono contrassegnate da TR31_B0_BASE_DERIVATION_KEY. `KeyUsage`

EMV MAC

> EMV MAC viene generalmente definito come una chiave di integrità nella documentazione EMV. Ricava una chiave utilizzando tecniche di derivazione EMV e quindi utilizza internamente ISO9797_ALGORITHM3. In genere viene utilizzato per inviare gli script dell'emittente a una chip card per la riprogrammazione. Le chiavi utilizzate con questa opzione sono contrassegnate da TR31_E2_EMV_MKEY_INTEGRITY. `KeyUsage` Se state inviando uno script e aggiornando un pin offline, verificate che esegua entrambe queste operazioni. GenerateMacEmvPinChange

Argomenti

- Genera un MAC
- Verifica MAC

Genera un MAC

L'API Generate MAC viene utilizzata per autenticare i dati relativi alle carte, come tenere traccia dei dati provenienti dalla banda magnetica di una scheda, utilizzando chiavi crittografiche note per generare un MAC (Message Authentication Code) per la convalida dei dati tra le parti che inviano e ricevono. I dati utilizzati per generare un MAC includono i dati dei messaggi, la chiave di crittografia MAC segreta e l'algoritmo MAC per generare un valore MAC univoco per la trasmissione. La parte ricevente del MAC utilizzerà gli stessi dati dei messaggi MAC, la stessa chiave di crittografia MAC e lo stesso algoritmo per riprodurre un altro valore MAC per il confronto e l'autenticazione dei dati. Anche se un carattere del messaggio cambia o la chiave MAC utilizzata per la verifica non è identica, il valore MAC risultante è diverso. L'API supporta ISO 9797-1 Algorithm 1 e ISO 9797-1 Algorithm 3 MAC (utilizzando una chiave MAC statica e una chiave DUKPT derivata), chiavi di crittografia MAC HMAC ed EMV MAC per questa operazione.

Il `message-data` valore di input per deve essere un dato HexBinary.

Per ulteriori informazioni su tutte le opzioni di questa API, consulta GenerateMace VerifyMac.

Il parametro opzionale mac-length consente di troncare il valore di output (sebbene ciò possa essere fatto anche all'interno del codice). Una lunghezza di 8 si riferisce a 8 byte o 16 caratteri esadecimali.

Le chiavi MAC possono essere create con AWS Payment Cryptography chiamando CreateKeyo importate chiamando. ImportKey

> **Note**
>
> Gli algoritmi CMAC e HMAC non richiedono il padding. Tutti gli altri richiedono che i dati vengano sommati alla dimensione del blocco dell'algoritmo, ovvero multipli di 8 byte (16 caratteri esadecimali) per TDES e 16 byte (32 caratteri esadecimali) per AES.

Esempi

- Genera HMAC
- Genera MAC usando l'algoritmo ISO 9797-1 3
- Genera MAC usando CMAC
- Genera MAC usando DUKPT CMAC

Genera HMAC

In questo esempio, genereremo un HMAC (Hash-Based Message Authentication Code) per l'autenticazione dei dati delle carte utilizzando l'algoritmo HMAC `HMAC_SHA256` e la chiave di crittografia HMAC. La chiave deve essere KeyUsage impostata su e su. `TR31_M7_HMAC_KEY` KeyModesOfUse `Generate` La lunghezza dell'hash (ad esempio 256) viene definita al momento della creazione della chiave e non può essere modificata.

Il parametro opzionale mac-length taglierà l'output MAC, sebbene ciò possa essere eseguito anche all'esterno del servizio. Questo valore è in byte, quindi un valore di 16 si aspetterà una stringa esadecimale di lunghezza 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"
}
```

Genera MAC usando l'algoritmo ISO 9797-1 3

In questo esempio, genereremo un MAC utilizzando l'algoritmo 3 ISO 9797-1 (Retail MAC) per l'autenticazione dei dati delle carte. La chiave deve essere KeyUsage impostata su e su. `TR31_M3_ISO_9797_3_MAC_KEY` KeyModesOfUse `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"
}
```

Genera MAC usando CMAC

CMAC è più comunemente usato quando le chiavi sono AES, ma supporta anche TDES. In questo esempio, genereremo un MAC utilizzando CMAC (ISO 9797-1 Algorithm 5) per l'autenticazione dei dati delle carte con una chiave AES. La chiave deve essere KeyUsage impostata su e su. `TR31_M6_ISO_9797_5_CMAC_KEY` KeyModesOfUse `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"
}
```

Genera MAC usando DUKPT CMAC

Questo esempio genera un MAC utilizzando DUKPT (Derived Unique Key Per Transaction) con CMAC per l'autenticazione dei dati delle carte. La chiave deve essere impostata su `TR31_B0_BASE_DERIVATION_KEY` e KeyUsage KeyModesOfUse `DeriveKey` impostata su true.

Le chiavi DUKPT derivano una chiave unica per ogni transazione utilizzando una Base Derivation Key (BDK) e un Key Serial Number (KSN).

Example

```
$ aws payment-cryptography-data generate-mac --key-identifier arn:aws:payment-
cryptography:us-east-2:111122223333:key/qnobl5lghrzunce6 --message-data
 "3b3130383834393030313037333934313533d323430383232363030303730303033f33" --generation-
attributes="DukptCmac={KeySerialNumber="932A6E954ABB32DD00000001",DukptKeyVariant=BIDIRECTIONAL
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6",
    "KeyCheckValue": "C1EB8F",
    "Mac": "1F8C36E63F91E4E93DF7842BF5E2E5F7"
}
```

Verifica MAC

L'API Verify MAC viene utilizzata per verificare il MAC (Message Authentication Code) per l'autenticazione dei dati relativi alla carta. Deve utilizzare la stessa chiave di crittografia utilizzata durante la generazione del MAC per riprodurre il valore MAC per l'autenticazione. La chiave di crittografia MAC può essere creata con AWS Payment Cryptography chiamando CreateKeyo importata chiamando. ImportKey L'API supporta le chiavi di crittografia DUKPT MAC, HMAC ed EMV MAC per questa operazione.

Se il valore è verificato, `MacDataVerificationSuccessful` verrà restituito il parametro di rispostaHttp/200, altrimenti `Http/400` con un messaggio che lo indica. `Mac verification failed`

Esempi

- Verifica HMAC
- Verifica il MAC utilizzando DUKPT CMAC

Verifica HMAC

In questo esempio, verificheremo un HMAC (Hash-Based Message Authentication Code) per l'autenticazione dei dati delle carte utilizzando l'algoritmo HMAC `HMAC_SHA256` e la chiave di crittografia HMAC. La chiave deve essere impostata su `TR31_M7_HMAC_KEY` e KeyUsage KeyModesOfUse `Verify` impostata su true.

Example

```
$ aws payment-cryptography-data verify-mac \
     --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
qnobl5lghrzunce6 \
     --message-data
 "3b343038383439303031303733393431353d323430383232363030303730303f33" \
     --mac ED87F26E961C6D0DDB78DA5038AA2BDDEA0DCE03E5B5E96BDDD494F4A7AA470C \
     --verification-attributes Algorithm=HMAC_SHA256
```

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

Verifica il MAC utilizzando DUKPT CMAC

In questo esempio, verificheremo un MAC utilizzando DUKPT (Derived Unique Key Per Transaction) con CMAC per l'autenticazione dei dati delle carte. La chiave deve essere impostata su `TR31_B0_BASE_DERIVATION_KEY` e KeyUsage KeyModesOfUse `DeriveKey` impostata su true. Le chiavi DUKPT derivano una chiave unica per ogni transazione utilizzando una Base Derivation Key (BDK) e un Key Serial Number (KSN). Il valore di DukptKeyVariant deve corrispondere tra mittente e destinatario. REQUEST verrà in genere utilizzato da terminale a backend, VERIFY da backend a terminale e BIDIRECTIONAL quando viene utilizzata una singola chiave in entrambe le direzioni.

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"
}
```

Chiavi valide per operazioni crittografiche

Alcuni tasti possono essere utilizzati solo per determinate operazioni. Inoltre, alcune operazioni possono limitare le modalità di utilizzo dei tasti. Consulta la seguente tabella per le combinazioni consentite.

> **Note**
>
> Alcune combinazioni, sebbene consentite, possono creare situazioni inutilizzabili, ad esempio la generazione di codici CVV (`generate`) ma l'impossibilità di verificarli. (`verify`)

Argomenti

- GenerateCardData
- VerifyCardData
- GeneratePinData (per VISA/ABA schemi)
- GeneratePinData (per IBM3624)
- VerifyPinData (per VISA/ABA schemi)
- VerifyPinData (per IBM3624)

- Decrittografia dei dati
- Encrypt Data (Crittografa dati)
- Traduci i dati dei pin
- Generate/Verify MAC
- GenerateMacEmvPinChange
- VerifyAuthRequestCryptogram
- Import/Export Chiave
- Tipi di chiavi non utilizzati

GenerateCardData

Endpoint API	Operazione o algoritmo crittografico	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
GenerateCardData	• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	CHIAVE_DI VERIFICA_ SCHEDA TR31_C0_C ARD_	• CHIAVE TDES_2 • CHIAVE TDES_3	{Genera = vero}, {Genera = vero, Verifica = vero}
GenerateCardData	• CARD_VERIFICATION_VALUE_1 • VALORE_DI VERIFICA_ CARTA_2	CHIAVE_DI _VERIFICA _SCHEDA TR31_C0_	• CHIAVE TDES_2	{Genera = vero}, {Genera = vero, Verifica = vero}
GenerateCardData	• CARDHOLDER_AUTHENTICATION_V	TR31_E6_E MV_MKEY_O THER	• TDES_2KEY	{= vero} DeriveKey

Endpoint API	Operazione o algoritmo crittografico	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
	ERIFICATION_VALUE			
GenerateCardData	• CODICE_DI_VERIFICA_SCHEDA DINAMICA	TR31_E4_EMV_MKEY_NUMBERS	• TDES_2KEY	{= vero} DeriveKey
GenerateCardData	• VALORE_DI_VERIFICA_SCHEDA DINAMICA	TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	{= vero} DeriveKey

VerifyCardData

Operazione o algoritmo crittografico	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
• AMEX_CARD_SECURITY_CODE_VERSION_1 • AMEX_CARD_SECURITY_CODE_VERSION_2	CHIAVE_DI VERIFICA_SCHEDA TR31_C0_CARD_	• CHIAVE TDES_2 • CHIAVE TDES_3	{Genera = vero}, {Genera = vero, Verifica = vero}
• CARD_VERIFICATION_VALUE_1	CHIAVE_DI_VERIFICA_SCHEDA TR31_C0_	• CHIAVE TDES_2	{Genera = vero}, {Genera = vero, Verifica = vero}

Operazione o algoritmo crittografico	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
• VALORE_DI VERIFICA_ CARTA_2			
• CARDHOLDE R_AUTHENT ICATION_V ERIFICATI ON_VALUE	TR31_E6_E MV_MKEY_OTHER	• TDES_2KEY	{= vero} DeriveKey
• CODICE_DI _VERIFICA _SCHEDA DINAMICA	TR31_E4_E MV_MKEY_N UMBERS	• TDES_2KEY	{= vero} DeriveKey
• VALORE_DI _VERIFICA _SCHEDA DINAMICA	TR31_E6_E MV_MKEY_OTHER	• TDES_2KEY	{= vero} DeriveKey

GeneratePinData (per VISA/ABA schemi)

`VISA_PIN or VISA_PIN_VERIFICATION_VALUE`

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
Chiave di crittografia PIN	TR31_P0_P IN_ENCRYP TION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Encrypt = vero, Wrap = vero} • {Encrypt = true, Decrypt = true,

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
			Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions
Chiave di generazione del PIN	TR31_V2_VISA_PIN_VERIFICATION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Genera = vero} • {Genera = vero, Verifica = vero}

GeneratePinData (per **IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
Chiave di crittografia PIN	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	Per IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET • {Encrypt = vero, Wrap = vero} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero}

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
			• {= vero} NoRestrictions Per IBM3624_PIN_OFFSET • {Encrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions
Chiave di generazione del PIN	TR31_V1_IBM3624_PIN_VERIFICATION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Genera = vero} • {Genera = vero, Verifica = vero}

VerifyPinData (per VISA/ABA schemi)

`VISA_PIN`

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiave di crittografia PIN	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Decrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true,

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
			Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions
Chiave di generazione del PIN	TR31_V2_VISA_PIN_VERIFICATION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Verifica = vero} • {Genera = vero, Verifica = vero}

VerifyPinData (per **IBM3624**)

`IBM3624_PIN_OFFSET,IBM3624_NATURAL_PIN,IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET)`

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiave di crittografia PIN	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	Per IBM3624_NATURAL_PIN, IBM3624_RANDOM_PIN, IBM3624_PIN_FROM_OFFSET • {Decrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero}

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
			• {= vero} NoRestrictions
Chiave di verifica del PIN	TR31_V1_IBM3624_PIN_VERIFICATION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Verifica = vero} • {Genera = vero, Verifica = vero}

Decrittografia dei dati

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
DISORDINATO	CHIAVE_DERIVAZIONE_BASE TR31_B0_BASE_DI_DERIVAZIONE	• CHIAVE TDES_2 • AES_128 • AES_192 • AES_256	• {= vero} DeriveKey • { NoRestrictions = vero}
EMV	TR31_E1_EMV_MKEY_RISERVATEZZA TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	• {= vero} DeriveKey
RSA	TR31_D1_CHIAVE_ASIMMETRICA_PER_CIFRATURA_DATI	• RSA_2048 • RSA_3072 • RSA_4096	• {Decrypt = true, unwrap=VERO} • {encrypt=True, wrap=True, Decrypt=true, unwrap=True}

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiavi simmetriche	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Decrypt = true, unwrap=True} • {encrypt=True, wrap=True, Decrypt=true, unwrap=True} • NoRestrictions {= vero}

Encrypt Data (Crittografa dati)

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
DISORDINATO	CHIAVE_DERIVAZIONE_BASE TR31_B0_BASE_DI_DERIVAZIONE	• CHIAVE TDES_2 • AES_128 • AES_192 • AES_256	• {= vero} DeriveKey • { NoRestrictions = vero}
EMV	TR31_E1_EMV_MKEY_RISERVATEZZA TR31_E6_EMV_MKEY_OTHER	• TDES_2KEY	• {= vero} DeriveKey
RSA	TR31_D1_CHIAVE_ASIMMETRICA_	• RSA_2048 • RSA_3072 • RSA_4096	• {Encrypt = true, wrap=True} • {encrypt=True, wrap=True,

Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
	PER_CIFRATURA_DATI		Decrypt=true, unwrap=True}
Chiavi simmetriche	TR31_D0_SYMMETRIC_DATA_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Encrypt=true, wrap=True} • {encrypt=True, wrap=True, Decrypt=true, unwrap=True} • NoRestrictions {= vero}

Traduci i dati dei pin

Direzione	Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Origine dati in entrata	DUKPT	CHIAVE_DERIVAZIONE_BASE TR31_B0_BASE_DI_DERIVAZIONE	• CHIAVE TDES_2 • AES_128 • AES_192 • AES_256	• {= vero} DeriveKey • { NoRestrictions = vero}
Origine dati in entrata	Non DUKPT (PEK, AWK, IWK, ecc.)	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192	• {Decrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap =

Direzione	Tipo di chiavi	Utilizzo delle chiavi consentito	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
			• AES_256	vero, Unwrap = vero} • {= vero} NoRestrictions
Obiettivo dati in uscita	DUPT	CHIAVE_DERIVAZIONE_BASE TR31_B0_BASE_DI_DERIVAZIONE	• CHIAVE TDES_2 • AES_128 • AES_192 • AES_256	• {= vero} DeriveKey • { NoRestrictions = vero}
Obiettivo dati in uscita	Non DUKPT (PEK, IWK, AWK, ecc.)	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Encrypt = vero, Wrap = vero} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions

Generate/Verify MAC

Le chiavi MAC vengono utilizzate per creare hash crittografici message/body di un dato. Non è consigliabile creare una chiave con modalità di utilizzo limitate poiché non sarà possibile eseguire l'operazione di abbinamento. Tuttavia, è possibile import/export utilizzare una chiave con una sola operazione se l'altro sistema è destinato a eseguire l'altra metà della coppia di operazioni.

Utilizzo consentito delle chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiave MAC	TR31_M1_ISO_9797_1_MAC_KEY	• TDES_2KEY • CHIAVE TDES_3	• {Genera = vero} • {Genera = vero, Verifica = vero} • {Verifica = vero} • {Genera = vero}
MAC Key (MAC per la vendita al dettaglio)	TR31_M1_ISO_9797_3_MAC_KEY	• TDES_2KEY • CHIAVE TDES_3	• {Genera = vero} • {Genera = vero, Verifica = vero} • {Verifica = vero} • {Genera = vero}
Chiave MAC (CMAC)	TR31_M6_ISO_9797_5_CMAC_KEY	• TDES_2KEY • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Genera = vero} • {Genera = vero, Verifica = vero} • {Verifica = vero} • {Genera = vero}
Chiave MAC (HMAC)	TR31_M7_HMAC_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Genera = vero} • {Genera = vero, Verifica = vero} • {Verifica = vero}
Chiave MAC (AS2805)	TR31_M0_ISO_16609_MAC_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3	• {Genera = vero} • {Genera = vero, Verifica = vero} • {Verifica = vero}

GenerateMacEmvPinChange

GenerateMacEmvPinChange combina la generazione MAC e la crittografia del PIN per le operazioni di modifica del PIN offline EMV. Questa operazione richiede due diversi tipi di chiave: una chiave di integrità per la generazione MAC e una chiave di riservatezza per la crittografia del PIN.

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiave di integrità della messaggistica sicura	TR31_E2_EMV_MKEY_INTEGRITY	• CHIAVE TDES_2	• {= vero} NoRestrictions
Chiave di riservatezza della messaggistica sicura	TR31_E1_EMV_MKEY_CONFIDENTIALITY	• TDES_2KEY	• { DeriveKey = vero}
PIN attuale PEK (chiave di crittografia PIN)	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Decrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions
Nuovo PIN PEK (chiave di crittografia PIN)	TR31_P0_PIN_ENCRYPTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Decrypt = true, Unwrap = true} • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero} • {= vero} NoRestrictions

Tipo di chiavi	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
Chiave ARQC **Note** Si applica solo agli schemi di derivazione Visa e Amex.	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• CHIAVE TDES_2	• {= vero} DeriveKey

VerifyAuthRequestCryptogram

Utilizzo delle chiavi consentito	Opzione EMV	Algoritmo a chiave consentita	Combinazione consentita di modalità di utilizzo chiave
• OPZIONE A • OPZIONE B	TR31_E0_EMV_MKEY_APP_CRYPTOGRAMS	• CHIAVE TDES_2	• {= vero} DeriveKey

Import/Export Chiave

Tipo di operazione	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
TR-31 Chiave di avvolgimento	CHIAVE DI PROTEZIONE TR31_K1_KEY_BLOCK_PROTECTION_KEY	• CHIAVE TDES_2 • CHIAVE TDES_3 • AES_128 • AES_192 • AES_256	• {Encrypt = true, Wrap = true} (solo esportazione)

Tipo di operazione	Utilizzo consentito delle chiavi	Algoritmo a chiave consentito	Combinazione consentita di modalità di utilizzo chiave
	TR31_K0_KEY_ENCRYPTION_KEY		• {Decrypt = true, Unwrap = true} (solo importazione) • {Encrypt = true, Decrypt = true, Wrap = vero, Unwrap = vero}
Importazione di CA attendibili	TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE	• RSA_2048 • RSA_3072 • RSA_4096	• {Verifica = vero}
Importazione del certificato a chiave pubblica per la crittografia asimmetrica	TR31_D1_ASYMMETRIC_KEY_FOR_DATA_ENCRYPTION	• RSA_2048 • RSA_3072 • RSA_4096	• {ENCRYPT=VERO, WRAP=VERO}
Chiave utilizzata per algoritmi di accordo chiave come ECDH	TR31_K3_ASYMMETRIC_KEY_FOR_KEY_AGREEMENT	• ECC_NIST_P256 • ECC_NIST_P384 • ECC_NIST_P521	• {= vero} DeriveKey

Tipi di chiavi non utilizzati

I seguenti tipi di chiave non sono attualmente utilizzati da AWS Payment Cryptography

- TR31_P1_PIN_GENERATION_KEY

Casi di utilizzo comune

AWS La crittografia dei pagamenti supporta molte operazioni crittografiche tipiche dei pagamenti. I seguenti argomenti fungono da guida su come utilizzare queste operazioni per i casi d'uso più comuni. Per un elenco di tutti i comandi, consulta l'API AWS Payment Cryptography.

Argomenti

- Emittenti e responsabili del trattamento degli emittenti
- Agevolatori di acquisizione e pagamento

Emittenti e responsabili del trattamento degli emittenti

I casi d'uso degli emittenti sono generalmente costituiti da alcune parti. Questa sezione è organizzata per funzione (ad esempio, utilizzo dei pin). In un sistema di produzione, le chiavi hanno in genere l'ambito di un determinato contenitore di schede e vengono create durante la configurazione del contenitore anziché in linea, come mostrato qui.

Argomenti

- Funzioni generali
- Funzioni specifiche della rete

Funzioni generali

Argomenti

- Genera un pin casuale e il PVV associato, quindi verifica il valore
- Genera o verifica un CVV per una determinata carta
- Genera o verifica un CVV2 per una carta specifica
- Genera o verifica un iCVV per una carta specifica
- Verifica un EMV ARQC e genera un ARPC
- Genera e verifica un MAC EMV
- Genera EMV MAC per la modifica del PIN

Genera un pin casuale e il PVV associato, quindi verifica il valore

Argomenti

- Crea la/e chiave/i
- Genera un PIN casuale, genera PVV e restituisci il PIN crittografato e il PVV
- Convalida il PIN crittografato utilizzando il metodo PVV
- PIN gestiti dall'utente

Crea la/e chiave/i

Per generare un pin casuale e il PVV, sono necessarie due chiavi, una PIN Verification Key (PVK) per generare il PVV e una PIN Encryption Key per crittografare il pin. Il pin stesso viene generato casualmente in modo sicuro all'interno del servizio e non è correlato crittograficamente a nessuna delle due chiavi.

Il PGK deve essere una chiave dell'algoritmo TDES_2KEY basato sull'algoritmo PVV stesso. Un PEK può essere TDES_2KEY, TDES_3KEY o AES_128. In questo caso, poiché il PEK è destinato all'uso interno del sistema, AES_128 sarebbe una buona scelta. Se un PEK viene utilizzato per l'interscambio con altri sistemi (ad esempio reti di carte, acquirer, ATM) o viene spostato nell'ambito di una migrazione, TDES_2KEY potrebbe essere la scelta più appropriata per motivi di compatibilità.

Crea il 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"}]'
```

La risposta riprende i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/ivi5ksfsuplneuyt Ne avrai bisogno nel passaggio successivo.

Crea il 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"}]'
```

La risposta riprende i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/ov6icy4ryas4zcza Ne avrai bisogno nel passaggio successivo.

Genera un PIN casuale, genera PVV e restituisci il PIN crittografato e il PVV

Example

In questo esempio, genereremo un nuovo pin (casuale) a 4 cifre in cui le uscite saranno crittografate `PIN block` (PinData.PinBlock) e (pin). PVV Data.VerificationValue Gli input chiave sono PAN il formato `Pin Verification Key` (noto anche come chiave di generazione del pin), `Pin Encryption Key` e il formato PIN Block.

Questo comando richiede che la chiave sia di tipo`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"
            }
        }
```

Convalida il PIN crittografato utilizzando il metodo PVV

Example

In questo esempio, convalideremo un PIN per un determinato PAN. Il PIN viene in genere fornito dal titolare della carta o dall'utente durante il momento della transazione per la convalida e viene confrontato con il valore in archivio (l'input del titolare della carta viene fornito come valore crittografato dal terminale o da un altro provider upstream). Per convalidare questo input, verranno forniti anche i seguenti valori in fase di esecuzione: il pin crittografato, la chiave utilizzata per crittografare il pin di input (spesso indicato come IWK) PAN e il valore da verificare (a PVV o`PIN offset`).

Se AWS Payment Cryptography è in grado di convalidare il PIN, viene restituito un. http/200 Se il PIN non viene convalidato, restituirà un. 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",
}
```

PIN gestiti dall'utente

Argomenti

- Panoramica di
- Crea le chiavi
- Imposta un PIN selezionabile dall'utente
- Reimposta un PIN (PIN verde)
- Rivela un PIN

Panoramica di

AWS La crittografia dei pagamenti supporta i flussi di PIN gestiti dall'utente (gestiti dal cliente), in cui un PIN viene scambiato in modo sicuro tra il dispositivo del titolare della carta e il servizio tramite il backend del cliente. Il dispositivo e il servizio utilizzano l'accordo chiave Elliptic Curve Diffie-Hellman (ECDH) per ricavare una chiave simmetrica condivisa su entrambi i lati, che protegge un blocco a 4 PIN in formato ISO. ??? Poiché la chiave è derivata anziché scambiata, il PIN chiaro e il blocco PIN chiaro non vengono mai trasmessi oltre il dispositivo.

Questi flussi coinvolgono tre parti. Il dispositivo titolare della carta genera la coppia di chiavi EC, ricava la chiave ECDH AES-128 , crittografa o decrittografa il blocco 4 PIN in formato ISO e detiene il PIN chiaro; non dispone di credenziali AWS. Il backend del cliente (l'emittente o il server del processore) detiene le credenziali AWS, chiama AWS Payment Cryptography, firma il certificato del dispositivo e archivia il PVV. ??? AWS Payment Cryptography esegue le operazioni crittografiche. Il dispositivo comunica solo con il backend del cliente e il backend del cliente effettua tutte le chiamate a Payment Cryptography. AWS

Questo argomento tratta tre flussi:

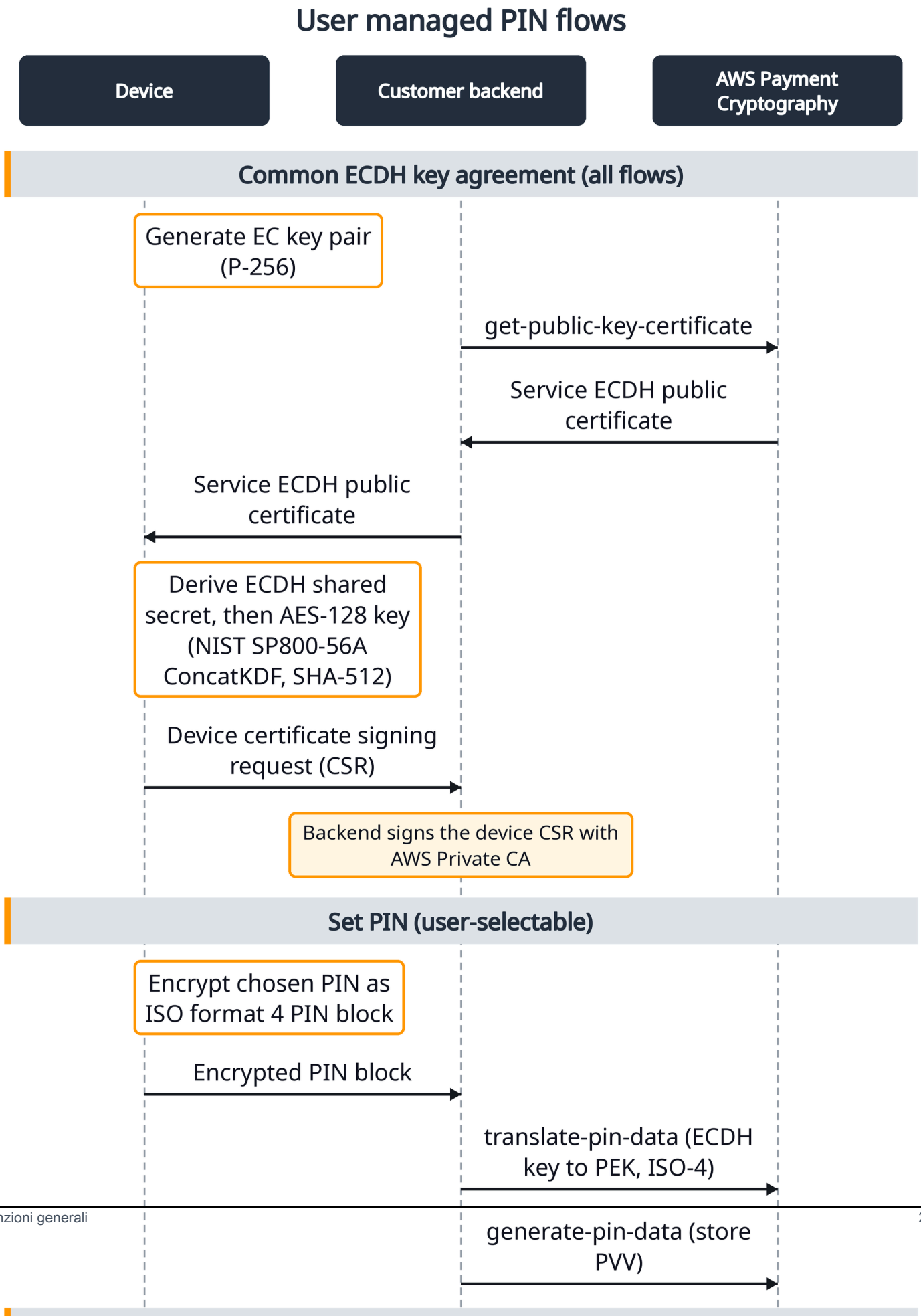
- User-selectable PIN (Set PIN): il titolare della carta sceglie il proprio PIN sul dispositivo.

- PIN verde (Reset PIN): il servizio genera un PIN casuale che viene inviato al titolare della carta, ad esempio durante la prima emissione o durante la reimpostazione del PIN dimenticato.
- Reveal PIN: un blocco PIN esistente viene restituito a un dispositivo autorizzato per casi d'uso ristretti.

Tutti e tre i flussi condividono la stessa configurazione delle chiavi e la stessa meccanica di accordo sui dispositivi ECDH, descritta in Creazione delle chiavi. ??? Per i dettagli dell'accordo chiave stesso, consulta Importazione di chiavi mediante tecniche asimmetriche (ECDH).

> **Note**
>
> Questi flussi gestiscono PIN trasparenti e blocchi di PIN solo sul dispositivo, mai in transito verso Payment Cryptography. AWS Esamina attentamente ogni flusso con il tuo team addetto alla sicurezza prima di implementarlo.

Il diagramma seguente mostra i flussi di PIN gestiti dall'utente.


User managed PIN flows
Device
Customer backend
AWS Payment Cryptography
Common ECDH key agreement (all flows)
Generate EC key pair (P-256)
get-public-key-certificate
Service ECDH public certificate
Service ECDH public certificate
Derive ECDH shared secret, then AES-128 key (NIST SP800-56A ConcatKDF, SHA-512)
Device certificate signing request (CSR)
Backend signs the device CSR with AWS Private CA
Set PIN (user-selectable)
Encrypt chosen PIN as ISO format 4 PIN block
Encrypted PIN block
translate-pin-data (ECDH key to PEK, ISO-4)
generate-pin-data (store PVV)

Per un esempio completo ed eseguibile di questi flussi utilizzando l'SDK AWS per Python, consulta l'esempio ECDH PIN set and reveal flows nell'archivio degli esempi su. AWS GitHub

> **Note**
>
> Il flusso tipico include un dispositivo titolare di carta che esegue l'accordo chiave ECDH e crittografa o decrittografa il blocco a 4 PIN in formato ISO. Se proteggi già i PIN tra il dispositivo e il backend del cliente con un altro mezzo, puoi utilizzare una versione modificata di questi flussi direttamente tra il backend del cliente e Payment Cryptography. AWS In questa variante, il backend del cliente esegue l'accordo chiave ECDH e le operazioni di blocco del PIN al posto del dispositivo.

Crea le chiavi

Tutti e tre i flussi utilizzano lo stesso set di chiavi: una chiave di accordo ECDH per l'accordo chiave con il dispositivo, una Pin Generation/Verification Key (PGK) per generare il PVV e una Pin Encryption Key (PEK) in cui il blocco PIN viene tradotto per l'archiviazione backend. Importi anche la chiave pubblica dell'autorità di certificazione del dispositivo (CA) in modo che AWS Payment Cryptography consideri attendibile i certificati del dispositivo presentati durante la traduzione.

Crea la chiave del contratto ECDH. Questa chiave viene utilizzata per derivare la chiave simmetrica condivisa con il dispositivo.

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

La risposta richiama i parametri della richiesta e include l'ARN della chiave ECDH.

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx",
        "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": "1A2B3C",
        "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"
    }
}
```

Crea la Generation/Verification chiave Pin (PGK) utilizzata per generare il PVV.

```
$ aws payment-cryptography create-key \
    --exportable \
    --key-attributes KeyAlgorithm=TDES_2KEY,KeyUsage=TR31_V2_VISA_PIN_VERIFICATION_KEY,
\
        KeyClass=SYMMETRIC_KEY,\
        KeyModesOfUse='{Generate=true,Verify=true}'
```

```
{
    "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"
    }
}
```

Crea la chiave di crittografia Pin (PEK) in cui viene tradotto il blocco PIN del dispositivo per l'archiviazione back-end.

```
$ 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}'
```

```
{
    "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": true,
                "Decrypt": true,
                "Wrap": true,
                "Unwrap": true,
                "Generate": false,
```

```
                "Sign": false,
                "Verify": false,
                "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"
    }
}
```

Importa la chiave pubblica della CA del dispositivo in modo che AWS Payment Cryptography possa verificare i certificati del dispositivo presentati nelle successive chiamate di traduzione. Sostituiscilo <base64-CA-cert> con il certificato CA con codifica base64.

```
$ aws payment-cryptography import-key \
    --key-material='{"RootCertificatePublicKey":{"KeyAttributes":
{"KeyAlgorithm":"ECC_NIST_P256","KeyClass":"PUBLIC_KEY","KeyModesOfUse":
{"Verify":true},"KeyUsage":"TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE"},"PublicKeyCertificat
CA-cert>"}}'
```

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "ECC_NIST_P256",
            "KeyModesOfUse": {
                "Verify": true
            }
        },
        "KeyCheckValue": "4D5E6F",
        "KeyCheckValueAlgorithm": "CMAC",
        "Enabled": true,
```

```
        "Exportable": false,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2023-06-05T06:41:46.648000-07:00",
        "UsageStartTimestamp": "2023-06-05T06:41:46.626000-07:00"
    }
}
```

Prendi nota di ogni reso. `KeyArn` Gli esempi che seguono utilizzano i seguenti segnaposti letterali:

- Chiave dell'accordo ECDH: `arn:aws:payment-cryptography:us-east-1:111122223333:key/xxxxxxxxxxxxxxxx`
- Chiave pubblica CA importata: `arn:aws:payment-cryptography:us-east-1:111122223333:key/yyyyyyyyyyyyyyyy`
- PGK: `arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza`
- PEK: `arn:aws:payment-cryptography:us-east-2:111122223333:key/ivi5ksfsuplneuyt`

Sul dispositivo titolare della carta (in genere un telefono cellulare o un dispositivo simile), vengono eseguiti i seguenti passaggi nel codice applicativo del dispositivo per ciascuno dei flussi Set PIN, Reset PIN e Reveal PIN: generare una coppia di chiavi EC su curve P-256; ottenere il certificato pubblico ECDH del servizio dal backend del cliente; derivare il segreto condiviso ECDH e quindi una chiave da 16 byte (utilizzando il NIST SP800-56A ConcatKDF (SHA-5 12) con il `SharedInformation` valore; e crittografare o decrittografare l'ISO formattare il blocco a 4 PIN con quella AES-128 chiave derivata. ??? Il dispositivo non dispone di credenziali AWS e non chiama direttamente AWS Payment Cryptography; scambia questi dati con il backend del cliente, che effettua le chiamate al servizio.

Poiché il dispositivo è in genere un telefono, la coppia di chiavi EC e il certificato vengono assegnati al primo utilizzo e successivamente aggiornati periodicamente. Il dispositivo genera la coppia di chiavi EC e una richiesta di firma del certificato (CSR) e il backend del cliente firma la CSR utilizzando o una CA simile. CA privata AWS Il backend del cliente utilizza il certificato firmato del dispositivo risultante nelle chiamate Translate-PIN Data e AWS Payment Cryptography lo verifica rispetto alla chiave pubblica CA importata.

Il backend del cliente ottiene il certificato pubblico ECDH del servizio e lo fornisce al dispositivo. Il certificato viene restituito come PEM con codifica base64 nel `KeyCertificate` campo, che viene

decodificato in base64 in un file PEM. Il dispositivo utilizza questo certificato come chiave pubblica del servizio quando ricava il segreto condiviso.

```
$ aws payment-cryptography get-public-key-certificate \
    --key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx
```

> **Note**
>
> `SharedInformation`è costituito da 32 byte casuali generati sul dispositivo e passati attraverso il backend del cliente a AWS Payment Cryptography come stringa esadecimale. Deve corrispondere al valore utilizzato dal dispositivo nel KDF, altrimenti le chiavi derivate non saranno d'accordo.

Imposta un PIN selezionabile dall'utente

Nel flusso Imposta PIN, il titolare della carta sceglie il proprio PIN sul dispositivo (ad esempio,). `1234` Il dispositivo ricava la chiave condivisa ECDH, crittografa il PIN scelto in un blocco a 4 PIN in formato ISO contenuto in tale chiave e invia il blocco ??? PIN crittografato, la relativa richiesta di firma del certificato (CSR) e le informazioni condivise al backend del cliente. Il backend del cliente firma il CSR e quindi esegue due chiamate sul piano dati. Questo esempio utilizza il campione. PAN `171234567890123` I valori del blocco PIN, del certificato e delle informazioni condivise mostrati sono segnaposto fittizi.

Innanzitutto, traduci il blocco ECDH-protected PIN nel PEK del server. La chiave in entrata è la chiave dell'accordo ECDH, racchiusa con Diffie-Hellman i parametri e il certificato del dispositivo firmato; la chiave in uscita è il PEK.

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block A1B2C3D4E5F60718 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --incoming-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
```

```
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

La risposta contiene il blocco PIN nuovamente crittografato con il PEK in formato ISO 4.

```
{
    "PinBlock": "F7E6D5C4B3A29180",
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "KeyCheckValue": "7CC9E2"
}
```

Quindi, genera il PVV per il PEK-encrypted blocco PIN. Il PGK genera il PVV e il PEK identifica la chiave con cui è crittografato il blocco PIN fornito.

```
$ aws payment-cryptography-data generate-pin-data \
    --generation-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_4 \
    --primary-account-number 171234567890123 \
    --generation-attributes
 'VisaPinVerificationValue={PinVerificationKeyIndex=1,EncryptedPinBlock=F7E6D5C4B3A29180}'
```

La risposta contiene l'ingresso PVV e il blocco PIN crittografato`PinData.VerificationValue`. Memorizza il PVV sul backend del cliente in modo da poter verificare il PIN in un secondo momento. Per la verifica, consulta Generare un pin casuale e il PVV associato, quindi verifica il valore.

```
{
    "GenerationKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "GenerationKeyCheckValue": "51A200",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
    "EncryptedPinBlock": "F7E6D5C4B3A29180",
```

```
    "PinData": {
        "VerificationValue": "5507"
    }
}
```

Per ulteriori informazioni, consulta TranslatePinData e GeneratePinData nella guida di riferimento alle API.

Reimposta un PIN (PIN verde)

Nel flusso Reset PIN (PIN verde), AWS Payment Cryptography genera un PIN casuale che il backend del cliente invia al dispositivo per mostrare al titolare della carta, ad esempio durante la prima emissione o durante la reimpostazione di un PIN dimenticato. Il dispositivo esegue la stessa configurazione ECDH descritta in Create the keys e il backend del cliente effettua le chiamate data-plane per suo conto. Questo esempio utilizza l'esempio PAN`171234567890123`; il blocco PIN e i valori delle informazioni condivise sono segnaposto fittizi.

Innanzitutto, genera un PIN casuale e il relativo PVV, crittografando il blocco PIN sotto il PEK.

```
$ aws payment-cryptography-data generate-pin-data \
    --generation-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ov6icy4ryas4zcza \
    --encryption-key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/ivi5ksfsuplneuyt \
    --pin-block-format ISO_FORMAT_4 \
    --primary-account-number 171234567890123 \
    --generation-attributes 'VisaPin={PinVerificationKeyIndex=1}'
```

La risposta contiene l'ingresso PVV `PinData.VerificationValue` e il blocco PEK-encrypted PIN (formato ISO 4). Memorizza il PVV sul backend del cliente per una verifica successiva.

```
{
    "GenerationKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ov6icy4ryas4zcza",
    "GenerationKeyCheckValue": "51A200",
    "EncryptionKeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt",
    "EncryptionKeyCheckValue": "7CC9E2",
    "EncryptedPinBlock": "F7E6D5C4B3A29180",
    "PinData": {
        "VerificationValue": "5507"
```

```
    }
}
```

Quindi, traduci il blocco PEK-encrypted PIN (formato ISO 4) sulla ECDH-derived chiave del dispositivo (formato ISO 4) in modo che solo il dispositivo richiedente possa leggerlo.

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block F7E6D5C4B3A29180 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --outgoing-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

La risposta contiene il blocco PIN nuovamente crittografato con la ECDH-derived chiave del dispositivo in formato ISO 4.

```
{
    "PinBlock": "A1B2C3D4E5F60718",
    "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx",
    "KeyCheckValue": "1A2B3C"
}
```

Il backend del cliente restituisce il blocco ECDH-encrypted PIN al dispositivo. Il dispositivo decodifica il blocco PIN restituito con la sua ECDH-derived chiave (nel codice dell'applicazione del dispositivo) e visualizza il PIN al titolare della carta. Poiché la chiave è derivata solo da quel dispositivo, nessun'altra parte può decrittografare il blocco PIN.

Per ulteriori informazioni, consulta GeneratePinData e TranslatePinData nella guida di riferimento alle API.

Rivela un PIN

Il flusso Reveal PIN restituisce un blocco PIN esistente a un dispositivo titolare di carta in modo che il dispositivo possa visualizzare o rivelare in altro modo il PIN. Il dispositivo esegue la stessa configurazione ECDH descritta in Creazione delle chiavi e il backend del cliente traduce nuovamente il blocco PEK-encrypted PIN memorizzato nella ECDH-derived chiave e lo restituisce al dispositivo. Questo esempio utilizza l'esempio PAN`171234567890123`; il blocco PIN e i valori delle informazioni condivise sono segnaposto fittizi.

> **Note**
>
> Valuta la possibilità di autenticare il titolare della carta e consentirgli di impostare un nuovo PIN invece di rivelare il PIN esistente. Una notevole eccezione è rappresentata dai paesi che utilizzano il PIN offline, ad esempio in Europa, dove il PIN è memorizzato anche sulla chip card. In questi casi, rivelare il PIN esistente può essere più semplice che riprogrammare la scheda.

Traduci il blocco PEK-encrypted PIN memorizzato (formato ISO 4) sulla ECDH-derived chiave del dispositivo (formato ISO 4).

```
$ aws payment-cryptography-data translate-pin-data \
    --encrypted-pin-block F7E6D5C4B3A29180 \
    --incoming-key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
ivi5ksfsuplneuyt \
    --incoming-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}' \
    --outgoing-key-identifier arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx \
    --outgoing-wrapped-key
 'WrappedKeyMaterial={DiffieHellmanSymmetricKey={CertificateAuthorityPublicKeyIdentifier=arn:aw
cryptography:us-east-1:111122223333:key/
yyyyyyyyyyyyyyyy,KeyAlgorithm=AES_128,KeyDerivationFunction=NIST_SP800,KeyDerivationHashAlgorit
signed-device-
cert>,SharedInformation=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef}}'
 \
    --outgoing-translation-attributes
 'IsoFormat4={PrimaryAccountNumber=171234567890123}'
```

La risposta contiene il blocco PIN crittografato nuovamente con la ECDH-derived chiave del dispositivo in formato ISO 4.

```
{
    "PinBlock": "A1B2C3D4E5F60718",
    "KeyArn": "arn:aws:payment-cryptography:us-east-1:111122223333:key/
xxxxxxxxxxxxxxx",
    "KeyCheckValue": "1A2B3C"
}
```

Il backend del cliente restituisce il blocco ECDH-encrypted PIN al dispositivo. Il dispositivo decodifica il blocco PIN restituito con la relativa ECDH-derived chiave (nel codice dell'applicazione del dispositivo) per rivelare il PIN. Poiché la chiave è derivata solo da quel dispositivo, nessun'altra parte può decrittografare il blocco PIN.

Per ulteriori informazioni, consulta la TranslatePinData guida di riferimento alle API.

Genera o verifica un CVV per una determinata carta

CVV o CVV1 è un valore tradizionalmente incorporato nella banda magnetica di una carta. Non è uguale al CVV2 (visibile al titolare della carta e da utilizzare per gli acquisti online).

Il primo passo è creare una chiave. Per questo tutorial, crei una chiave 3DES (2KEY TDES) CVK a doppia lunghezza.

> **Note**
>
> CVV, CVV2 e iCVV utilizzano tutti algoritmi simili se non identici ma variano i dati di input. Tutti utilizzano lo stesso tipo di chiave TR31_C0_CARD_VERIFICATION_KEY ma si consiglia di utilizzare chiavi separate per ogni scopo. Queste possono essere distinte utilizzando i tag alias come nell'esempio seguente. and/or

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```


Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/r52o3wbqxyf6qlqr Ne avrai bisogno nel passaggio successivo.

Genera un CVV

Example

In questo esempio, genereremo un CVV per un determinato PAN con input diPAN, un codice di servizio (come definito da ISO/IEC 7813) di 121 e la data di scadenza della carta.

Per tutti i parametri disponibili, consulta la guida di riferimento CardVerificationValue1 alle 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"
                }
```

Convalida il CVV

Example

In questo esempio, verificheremo un CVV per un determinato PAN inserendo un CVK, un codice di servizio 121PAN, la data di scadenza della carta e il CVV fornito durante la transazione per la convalida.

Per tutti i parametri disponibili, consulta la guida di riferimento alle API. CardVerificationValue1

> **Note**
>
> CVV non è un valore inserito dall'utente (come CVV2) ma in genere è incorporato in una banda magnetica. Si dovrebbe valutare se debba sempre essere convalidato quando viene fornito.

```
$ 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"
}
```

Genera o verifica un CVV2 per una carta specifica

Il CVV2 è un valore che viene tradizionalmente fornito sul retro di una carta e viene utilizzato per gli acquisti online. Per le carte virtuali, potrebbe essere visualizzato anche su un'app o su una schermata. Dal punto di vista crittografico, è uguale a CVV1 ma con un valore del codice di servizio diverso.

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/7f7g4spf3xcklhzu Ne avrai bisogno nel passaggio successivo.

Genera un CVV2

Example

In questo esempio, genereremo un CVV2 per un determinato PAN con input e data di scadenza della carta. PAN

Per tutti i parametri disponibili, consulta la guida di riferimento CardVerificationValue2 delle 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"
                    }
```

Convalida un CVV2

Example

In questo esempio, verificheremo un CVV2 per un determinato PAN inserendo un CVK, la data di scadenza della carta PAN e il CVV fornito durante la transazione per la convalida.

Per tutti i parametri disponibili, consulta la guida di riferimento alle API. CardVerificationValue2

> **Note**
>
> CVV2 e gli altri input sono valori inseriti dall'utente. Pertanto, non è necessariamente indice di un problema il fatto che ciò non venga convalidato periodicamente.

```
$ 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"
   }
```

Genera o verifica un iCVV per una carta specifica

iCVV utilizza lo stesso algoritmo, CVV/CVV2 ma iCVV è incorporato in una chip card. Il suo codice di servizio è 999.

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
               }
           }
```


Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/c7dsi763r6s7lfp3 Ne avrai bisogno nel passaggio successivo.

Genera un iCVV

Example

In questo esempio, genereremo un iCVV per un determinato PAN con input diPAN, un codice di servizio (come definito da ISO/IEC 7813) di 999 e la data di scadenza della carta.

Per tutti i parametri disponibili, consulta la guida di riferimento CardVerificationValue1 alle 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"
                    }
```

Convalida iCVV

Example

Per la convalida, gli input sono CVK, un codice di servizio 999PAN, la data di scadenza della carta e l'iCVV fornito durante la transazione per la convalida.

Per tutti i parametri disponibili, consulta la guida di riferimento alle API. CardVerificationValue1

> **Note**
>
> iCVV non è un valore inserito dall'utente (come CVV2) ma in genere è incorporato in una scheda. EMV/chip Si dovrebbe valutare se debba sempre essere convalidato quando viene fornito.

```
$ 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"
    }
```

Verifica un EMV ARQC e genera un ARPC

ARQC (Authorization Request Cryptogram) è un crittogramma generato da una carta EMV (chip) e utilizzato per convalidare i dettagli della transazione e l'uso di una carta autorizzata. Incorpora i dati della carta, del terminale e della transazione stessa.

Al momento della convalida sul backend, vengono forniti gli stessi input a AWS Payment Cryptography, il crittogramma viene ricreato internamente e confrontato con il valore fornito con la transazione. In questo senso, è simile a un MAC. Il libro 2 di EMV 4.4 definisce tre aspetti di questa funzione: metodi di derivazione delle chiavi (noti come chiave di sessione comune - CSK) per generare chiavi di transazione una tantum, un payload minimo e metodi per generare una risposta (ARPC).

Gli schemi delle singole carte possono specificare campi transazionali aggiuntivi da incorporare o l'ordine in cui tali campi vengono visualizzati. Esistono anche altri schemi di derivazione specifici dello schema (generalmente obsoleti) che sono trattati altrove in questa documentazione.

Per ulteriori informazioni, consulta la guida VerifyCardValidationData alle API.

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/pw3s6nl62t5ushfk Ne avrai bisogno nel passaggio successivo.

Genera un ARQC

L'ARQC viene generato esclusivamente da una scheda EMV. Pertanto, AWS Payment Cryptography non è in grado di generare un tale payload. A scopo di test, sono disponibili online numerose librerie in grado di generare un payload appropriato oltre a valori noti generalmente forniti dai vari schemi.

Convalida un ARQC

Example

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un. http/200
Un ARPC (risposta) può essere fornito facoltativamente e incluso nella risposta dopo la convalida dell'ARQC.

```
$ 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"
}
```

Genera e verifica un MAC EMV

EMV MAC è un MAC che utilizza un input di una chiave derivata EMV e quindi esegue un MAC ISO9797-3 (Retail) sui dati risultanti. EMV MAC viene in genere utilizzato per inviare comandi a una scheda EMV come gli script di sblocco.

> **Note**
>
> AWS Payment Cryptography non convalida il contenuto dello script. Consulta il manuale dello schema o della carta per i dettagli sui comandi specifici da includere.

Per ulteriori informazioni, MacAlgorithmEmv consulta la guida alle API.

Argomenti

- Crea la chiave
- Genera un MAC EMV

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```


Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/pw3s6nl62t5ushfk Ne avrai bisogno nel passaggio successivo.

Genera un MAC EMV

Il flusso tipico prevede che un processo di backend generi uno script EMV (ad esempio card unblock), lo firmi utilizzando questo comando (che ricava una chiave monouso specifica per una particolare scheda) e quindi restituisca il MAC. Quindi il comando + MAC vengono inviati alla

scheda da applicare. L'invio del comando alla carta non rientra nell'ambito della crittografia dei AWS pagamenti.

> **Note**
>
> Questo comando è pensato per i comandi in cui non vengono inviati dati crittografati (come il PIN). EMV Encrypt può essere combinato con questo comando per aggiungere dati crittografati allo script dell'emittente prima di chiamare questo comando

Dati dei messaggi

I dati dei messaggi includono l'intestazione e il comando APDU. Sebbene ciò possa variare in base all'implementazione, questo esempio è l'intestazione APDU per lo sblocco (84 24 00 00 08), seguita da ATC (0007) e quindi da ARQC della transazione precedente (999E57FD0F47CACE). Il servizio non convalida il contenuto di questo campo.

Modalità di derivazione delle chiavi di sessione

Questo campo definisce come viene generata la chiave di sessione. EMV_COMMON_SESSION_KEY viene generalmente utilizzato per le nuove implementazioni, mentre può essere utilizzato anche EMV2000 | AMEX | MASTERCARD_SESSION_KEY | VISA.

MajorKeyDerivationMode

EMV definisce la modalità A, B o C. La modalità A è la più comune e Payment Cryptography attualmente supporta la modalità A o la modalità B. AWS

PAN

Il numero di conto, in genere disponibile nel campo chip 5A o nel campo 2 ISO8583, ma può anche essere recuperato dal sistema della carta.

PSN

Il numero di sequenza della carta. Se non viene utilizzato, immettere 00.

SessionKeyDerivationValue

Si tratta dei dati di derivazione per sessione. Può essere l'ultimo ARQC (ApplicationCryptogram) dal campo 9F26 o l'ultimo ATC da 9F36 a seconda dello schema di derivazione.

Padding

Il padding viene applicato automaticamente e utilizza il metodo di riempimento 9797-1 2. ISO/IEC

Example

```
$ aws payment-cryptography-data generate-mac --message-data
 84240000080007999E57FD0F47CACE --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"
}
```

Genera EMV MAC per la modifica del PIN

La modifica del PIN EMV combina due operazioni: la generazione di un MAC per lo script dell'emittente e la crittografia di un nuovo PIN per la modifica del PIN offline su una scheda con chip EMV. Questo comando è necessario solo in alcuni paesi in cui il PIN è memorizzato sulla chip card (questo è comune per i paesi europei). Viene comunemente utilizzato quando il titolare della carta deve modificare il proprio PIN e il nuovo PIN deve essere trasmesso in modo sicuro alla carta insieme a un MAC per verificare l'autenticità del comando.

> **Note**
>
> Se devi solo inviare comandi alla scheda ma non modificare il PIN, prendi in considerazione l'utilizzo dei comandi ARPC CSU o Generate EMV MAC.

Per ulteriori informazioni, consulta la guida GenerateMacEmvPinChange alle API.

Genera EMV MAC e PIN crittografato per la modifica del PIN

Questa operazione richiede due chiavi: una chiave di integrità EMV (: TR31_E2_EMV_MKEY_INTEGRITY) per la generazione MAC e una chiave di riservatezza EMV (KeyUsage: TR31_E4_EMV_MKEY_CONFIDENTIALITY) per la crittografia del PIN. KeyUsage Il flusso tipico è che un processo di backend generi uno script di modifica del PIN EMV, che include sia il MAC per lo script dell'emittente sia il nuovo PIN crittografato. Il comando e il PIN crittografato vengono quindi inviati alla scheda per aggiornare il PIN offline. L'invio del comando alla carta non rientra nell'ambito della crittografia dei AWS pagamenti.

Dati del messaggio

I dati dei messaggi includono il comando APDU per lo script dell'emittente. Il servizio non convalida il contenuto di questo campo.

Nuovo blocco PIN crittografato

Il nuovo blocco PIN crittografato che verrà inviato alla carta. Questo deve essere fornito come valore crittografato utilizzando una chiave di crittografia PIN.

Nuovo identificatore PIN PEK

La chiave utilizzata per crittografare il nuovo PIN prima che venga passato a questa API.

Chiave di integrità della messaggistica sicura

La chiave di integrità EMV (KeyUsage: TR31_E2_EMV_MKEY_INTEGRITY) utilizzata per la generazione MAC.

Chiave di riservatezza della messaggistica sicura

La chiave di riservatezza EMV (KeyUsage: TR31_E4_EMV_MKEY_CONFIDENTIALITY) utilizzata per la crittografia del PIN.

MajorKeyDerivationMode

EMV definisce la modalità A, B o C. La modalità A è la più comune e Payment Cryptography attualmente supporta la modalità A o la modalità B. AWS

Modalità

La modalità di crittografia, in genere CBC per le operazioni di modifica del PIN.

PAN

Il numero di conto, in genere disponibile nel campo chip 5A o nel campo 2 ISO8583, ma può anche essere recuperato dal sistema della carta.

PanSequenceNumber

Il numero di sequenza della carta. Se non viene utilizzato, immettere 00.

ApplicationCryptogram

Si tratta dei dati di derivazione per sessione, in genere l'ultimo ARQC dal campo 9F26.

PinBlockLengthPosition

Specifica dove è codificata la lunghezza del blocco PIN. In genere impostato su NESSUNO. Controlla le specifiche dello schema della tua carta se non sei sicuro.

PinBlockPaddingType

Specifica il tipo di spaziatura interna per il blocco PIN. In genere impostato su NO_PADDING. Controlla le specifiche dello schema della tua carta se non sei sicuro.

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"
}
```

Funzioni specifiche della rete

Argomenti

- Funzioni specifiche di Visa
- Funzioni specifiche di Mastercard

- Funzioni specifiche di American Express
- Funzioni specifiche JCB
- UnionPay funzioni specifiche

Funzioni specifiche di Visa

Argomenti

- ARQC - CVN18/CVN22
- ARQC - CVN10
- 3DS CAVV V7
- DCvv (valore dinamico di verifica della carta) - CVN17

ARQC - CVN18/CVN22

CVN18 e CVN22 utilizzano il metodo CSK di derivazione delle chiavi. ??? I dati esatti delle transazioni variano tra questi due metodi: consulta la documentazione dello schema per i dettagli sulla costruzione del campo dei dati delle transazioni.

ARQC - CVN10

CVN10 è un vecchio metodo Visa per le transazioni EMV che utilizza la derivazione della chiave per carta anziché la derivazione di sessione (per transazione) e utilizza anche un payload diverso. Per informazioni sul contenuto del payload, contatta lo schema per ulteriori dettagli.

Crea chiave

```
$ 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"}]'
```

La risposta riprende i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/pw3s6nl62t5ushfk Ne avrai bisogno nel passaggio successivo.

Convalida l'ARQC

Example

In questo esempio, convalideremo un ARQC generato utilizzando Visa CVN10.

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un. http/200 Se l'arqc non è convalidato, restituirà una risposta. 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

Per le transazioni Visa Secure (3DS), viene generato un CAVV (Cardholder Authentication Verification Value) dall'emittente Access Control Server (ACS). Il CAVV è la prova che l'autenticazione del titolare della carta è avvenuta, è unico per ogni transazione di autenticazione e viene fornito dall'acquirente nel messaggio di autorizzazione. CAVV v7 associa i dati aggiuntivi sulla transazione all'approvazione, inclusi elementi quali il nome del commerciante, l'importo dell'acquisto e la data di acquisto. In questo modo, è effettivamente un hash crittografico del payload della transazione.

Crittograficamente, CAVV V7 utilizza l'algoritmo CVV ma gli input lo sono stati tutti. changed/ repurposed Si prega di consultare la terza party/Visa documentazione appropriata su come produrre gli input per generare un payload CAVV V7.

Crea la chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/dnaeyrjgdjjtw6dk Ne avrai bisogno nel passaggio successivo.

Genera un CAVV V7

Example

In questo esempio, genereremo un CAVV V7 per una determinata transazione con input come specificato nelle specifiche. Nota che per questo algoritmo, i campi possono essere used/re riutilizzati, quindi non si deve presumere che le etichette dei campi corrispondano agli input.

Per tutti i parametri disponibili, consulta la guida CardVerificationValue1 di riferimento alle 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"
            }
```

Convalida CAVV V7

Example

Per la convalida, gli input sono CVK, i valori di input calcolati e il CAVV fornito durante la transazione per la convalida.

Per tutti i parametri disponibili, consulta la guida di riferimento delle API. CardVerificationValue1

> **Note**
>
> CAVV non è un valore inserito dall'utente (come CVV2) ma viene calcolato dall'emittente ACS. Si dovrebbe valutare se debba sempre essere convalidato quando viene fornito.

```
$ 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 (valore dinamico di verifica della carta) - CVN17

DCVv (dynamic Card Verification Value) è un crittogramma dinamico utilizzato per le transazioni EMV senza Visa-specific contatto. È noto come Early EMV e fornisce una maggiore sicurezza generando

un valore di verifica univoco per ogni transazione. Il dCvv utilizza input tra cui il Primary Account Number (PAN), il PAN Sequence Number (PSN), l'Application Transaction Counter (ATC), la data di scadenza della carta e il codice di servizio. È ancora utilizzato in alcuni luoghi, ma è stato per lo più sostituito da altri algoritmi come CVN18.

Per tutti i parametri disponibili, consulta la guida di riferimento DynamicCardVerificationValue alle API.

Gli input da `DynamicCardVerificationValue` mappare ai seguenti dati relativi all'autorizzazione Visa e alle transazioni EMV:

`ApplicationTransactionCounter`(ATC)

Il contatore delle transazioni gestito dal chip, incrementato per ogni transazione. Nei dati EMV questo è il tag. `9F36`

`PanSequenceNumber`

Il numero di sequenza PAN che distingue le carte che condividono le stesse carte. PAN Nei dati EMV questo è il tag. `5F34`

`CardExpiryDate`

La data di scadenza della richiesta della carta (tag EMV`5F24`), fornita nello stesso formato utilizzato dalle altre chiamate di verifica della carta.

`ServiceCode`

Il codice di servizio a tre cifre riportato nei dati della traccia 2 (equivalente) della carta.

Crea chiave

```
$ 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"}]'
```

La risposta riprende i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/mw7dn3qxvkfh8ztc Ne avrai bisogno nel passaggio successivo.

Genera un DCvv

Example

In questo esempio, genereremo un DCvv per una transazione EMV contactless. Gli input includono il PAN, il numero di sequenza PAN, il contatore delle transazioni dell'applicazione, la data di scadenza della carta e il codice di servizio.

```
$ aws payment-cryptography-data generate-card-validation-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/mw7dn3qxvkfh8ztc \
    --primary-account-number=4111112627662122 \
```

```
    --generation-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,CardExpir
 \
    --validation-data-length 5
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2",
    "ValidationData": "36667"
}
```

Convalida dCvv

Example

In questo esempio, convalideremo un DCvv fornito durante una transazione. Gli stessi input utilizzati per la generazione devono essere forniti per la convalida.

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un. http/200 Se il valore non viene convalidato, restituirà una risposta. 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=4111112627662122 \
    --validation-data=36667 \
    --verification-attributes
 DynamicCardVerificationValue='{ApplicationTransactionCounter=01,PanSequenceNumber=00,CardExpir
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
mw7dn3qxvkfh8ztc",
    "KeyCheckValue": "A8E4D2"
}
```

Funzioni specifiche di Mastercard

Argomenti

- DCVC3
- ARQC -/ CVN14CVN15

- ARQC -/ CVN12CVN13
- 3DS AAV SPA2

DCVC3

DCVC3 è antecedente agli CVN12 schemi EMV CSK e Mastercard e rappresenta un altro approccio all'utilizzo delle chiavi dinamiche. A volte viene riutilizzato anche per altri casi d'uso. In questo schema, gli input sono dati PAN, PSN, Track1/Track2, un numero imprevedibile e un contatore di transazioni (ATC).

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/hrh6qgbi3sk4y3wq. Ne hai bisogno nel passaggio successivo.

Genera un DCVC3

Example

Sebbene DCVC3 sia generalmente generato da una chip card, può anche essere generato manualmente, come in questo esempio

```
$ 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"
  }
```

Convalida il DCVC3

Example

In questo esempio, convalideremo un. DCVC3 Nota che ATC dovrebbe essere fornito come numero esadecimale, ad esempio un contatore di 11 dovrebbe essere rappresentato come 000B. Il servizio prevede un valore a 3 cifre DCVC3, quindi se hai memorizzato un valore di 4 (o 5) cifre, tronca semplicemente i caratteri a sinistra fino a ottenere 3 cifre (ad esempio 15321 dovrebbe avere un valore dei dati di convalida pari a 321).

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un http/200. Se il valore non è convalidato, restituirà una risposta 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 e CVN15 utilizza il metodo EMV CSK di derivazione delle chiavi. I dati esatti delle transazioni variano tra questi due metodi: consulta la documentazione dello schema per i dettagli sulla costruzione del campo dei dati delle transazioni.

ARQC -/ CVN12CVN13

CVN12 e CVN13 sono un vecchio metodo specifico di MasterCard per le transazioni EMV che incorporano un numero imprevedibile nella derivazione per transazione e utilizzano anche un payload diverso. Per informazioni sul contenuto del payload, si prega di contattare lo schema.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/pw3s6nl62t5ushfk. Ne hai bisogno nel passaggio successivo.

Convalida l'ARQC

Example

In questo esempio, convalideremo un ARQC generato utilizzando Mastercard. CVN12

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un http/200. Se l'arqc non è convalidato, restituirà una risposta 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
 0000000017000000000000000840008000800008401605170000000093800000B1F2201030000000000000000000
 \
  --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(Secure Payment Application) L'AAV (Account Authentication Value) viene utilizzato per le transazioni con Mastercard 3DS (noto anche come Mastercard Idenity Check). Fornisce l'autenticazione crittografica per le transazioni di e-commerce utilizzando la generazione MAC basata su HMAC. L'AAV viene generato utilizzando dati specifici della transazione e una chiave segreta condivisa.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di *`KeyArn`* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-west- 2:111122223333:key/q5vjtshsg67cz5gn. Ne hai bisogno nella fase successiva.

Genera SPA2 AAV

Example

In questo esempio, genereremo il componente Issuer Authentication Value (SPA2 IAV) dell'AAV utilizzando la generazione MAC HMAC. I dati del messaggio contengono le informazioni specifiche della transazione che verranno autenticate. Il formato dei dati dei messaggi deve seguire le SPA2 specifiche di Mastercard e non è trattato in questo esempio.

> **Note**
>
> Consulta le specifiche della tua Mastercard per la formattazione necessaria per inserire lo IAV nel valore 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"
}
```

SPA2 Verifica AAV

Example

In questo esempio, verificheremo un SPA2 AAV. Per la verifica vengono forniti gli stessi dati del messaggio e lo stesso valore MAC.

Se AWS Payment Cryptography è in grado di convalidare il MAC, viene restituito un http/200. Se il MAC non è convalidato, restituirà una risposta 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"
}
```

Funzioni specifiche di American Express

Argomenti

- CSC1
- CSC2
- iCSC
- 3DS AVV

CSC1

La versione 1 di CSC è anche nota come algoritmo CSC classico. Il servizio può fornirlo come numero di 3,4 o 5 cifre.

Per tutti i parametri disponibili, vedi AmexCardSecurityCodeVersion1 nella guida di riferimento dell'API.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
            }
        }
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/esh6hn7pxdtttzgq. Ne hai bisogno nella fase successiva.

Genera un 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"
  }
```

Convalida il CSC1

Example

In questo esempio, convalideremo un. CSC1

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un http/200. Se il valore non è convalidato, restituirà una risposta 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

La versione 2 di CSC è anche nota come algoritmo CSC avanzato. Il servizio può fornirlo come numero di 3,4 o 5 cifre. Il codice di servizio per CSC2 è in genere 000.

Per tutti i parametri disponibili, vedi AmexCardSecurityCodeVersion2 nella guida di riferimento delle API.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/erlm445qvunmvoda. Ne hai bisogno nella fase successiva.

Genera un CSC2

In questo esempio, genereremo un CSC2 con una lunghezza di 4. CSC può essere generato con una lunghezza di 3,4 o 5. Per American Express, PANs dovrebbe essere composto da 15 cifre e iniziare con 34 o 37. La data di scadenza è in genere formattata come YYMM. Il codice di servizio può variare: consulta il manuale, ma i valori tipici sono 000, 201 o 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"
}
```

Convalida il CSC2

Example

In questo esempio, convalideremo un. CSC2

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un http/200. Se il valore non è convalidato, restituirà una risposta 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 è anche noto come algoritmo CSC statico e viene calcolato utilizzando CSC versione 2. Il servizio può fornirlo come numero di 3,4 o 5 cifre.

Usa il codice di servizio 999 per calcolare iCSC per una scheda di contatto. Usa il codice di servizio 702 per calcolare iCSC per una carta contactless.

Per tutti i parametri disponibili, vedi AmexCardSecurityCodeVersion2 nella guida di riferimento delle API.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```

Prendi nota di *KeyArn* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 1:111122223333:key/7vrybrbvjcvwtunv. Ne hai bisogno nella fase successiva.

Genera un iCSC

In questo esempio, genereremo un iCSC con una lunghezza di 4, per una carta contactless utilizzando il codice di servizio 702. CSC può essere generato con una lunghezza di 3,4 o 5. Per American Express, PANs dovrebbe essere composto da 15 cifre e iniziare con 34 o 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"
}
```

Convalida l'iCSC

Example

In questo esempio, convalideremo un iCSC.

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un http/200. Se il valore non è convalidato, restituirà una risposta 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 AVV

3DS AEVV (3-D Secure Account Verification Value) viene utilizzato per l'autenticazione American Express 3-D Secure. Utilizza lo stesso algoritmo CSC2 ma con parametri di input diversi. Il campo della data di scadenza deve essere compilato con un numero imprevedibile (casuale) e il codice di servizio è composto dal codice dei risultati di autenticazione AEVV (1 cifra) più il codice di autenticazione a due fattori (2 cifre). La lunghezza di uscita deve essere di 3 cifre.

Per tutti i parametri disponibili, vedi AmexCardSecurityCodeVersion2 nella guida di riferimento delle API.

Crea chiave

```
$ 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"}]'
```

La risposta richiama i parametri della richiesta, tra cui un ARN per le chiamate successive e un Key Check Value (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"
        }
    }
```


Prendi nota di *`KeyArn`* ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:key/kw8djn5qxvfh3ztm. Ne hai bisogno nel passaggio successivo.

Genera un AEVV 3DS

In questo esempio, genereremo un AEVV 3DS con una lunghezza di 3. Il campo della data di scadenza contiene un numero imprevedibile (casuale) (ad esempio 1234) e il codice di servizio è composto dal codice dei risultati di autenticazione AEVV (1 cifra) più il codice di autenticazione di

secondo fattore (2 cifre), ad esempio 543 dove 5 è il codice dei risultati di autenticazione e 43 è il codice di autenticazione di secondo fattore. Per American Express, PANs deve essere composto da 15 cifre e iniziare con 34 o 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"
}
```

Convalida il 3DS AEVV

Example

In questo esempio, convalideremo un AEVV 3DS.

Se AWS Payment Cryptography è in grado di convalidare, viene restituito un http/200. Se il valore non è convalidato, restituirà una risposta 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
}
```

Funzioni specifiche JCB

Argomenti

- ARQC - CVN04
- ARQC - CVN01

ARQC - CVN04

JCB CVN04 utilizza il metodo CSK di derivazione delle chiavi. ??? Consulta la documentazione dello schema per i dettagli sulla costruzione del campo dei dati delle transazioni.

ARQC - CVN01

CVN01 è un vecchio metodo JCB per le transazioni EMV che utilizza la derivazione della chiave per carta anziché la derivazione di sessione (per transazione) e utilizza anche un payload diverso. Questo messaggio viene utilizzato anche da Visa, quindi il nome dell'elemento ha quel nome anche se viene utilizzato anche per JCB. Per informazioni sul contenuto del payload, si prega di contattare la documentazione dello schema.

Crea chiave

```
$ 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"}]'
```

La risposta riprende i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
                    }
                }
```

Prendi nota di `KeyArn` ciò che rappresenta la chiave, ad esempio arn:aws:payment-cryptography:us-east- 2:111122223333:. key/pw3s6nl62t5ushfk Ne avrai bisogno nel passaggio successivo.

Convalida l'ARQC

Example

In questo esempio, convalideremo un ARQC generato utilizzando JCB CVN01. Questo utilizza le stesse opzioni del metodo Visa, da cui il nome del parametro.

Se AWS Payment Cryptography è in grado di convalidare l'ARQC, viene restituito un. http/200 Se l'arqc non è convalidato, restituirà una risposta. 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"
            }
```

UnionPay funzioni specifiche

Gli esempi seguenti riguardano UnionPay International (UPI), a volte indicato anche come Cina UnionPay (CUP).

Argomenti

- Convalida ARQC
- CVN2 (numero di verifica della carta)
- AVN (numero di verifica dell'autenticazione)

Convalida ARQC

UnionPay La convalida ARQC utilizza la modalità di derivazione della chiave EMV_OPTION_A. La chiave di sessione è derivata dal numero di sequenza PAN e dal contatore delle PrimaryAccountNumber transazioni dell'applicazione utilizzando uno schema. UnionPay-specific Per informazioni sul formato dei dati delle transazioni e sull'ordine dei campi, consulta la documentazione dello schema.

Convalida l'ARQC

Example Convalida un ARQC utilizzando la derivazione della chiave UnionPay

Questo esempio mostra come convalidare un ARQC generato utilizzando la derivazione della chiave. UnionPay

Se AWS Payment Cryptography convalida l'ARQC, restituisce una risposta HTTP 200. Se la convalida ARQC non supera la convalida, AWS Payment Cryptography restituisce una risposta HTTP 400 con un codice di motivazione.

```
$ aws payment-cryptography-data verify-auth-request-cryptogram --auth-request-
cryptogram A12345B1A12345B1 \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
pw3s6nl62t5ushfk \
  --major-key-derivation-mode EMV_OPTION_A \
  --transaction-data 123456789ABCDEF80000000000000000 \
```

```
  --session-key-derivation-
attributes='UnionPay={PrimaryAccountNumber=4123412341234123,PanSequenceNumber=01,ApplicationTra
```

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

Se l'ARQC non può essere convalidato, AWS Payment Cryptography restituisce una risposta di errore:

```
An error occurred (VerificationFailedException) when calling the
 VerifyAuthRequestCryptogram operation: Auth request cryptogram verification failed.

Reason: INVALID_AUTH_REQUEST_CRYPTOGRAM
```

CVN2 (numero di verifica della carta)

CVN2 è l'equivalente di CVV2. UnionPay È un numero di verifica della carta utilizzato per le transazioni online e per le transazioni con carta non presente. Crittograficamente, CVN2 utilizza lo stesso algoritmo di CVV2 (). CardVerificationValue2 Gli input sono il numero di conto primario (PAN) e la data di scadenza della carta (4 cifre in formato MMYY).

Crea la chiave

Per creare una chiave CVN2, usa il `create-key` comando con l'uso e l'algoritmo`TR31_C0_CARD_VERIFICATION_KEY`. `TDES_2KEY`

```
$ 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":"CVN2"},{"Key":"CARD_BIN","Value":"47613499"}]'
```

La risposta riproduce i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
    }
}
```

Prendi nota del `KeyArn` valore, ad esempio arn:aws:payment-cryptography:us-east-2:111122223333:. key/7f7g4spf3xcklhzu Ne hai bisogno nei seguenti passaggi.

Genera un CVN2

Example Genera un CVN2 per un determinato PAN

Questo esempio genera un CVN2 per un determinato PAN con immissioni e data di scadenza della carta. `PAN`

Per tutti i parametri disponibili, consulta la guida di riferimento alle CardVerificationValue2 API.

```
$ aws payment-cryptography-data generate-card-validation-data \
```

```
  --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu \
  --primary-account-number=4761349999991234 \
  --generation-attributes CardVerificationValue2='{CardExpiryDate=1226}'
```

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

Convalida un CVN2

Example Convalida un CVN2 per un determinato PAN

Questo esempio convalida un CVN2 per un determinato PAN inserendo i valoriPAN, la data di scadenza della carta e il CVN2 fornito durante la transazione.

Per tutti i parametri disponibili, consulta la guida di riferimento alle API. CardVerificationValue2

> **Note**
>
> CVN2 è un valore immesso dall'utente. Gli errori di convalida periodici non indicano necessariamente un problema.

```
$ aws payment-cryptography-data verify-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-
east-2:111122223333:key/7f7g4spf3xcklhzu \
  --primary-account-number=4761349999991234 \
  --verification-attributes CardVerificationValue2='{CardExpiryDate=1226}' \
  --validation-data 321
```

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

AVN (numero di verifica dell'autenticazione)

AVN (Authentication Verification Number) è il crittogramma UnionPay 3DS. Utilizza l'algoritmo CVV (CardVerificationValue1) con campi di input riutilizzati come di seguito, ma controlla la documentazione dello schema per i dettagli definitivi.

- PAN — Numero di conto primario (16-19 cifre, dall'area Q)
- CardExpiryDate— Le quattro cifre meno significative di un valore derivato dall'ID della transazione DS (DSTRansID)
- ServiceCode— 3 cifre: risultato della transazione di autenticazione (1 cifra) + bit di riempimento '0' (1 cifra) + indicatore del generatore AV (1 cifra)

Crea la chiave

Per creare una chiave AVN, usa il `create-key` comando con l'`TR31_C0_CARD_VERIFICATION_KEY`uso e l'`TDES_2KEY`algoritmo.

```
$ 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":"AVN"},{"Key":"CARD_BIN","Value":"47613499"}]'
```

La risposta riproduce i parametri della richiesta, incluso un ARN per le chiamate successive e un Key Check Value (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"
    }
}
```

Prendi nota del `KeyArn` valore, ad esempio arn:aws:payment-cryptography:us-east-2:111122223333:. key/dnaeyrjgdjjtw6dk Ne hai bisogno nei seguenti passaggi.

Genera un AVN

Example Genera un AVN per un determinato PAN

Questo esempio genera un AVN per un determinato PAN. Il `CardExpiryDate` campo contiene le quattro cifre meno significative derivate dal DSTRansID. Il `ServiceCode` campo contiene il risultato dell'autenticazione, il bit di riempimento e l'indicatore del generatore.

Per tutti i parametri disponibili, CardVerificationValue1 consulta la guida di riferimento alle API.

```
$ aws payment-cryptography-data generate-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk \
  --primary-account-number=4761349999991234 \
  --generation-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=201}'
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
    "KeyCheckValue": "F3FB13",
    "ValidationData": "491"
```

```
}
```

Convalida un AVN

Example Convalida un AVN per un determinato PAN

Questo esempio convalida un AVN per un determinato PAN utilizzando gli stessi campi di input che lo hanno generato, insieme al valore AVN da verificare.

Per tutti i parametri disponibili, consulta la guida CardVerificationValue1 di riferimento alle API.

```
$ aws payment-cryptography-data verify-card-validation-data \
  --key-identifier arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk \
  --primary-account-number=4761349999991234 \
  --verification-attributes
 CardVerificationValue1='{CardExpiryDate=9431,ServiceCode=201}' \
  --validation-data 491
```

```
{
    "KeyArn": "arn:aws:payment-cryptography:us-east-2:111122223333:key/
dnaeyrjgdjjtw6dk",
    "KeyCheckValue": "F3FB13"
}
```

Agevolatori di acquisizione e pagamento

Gli acquirenti PSPs e i facilitatori di pagamento hanno in genere una serie di requisiti crittografici diversi rispetto agli emittenti. Casi di utilizzo comune comprendono:

Decrittografia dei dati

I dati (in particolare i dati pan) possono essere crittografati da un terminale di pagamento e devono essere decrittografati dal backend. Decrypt Data e Encrypt Data supportano una varietà di metodi, tra cui tecniche di derivazione TDES, AES e DUKPT. Il servizio AWS Payment Cryptography stesso è inoltre conforme allo standard PCI P2PE ed è registrato come componente di decrittografia PCI P2PE.

TranslatePin

Per mantenere la conformità al PCI PIN, i sistemi di acquisizione non devono avere i PIN del titolare della carta in chiaro dopo che sono stati inseriti su un dispositivo sicuro. Pertanto, per

trasferire il pin dal terminale a un sistema a valle (come una rete di pagamento o un emittente), è necessario ricrittografarlo utilizzando una chiave diversa da quella utilizzata dal terminale di pagamento. Translate Pin lo fa convertendo un pin crittografato da una chiave all'altra in modo sicuro con il servicebbb. Utilizzando questo comando, i pin possono essere convertiti tra vari schemi come la derivazione TDES, AES e DUKPT e formati di blocchi di pin come ISO-0, ISO-3 e ISO-4.

VerifyMac

I dati di un terminale di pagamento possono essere sottoposti a MAC per garantire che non siano stati modificati durante il transito. Verify Mac GenerateMac supporta una varietà di tecniche che utilizzano chiavi simmetriche, tra cui tecniche di derivazione TDES, AES e DUKPT da utilizzare con l'algoritmo 1 ISO-9797-1, l'algoritmo 3 ISO-9797-1 (Retail MAC) e le tecniche CMAC.

Argomenti aggiuntivi

- Uso dei tasti dinamici

Uso dei tasti dinamici

Dynamic Keys consente di utilizzare chiavi monouso o a uso limitato per operazioni crittografiche come. `EncryptData` Questo flusso può essere utilizzato quando il materiale chiave ruota frequentemente (ad esempio in ogni transazione con carta) e si desidera evitare di importare il materiale chiave nel servizio. Le chiavi di breve durata possono essere utilizzate come parte di SoftPOS/mPOC o di altre soluzioni.

> **Note**
>
> Questo può essere utilizzato al posto del flusso tipico che utilizza la crittografia dei AWS pagamenti, in cui le chiavi crittografiche vengono create o importate nel servizio e le chiavi vengono specificate utilizzando un alias di chiave o una chiave arn.

Le seguenti operazioni supportano Dynamic Keys:

- EncryptData
- DecryptData
- ReEncryptData

- TranslatePin

Decrittografia dei dati

L'esempio seguente mostra l'utilizzo di Dynamic Keys insieme al comando decrypt. L'identificatore di chiave in questo caso è la chiave di avvolgimento (KEK) che protegge la chiave di decrittografia (fornita nel parametro wrapped-key in formato TR-31). La chiave avvolta deve essere lo scopo principale di D0 da utilizzare con il comando decrypt insieme a una modalità di utilizzo di B o D.

Example

```
$ aws payment-cryptography-data decrypt-data --key-identifier
 arn:aws:payment-cryptography:us-east-2:111122223333:key/ov6icy4ryas4zcza
 --cipher-text 1234123412341234123412341234123A --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"
}
```

Traduzione di un pin

L'esempio seguente mostra l'utilizzo di Dynamic Keys insieme al comando translate pin per tradurre da una chiave dinamica a una chiave di lavoro Acquirer semi-statica (AWK). L'identificatore della chiave in ingresso in questo caso è la chiave di avvolgimento (KEK) che protegge la chiave di crittografia dinamica dei pin (PEK) fornita nel formato TR-31. La chiave incapsulata deve essere lo scopo principale `P0` insieme alla modalità di utilizzo di B o D. L'identificatore di chiave in uscita è una chiave di tipo e una modalità di utilizzo di Encrypt=True, wrap=True `TR31_P0_PIN_ENCRYPTION_KEY`

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"
}
```

Caratteristiche specifiche della regione per AWS Crittografia dei pagamenti

Alcune funzionalità possono essere specifiche della regione e non essere utilizzate in altro modo. Queste funzionalità sono descritte più dettagliatamente in questa sezione.

AS2805

L'Australia Standard 2805 (AS2805) è uno standard per i trasferimenti elettronici di fondi utilizzato principalmente per le transazioni di pagamento basate su carta. È gestito da Standards Australia. https://www.standards.org.au/ Lo standard è composto da 6 libri che trattano numerosi argomenti, dal formato dei messaggi agli standard di crittografia.

La parte 6 fornisce indicazioni sulla gestione delle chiavi, compresa la comunicazione da host a host (da nodo a nodo) e i relativi requisiti crittografici, mentre altri aspetti sono trattati in altre parti. Tutta la crittografia di questo standard è attualmente basata su TDES.

AS2805 presenta una serie di differenze rispetto ad altre implementazioni, che sono riassunte di seguito.

Protezione delle chiavi

> Si basa su varianti chiave anziché su blocchi di tasti come in. TR-31/X9.143 AWS Payment Cryptography memorizza internamente tutte le chiavi come blocchi chiave, ma consente l'importazione, l'esportazione e il calcolo utilizzando le varianti definite da AS2805.

Chiavi unidirezionali

> AS2805 impone l'uso di chiavi unidirezionali. Se entrambi i nodi devono generare codici di autenticazione dei messaggi (MAC), utilizzano due chiavi.

Blocchi Pin

> AS2805 definisce una tecnica di derivazione delle chiavi di crittografia dei pin uniche per transazione. Può essere utilizzata al posto di DUKPT. Lo schema AS2805 si basa sui dati delle transazioni (numero di traccia e importo della transazione) rispetto all'uso del contatore delle transazioni da parte del DUKPT.

Convalida dello scambio di chiavi

Definisce un processo per convalidare KEK prima di iniziare a scambiare chiavi funzionanti come le chiavi PIN. In altri schemi, i KEK vengono scambiati raramente e convalidati utilizzando KCV.

AS2805 utilizza il concetto di varianti di chiave anziché di blocchi di chiavi per garantire che le chiavi vengano utilizzate solo per lo scopo previsto (e unico). Di seguito viene illustrato il modo in cui AWS Payment Cryptography esegue la mappatura tra varianti e blocchi di tasti durante l'importazione, l'esportazione o l'esecuzione di altre funzioni crittografiche con le chiavi.

Tipo di chiave AS2805	AWS Tipo di chiave di crittografia dei pagamenti
TERMINAL_MAJOR_KEY_VARIANT_00	TR31_K0_KEY_ENCRYPTION_KEY
PIN_ENCRYPTION_KEY_VARIANT_28	CHIAVE_DI CRITTOGRAFIA TR31_P0_PIN_
MESSAGE_AUTHENTICATION_KEY_ VARIANT_24	TR31_M0_ISO_16609_MAC_KEY
DATA_ENCRYPTION_KEY_VARIANT_22	CHIAVE_SIMMETRIC_DATA_ENCRY PTION_TR31_D0_SYMMETRIC_DATA_
VARIANT_MASK_82, VARIANT_MASK_82C0	Opzioni disponibili come parte del processo di convalida KEK. Questi tipi di chiavi sono effimeri e non vengono memorizzati dal servizio.

Dati due nodi, node1 e node2, gli esempi seguenti sono dal punto di vista di node1. AWS Payment Cryptography supporta le API da entrambi i lati del processo.

Argomenti

- Scambio di chiavi iniziali (KEK)
- Convalida del KEK
- Creazione e trasmissione di chiavi funzionanti
- Esportazione di chiavi di lavoro
- Pin Translation

- Generazione e convalida Mac

Scambio di chiavi iniziali (KEK)

In AS2805, ciascuna parte ha il proprio KEK. KEK (s) si riferisce alla chiave laterale di invio che verrà utilizzata ogni volta che il lato mittente ha bisogno di protect/wrap chiavi e le invia al nodo2. KEK (r) è la chiave creata dal lato opposto (nodo2).

> **Note**
>
> Questi termini sono relativi: una parte crea una chiave (parte mittente) e l'altra la riceve. Quindi, dato KEY1, viene indicato sul nodo1 come KEK (s) e sul nodo2 come KEK (r).

I KEK per AS2805 sono sempre di tipo di chiave = TR31_K0_KEY_ENCRYPTION_KEY poiché vengono utilizzati per proteggere i crittogrammi e non i blocchi di chiavi. Questo corrisponde a TERMINAL_MAJOR_KEY_VARIANT_00 come definito in AS2805 6.1

Fasi:

1.Creare una chiave

Crea una chiave usando l'CreateKeyapi. Creerai una chiave di tipo TR31_K0_KEY_ENCRYPTION_KEY

2.Determina il metodo per scambiare le chiavi con node2

Determina come scambiare KEK con la controparte. Per AS2805, il metodo più comune e interoperabile è RSA Wrap.

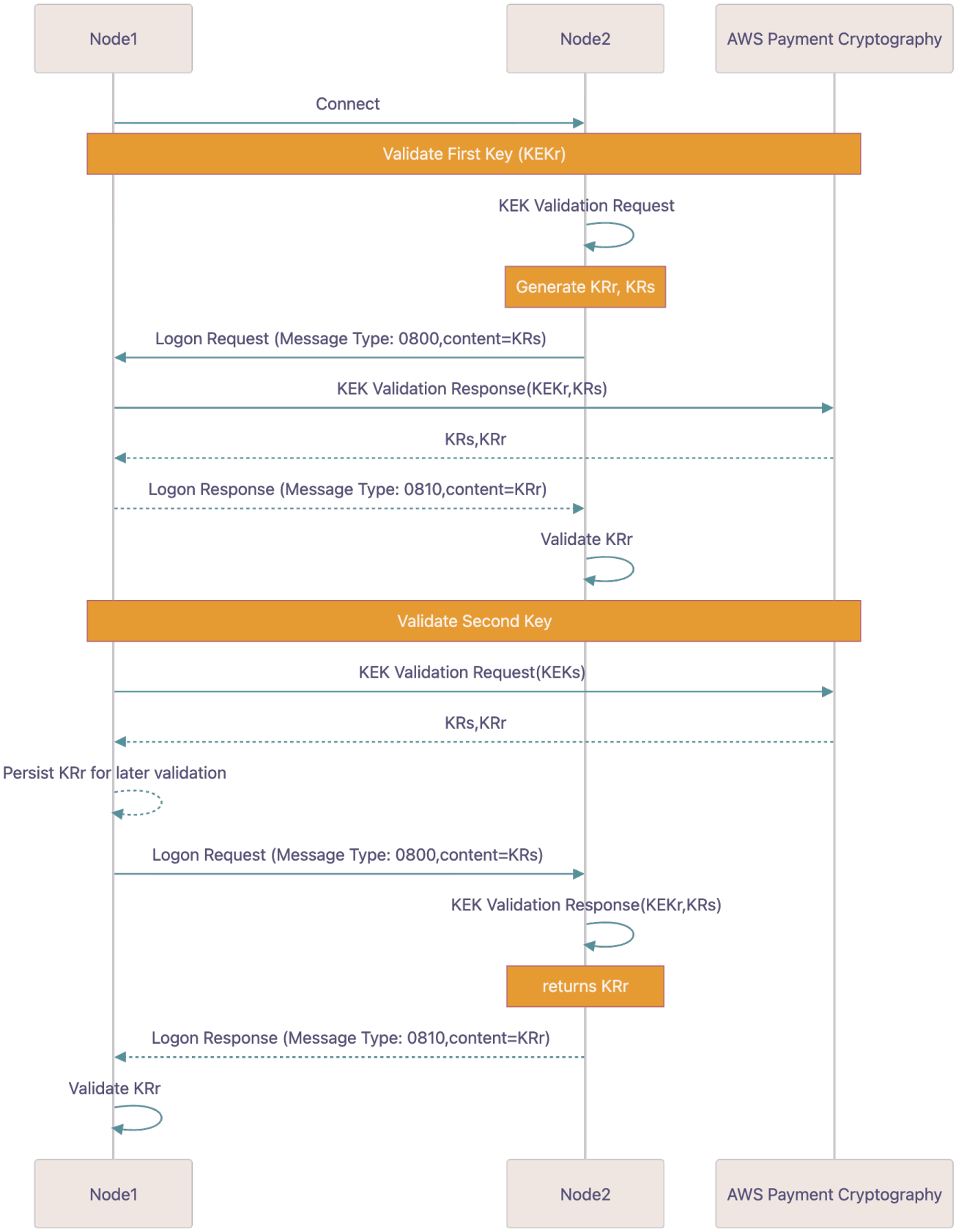
3. Esporta KEK

In base alla selezione effettuata sopra, riceverai un certificato con chiave pubblica da node2. Eseguirai l'esportazione utilizzando quel certificato per proteggere la chiave (o derivarne una se usi ECDH).

4. Importa KeKR

In base alla selezione effettuata sopra, invierai un certificato a chiave pubblica a node2. Eseguirai l'importazione utilizzando quel certificato per caricare il KeKR del nodo 2 nel servizio.

Convalida del KEK



Quando il servizio (node1) si connette al nodo2, ciascuna parte si assicurerà di utilizzare lo stesso KEK per le operazioni successive utilizzando un processo chiamato KEK Validation.

1. Passaggi per convalidare la prima chiave

1.1 Ricevi KRs

Node2 genererà un KRs e te lo invierà come parte del processo di accesso. Possono utilizzare la crittografia dei AWS pagamenti per generare questo valore o un'altra soluzione.

1.2 Genera una risposta di convalida KEK

Il tuo nodo genererà una risposta di convalida KEK con input come KEK (r) e KR forniti nel passaggio 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 Rendimento calcolato in KRr

Restituisci il kRr calcolato al nodo2. Quel nodo lo confronterà con il valore calcolato nel passaggio 1.

2. Passaggi per convalidare la seconda chiave

2.1 Genera kRr e KRs

Il tuo nodo genererà un valore casuale e una copia invertita (invertita) di questo valore utilizzando la crittografia dei pagamenti. AWS Il servizio restituirà entrambi questi valori racchiusi nei KEK. Questi sono noti come KR (s) e 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 Invia KRs al nodo2

Invia i KR al nodo2. Conserva il kRr per una successiva convalida.

2.3 Il nodo 2 genera una risposta di convalida KEK

Node2 utilizza KeKr e KRs, genera il kRr e lo invia nuovamente al tuo servizio.

2.4 Convalida la risposta

Confronta kRr del passaggio 1 e il valore restituito dal passaggio 3. Se corrispondono, procedi.

Creazione e trasmissione di chiavi funzionanti

Le chiavi di lavoro tipiche utilizzate in AS2805 includono due set di chiavi:

Chiavi tra nodi come: chiave pin di zona (ZPK), chiave di crittografia di zona (ZEK) e chiave di autenticazione di zona (ZAK).

Chiavi tra terminali e nodi come: chiave principale del terminale (TMK) e chiave pin del terminale (TPK) se non si utilizza DUKPT.

> **Note**
>
> Consigliamo di ridurre al minimo la chiave per chiave terminale e di sfruttare tecniche come DUKPT, laddove possibile, che TR-34 utilizzano un numero inferiore di chiavi.

Example

In questo esempio, abbiamo utilizzato tag opzionali per tracciare lo scopo e l'uso di questa chiave. I tag non vengono utilizzati come parte della funzione crittografica del sistema, ma possono essere utilizzati per la categorizzazione, il monitoraggio finanziario e possono essere utilizzati per applicare le politiche 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 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"
 }
}
```

Esportazione di chiavi di lavoro

Per mantenere la compatibilità con altre parti, AWS Payment Cryptography supporta le tecniche di key wrapping simmetrico AS2805 che utilizzano varianti di chiave anziché blocchi di tasti come. TR-31 Se più chiavi sono condivise tra le parti, ciascuna deve essere esportata singolarmente. Se i dati vengono inviati in modo bidirezionale, potrebbero esserci due chiavi tra parti dello stesso tipo, ad esempio ZAK (s) e ZAK (r), utilizzate da ciascuna parte per generare codici di autenticazione dei messaggi.

I parametri aggiuntivi da importare ed esportare in questi formati sono specificati nei comandi.

```
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 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"
    }
}
```

Pin Translation

AS2805 descrive una modalità di derivazione delle chiavi specifica per una sessione nella sezione 6.4. Ha uno scopo simile a DUKPT ed entrambi gli algoritmi possono essere utilizzati poiché DUKPT è trattato nella sezione 6.7. In questo schema, una chiave pin di sessione (nota come KPE) viene derivata dalla chiave PIN del terminale utilizzando SystemTraceAuditNumber (STAN) e come dati di derivazione. TransactionAmount

Translate pin è una funzione comune in grado di tradurre to/from una varietà di formati. In questo esempio, traduciamo un pin da un KPE in una chiave di crittografia dei pin (PEK), ad esempio quando si invia un pin a una rete di pagamento.

```
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"
    }
}
```

Generazione e convalida Mac

I comandi MAC di generazione e verifica supportano una varietà di MAC tra cui HMAC, CMAC, EMV MAC, ecc. Per AS2805, esiste una variante aggiuntiva definita in. AS2805.4.1 In genere in AS2805, i messaggi in arrivo vengono verificati utilizzando questo MAC e anche i messaggi in uscita includono un MAC.

```
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"
}
```

Sicurezza in AWS Crittografia dei pagamenti

La sicurezza nel cloud AWS è la massima priorità. Come AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS voi. Il modello di responsabilità condivisa descrive questo modello come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud:AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in sicurezza. Third-party gli auditor testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito dei Programmi di conformità Programmi di AWS conformità Programmi Per ulteriori informazioni sui programmi di conformità che si applicano alla crittografia dei AWS pagamenti, consulta AWS Services in Scope by Compliance Program
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. L'utente è anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della propria azienda e le leggi e normative vigenti.

Questo argomento ti aiuta a capire come applicare il modello di responsabilità condivisa quando usi la crittografia dei AWS pagamenti. Mostra come configurare la crittografia dei AWS pagamenti per soddisfare i tuoi obiettivi di sicurezza e conformità. Imparerai anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse AWS di crittografia dei pagamenti.

Argomenti

- Protezione dei dati in AWS Crittografia dei pagamenti
- Resilienza in AWS Crittografia dei pagamenti
- Sicurezza dell'infrastruttura in AWS Payment Cryptography
- Connessione a AWS Crittografia dei pagamenti tramite un endpoint VPC
- Utilizzo del protocollo TLS post-quantistico ibrido
- Le migliori pratiche di sicurezza per AWS Crittografia dei pagamenti

Protezione dei dati in AWS Crittografia dei pagamenti

Il modello di responsabilità AWS https://aws.amazon.com/compliance/shared-responsibility-model/ si applica alla protezione dei dati nella crittografia dei AWS pagamenti. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, vedi Domande frequenti sulla privacy dei dati Termini legali della regione . Per ulteriori informazioni sulla protezione dei dati in Europa, consulta il Centro generale sulla protezione dei dati (GDPR).

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con Centro identità AWS IAM o AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- SSL/TLS Usalo per comunicare con AWS le risorse. È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'uso dei CloudTrail percorsi per acquisire AWS le attività, consulta Lavorare con i CloudTrail percorsi nella Guida per l'AWS CloudTrail utente.
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-3 per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il Federal Information Processing Standard (FIPS) 140-3.

Ti consigliamo di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con AWS Payment Cryptography o altro Servizi AWS utilizzando la console, l'API o gli SDK. AWS CLI AWS I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Quando si fornisce un URL a un server esterno, suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la richiesta al server.

AWS Payment Cryptography archivia e protegge le chiavi di crittografia dei pagamenti per renderle altamente disponibili e fornirti un controllo degli accessi forte e flessibile.

Argomenti

- Protezione del materiale della chiave
- Crittografia dei dati
- Crittografia dei dati a riposo
- Crittografia dei dati in transito
- Riservatezza del traffico Internet

Protezione del materiale della chiave

Per impostazione predefinita, AWS Payment Cryptography protegge il materiale delle chiavi crittografiche per le chiavi di pagamento gestite dal servizio. Inoltre, AWS Payment Cryptography offre opzioni per importare materiale chiave creato all'esterno del servizio. Per dettagli tecnici sulle chiavi di pagamento e sul materiale chiave, consulta AWS Payment Cryptography Cryptographic Details.

Crittografia dei dati

I dati in AWS Payment Cryptography sono costituiti dalle chiavi AWS Payment Cryptography, dal materiale chiave di crittografia che rappresentano e dai relativi attributi di utilizzo. Il materiale chiave è disponibile in testo normale solo all'interno dei moduli di sicurezza hardware (HSM) di AWS Payment Cryptography e solo quando è in uso. Altrimenti, il materiale e gli attributi chiave vengono crittografati e archiviati in uno storage persistente durevole.

Il materiale chiave generato o caricato da AWS Payment Cryptography per le chiavi di pagamento non esce mai dai confini degli HSM di AWS Payment Cryptography non crittografati. Può essere esportato crittografato tramite le operazioni dell'API AWS Payment Cryptography.

Crittografia dei dati a riposo

AWS Payment Cryptography genera materiale chiave per le chiavi di pagamento negli HSM PCI PTS. HSM-listed Quando non in uso, il materiale della chiave viene crittografato da una chiave HSM e scritto in uno storage persistente e durevole. Il materiale chiave per le chiavi di crittografia dei

pagamenti e le chiavi di crittografia che proteggono il materiale chiave non lascia mai gli HSM in formato di testo normale.

La crittografia e la gestione del materiale chiave per le chiavi di crittografia dei pagamenti sono gestite interamente dal servizio.

Per maggiori dettagli, consulta i dettagli crittografici di AWS Key Management Service.

Crittografia dei dati in transito

Il materiale chiave generato o caricato da AWS Payment Cryptography per le chiavi di pagamento non viene mai esportato o trasmesso nelle operazioni dell'API di AWS Payment Cryptography in chiaro. AWS Payment Cryptography utilizza identificatori chiave per rappresentare le chiavi nelle operazioni API.

Tuttavia, alcune operazioni API esportano chiavi crittografate da una chiave di scambio di chiavi precedentemente condivisa o asimmetrica. Inoltre, i clienti possono utilizzare le operazioni API per importare materiale chiave crittografato per le chiavi di pagamento.

Tutte le chiamate API AWS di Payment Cryptography devono essere firmate e trasmesse utilizzando Transport Layer Security (TLS). AWS La crittografia dei pagamenti richiede versioni TLS e suite di crittografia definite da PCI come «crittografia avanzata». Tutti gli endpoint di servizio supportano TLS 1.2—1.3 e TLS ibrido post-quantistico.

Per maggiori dettagli, consulta i dettagli crittografici di AWS Key Management Service.

Riservatezza del traffico Internet

AWS Payment Cryptography supporta una console di gestione AWS e una serie di operazioni API che consentono di creare e gestire chiavi di pagamento e utilizzarle in operazioni crittografiche.

AWS Payment Cryptography supporta due opzioni di connettività di rete dalla rete privata ad AWS.

- Una connessione VPN IPSec su Internet.
- AWS Direct Connect, che collega la rete interna a una posizione AWS Direct Connect tramite un cavo in fibra ottica Ethernet standard.

Tutte le chiamate API Payment Cryptography devono essere firmate e trasmesse tramite Transport Layer Security (TLS). Le chiamate richiedono anche una moderna suite di cifratura che supporta la perfect forward secrecy. Il traffico verso i moduli di sicurezza hardware (HSM) che memorizzano

il materiale chiave per le chiavi di pagamento è consentito solo da host API AWS Payment Cryptography noti sulla rete interna AWS.

Per connetterti direttamente ad AWS Payment Cryptography dal tuo cloud privato virtuale (VPC) senza inviare traffico sulla rete Internet pubblica, utilizza gli endpoint VPC, basati su AWS. PrivateLink Per ulteriori informazioni, consulta Connessione a AWS Payment Cryptography tramite un endpoint VPC.

AWS Payment Cryptography supporta anche un'opzione ibrida di scambio di chiavi post-quantistiche per il protocollo di crittografia di rete Transport Layer Security (TLS). Puoi utilizzare questa opzione con TLS quando ti connetti agli endpoint dell'API AWS Payment Cryptography.

Resilienza in AWS Crittografia dei pagamenti

AWS l'infrastruttura globale è costruita attorno alle AWS regioni e alle zone di disponibilità. Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, connesse tramite reti altamente ridondanti, a bassa latenza e throughput elevato. Con le zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni su AWS regioni e zone di disponibilità, consulta AWS Global Infrastructure.

Isolamento regionale

AWS Payment Cryptography è un servizio regionale disponibile in più regioni.

Il design isolato a livello regionale di AWS Payment Cryptography garantisce che un problema di disponibilità in una regione AWS non possa influire sul funzionamento di AWS Payment Cryptography in nessun'altra regione. AWS Payment Cryptography è progettato per garantire zero tempi di inattività pianificati, con tutti gli aggiornamenti del software e le operazioni di scalabilità eseguite senza interruzioni e in modo impercettibile.

L'AWS Payment Cryptography Service Level Agreement (SLA) include un impegno di servizio del 99,99% per tutte le API di crittografia di pagamento. Per adempiere a questo impegno, AWS Payment Cryptography garantisce che tutti i dati e le informazioni di autorizzazione necessari per eseguire una richiesta API siano disponibili su tutti gli host regionali che ricevono la richiesta.

L'infrastruttura AWS Payment Cryptography è replicata in almeno tre zone di disponibilità (AZ) in ciascuna regione. Per garantire che i guasti di più host non influiscano sulle prestazioni di AWS

Payment Cryptography, AWS Payment Cryptography è progettato per servire il traffico dei clienti proveniente da qualsiasi AZ in una regione.

Le modifiche apportate alle proprietà o alle autorizzazioni di una chiave di pagamento vengono replicate su tutti gli host della regione per garantire che le richieste successive possano essere elaborate correttamente da qualsiasi host della regione. Le richieste di operazioni crittografiche che utilizzano la chiave di pagamento vengono inoltrate a una flotta di moduli di sicurezza hardware (HSM) di AWS Payment Cryptography, ognuno dei quali può eseguire l'operazione con la chiave di pagamento.

Multi-tenant design

Il design multi-tenant di AWS Payment Cryptography consente di soddisfare gli SLA di disponibilità e di sostenere tassi di richiesta elevati, proteggendo al contempo la riservatezza delle chiavi e dei dati.

Vengono implementati diversi meccanismi di rafforzamento dell'integrità per garantire che la chiave di pagamento specificata per l'operazione crittografica sia sempre quella utilizzata.

Il materiale chiave in testo semplice per le tue chiavi di crittografia dei pagamenti è ampiamente protetto. Il materiale chiave viene crittografato nell'HSM non appena viene creato e il materiale crittografato viene immediatamente trasferito in un archivio sicuro. La chiave crittografata viene recuperata e decrittografata all'interno del modulo HSM solo nel momento in cui viene utilizzata. La chiave in testo normale rimane nella memoria HSM solo per il tempo necessario al completamento dell'operazione di crittografia. Il materiale della chiave in testo normale non lascia mai i moduli HSM e non viene mai scritto su storage persistente.

Per ulteriori informazioni sui meccanismi utilizzati da AWS Payment Cryptography per proteggere le tue chiavi, consulta AWS Payment Cryptography Cryptographic Details.

Sicurezza dell'infrastruttura in AWS Payment Cryptography

In quanto servizio gestito, AWS Payment Cryptography è protetto dalle procedure AWS globali di sicurezza della rete descritte nel white paper Amazon Web Services: Overview of Security Processes.

Utilizzi le chiamate API AWS pubblicate per accedere AWS Payment Cryptography tramite la rete. I client devono supportare Transport Layer Security (TLS) 1.2 o versioni successive. I clienti devono inoltre supportare suite di crittografia con Perfect Forward Secrecy (PFS) come Ephemeral (DHE)

o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Diffie-Hellman La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate tramite un ID chiave di accesso e una chiave di accesso segreta associata a un principal IAM. In alternativa, è possibile utilizzare Servizio di token di sicurezza AWS (AWS STS) per generare le credenziali di sicurezza temporanee per firmare le richieste.

Isolamento degli host fisici

La sicurezza dell'infrastruttura fisica utilizzata da AWS Payment Cryptography è soggetta ai controlli descritti nella sezione Sicurezza fisica e ambientale di Amazon Web Services: panoramica dei processi di sicurezza. Puoi trovare altri dettagli nei report di conformità e nei risultati degli audit di terze parti elencati nella sezione precedente.

AWS Payment Cryptography è supportata da moduli di sicurezza hardware (HSM) PCI PTS HSM-listed dedicati e disponibili in commercio. Il materiale chiave per le chiavi AWS Payment Cryptography viene archiviato solo nella memoria volatile sugli HSM e solo mentre la chiave di crittografia di pagamento è in uso. Gli HSM si trovano in rack ad accesso controllato all'interno dei data center Amazon che impongono il doppio controllo per qualsiasi accesso fisico. Per informazioni dettagliate sul funzionamento degli HSM di AWS Payment Cryptography, consulta AWS Payment Cryptography Cryptographic Details.

Connessione a AWS Crittografia dei pagamenti tramite un endpoint VPC

Puoi connetterti direttamente alla crittografia dei AWS pagamenti tramite un endpoint di interfaccia privato nel tuo cloud privato virtuale (VPC). Quando utilizzi un endpoint VPC di interfaccia, la comunicazione tra il tuo VPC e AWS Payment Cryptography viene condotta interamente all'interno della rete. AWS

AWS Payment Cryptography supporta gli endpoint Amazon Virtual Private Cloud (Amazon VPC) basati su. AWS PrivateLink Ogni endpoint VPC è rappresentato da una o più interfacce di rete elastiche (ENI) con indirizzi IP privati nelle sottoreti del VPC.

L'endpoint VPC dell'interfaccia collega il tuo VPC direttamente a AWS Payment Cryptography senza un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. AWS Direct Connect Le istanze del tuo VPC non necessitano di indirizzi IP pubblici per comunicare con Payment Cryptography. AWS

Regioni

AWS Payment Cryptography supporta gli endpoint VPC e le policy sugli endpoint VPC in tutti i casi in cui è supportata la crittografia di pagamento. Regioni AWS AWS

Argomenti

- Considerazioni per AWS Endpoint VPC per la crittografia dei pagamenti
- Creazione di un endpoint VPC per AWS Crittografia dei pagamenti
- Connessione a un AWS Endpoint VPC per la crittografia dei pagamenti
- Controllo dell'accesso all'endpoint VPC
- Utilizzo di un endpoint VPC in un'istruzione di policy
- Registrazione dell'endpoint VPC

Considerazioni per AWS Endpoint VPC per la crittografia dei pagamenti

> **Note**
>
> Sebbene gli endpoint VPC consentano di connettersi al servizio in una sola zona di disponibilità (AZ), consigliamo di collegarsi a tre zone di disponibilità per scopi di alta disponibilità e ridondanza.

Prima di configurare un endpoint VPC di interfaccia per la crittografia dei AWS pagamenti, consulta l'argomento Proprietà e limitazioni degli endpoint di interfaccia nella Guida. AWS PrivateLink

AWS Il supporto per la crittografia dei pagamenti per un endpoint VPC include quanto segue.

- Puoi utilizzare il tuo endpoint VPC per richiamare tutte le operazioni del AWS Payment Cryptography Control Plane e AWS le operazioni del Payment Cryptography Data Plane da un VPC.
- Puoi creare un endpoint VPC di interfaccia che si connette a un endpoint della regione Payment Cryptography. AWS
- AWS La crittografia dei pagamenti è costituita da un piano di controllo e un piano dati. Puoi scegliere di configurare uno o entrambi i servizi secondari, AWS PrivateLink ma ognuno è configurato separatamente.

- Puoi utilizzare AWS CloudTrail i log per verificare l'utilizzo delle chiavi di crittografia dei AWS pagamenti tramite l'endpoint VPC. Per informazioni dettagliate, vedi Registrazione dell'endpoint VPC.

Creazione di un endpoint VPC per AWS Crittografia dei pagamenti

Puoi creare un endpoint VPC per la crittografia dei AWS pagamenti utilizzando la console Amazon VPC o l'API Amazon VPC. Per ulteriori informazioni, consulta la sezione Creazione di un endpoint di interfaccia nella Guida per l'utente di AWS PrivateLink .

- Per creare un endpoint VPC per la crittografia dei AWS pagamenti, utilizza i seguenti nomi di servizio:

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

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

  Ad esempio, nella regione degli Stati Uniti occidentali (Oregon) (`us-west-2`), i nomi dei servizi sarebbero:

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

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

Per semplificare l'utilizzo dell'endpoint VPC, puoi abilitare un nome DNS privato per l'endpoint VPC. Se si seleziona l'opzione Abilita nome DNS, il nome host DNS standard di AWS Payment Cryptography si risolve nell'endpoint VPC. Ad esempio, `https://controlplane.payment-cryptography.us-west-2.amazonaws.com` si risolverebbe in un endpoint VPC connesso al nome del servizio `com.amazonaws.us-west-2.payment-cryptography.controlplane`.

Questa opzione rende più semplice utilizzare l'endpoint VPC. Gli AWS SDK AWS CLI utilizzano il nome host DNS standard di AWS Payment Cryptography per impostazione predefinita, quindi non è necessario specificare l'URL dell'endpoint VPC nelle applicazioni e nei comandi.

Per ulteriori informazioni, consulta la sezione Accesso a un servizio tramite un endpoint di interfaccia nella Guida di AWS PrivateLink .

Connessione a un AWS Endpoint VPC per la crittografia dei pagamenti

Puoi connetterti alla crittografia dei AWS pagamenti tramite l'endpoint VPC utilizzando un SDK, o. AWS AWS CLI AWS Strumenti per PowerShell Per specificare l'endpoint VPC, utilizzare il nome DNS.

Ad esempio, il comando list-keys utilizza il parametro `endpoint-url` per specificare l'endpoint VPC. Per utilizzare un comando come questo, sostituisci l'ID dell'endpoint VPC con uno presente nel tuo account.

```
$ aws payment-cryptography list-keys --endpoint-url https://
vpce-1234abcdf5678c90a-09p7654s-us-east-1a.ec2.us-east-1.vpce.amazonaws.com
```

Se hai attivato nomi host privati al momento della creazione dell'endpoint VPC, non è necessario specificare l'URL dell'endpoint VPC nella configurazione dell'applicazione o nei comandi della CLI. Il nome host DNS standard AWS di Payment Cryptography si risolve nell'endpoint VPC. Gli SDK AWS CLI e gli SDK utilizzano questo nome host per impostazione predefinita, quindi puoi iniziare a utilizzare l'endpoint VPC per connetterti a un endpoint regionale di AWS Payment Cryptography senza modificare nulla negli script e nelle applicazioni.

Per utilizzare nomi host privati, gli attributi `enableDnsHostnames` e `enableDnsSupport` del VPC devono essere impostati su `true`. Per impostare questi attributi, utilizzate l'operazione. ModifyVpcAttribute Per informazioni dettagliate, consulta la sezione Visualizzazione e aggiornamento degli attributi DNS per il VPC nella Guida per l'utente di Amazon VPC.

Controllo dell'accesso all'endpoint VPC

Per controllare l'accesso al tuo endpoint VPC per la crittografia dei AWS pagamenti, allega una policy sugli endpoint VPC all'endpoint VPC. La policy sugli endpoint determina se i committenti possono utilizzare l'endpoint VPC per chiamare le operazioni di Payment Cryptography con risorse specifiche di Payment Cryptography. AWS AWS

Puoi creare una policy di endpoint VPC quando crei l'endpoint e puoi modificare la policy di endpoint VPC in qualsiasi momento. Utilizza la console di gestione VPC o le operazioni or. CreateVpcEndpoint ModifyVpcEndpoint Puoi anche creare e modificare una policy per gli endpoint VPC utilizzando un modello. AWS CloudFormation Per informazioni sull'utilizzo della console di gestione VPC, consulta la sezione Creazione di un endpoint di interfaccia e Modifica di un endpoint di interfaccia nella Guida di AWS PrivateLink .

Per informazioni sulla scrittura e sulla formattazione di un documento di policy JSON, consulta la Documentazione di riferimento sulla policy IAM JSON nella Guida per l'utente di IAM.

Argomenti

- Informazioni sulle policy di endpoint VPC
- Policy di endpoint VPC predefinita
- Creazione di una policy degli endpoint VPC
- Visualizzazione di una policy di endpoint VPC

Informazioni sulle policy di endpoint VPC

Affinché una richiesta AWS di crittografia dei pagamenti che utilizza un endpoint VPC abbia esito positivo, il committente richiede le autorizzazioni di due fonti:

- Una policy basata sull'identità deve concedere al principale il permesso di richiamare l'operazione sulla risorsa (chiavi o alias di Payment Cryptography)AWS .
- Una policy di endpoint VPC deve concedere l'autorizzazione al principale per utilizzare l'endpoint per effettuare la richiesta.

Ad esempio, una policy chiave potrebbe concedere al principale l'autorizzazione a chiamare Decrypt su una particolare chiave di crittografia di pagamento. AWS Tuttavia, la politica degli endpoint VPC potrebbe non consentire al principale di utilizzare le chiavi di crittografia `Decrypt` dei AWS pagamenti utilizzando l'endpoint.

Oppure una policy sugli endpoint VPC potrebbe consentire a un principale di utilizzare l'endpoint per richiamare determinate chiavi di crittografia dei pagamenti. StopKeyUsage AWS Ma se il principale non dispone delle autorizzazioni previste da una policy IAM, la richiesta ha esito negativo.

Policy di endpoint VPC predefinita

Ogni endpoint VPC dispone di una policy di endpoint VPC, ma non è necessario specificare la policy. Se non specifichi una policy, la policy di endpoint predefinita consente tutte le operazioni effettuate da tutte i principali su tutte le risorse dell'endpoint.

Tuttavia, per le risorse AWS di Payment Cryptography, il principale deve inoltre disporre dell'autorizzazione a richiamare l'operazione da una policy IAM. Pertanto, in pratica, la policy

predefinita indica che se un principale dispone dell'autorizzazione per chiamare un'operazione su una risorsa, può anche chiamarla utilizzando l'endpoint.

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

Per permettere ai principali di utilizzare l'endpoint VPC solo per un sottoinsieme di operazioni consentite, crea o modifica la policy di endpoint VPC.

Creazione di una policy degli endpoint VPC

Una policy di endpoint VPC determina se un principale dispone dell'autorizzazione per utilizzare l'endpoint VPC per eseguire operazioni su una risorsa. Per le risorse AWS di Payment Cryptography, il principale deve inoltre disporre dell'autorizzazione per eseguire le operazioni in base a una policy IAM.

Ogni istruzione della policy di endpoint VPC richiede i seguenti elementi:

- Il principale che può eseguire operazioni.
- Le azioni che possono essere eseguite
- Le risorse sui cui si possono eseguire le azioni

L'istruzione della policy non specifica l'endpoint VPC. Si applica invece a qualsiasi endpoint VPC a cui è collegata la policy. Per ulteriori informazioni, consulta Controllo degli accessi ai servizi con endpoint VPC nella Guida per l'utente di Amazon VPC.

Di seguito è riportato un esempio di policy sugli endpoint VPC per AWS la crittografia dei pagamenti. Se collegata a un endpoint VPC, questa politica consente di utilizzare l'endpoint VPC `ExampleUser` per chiamare le operazioni specificate sulle chiavi di crittografia di pagamento specificate. AWS Prima di utilizzare una policy come questa, sostituisci l'identificatore principale e chiave di esempio con valori validi del tuo account.

```
{
   "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 registra tutte le operazioni che utilizzano l'endpoint VPC. Tuttavia, i tuoi CloudTrail registri non includono le operazioni richieste dai titolari in altri account o le operazioni relative alle chiavi di crittografia dei AWS pagamenti in altri account.

Pertanto, potresti voler creare una policy sugli endpoint VPC che impedisca ai titolari degli account esterni di utilizzare l'endpoint VPC per chiamare qualsiasi operazione di crittografia dei AWS pagamenti su qualsiasi chiave dell'account locale.

L'esempio seguente utilizza la chiave di condizione aws: PrincipalAccount global per negare l'accesso a tutti i principali per tutte le operazioni su tutte le chiavi di crittografia dei AWS pagamenti, a meno che il principale non si trovi nell'account locale. Prima di utilizzare una policy come questa, sostituisci l'ID account dell'esempio con uno valido.

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

```
          "aws:PrincipalAccount": "111122223333"
        }
      }
    }
  ]
}
```

Visualizzazione di una policy di endpoint VPC

Per visualizzare la politica degli endpoint VPC per un endpoint, usa la console di gestione VPC o l'operazione. DescribeVpcEndpoints

Il AWS CLI comando seguente ottiene la policy per l'endpoint con l'ID dell'endpoint VPC specificato.

Prima di eseguire questo comando, sostituisci l'ID endpoint dell'esempio con un ID valido del tuo account.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpce-1234abcdf5678c90a`].[PolicyDocument]'
--output text
```

Utilizzo di un endpoint VPC in un'istruzione di policy

È possibile controllare l'accesso alle risorse e alle operazioni AWS di Payment Cryptography quando la richiesta proviene da VPC o utilizza un endpoint VPC. A tale scopo, utilizzane una policy IAM ???

- Usa la chiave di condizione `aws:sourceVpce` per concedere o limitare l'accesso in base all'endpoint VPC.
- Usa la chiave di condizione `aws:sourceVpc` per concedere o limitare l'accesso in base al VPC che ospita l'endpoint privato.

> **Note**
>
> La chiave `aws:sourceIP` condizionale non è efficace quando la richiesta proviene da un endpoint https://docs.aws.amazon.com/vpc/latest/userguide/vpc-endpoints.html Amazon VPC. Per limitare le richieste a un endpoint VPC, utilizza il comando `aws:sourceVpce` o le chiavi di condizione `aws:sourceVpc`. Per ulteriori informazioni, consulta la sezione Gestione delle identità e degli accessi per endpoint VPC e servizi endpoint VPC nella Guida di AWS PrivateLink .

Puoi utilizzare queste chiavi di condizione globali per controllare l'accesso alle chiavi AWS di crittografia di pagamento, agli alias e a operazioni del genere CreateKey che non dipendono da alcuna risorsa particolare.

Ad esempio, la seguente politica chiave di esempio consente a un utente di eseguire particolari operazioni crittografiche con una chiave di crittografia di AWS pagamento solo quando la richiesta utilizza l'endpoint VPC specificato, bloccando l'accesso sia da Internet che dalle connessioni (se configurate AWS PrivateLink). Quando un utente effettua una richiesta a AWS Payment Cryptography, l'ID dell'endpoint VPC nella richiesta viene confrontato con il valore della chiave di condizione nella `aws:sourceVpce` policy. Se non corrisponde, la richiesta viene rifiutata.

Per utilizzare una policy come questa, sostituisci l'ID segnaposto e gli Account AWS ID degli endpoint VPC con valori validi per il tuo account.

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"
                }
            }
        }

    ]
}
```

Puoi anche utilizzare la chiave `aws:sourceVpc` condizionale per limitare l'accesso alle tue chiavi di crittografia dei AWS pagamenti in base al VPC in cui risiede l'endpoint VPC.

La seguente politica chiave di esempio consente i comandi che gestiscono le chiavi di crittografia dei AWS pagamenti solo quando provengono da. `vpc-12345678` Inoltre, consente i comandi che utilizzano le chiavi AWS di crittografia dei pagamenti per operazioni crittografiche solo quando provengono da. `vpc-2b2b2b2b` Puoi usare una policy come questa se un'applicazione è in esecuzione in un VPC, ma devi utilizzare un secondo VPC separato per le funzioni di gestione.

Per utilizzare una policy come questa, sostituisci l'ID segnaposto e gli Account AWS ID degli endpoint VPC con valori validi per il tuo account.

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": "*"
        }
    ]
}
```

Registrazione dell'endpoint VPC

AWS CloudTrail registra tutte le operazioni che utilizzano l'endpoint VPC. Quando una richiesta a AWS Payment Cryptography utilizza un endpoint VPC, l'ID dell'endpoint VPC viene visualizzato nella voce di registro che registra la AWS CloudTrail richiesta. Puoi utilizzare l'ID dell'endpoint per verificare l'uso del tuo endpoint Payment Cryptography VPC. AWS

Per proteggere il tuo VPC, le richieste che vengono rifiutate da una policy sugli endpoint VPC, ma che altrimenti sarebbero state consentite, non vengono registrate. AWS CloudTrail

Ad esempio, questa voce di log di esempio registra una richiesta GenerateMac che utilizza l'endpoint VPC. Il campo `vpcEndpointId` viene visualizzato alla fine della voce di 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,
```

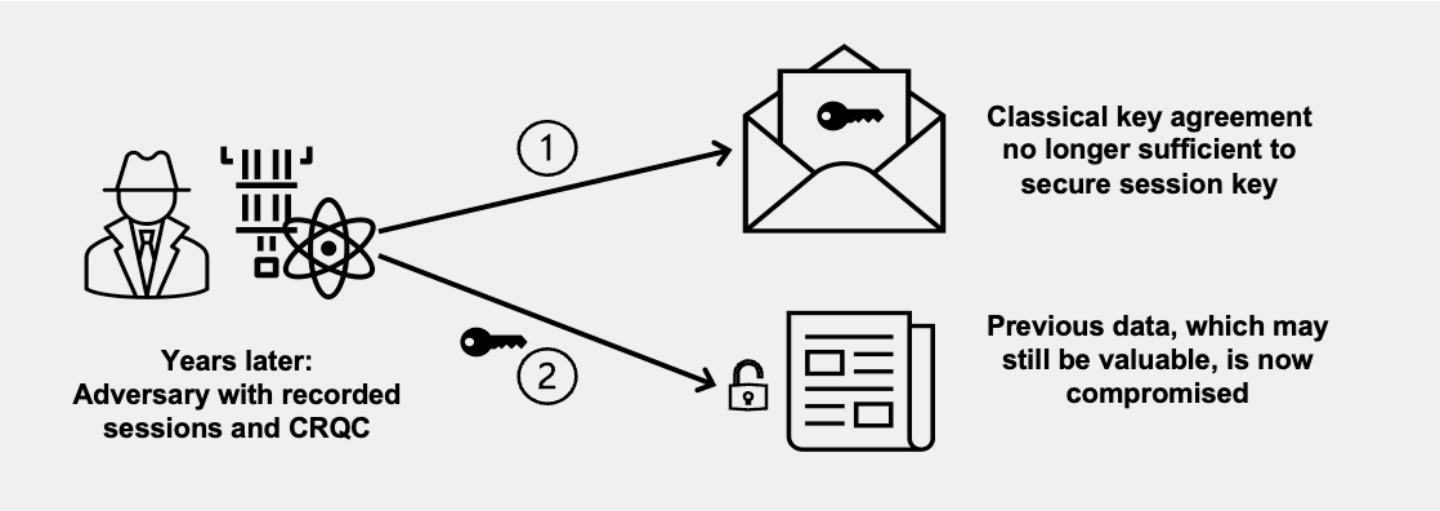
```
            "exportable": true,
            "keyState": "CREATE_COMPLETE",
            "keyOrigin": "AWS_PAYMENT_CRYPTOGRAPHY",
            "createTimestamp": "May 27, 2024, 7:49:54 PM",
            "usageStartTimestamp": "May 27, 2024, 7:49:54 PM"
        }
    },
    "requestID": "f3020b3c-4e86-47f5-808f-14c7a4a99161",
    "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"
    }
}
```

Utilizzo del protocollo TLS post-quantistico ibrido

AWS Payment Cryptography e molti altri servizi supportano un'opzione ibrida di scambio di chiavi post-quantistiche per il protocollo di crittografia di rete Transport Layer Security (TLS). Puoi usare questa opzione TLS quando ti connetti agli endpoint API o quando usi gli SDK. AWS Queste caratteristiche opzionali di scambio di chiavi post-quantistiche ibride sono sicure almeno quanto la crittografia TLS che utilizziamo oggi e potrebbero fornire ulteriori vantaggi per la sicurezza a lungo termine.

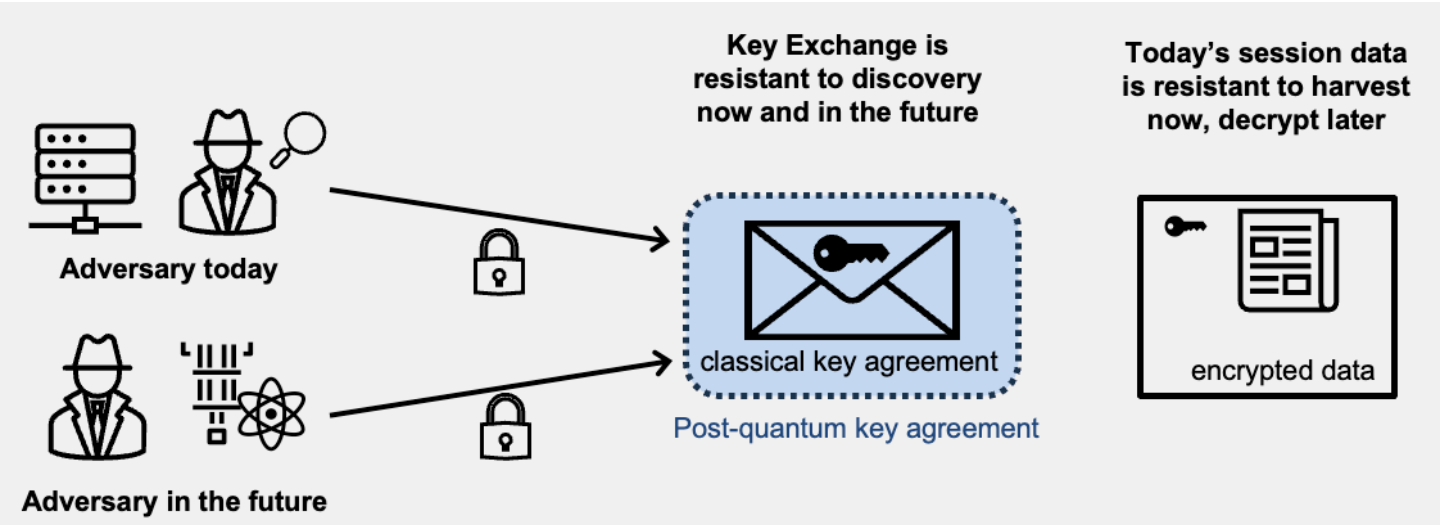
I dati inviati ai servizi abilitati sono protetti in transito dalla crittografia fornita da una connessione Transport Layer Security (TLS). Le classiche suite di crittografia basate su RSA ed ECC supportate da AWS Payment Cryptography per le sessioni TLS rendono impossibili gli attacchi di forza bruta ai meccanismi di scambio delle chiavi con la tecnologia attuale. Tuttavia, se in futuro i computer quantistici (CRQC) su larga scala o crittograficamente rilevanti diventeranno pratici, i meccanismi di scambio di chiavi TLS esistenti saranno suscettibili a questi attacchi. È possibile che gli avversari inizino a raccogliere dati crittografati ora con la speranza di poterli decrittografare in futuro (raccogli ora, decifrali dopo). Se state sviluppando applicazioni che si basano sulla riservatezza a lungo

termine dei dati trasmessi tramite una connessione TLS, dovreste prendere in considerazione un piano di migrazione alla crittografia post-quantistica prima che i computer quantistici su larga scala diventino disponibili all'uso. AWS sta lavorando per prepararsi per questo futuro e vogliamo che anche tu sia ben preparato.



Per proteggere i dati crittografati oggi da potenziali attacchi futuri, AWS sta partecipando con la comunità crittografica allo sviluppo di algoritmi quantistici resistenti o post-quantistici. AWS ha implementato suite ibride di crittografia post-quantistica per lo scambio di chiavi che combinano elementi classici e post-quantistici per garantire che la connessione TLS sia almeno altrettanto forte di quella delle suite di crittografia classiche.

Queste suite di crittografia ibride sono disponibili per l'uso sui carichi di lavoro di produzione quando si utilizzano le versioni recenti degli SDK. AWS Per ulteriori informazioni su enable/disable questo comportamento, consulta Abilitazione del TLS post-quantistico ibrido.



Informazioni sullo scambio di chiavi post-quantistiche ibride in TLS

Gli algoritmi AWS utilizzati sono un ibrido che combina Elliptic Curve Diffie-Hellman (ECDH), un classico algoritmo di scambio di chiavi utilizzato oggi in TLS, con Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM), un algoritmo di crittografia a chiave pubblica e definizione di chiavi che il National Institute for Standards and Technology (NIST) ha designato come primo algoritmo standard di accordo di chiavi post-quantistico. https://csrc.nist.gov/pubs/fips/203/final Questo algoritmo ibrido utilizza ciascuno degli algoritmi in modo indipendente per generare una chiave. Quindi combina crittograficamente le due chiavi.

Come utilizzarlo

Per informazioni sull'abilitazione del TLS post-quantistico ibrido negli AWS SDK, vedi Abilitazione del TLS post-quantistico ibrido.

Scopri di più su PQC

Per informazioni sul progetto di crittografia post-quantistica presso il National Institute for Standards and Technology (NIST), vedere Cryptography. Post-Quantum

Per informazioni sulla standardizzazione della crittografia post-quantistica del NIST, vedere Cryptography Standardization. Post-Quantum

Abilitazione del TLS ibrido post-quantistico

Questo argomento è stato spostato nella Guida di riferimento per AWS SDK e strumenti.

Le migliori pratiche di sicurezza per AWS Crittografia dei pagamenti

AWS Payment Cryptography supporta molte funzionalità di sicurezza integrate o che puoi implementare opzionalmente per migliorare la protezione delle chiavi di crittografia e garantire che vengano utilizzate per lo scopo previsto, tra cui le politiche IAM, un ampio set di chiavi di condizione delle policy per perfezionare le policy e le policy IAM e l'applicazione integrata delle regole PCI PIN relative ai blocchi di chiavi.

> **Important**
>
> Le linee guida generali fornite non rappresentano una soluzione di sicurezza completa. Poiché non tutte le best practice sono appropriate per tutte le situazioni, non sono prescrittive.

- Utilizzo delle chiavi e modalità di utilizzo: la crittografia dei AWS pagamenti segue e applica le restrizioni sull'utilizzo delle chiavi e sulle modalità di utilizzo, come descritto nella specifica ANSI X9 TR 31-2018 Interoperable Secure Key Exchange Key Block e coerente con il requisito di sicurezza PCI PIN 18-3. Ciò limita la possibilità di utilizzare una singola chiave per più scopi e lega crittograficamente i metadati della chiave (come le operazioni consentite) al materiale della chiave stessa. AWS La crittografia dei pagamenti impone automaticamente queste restrizioni, ad esempio una chiave di crittografia a chiave (TR31_K0_KEY_ENCRYPTION_KEY) non può essere utilizzata anche per la decrittografia dei dati. Per ulteriori dettagli, consulta Comprendere gli attributi chiave per AWS Chiave di crittografia dei pagamenti.
- Limita la condivisione del materiale delle chiavi simmetriche: condividete il materiale chiave simmetrico (come le chiavi di crittografia dei pin o le chiavi di crittografia delle chiavi) solo con un massimo di un'altra entità. Se è necessario trasmettere materiale sensibile a più entità o partner, crea chiavi aggiuntive. AWS La crittografia dei pagamenti non espone mai in chiaro il materiale delle chiavi simmetriche o le chiavi private asimmetriche.
- Usa alias o tag per associare le chiavi a determinati casi d'uso o partner: gli alias possono essere utilizzati per indicare facilmente il caso d'uso associato a una chiave, ad esempio per indicare una chiave di verifica della carta associata alias/BIN_12345_CVK al BIN 12345. Per offrire maggiore flessibilità, prendi in considerazione la creazione di tag come bin=12345, use_case=acquiring, country=us, partner=foo. Gli alias e i tag possono essere utilizzati anche per limitare l'accesso, ad esempio applicando i controlli di accesso tra i casi d'uso di emissione e quelli di acquisizione.
- Pratica l'accesso con privilegi minimi: l'IAM può essere utilizzato per limitare l'accesso di produzione ai sistemi anziché ai singoli utenti, ad esempio vietando ai singoli utenti di creare chiavi o eseguire operazioni crittografiche. L'IAM può essere utilizzato anche per limitare l'accesso sia ai comandi che alle chiavi che potrebbero non essere applicabili al caso d'uso, ad esempio per limitare la capacità di generare o convalidare i pin per un acquirente. Un altro modo per utilizzare l'accesso con privilegi meno privilegiati consiste nel limitare le operazioni sensibili (come l'importazione di chiavi) a specifici account di servizio. Per esempi, consulta AWS Esempi di politiche basate sull'identità della crittografia dei pagamenti.

Consulta anche

- Gestione delle identità e degli accessi per AWS Crittografia dei pagamenti
- Best practice per la sicurezza in IAM nella Guida per l'utente di IAM

Convalida della conformità per la crittografia AWS dei pagamenti

Come per altri AWS servizi, i clienti richiedono una chiara comprensione del modello di responsabilità condivisa per la sicurezza e la conformità. Trattandosi di un servizio che supporta specificamente i pagamenti, la conformità agli standard PCI applicabili è particolarmente importante per i clienti di AWS Payment Cryptography. AWS Le valutazioni PCI DSS e PCI 3DS includono la crittografia dei pagamenti. AWS Potrebbero esserci riferimenti al servizio nelle Guide sulla responsabilità condivisa, disponibili su, per questi report. AWS Artifact Le valutazioni P2PE (PIN Security and Point-to-Point Encryption) PCI sono specifiche per la crittografia dei pagamenti. AWS

Questa sezione fornisce informazioni sullo stato e l'ambito della conformità del servizio e informazioni utili per pianificare le valutazioni PCI PIN Security e PCI P2PE delle applicazioni.

Argomenti

- Conformità del servizio
- Pianificazione della conformità ai PIN
- Utilizzo del componente di decrittografia dei AWS pagamenti nelle soluzioni P2PE

Conformità del servizio

I revisori esterni valutano la sicurezza e la conformità della crittografia dei AWS pagamenti nell'ambito di diversi programmi di AWS conformità. Questi includono SOC, PCI e altri.

AWS La crittografia dei pagamenti è stata valutata per diversi standard PCI oltre a PCI DSS e PCI 3DS. Questi includono la crittografia PCI PIN Security (PCI PIN) e la crittografia PCI (P2PE). Point-to-Point Consulta le attestazioni e le AWS Artifact guide di conformità disponibili.

Per un elenco di AWS servizi nell'ambito di programmi di conformità specifici, consulta Servizi AWS nell'ambito del programma di conformità . Per informazioni generali, consulta Programmi per la conformità di AWS.

Puoi scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

La tua responsabilità in materia di conformità quando utilizzi la crittografia dei AWS pagamenti è determinata dalla sensibilità dei tuoi dati, dagli obiettivi di conformità della tua azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide rapide su sicurezza e conformità Guide introduttive implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla sicurezza e sulla conformità. AWS
- AWS Risorse per la conformità Risorse per AWS : questa raccolta di cartelle di lavoro e guide può essere valida per il settore e la località in cui operi.
- Evaluating Resources with Rules nella AWS Config Developer Guide:AWS Config valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida del settore e alle normative.
- AWS Security Hub CSPM—Questo AWS servizio offre una visione completa dello stato di sicurezza dell'utente, AWS che consente di verificare la conformità agli standard e alle best practice del settore della sicurezza.

Pianificazione della conformità ai PIN

Questa guida descrive la documentazione e le prove necessarie per preparare una valutazione del PIN PCI dell'applicazione di elaborazione del PIN che utilizza la crittografia dei AWS pagamenti.

Come per altri Servizi AWS standard di conformità, è responsabilità dell'utente utilizzare il servizio in modo sicuro, configurando il controllo degli accessi e utilizzando i parametri di sicurezza in linea con i requisiti del PIN PCI. Questa guida illustrerà tali configurazioni laddove appropriate per soddisfare un requisito.

Argomenti

- Argomenti comuni
- Ambito di valutazione
- Operazioni di elaborazione delle transazioni

Argomenti comuni

La migrazione delle applicazioni dalla connessione a HSM a un servizio gestito come AWS Payment Cryptography solleva problemi e concetti comuni per i clienti e i loro valutatori. Questa sezione fornisce informazioni per chiarire in che modo l'uso sicuro del servizio affronta queste situazioni.

Argomenti

- Responsabilità condivisa
- Configurazione HSM minima
- Scambio di chiavi tra cliente e APC

Responsabilità condivisa

I clienti che si sono assunti la completa responsabilità della sicurezza e della conformità delle applicazioni ristruttureranno la conformità per sfruttare la gestione delle chiavi, i controlli di sicurezza e le funzionalità HSM gestite di AWS Payment Cryptography («il servizio»). Ciò sposterà completamente alcuni requisiti AWS, come attestato dalle valutazioni di terze parti di AWS Payment Cryptography. Alcuni requisiti verranno condivisi tra l'applicazione del cliente e il servizio. Un'applicazione è responsabile di:

- Fornire informazioni accurate al servizio
- Utilizzo dei controlli di sicurezza in base alle raccomandazioni del servizio e ai requisiti di sicurezza PCI PIN
- Implementazione dei controlli di sicurezza richiesti utilizzando gli strumenti forniti dal servizio

I clienti e i relativi valutatori utilizzeranno guide di implementazione e responsabilità condivise pubblicate con attestati di conformità AWS Artifact per implementare i controlli e il monitoraggio del controllo, quindi pianificare e completare le valutazioni.

Configurazione HSM minima

Lo standard di sicurezza dei dati PCI, lo standard fondamentale per altri standard PCI, richiede che tutti i sistemi siano configurati con le funzionalità minime necessarie per il loro funzionamento. PCI PIN, P2PE e altri standard di soluzione applicano questo requisito alla soluzione. HSMs HSMs deve abilitare solo le funzioni necessarie per la soluzione.

AWS i servizi devono essere trattati come sistemi e configurati per le funzionalità minime richieste. Payment Card Industry Data Security Standard (PCI DSS) v4.0 su AWS consiglia di utilizzare IAM per configurare funzionalità minime per ogni servizio AWS utilizzato dalla soluzione. Questo vale anche per la crittografia dei pagamenti AWS . Le policy IAM consentono autorizzazioni granulari per limitare le funzioni crittografiche solo ai componenti dell'applicazione che si basano su di esse.

Scambio di chiavi tra cliente e APC

PIN PIN I requisiti di sicurezza 8-4 e 15-2 richiedono che le chiavi pubbliche per lo scambio e il caricamento delle chiavi siano autenticate e protette dall'integrità. Per il caricamento remoto delle chiavi dei POI, descritto funzionalmente in ANSI/ASC X9 TR-34 e regolato dal PIN PCI Allegato A, le chiavi pubbliche vengono spesso trasmesse in certificati firmati da un'autorità di certificazione conforme all'Allegato A2. Per gli scambi tra organizzazioni, le chiavi pubbliche utilizzano altri meccanismi di autenticità e integrità.

Tutte le interazioni tra cliente e AWS avvengono tramite AWS APIs, che autentica reciprocamente ogni chiamata API e garantisce l'integrità delle chiamate e delle risposte tramite TLS. L'autenticazione dell'applicazione del cliente è gestita da AWS Identity and Access Management con meccanismi come Security Tokens e SigV4. Gli endpoint delle API AWS vengono autenticati dal cliente utilizzando l'autenticazione del server TLS, integrata in AWS. SDKs Quindi TLS assicura la riservatezza e l'integrità di tutti i dati trasmessi tra il cliente e ciascuna API AWS.

APC APIs GetParametersForImport e ImportKey implementano un trasferimento di chiavi dal cliente al servizio. Sebbene l'Autorità di certificazione (CA) fornita da non GetParametersForImport sia conforme all'allegato A2, è sicura e unica per l'account. Sebbene non sia possibile fare affidamento sulla conformità ai requisiti 8-4 e 15-2, questa CA fornisce la verifica dell'integrità della chiave importata. Puoi anche utilizzare la tua CA sfruttando l'API. GetCertificateSigningRequest

I meccanismi che forniscono l'autenticazione a chiave pubblica e la garanzia dell'integrità sono:

- Autenticazione fornita dall'autenticazione API AWS
- L'integrità della chiave è garantita dalla funzionalità MAC del certificato fornito da GetParametersForImport, anche se le informazioni sull'identità contenute nel certificato non sono attendibili. L'integrità della chiave è garantita anche dal MAC utilizzato da TLS che protegge la sessione tra il cliente e AWS.

I certificati e i blocchi chiave forniti da APC sono conformi all'allegato A1, che specifica i requisiti per i certificati e la protezione delle chiavi con metodi asimmetrici.

Ambito di valutazione

Il primo passo nella pianificazione di qualsiasi valutazione è documentarne l'ambito. Per quanto riguarda il PCI PIN, rientrano i sistemi e i processi che proteggono i PIN, compresa la protezione delle chiavi crittografiche e dei dispositivi che li proteggono: terminali di pagamento, detti anche punti di interazione (POI), HSM e altri dispositivi crittografici sicuri (SCD).

Non risponderemo ai requisiti per i quali l'utente si assume la piena responsabilità, poiché tali requisiti non rientrano nell'ambito del servizio. Ad esempio, configurazione e fornitura di terminali di pagamento. Consultate la AWS Payment Cryptography Shared Responsibility Guide for PCI PIN, disponibile su AWS Artifact

Argomenti

- Responsabilità condivisa
- High-Level Diagrammi di rete
- Tabella chiave
- Riferimenti ai documenti

Responsabilità condivisa

AWS Payment Cryptography è un'Encryption and Support Organization (ESO) e un PIN-Acquiring Third-Party Servicer (TPS), come definito dal Visa PIN Security Program ed elencato nel Visa Global Service Provider Registry, alla voce «Amazon Web Services, LLC». Ciò significa che Visa consente l'utilizzo del servizio dal PIN-Acquiring Third-Party VisaNet Processore (VNP), dal Processore VisaNet del PIN-Acquiring Cliente che agisce come fornitore di servizi e da altri fornitori di TPS ed ESO senza richiedere un'ulteriore valutazione da parte dei valutatori del PIN dei clienti (PCI Qualified PIN Assessors o PCI QPA).

Altri marchi di carte o fornitori di reti di pagamento possono fare affidamento sul Visa PIN Security Program o disporre di propri programmi. Contattaci Supporto AWS per domande sulla conformità del servizio per altri programmi di rete di pagamento.

AWS fornisce l'attestato di conformità PCI PIN Security (AOC) e la Guida alla responsabilità condivisa per la crittografia dei AWS pagamenti in. AWS Artifact L'uso di fornitori di servizi nell'elaborazione del PIN è comune da molti anni, tuttavia lo standard di sicurezza PCI PIN, fino alla versione 3.1, non riguarda la gestione dei provider di servizi di terze parti. Nemmeno il Visa PIN Security Program. Customer QPA ha seguito il modello stabilito con il PCI DSS, l'AOC e la Shared Responsibility Guide, secondo cui la AWS«conformità» ha superato il test per i requisiti applicabili.

High-Level Diagrammi di rete

Il modello di segnalazione PCI PIN richiede: «Per le entità impegnate nell'elaborazione di transazioni basate sul PIN, fornite uno schema di rete che descriva i flussi di transazione basati sul PIN con l'utilizzo del tipo di chiave associato. Inoltre, i KIF e le entità impegnate nella distribuzione remota delle chiavi utilizzando tecniche asimmetriche dovrebbero fornire flussi di materiale chiave»

AWS Payment Cryptography ha riportato la struttura interna del servizio per la nostra valutazione del PIN PCI. I diagrammi illustreranno la chiamata alle API di servizio per l'elaborazione del PIN.

Esempio di diagramma di rete di alto livello per applicazioni PIN che utilizzano la crittografia dei pagamenti: AWS

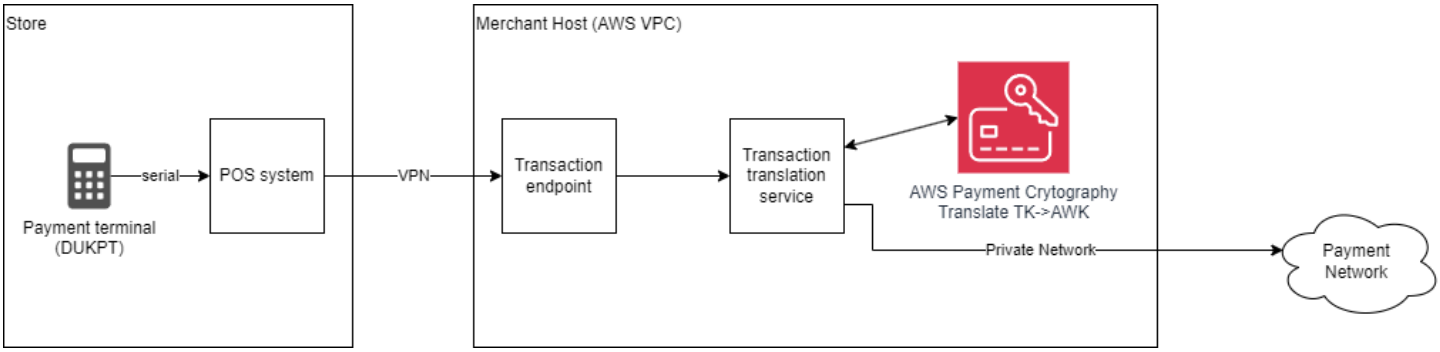


Tabella chiave

Il rapporto richiede che siano elencate tutte le chiavi che proteggono i PIN, direttamente o indirettamente. Tutte le chiavi esistenti nel servizio possono essere elencate con l'ListKeysAPI.

Assicurati di fornire l'elenco delle chiavi per tutte le regioni e gli account che possiedono le chiavi per la tua applicazione.

Riferimenti ai documenti

La documentazione e i consigli del fornitore per l'uso sicuro della crittografia dei AWS pagamenti sono disponibili nella Guida per l'utente e nel riferimento API. Questi sono collegati, se del caso, in questa guida.

Operazioni di elaborazione delle transazioni

I requisiti PCI PIN sono organizzati in Obiettivi di controllo. Ogni obiettivo di controllo raggruppa i requisiti per garantire un aspetto della sicurezza dei PIN.

Argomenti

- Obiettivo di controllo 1: I PIN utilizzati nelle transazioni regolate da questi requisiti vengono elaborati utilizzando apparecchiature e metodologie che ne garantiscono la sicurezza.
- Obiettivo di controllo 2: Le chiavi crittografiche utilizzate per il PIN encryption/decryption e la relativa gestione delle chiavi vengono create utilizzando processi che assicurano che non sia possibile prevedere alcuna chiave o determinare che determinate chiavi sono più probabili di altre chiavi.

- Obiettivo di controllo 3: le chiavi vengono convogliate o trasmesse in modo sicuro.
- Obiettivo di controllo 4: Key-loading gli HSM e PIN-acceptance i dispositivi POI vengono gestiti in modo sicuro.
- Obiettivo di controllo 5: Le chiavi vengono utilizzate in modo da prevenire o rilevare il loro utilizzo non autorizzato.
- Obiettivo di controllo 6: Le chiavi vengono amministrate in modo sicuro.
- Obiettivo di controllo 7: le apparecchiature utilizzate per elaborare PIN e chiavi sono gestite in modo sicuro.

Obiettivo di controllo 1: I PIN utilizzati nelle transazioni regolate da questi requisiti vengono elaborati utilizzando apparecchiature e metodologie che ne garantiscono la sicurezza.

Requisito 1: gli HSM utilizzati dalla crittografia dei AWS pagamenti sono stati valutati nell'ambito della nostra valutazione del PIN PCI. Per i clienti che utilizzano il servizio, i requisiti 1-3 e 1-4 sono «in vigore» rispetto all'HSM gestito dal servizio. I risultati relativi a HSM confermeranno che i test sono stati confermati dal QPA. AWS È possibile fare riferimento all'attestato di conformità PIN. AWS Artifact Gli altri SCD, come i POI, presenti nella soluzione in uso dovranno essere inventariati e referenziati.

Requisito 2: la documentazione delle procedure deve specificare in che modo i PIN dei titolari di carta sono protetti per quanto riguarda la divulgazione al personale, i protocolli di traduzione dei PIN implementati e la protezione durante l'elaborazione in linea e offline. Inoltre, la documentazione deve contenere un riepilogo dei metodi di gestione delle chiavi crittografiche utilizzati all'interno di ciascuna zona.

Requisito 3: I POI devono essere configurati per la crittografia e la trasmissione sicure del PIN. AWS Payment Cryptography supporta solo le traduzioni dei blocchi PIN specificate nel Requisito 3-3.

Requisito 4: l'applicazione non deve memorizzare blocchi PIN. I blocchi PIN, anche criptati, non devono essere conservati nei diari o nei registri delle transazioni. Il servizio non memorizza i blocchi PIN e la valutazione del PIN verifica che non siano presenti nei registri.

Si noti che lo standard di sicurezza PCI PIN si applica all'acquisizione «della gestione, dell'elaborazione e della trasmissione sicure dei dati del numero di identificazione personale (PIN) durante l'elaborazione delle transazioni online e offline con carte di pagamento presso bancomat e terminali POS», come indicato nello standard. Tuttavia, lo standard viene spesso utilizzato per

valutare la gestione delle chiavi crittografiche per i pagamenti che non rientrano nell'ambito previsto. Ciò può includere casi d'uso da parte dell'emittente in cui vengono archiviati i PIN. Le eccezioni ai requisiti per questi casi devono essere concordate con i destinatari della valutazione.

Obiettivo di controllo 2: Le chiavi crittografiche utilizzate per il PIN encryption/decryption e la relativa gestione delle chiavi vengono create utilizzando processi che assicurano che non sia possibile prevedere alcuna chiave o determinare che determinate chiavi sono più probabili di altre chiavi.

Requisito 5: la generazione di chiavi mediante AWS Payment Cryptography è stata valutata nell'ambito della nostra valutazione PCI PIN. Questo può essere specificato nella colonna «Generato da» della tabella chiave.

Requisito 6: I controlli di sicurezza per le chiavi contenute nella crittografia dei AWS pagamenti sono stati valutati nell'ambito della valutazione del PIN PCI del servizio. Includi le descrizioni dei controlli di sicurezza relativi alla generazione delle chiavi all'interno dell'applicazione e con qualsiasi altro fornitore di servizi.

Requisito 7: È necessario disporre di una documentazione sulla politica di generazione delle chiavi che specifichi come vengono generate le chiavi e tutte le parti interessate devono esserne a conoscenza. procedures/policies Le procedure per la creazione di chiavi utilizzando l'API APC devono includere l'uso di ruoli con autorizzazioni e approvazioni per la creazione di chiavi per l'esecuzione di script o altro codice che crea chiavi. AWS CloudTrail i log contengono tutti CreateKeygli eventi con data e ora, ARN chiave e ID utente. I numeri di serie e i registri HSM per l'accesso ai supporti fisici sono stati valutati nell'ambito della valutazione del PIN del servizio.

Obiettivo di controllo 3: le chiavi vengono convogliate o trasmesse in modo sicuro.

Requisito 8: La trasmissione delle chiavi con la crittografia dei AWS pagamenti è stata valutata nell'ambito della nostra valutazione del PIN PCI. Dovrai documentare i meccanismi di protezione chiave per i trasferimenti prima dell'importazione e dopo l'esportazione da Payment Cryptography. AWS Il servizio fornisce valori di controllo chiave per tutte le chiavi per convalidare la corretta trasmissione.

Il requisito 8-4 richiede che le chiavi pubbliche vengano trasmesse in modo da proteggerne l'integrità e l'autenticità. La comunicazione tra l'applicazione e l'altra AWS è controllata dall'autenticazione dell'applicazione AWS, utilizzando AWS Identity and Access Management metodi, AWS l'autenticazione degli endpoint API all'applicazione tramite certificati del server TLS. Inoltre, le chiavi pubbliche esportate o importate in AWS Payment

Cryptography hanno certificati firmati da CA temporanei e specifici del cliente (vedi, e). GetPublicKeyCertificateGetParametersForImportGetParametersForExport Queste CA non possono essere utilizzate come unico metodo di autenticazione, in quanto non sono conformi allo standard PCI PIN Security Annex A2. Tuttavia, i certificati forniscono ancora la garanzia di integrità per le chiavi pubbliche e IAM fornisce l'autenticazione.

Quando si scambiano chiavi pubbliche con i partner commerciali utilizzando metodi asimmetrici, è necessario provvedere all'autenticazione dell'azienda tramite il canale di comunicazione, ad esempio utilizzando un sito Web sicuro per lo scambio di file.

Requisito 9: il servizio non utilizza né supporta direttamente componenti chiave in testo non crittografato.

Requisito 10: Il servizio impone la forza fondamentale relativa della protezione delle chiavi per il trasporto. L'utente è responsabile della trasmissione delle chiavi prima dell'importazione e dopo l'esportazione da AWS Payment Cryptography e dell'utilizzo di API e TR-31 parametri accurati per l'importazione, l'esportazione e la generazione delle chiavi. È necessario disporre di procedure documentate per descrivere i meccanismi di trasmissione delle chiavi e l'elenco delle chiavi crittografiche utilizzate per la trasmissione.

Requisito 11: La documentazione delle procedure deve specificare come vengono trasmesse le chiavi. Le procedure per la trasmissione delle chiavi mediante l'API AWS Payment Cryptography devono includere l'uso di ruoli con autorizzazioni e approvazioni di importazione ed esportazione delle chiavi per l'esecuzione di script o altro codice che crea le chiavi. AWS CloudTrail i registri contengono tutti gli eventi. ImportKeyExportKey

Obiettivo di controllo 4: Key-loading gli HSM e PIN-acceptance i dispositivi POI vengono gestiti in modo sicuro.

Requisito 12: L'utente è responsabile del caricamento delle chiavi dai componenti o dalle condivisioni. La gestione delle chiavi principali HSM è stata valutata nell'ambito della valutazione del PIN del servizio. AWS Payment Cryptography non carica le chiavi da singole azioni o componenti. Consulta la sezione Dettagli crittografici.

Requisiti 13 e 14: è necessario descrivere la protezione delle chiavi per i trasferimenti prima dell'importazione e dopo l'esportazione dal servizio.

Requisito 15: La crittografia dei AWS pagamenti fornisce valori di controllo chiave per tutte le chiavi del servizio e garantisce l'integrità per le chiavi pubbliche. L'applicazione è responsabile dell'utilizzo

di questi controlli per convalidare le chiavi dopo l'importazione o l'esportazione dal servizio. È necessario documentare le procedure per garantire l'esistenza di un meccanismo di convalida.

Il requisito 15-2 richiede che le chiavi pubbliche siano caricate in modo da proteggerne l'integrità e l'autenticità. ImportKey, insieme a GetParametersForImport, prevede la convalida dei certificati di firma forniti. Se i certificati forniti sono autofirmati, l'autenticazione deve essere fornita mediante un meccanismo separato, ad esempio lo scambio sicuro di file.

Requisito 16: la documentazione delle procedure deve specificare in che modo le chiavi vengono caricate nel servizio. Le procedure per l'importazione delle chiavi tramite l'API devono includere l'uso di ruoli con autorizzazioni e approvazioni di importazione delle chiavi per l'esecuzione di script o altro codice che carica le chiavi. AWS CloudTrail i log contengono tutti gli eventi. ImportKey È necessario includere i meccanismi di registrazione nella documentazione. Il servizio fornisce valori di controllo chiave per tutte le chiavi per convalidare il corretto caricamento delle chiavi.

Obiettivo di controllo 5: Le chiavi vengono utilizzate in modo da prevenire o rilevare il loro utilizzo non autorizzato.

Requisito 17: il servizio fornisce meccanismi, come tag e alias, per le chiavi che consentono il monitoraggio delle relazioni di condivisione delle chiavi. Inoltre, i valori di controllo delle chiavi devono essere conservati separatamente per dimostrare che i valori delle chiavi noti o predefiniti non vengono utilizzati quando le chiavi vengono condivise.

Requisito 18: il servizio fornisce controlli di integrità delle chiavi, tramite GetKeye ListKeys, tramite eventi di gestione delle chiavi AWS CloudTrail, che possono essere utilizzati per rilevare sostituzioni non autorizzate o monitorare la sincronizzazione delle chiavi tra le parti. Il servizio archivia le chiavi esclusivamente in blocchi chiave. L'utente è responsabile dell'archiviazione e dell'utilizzo delle chiavi prima dell'importazione e dopo l'esportazione da AWS Payment Cryptography.

È necessario disporre di procedure per un'indagine immediata in caso di discrepanza durante l'elaborazione di transazioni basate sul PIN o di eventi imprevisti di gestione delle chiavi.

Requisito 19: il servizio utilizza le chiavi esclusivamente nei blocchi chiave, nell' KeyUsageimposizione e in altri attributi chiave per tutte le operazioni. KeyModeOfUse Ciò include restrizioni sulle operazioni con chiavi private. È necessario utilizzare le chiavi pubbliche per un unico scopo: crittografia o verifica della firma digitale, ma non per entrambi. È necessario utilizzare account separati per la produzione e test/development i sistemi.

Requisito 20: l'utente si assume la responsabilità di questo requisito.

Obiettivo di controllo 6: Le chiavi vengono amministrate in modo sicuro.

Requisito 21: la memorizzazione e l'uso delle chiavi con la crittografia dei AWS pagamenti sono stati valutati nell'ambito della valutazione del PIN PCI del servizio. Per i requisiti di archiviazione relativi ai componenti chiave, è responsabilità dell'utente archiviarli come indicato ai punti 21-2 e 21-3. È necessario descrivere i principali meccanismi di protezione nella documentazione relativa alla policy prima dell'importazione e dopo l'esportazione dal servizio.

Requisito 22: le principali procedure di compromesso per la crittografia dei AWS pagamenti sono state valutate nell'ambito della valutazione del PIN PCI del servizio. Dovrai descrivere le principali procedure di rilevamento e risposta alle compromissioni, tra cui il monitoraggio e la risposta alle notifiche di. AWS

Requisito 23: La crittografia dei AWS pagamenti non supporta varianti o altri metodi di calcolo delle chiavi reversibili. Le chiavi principali APC o le chiavi da esse cifrate non sono mai disponibili per i clienti. L'uso del calcolo reversibile delle chiavi è stato valutato nell'ambito della valutazione del PIN PCI del servizio.

Requisito 24: pratiche di distruzione delle chiavi private e segrete interne La crittografia dei AWS pagamenti è stata valutata nell'ambito della valutazione del PIN PCI del servizio. È necessario descrivere la procedura di distruzione delle chiavi prima dell'importazione e dopo l'esportazione da APC. I requisiti di distruzione relativi ai componenti chiave (24-2.2 e 24-2.3) restano di vostra responsabilità.

Requisito 25: L'accesso alle chiavi segrete e private all'interno di AWS Payment Cryptography è stato valutato nell'ambito della valutazione del PIN PCI del servizio. Dovrai disporre di un processo e di una documentazione per i controlli di accesso alle chiavi prima dell'importazione e dopo l'esportazione da AWS Payment Cryptography.

Requisito 26: è necessario descrivere la registrazione per qualsiasi accesso a chiavi, componenti chiave o materiali correlati utilizzati al di fuori del servizio. I registri per tutte le attività di gestione delle chiavi eseguite dall'applicazione con il servizio sono disponibili tramite. AWS CloudTrail

Requisito 27: è necessario descrivere le procedure di backup per chiavi, componenti chiave o materiali correlati utilizzati al di fuori del servizio.

Requisito 28: le procedure per l'amministrazione di tutte le chiavi che utilizzano l'API devono includere l'uso di ruoli con autorizzazioni e approvazioni di amministrazione chiave per l'esecuzione di script o altro codice che gestisce le chiavi. AWS CloudTrail i log contengono tutti gli eventi di amministrazione chiave

Obiettivo di controllo 7: le apparecchiature utilizzate per elaborare PIN e chiavi sono gestite in modo sicuro.

Requisito 29: i requisiti di protezione fisica e logica per gli HSM sono soddisfatti mediante l'uso della crittografia dei AWS pagamenti.

Requisito 30: l'applicazione si assumerà la responsabilità di tutti i requisiti di protezione fisica e logica dei dispositivi POI.

Requisito 31: La protezione dei dispositivi crittografici sicuri (SCD) utilizzati da AWS Payment Cryptography è stata valutata nell'ambito della valutazione del PIN PCI del servizio. Dovrai dimostrare la protezione di tutti gli altri SCD utilizzati dall'applicazione.

Requisito 32: L'uso degli SCD utilizzati da AWS Payment Cryptography è stato valutato nell'ambito della valutazione del PIN PCI del servizio. Dovrai dimostrare il controllo dell'accesso e la protezione di tutti gli altri SCD utilizzati dall'applicazione.

Requisito 33: Dovrete descrivere le protezioni di tutte le PIN-processing apparecchiature sotto il vostro controllo.

Utilizzo del componente di decrittografia dei AWS pagamenti nelle soluzioni P2PE

Le soluzioni PCI P2PE possono utilizzare il componente di decrittografia della crittografia dei pagamenti.AWS Ciò è documentato nella Point-to-Point Crittografia PCI: requisiti di sicurezza e procedure di test, sezione Soluzioni P2PE e uso di fornitori di componenti and/or P2PE di terze parti: «Un fornitore di soluzioni (o un commerciante come fornitore di soluzioni) può esternalizzare determinate funzioni P2PE a fornitori di componenti P2PE elencati PCI e segnalare l'uso dei componenti P2PE elencati PCI nel proprio rapporto P2PE sulla convalida (P-ROV)», disponibile sul sito Web PCI.

Come per altri servizi AWS e standard di conformità, è tua responsabilità utilizzare il servizio in modo sicuro, configurando il controllo degli accessi e utilizzando parametri di sicurezza in linea con i requisiti PCI P2PE. La Guida per l'utente del componente di decrittografia P2PE di AWS Payment Cryptography, disponibile su AWS Artifact, contiene istruzioni dettagliate per l'integrazione della crittografia dei AWS pagamenti con la soluzione PCI P2PE e il rapporto annuale sui componenti di decrittografia, necessario per i report di conformità.

Gestione delle identità e degli accessi per AWS Crittografia dei pagamenti

AWS Identity and Access Management (IAM) è uno strumento Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare AWS le risorse Payment Cryptography. IAM è uno strumento Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Argomenti

- Destinatari
- Autenticazione con identità
- Gestione dell'accesso tramite policy
- In che modo AWS La crittografia dei pagamenti funziona con IAM
- AWS Esempi di politiche basate sull'identità della crittografia dei pagamenti
- Resource-based politiche per AWS Crittografia dei pagamenti
- Multi-party approvazione per la crittografia AWS dei pagamenti
- Risoluzione dei problemi AWS Identità e accesso alla crittografia dei pagamenti

Destinatari

Il modo in cui utilizzi AWS Identity and Access Management (IAM) varia in base al tuo ruolo:

- Utente del servizio: richiedi le autorizzazioni all'amministratore se non riesci ad accedere alle funzionalità (consulta Risoluzione dei problemi AWS Identità e accesso alla crittografia dei pagamenti)
- Amministratore del servizio: determina l'accesso degli utenti e invia le richieste di autorizzazione (consulta In che modo AWS La crittografia dei pagamenti funziona con IAM)
- Amministratore IAM: scrivi policy per gestire l'accesso (consulta AWS Esempi di politiche basate sull'identità della crittografia dei pagamenti)

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi autenticarti come utente IAM o assumendo un ruolo IAM. Utente root dell'account AWS

Puoi accedere come identità federata utilizzando credenziali provenienti da una fonte di identità come Centro identità AWS IAM (IAM Identity Center), autenticazione Single Sign-On o credenziali. Google/ Facebook Per ulteriori informazioni sull'accesso, consulta Come accedere all' Account AWS nella Guida per l'utente di Accedi ad AWS .

Per l'accesso programmatico, AWS fornisce un SDK e una CLI per firmare crittograficamente le richieste. Per ulteriori informazioni, consulta AWS Signature Version 4 per le richieste API nella Guida per l'utente di IAM.

Account AWS utente root

Quando si crea un Account AWS, si inizia con un'identità di accesso denominata utente Account AWS root che ha accesso completo a tutte Servizi AWS le risorse. Consigliamo vivamente di non utilizzare l'utente root per le attività quotidiane. Per le attività che richiedono le credenziali come utente root, consulta Attività che richiedono le credenziali dell'utente root nella Guida per l'utente di IAM.

Utenti e gruppi IAM

Un utente IAM è una identità che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ti consigliamo di utilizzare credenziali temporanee invece di utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee nella Guida per l'utente IAM.

Un gruppo IAM specifica una raccolta di utenti IAM e semplifica la gestione delle autorizzazioni per gestire gruppi di utenti di grandi dimensioni. Per ulteriori informazioni, consulta Casi d'uso per utenti IAM nella Guida per l'utente di IAM.

Ruoli IAM

Un ruolo IAM è un'identità con autorizzazioni specifiche che fornisce credenziali temporanee. Puoi assumere un ruolo passando da un ruolo utente a un ruolo IAM (console) o chiamando un'operazione AWS CLI o AWS API. Per ulteriori informazioni, consulta Metodi per assumere un ruolo nella Guida per l'utente di IAM.

I ruoli IAM sono utili per l'accesso degli utenti federati, le autorizzazioni utente IAM temporanee, l'accesso multi-account, l'accesso multi-servizio e le applicazioni in esecuzione su Amazon EC2. Per maggiori informazioni, consultare Accesso a risorse multi-account in IAM nella Guida per l'utente IAM.

Gestione dell'accesso tramite policy

Puoi controllare l'accesso AWS creando policy e associandole a AWS identità o risorse. Una policy definisce le autorizzazioni quando è associata a un'identità o a una risorsa. AWS valuta queste politiche quando un preside effettua una richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per maggiori informazioni sui documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente IAM.

Utilizzando le policy, gli amministratori specificano chi ha accesso a cosa definendo quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Un amministratore IAM crea le policy IAM e le aggiunge ai ruoli, che gli utenti possono quindi assumere. Le policy IAM definiscono le autorizzazioni indipendentemente dal metodo utilizzato per eseguirle.

Identity-based politiche

Identity-based le politiche sono documenti di policy sulle autorizzazioni JSON che alleghi a un'identità (utente, gruppo o ruolo). Tali policy controllano le operazioni autorizzate per l'identità, nonché le risorse e le condizioni in cui possono essere eseguite. Per informazioni su come creare una policy basata su identità, consultare Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente nella Guida per l'utente IAM.

Identity-based le politiche possono essere politiche in linea (incorporate direttamente in una singola identità) o politiche gestite (politiche autonome collegate a più identità). Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scegliere tra policy gestite e policy in linea nella Guida per l'utente di IAM.

Resource-based politiche

Resource-based le politiche sono documenti di policy JSON allegati a una risorsa. Gli esempi includono le policy di trust dei ruoli IAM e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare

l'accesso a una risorsa specifica. In una policy basata sulle risorse è obbligatorio specificare un'entità principale.

Resource-based le politiche sono politiche in linea che si trovano in quel servizio. Non è possibile utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Liste di controllo degli accessi (ACL)

Le liste di controllo degli accessi (ACL) controllano quali entità principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni per accedere a una risorsa. Le ACL sono simili alle policy basate su risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano gli ACL. AWS WAF Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi che possono impostare le autorizzazioni massime concesse da tipi di policy più comuni:

- Limiti delle autorizzazioni: imposta il numero massimo di autorizzazioni che una policy basata su identità ha la possibilità di concedere a un'entità IAM. Per ulteriori informazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente di IAM.
- Policy di controllo dei servizi (SCP): specifica il numero massimo di autorizzazioni per un'organizzazione o un'unità organizzativa (OU) in AWS Organizations. Per ulteriori informazioni, consultare Policy di controllo dei servizi nella Guida per l'utente di AWS Organizations .
- Policy di controllo delle risorse (RCP): imposta le autorizzazioni massime disponibili per le risorse degli account. Per ulteriori informazioni, consulta Policy di controllo delle risorse (RCP) nella Guida per l'utente di AWS Organizations .
- Policy di sessione: policy avanzate passate come parametro quando si crea una sessione temporanea per un ruolo o un utente federato. Per maggiori informazioni, consultare Policy di sessione nella Guida per l'utente IAM.

Più tipi di policy

Quando a una richiesta si applicano più tipi di policy, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire o meno una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella IAM User Guide.

In che modo AWS La crittografia dei pagamenti funziona con IAM

Prima di utilizzare IAM per gestire l'accesso alla crittografia dei AWS pagamenti, è necessario comprendere quali funzionalità IAM sono disponibili per l'uso con AWS Payment Cryptography. Per avere una visione di alto livello di come AWS Payment Cryptography e altri AWS servizi funzionano con IAM, consulta AWS Services That Work with IAM nella IAM User Guide.

Argomenti

- AWS Politiche di crittografia dei pagamenti Identity-based
- Autorizzazione basata su AWS Tag di crittografia dei pagamenti

AWS Politiche di crittografia dei pagamenti Identity-based

Con le policy basate sull'identità IAM, puoi specificare azioni e risorse consentite o negate, nonché le condizioni in base alle quali le azioni sono consentite o negate. AWS La crittografia dei pagamenti supporta azioni, risorse e chiavi di condizione specifiche. Per informazioni su tutti gli elementi utilizzati in una policy JSON, consulta Documentazione di riferimento degli elementi delle policy JSON IAM nella Guida per l'utente IAM.

Azioni

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale entità principale può eseguire operazioni su quali risorse e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso in una policy. Includere le operazioni in una policy per concedere le autorizzazioni di eseguire l'operazione associata.

Le azioni politiche in AWS Payment Cryptography utilizzano il seguente prefisso prima dell'azione:. `payment-cryptography:` Ad esempio, per concedere a qualcuno il permesso di eseguire un'operazione dell'`VerifyCardData`API AWS Payment Cryptography, includi l'`payment-cryptography:VerifyCardData`azione nella sua politica. Le istruzioni della policy devono

includere un elemento `Action` o `NotAction`. AWS Payment Cryptography definisce una propria serie di azioni che descrivono le attività che è possibile eseguire con questo servizio.

Per specificare più azioni in una sola istruzione, separa ciascuna di esse con una virgola come mostrato di seguito:

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

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `List` (come `ListKeys` e`ListAliases`), includi l'azione seguente:

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

Per visualizzare un elenco di azioni di crittografia dei AWS pagamenti, consulta Azioni definite dalla crittografia dei AWS pagamenti nella Guida per l'utente IAM.

Resources

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale entità principale può eseguire operazioni su quali risorse e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'operazione. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). Per le azioni che non supportano le autorizzazioni a livello di risorsa, si utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

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

La risorsa chiave di crittografia dei pagamenti ha il seguente ARN:

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

Per ulteriori informazioni sul formato degli ARN, consulta Amazon Resource Names (ARNs) e AWS Service Namespaces.

Ad esempio, per specificare l'istanza `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` nell'istruzione, utilizza il seguente ARN:

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

Per specificare tutte le chiavi che appartengono a un account specifico, usa il carattere jolly (*):

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

Alcune azioni AWS di crittografia dei pagamenti, come quelle per la creazione di chiavi, non possono essere eseguite su una risorsa specifica. In questi casi, è necessario utilizzare il carattere jolly (*).

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

Per specificare più risorse in una singola istruzione, utilizzate una virgola come illustrato di seguito:

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

Esempi

Per visualizzare esempi di politiche basate sull'identità della crittografia dei AWS pagamenti, consulta. AWS Esempi di politiche basate sull'identità della crittografia dei pagamenti

Autorizzazione basata su AWS Tag di crittografia dei pagamenti

Puoi allegare tag alle risorse di crittografia dei AWS pagamenti o passare i tag di una richiesta a AWS Payment Cryptography. Per controllare l'accesso basato su tag, fornire informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `payment-cryptography:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

AWS Esempi di politiche basate sull'identità della crittografia dei pagamenti

Per impostazione predefinita, gli utenti e i ruoli IAM non sono autorizzati a creare o modificare risorse AWS Payment Cryptography. Inoltre, non possono eseguire attività utilizzando l' AWS API Console di gestione AWS AWS CLI, o. Un amministratore IAM deve creare policy IAM che concedono a utenti e ruoli l'autorizzazione per eseguire operazioni API specifiche sulle risorse specificate di cui

hanno bisogno. L'amministratore deve quindi allegare queste policy a utenti o IAM che richiedono tali autorizzazioni.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy nella scheda JSON nella Guida per l'utente IAM.

Argomenti

- Best practice delle policy
- Utilizzo di AWS Console di crittografia dei pagamenti
- Consentire agli utenti di visualizzare le loro autorizzazioni
- Capacità di accedere a tutti gli aspetti di AWS Crittografia dei pagamenti
- Possibilità di chiamare le API utilizzando chiavi specifiche
- Capacità di negare specificamente una risorsa

Best practice delle policy

Identity-based le politiche determinano se qualcuno può creare, accedere o eliminare le risorse AWS di crittografia dei pagamenti nel tuo account. Queste operazioni possono comportare costi aggiuntivi per l' Account AWS. Quando si creano o modificano policy basate sull'identità, seguire queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le policy AWS gestite che concedono le autorizzazioni per molti casi d'uso comuni. Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per maggiori informazioni, consulta Policy gestite da AWS o Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente di IAM.
- Applicazione delle autorizzazioni con privilegio minimo - Quando si impostano le autorizzazioni con le policy IAM, concedere solo le autorizzazioni richieste per eseguire un'attività. È possibile farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegio minimo. Per maggiori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente di IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso - Per limitare l'accesso ad azioni e risorse è possibile aggiungere una condizione alle policy. Ad esempio, è possibile

scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio CloudFormation. Per maggiori informazioni, consultare la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.

- Utilizzo dello strumento di analisi degli accessi IAM per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali - Lo strumento di analisi degli accessi IAM convalida le policy nuove ed esistenti in modo che aderiscano al linguaggio (JSON) della policy IAM e alle best practice di IAM. Lo strumento di analisi degli accessi IAM offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per maggiori informazioni, consultare Convalida delle policy per il Sistema di analisi degli accessi IAM nella Guida per l'utente di IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungere le condizioni MFA alle policy. Per maggiori informazioni, consultare Protezione dell'accesso API con MFA nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Utilizzo di AWS Console di crittografia dei pagamenti

Per accedere alla console AWS Payment Cryptography, devi disporre di un set minimo di autorizzazioni. Queste autorizzazioni devono consentirti di elencare e visualizzare i dettagli sulle risorse di crittografia dei AWS pagamenti presenti nel tuo account. AWS Se crei una policy basata su identità più restrittiva rispetto alle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per le entità (utenti e ruoli IAM) associate a tale policy.

Per garantire che tali entità possano ancora utilizzare la console AWS di crittografia dei pagamenti, allega anche la seguente politica AWS gestita alle entità. Per ulteriori informazioni, consulta Aggiunta di autorizzazioni a un utente nella Guida per l'utente IAM.

Non è necessario consentire autorizzazioni minime di console per gli utenti che effettuano chiamate solo verso AWS CLI o l' AWS API. Al contrario, è possibile accedere solo alle operazioni che soddisfano l'operazione API che stai cercando di eseguire.

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono collegate alla relativa identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando l'API o a livello di codice. 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": "*"
        }
    ]
}
```

Capacità di accedere a tutti gli aspetti di AWS Crittografia dei pagamenti

> **Warning**
>
> Questo esempio fornisce autorizzazioni ampie e non è consigliato. Considerate invece i modelli di accesso meno privilegiati.

In questo esempio, vuoi concedere a un utente IAM del tuo AWS account l'accesso a tutte le tue chiavi di crittografia dei AWS pagamenti e la possibilità di chiamare tutte le API di crittografia dei AWS pagamenti, comprese entrambe le operazioni. `ControlPlane` `DataPlane`

JSON

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

Possibilità di chiamare le API utilizzando chiavi specifiche

In questo esempio, vuoi concedere a un utente IAM del tuo AWS account l'accesso a una delle tue chiavi di crittografia dei AWS pagamenti, quindi utilizzare questa risorsa in due API `arn:aws:payment-cryptography:us-east-2:111122223333:key/kwapwa6qaifllw2h` e. `GenerateCardValidationData` `VerifyCardValidationData` Al contrario, l'utente IAM non avrà accesso all'uso di questa chiave per altre operazioni come o `DeleteKey` `ExportKey`

Le risorse possono essere chiavi con prefisso `key` o alias con prefisso. `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"
               ]
            }
      ]
   }
```

Capacità di negare specificamente una risorsa

> **Warning**
>
> Valuta attentamente le implicazioni della concessione dell'accesso con caratteri jolly. Considerate invece un modello con privilegi minimi.

In questo esempio, desideri consentire a un utente IAM del tuo AWS account di accedere a una qualsiasi delle tue chiavi di crittografia dei AWS pagamenti, ma desideri negare le autorizzazioni a una chiave specifica. L'utente avrà accesso a `VerifyCardData` e `GenerateCardData` con tutte le chiavi ad eccezione di quella specificata nella dichiarazione di negazione.

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"
              ]
          }
      ]
  }
```

Resource-based politiche per AWS Crittografia dei pagamenti

Resource-based le politiche sono documenti di policy JSON che alleghi a una risorsa, come una chiave di crittografia dei pagamenti. AWS In una politica basata sulle risorse, si specifica chi può accedere alla chiave e le azioni che può eseguire su di essa. Le policy basate su risorse possono essere utilizzate per:

- Concedi l'accesso a una singola chiave a più utenti e ruoli.
- Concedi l'accesso a utenti o ruoli in altri AWS account.

Argomenti

- Considerazioni
- Gestione delle politiche basate sulle risorse
- Resource-based esempi di politiche

Quando colleghi una policy basata sulle risorse a una chiave di crittografia dei AWS AWS pagamenti, Payment Cryptography utilizza la logica di valutazione delle policy IAM per determinare se una determinata entità è autorizzata a eseguire l'azione richiesta. Per abilitare l'accesso tra più account, puoi specificare un intero account o entità IAM in un altro account come principale in una policy basata sulle risorse. Cross-account l'accesso richiede due politiche:

1. Resource-based policy (account del proprietario della chiave): il proprietario della chiave consente di `PutResourcePolicy` concedere l'accesso all'account del chiamante o al responsabile IAM.
2. Identity-based policy (account del chiamante): l'amministratore IAM del chiamante deve inoltre consentire l'azione AWS Payment Cryptography (ad esempio,`payment-cryptography:EncryptData`) nella policy IAM del chiamante.

Entrambe le politiche devono consentire l'azione. Se manca una delle due, la richiesta tra più account viene rifiutata con`AccessDeniedException`.

Se una politica basata sulle risorse consente l'accesso a un principale nello stesso account, non è richiesta alcuna politica aggiuntiva basata sull'identità. Per ulteriori informazioni, consulta In che modo i ruoli IAM differiscono dalle Resource-based politiche nella Guida per l'utente IAM.

> ⚠ Operazioni del piano di controllo delle politiche delle risorse
>
> Resource-based i criteri non si applicano alle operazioni del piano di controllo delle politiche delle risorse come `PutResourcePolicy``GetResourcePolicy`, e `DeleteResourcePolicy`. Ciò impedisce potenziali scenari di blocco in cui una politica delle risorse potrebbe negare la possibilità di modificare o rimuovere la politica stessa. L'accesso a queste operazioni del piano di controllo è regolato esclusivamente da policy basate sull'identità IAM.

Considerazioni

Tieni presente quanto segue quando utilizzi politiche basate sulle risorse con Payment Cryptography. AWS

- AWS La crittografia dei pagamenti non impone automaticamente l'accesso pubblico alle chiavi. Non è possibile creare una politica basata sulle risorse che garantisca l'accesso a destinatari anonimi o pubblici. Tutti gli accessi alle chiavi AWS di crittografia dei pagamenti richiedono AWS numeri principali autenticati e l'accesso pubblico è sempre bloccato.

- Resource-based le politiche vengono applicate per chiave. A ogni chiave AWS di crittografia dei pagamenti può essere associata al massimo una politica basata sulle risorse.
- Resource-based le politiche non si applicano agli alias. Quando si fa riferimento a una chiave tramite il relativo alias, viene valutata la politica delle risorse associata alla chiave sottostante.
- Resource-based le politiche non si applicano alle chiavi Replica Region di sola lettura create utilizzando Multi-Region la replica delle chiavi in questo momento. Le politiche relative alle risorse possono essere allegate solo alla chiave della regione primaria.
- L'`Resource`elemento di una policy basata sulle risorse deve essere `"*"` o corrispondere esattamente all'ARN della chiave a cui è associata la policy. L'utilizzo `"*"` è consigliato perché consente di riutilizzare lo stesso documento di policy su più chiavi.
- Le API di gestione delle politiche delle risorse (`PutResourcePolicy`GetResourcePolicy, e`DeleteResourcePolicy`) sono limitate a chi possiede Account AWS la chiave. Solo i responsabili all'interno dell'account del proprietario della chiave possono gestire le politiche relative alle risorse.

Gestione delle politiche basate sulle risorse

Puoi gestire le politiche basate sulle risorse per le chiavi di crittografia dei AWS pagamenti utilizzando l'API or. AWS CLI AWS Per utilizzare questo comando, sostituisci il comando *`italicized placeholder text`* nell'esempio con le tue informazioni.

Allega una politica basata sulle risorse

Utilizza l'azione `PutResourcePolicy`API o il comando put-resource-policyCLI per allegare una policy basata sulle risorse a una chiave. Se una policy esiste già, il comando la sostituisce.

L'esempio seguente collega una politica basata sulle risorse da un file JSON a una chiave.

```
aws payment-cryptography put-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h \
    --policy file://policy.json
```

Recupera una politica basata sulle risorse

Utilizza l'azione `GetResourcePolicy`API o il comando get-resource-policyCLI per recuperare la policy basata sulle risorse allegata a una chiave.

L'esempio seguente recupera la policy basata sulle risorse allegata a una chiave.

```
aws payment-cryptography get-resource-policy \
    --resource-arn arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h
```

La risposta restituisce il documento relativo alla politica:

```
{
    "Policy": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
                },
                "Action": [
                    "payment-cryptography:EncryptData",
                    "payment-cryptography:DecryptData"
                ],
                "Resource": "*"
            }
        ]
    }
}
```

Eliminare una politica basata sulle risorse

Utilizza l'azione DeleteResourcePolicyAPI o il comando delete-resource-policyCLI per rimuovere la policy basata sulle risorse da una chiave.

L'esempio seguente elimina la policy basata sulle risorse allegata a una chiave.

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

Resource-based esempi di politiche

Concedi l'accesso a una chiave su più account

La seguente politica basata sulle risorse concede a un ruolo in un altro AWS account l'autorizzazione a utilizzare una chiave di crittografia dei AWS pagamenti per operazioni crittografiche.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/ExampleRole"
            },
            "Action": [
                "payment-cryptography:GenerateCardValidationData",
                "payment-cryptography:VerifyCardValidationData"
            ],
            "Resource": "*"
        }
    ]
}
```

Concedi autorizzazioni diverse a diversi account

La seguente politica basata sulle risorse mostra come concedere autorizzazioni diverse ai principali in account separati. In questo esempio, un 3DS Access Control Server (ACS) in un account può generare dati di convalida delle carte, mentre un servizio di autorizzazione dei pagamenti in un altro account può convalidare solo i crittogrammi 3DS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow3DSACSToGenerate",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/3dsAcsRole"
            },
            "Action": [
                "payment-cryptography:GenerateCardValidationData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowPaymentAuthToVerify",
            "Effect": "Allow",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::444455556666:role/PaymentAuthRole"
            },
            "Action": [
                "payment-cryptography:VerifyAuthRequestCryptogram"
            ],
            "Resource": "*"
        }
    ]
}
```

Multi-party approvazione per la crittografia AWS dei pagamenti

AWS La crittografia dei pagamenti si integra con Multi-party l'approvazione (MPA), una funzionalità di Amazon Web Services Organizations, per aiutare a proteggere le operazioni critiche attraverso un processo di approvazione distribuito. Con MPA, puoi richiedere che più persone fidate approvino specifiche operazioni di crittografia dei AWS pagamenti prima che vengano eseguite.

Argomenti

- Panoramica di
- Operazioni protette
- Prerequisiti
- Abilitazione e disabilitazione di MPA
- Nozioni di base
- Esempio: importazione di un certificato root con MPA abilitato
- AWS CloudTrail registrazione degli eventi MPA
- Verifica dello stato della richiesta e gestione degli errori

Panoramica di

Multi-party l'approvazione aggiunge un ulteriore livello di sicurezza alle operazioni sensibili di crittografia dei AWS pagamenti richiedendo l'approvazione di un gruppo di persone fidate prima che l'operazione possa procedere. Ciò aiuta a proteggere da modifiche non autorizzate se un singolo set di credenziali viene compromesso e impedisce a un singolo individuo di apportare una modifica unilateralmente.

Un team di approvazione è un gruppo di approvatori all'interno dell'organizzazione designati per approvare o rifiutare le richieste operative protette. Il processo di approvazione è interamente gestito

dagli approvatori dell'organizzazione. Nessun AWS personale è coinvolto nell'approvazione o nel rifiuto delle richieste.

Quando MPA è abilitato per un'operazione protetta, si verifica quanto segue:

1. Un richiedente avvia l'operazione protetta.
2. MPA crea una sessione di approvazione e invia una notifica ai membri del team di approvazione.
3. I membri del team di approvazione esaminano la richiesta e la approvano o la rifiutano tramite il portale MPA.
4. Una volta raggiunta la soglia minima di approvazioni richiesta, l'operazione procede. Se la richiesta viene rifiutata dal team di approvazione o il tempo consentito per la sessione scade prima del raggiungimento della soglia di approvazione, l'operazione non viene eseguita. In entrambi i casi, il richiedente deve inviare una nuova richiesta per riprovare l'operazione.

> **Note**
>
> Quando si importa un certificato CA principale con MPA abilitato, il `RequesterComment` parametro è obbligatorio. Questo commento è incluso nella notifica di approvazione inviata al team di approvazione e fornisce il contesto per la richiesta.

Operazioni protette

AWS Payment Cryptography supporta MPA per le seguenti operazioni:

- `ImportKey`con materiale `RootCertificatePublicKey` chiave: l'importazione di un certificato a chiave pubblica principale è un'operazione fondamentale perché i certificati radice stabiliscono il punto di riferimento per tutte le successive importazioni ed esportazioni di chiavi mediante lo scambio asimmetrico di chiavi, ad esempio. TR-34 Richiedere l'approvazione di più parti per questa operazione aiuta a garantire che nessun singolo individuo possa stabilire o modificare unilateralmente l'affidabilità delle chiavi di crittografia dei pagamenti. AWS

Prerequisiti

Prima di poter utilizzare MPA con la crittografia dei AWS pagamenti, è necessario completare i seguenti prerequisiti:

- Configura MPA nel tuo ambiente Amazon Web Services Organizations. Per istruzioni, consulta Cos'è Multi-party l'approvazione? nella Guida per Multi-party l'utente per l'approvazione.
- Crea almeno un team di approvazione con gli approvatori richiesti.
- Condividi il team di approvazione con Account AWS quello che contiene le tue chiavi AWS di crittografia dei pagamenti utilizzando. AWS Resource Access Manager
- L'account di gestione della tua organizzazione deve essere abilitato per Multi-party l'approvazione.

Abilitazione e disabilitazione di MPA

Dopo aver creato un team di approvazione, puoi abilitare MPA for AWS Payment Cryptography associando il team al tuo account. Puoi anche disabilitare MPA dissociando il team, sebbene la disassociazione richieda l'approvazione del team di approvazione attualmente associato.

Abilita MPA

Utilizza l'azione `AssociateMpaTeam` API o il comando associate-mpa-team CLI per associare un team di approvazione al tuo account AWS Payment Cryptography. Una volta associate, le operazioni protette richiedono l'approvazione del team prima di poter procedere.

```
aws payment-cryptography associate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

Disattiva MPA

Utilizza l'azione `DisassociateMpaTeam` API o il comando disassociate-mpa-team CLI per rimuovere l'associazione del team di approvazione. La dissociazione di un team è di per sé un'operazione protetta che richiede l'approvazione del team di approvazione attualmente associato.

```
aws payment-cryptography disassociate-mpa-team \
    --team-arn arn:aws:mpa:us-east-1:111122223333:team/my-approval-team
```

> **Important**
>
> La disabilitazione dell'MPA richiede l'approvazione del team di approvazione attualmente associato. Ciò garantisce che nessun singolo individuo possa rimuovere unilateralmente la protezione di approvazione multipartitica.

> **Note**
>
> Il `--requester-comment` parametro è facoltativo per e. associate-mpa-team disassociate-mpa-team

Nozioni di base

Per iniziare a usare MPA for AWS Payment Cryptography, consulta la Guida per Multi-party l'utente di approvazione per istruzioni di configurazione dettagliate, tra cui come creare team di approvazione, configurare le politiche di approvazione e gestire le sessioni di approvazione.

Esempio: importazione di un certificato root con MPA abilitato

Dopo che MPA è stato abilitato e un team di approvazione è stato associato all'`ImportKey`operazione per`RootCertificatePublicKey`, la richiesta di importazione richiede l'approvazione prima di poter procedere.

1. Un richiedente chiama `import-key` per importare un certificato a chiave pubblica principale. Per utilizzare questo comando, sostituite il comando contenuto *`italicized placeholder text`* nell'esempio con le vostre informazioni.

   ```
   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": "LS0tLS1CRUdJTi..."}}' \
       --requester-comment "Importing new root CA certificate for TR-34 key exchange
    with partner XYZ"
   ```

   La risposta restituisce una chiave `KeyState` impostata su`CREATE_IN_PROGRESS`, che indica che la richiesta è in attesa di approvazione. La risposta include `MpaStatus` anche dettagli sulla sessione di approvazione:

   ```
   {
       "Key": {
   ```

```
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_IN_PROGRESS",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-
east-1:111122223333:session/abc123def456",
            "Status": "PENDING",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

2. Poiché MPA è abilitato, la richiesta non viene completata immediatamente. AWS Payment Cryptography crea invece una sessione di approvazione e restituisce una risposta indicante che l'approvazione è in sospeso.
3. I membri del team di approvazione ricevono una notifica e esaminano la richiesta tramite il portale MPA. Una volta che il numero richiesto di approvatori ha approvato la richiesta, l'operazione di importazione procede e il certificato radice viene importato.

AWS CloudTrail registrazione degli eventi MPA

Quando MPA è abilitato, AWS Payment Cryptography registra gli eventi del servizio al termine di una sessione di approvazione AWS CloudTrail. Questi eventi registrano l'esito del processo di approvazione, incluso se la richiesta è stata approvata o meno. È possibile utilizzare questi registri per controllare l'attività MPA e tenere traccia dello stato delle operazioni protette.

MPA-related CloudTrail gli eventi includono i seguenti campi in: `serviceEventDetails`

- `keyArn`— L'ARN della chiave interessata dall'operazione.
- `operation`— L'operazione protetta richiesta.
- `mpaSessionArn`— L'ARN della sessione di approvazione dell'MPA.

- `sessionStatus`— L'esito della sessione di approvazione (`APPROVEDoFAILED`).

Richiesta approvata

L'esempio seguente mostra un CloudTrail evento relativo a una `ImportKey` richiesta approvata dal team MPA:

```
{
    "eventVersion": "1.11",
    "eventTime": "2026-04-28T18:49:51Z",
    "eventName": "ImportKey",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventType": "AwsServiceEvent",
    "eventCategory": "Management",
    "awsRegion": "us-east-1",
    "readOnly": false,
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "resources": [
        {
            "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/spa2dclzmsihlj4o",
            "accountId": "111122223333",
            "type": "AWS::PaymentCryptography::Key"
        }
    ],
    "serviceEventDetails": {
        "keyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/spa2dclzmsihlj4o",
        "operation": "ImportKey",
        "mpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/my-approval-
team/44c76e07-8937-4d7d-bb9a-a646322e2a1e",
        "sessionStatus": "APPROVED"
    }
}
```

Richiesta non riuscita

L'esempio seguente mostra un CloudTrail evento per una `ImportKey` richiesta che è stata rifiutata o scaduta:

```
{
    "eventVersion": "1.11",
    "eventTime": "2026-04-28T18:50:35Z",
    "eventName": "ImportKey",
    "eventSource": "payment-cryptography.amazonaws.com",
    "eventType": "AwsServiceEvent",
    "eventCategory": "Management",
    "awsRegion": "us-east-1",
    "readOnly": false,
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "payment-cryptography.amazonaws.com"
    },
    "resources": [
        {
            "ARN": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/qj46ku4qimypxdo7",
            "accountId": "111122223333",
            "type": "AWS::PaymentCryptography::Key"
        }
    ],
    "serviceEventDetails": {
        "keyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/qj46ku4qimypxdo7",
        "operation": "ImportKey",
        "mpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/my-approval-team/
b0ac1994-14e1-47a6-bf1a-0b6fc0b845f2",
        "sessionStatus": "FAILED"
    }
}
```

Per ulteriori informazioni AWS CloudTrail, consulta la Guida per l'AWS CloudTrail utente.

Verifica dello stato della richiesta e gestione degli errori

È possibile verificare lo stato di una richiesta MPA in sospeso chiamando. `GetKey` La risposta include il `MpaStatus` campo con i dettagli della sessione di approvazione corrente. Per utilizzare

questo comando, sostituisci il *italicized placeholder text* comando dell'esempio con le tue informazioni.

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

Mentre la richiesta è in attesa di approvazione, la risposta viene mostrata `KeyState` come `CREATE_IN_PROGRESS` e `MpaStatus.Status` come`PENDING`:

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_IN_PROGRESS",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "PENDING",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

Una volta che il numero richiesto di approvatori ha approvato la richiesta, `KeyState` passa alla `CREATE_COMPLETE` fase successiva. `MpaStatus.Status` `APPROVED` La chiave è ora pronta per l'uso:

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
```

```
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_COMPLETE",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "APPROVED",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00"
        }
    }
}
```

Se la richiesta viene rifiutata dal team di approvazione o la sessione scade prima che venga raggiunta la soglia di approvazione, la `KeyState` modifica `CREATE_FAILED` e le modifiche `MpaStatus.Status` apportate a`FAILED`:

```
{
    "Key": {
        "KeyArn": "arn:aws:payment-cryptography:us-
east-2:111122223333:key/kwapwa6qaifllw2h",
        "KeyAttributes": {
            "KeyUsage": "TR31_S0_ASYMMETRIC_KEY_FOR_DIGITAL_SIGNATURE",
            "KeyClass": "PUBLIC_KEY",
            "KeyAlgorithm": "RSA_4096"
        },
        "Enabled": true,
        "KeyState": "CREATE_FAILED",
        "KeyOrigin": "EXTERNAL",
        "CreateTimestamp": "2026-04-27T10:15:30.000000+00:00",
        "UsageStartTimestamp": "2026-04-27T10:15:29.926000+00:00",
        "MpaStatus": {
            "MpaSessionArn": "arn:aws:mpa:us-east-1:111122223333:session/abc123def456",
            "Status": "FAILED",
            "InitiationDate": "2026-04-27T10:15:30.000000+00:00",
            "StatusMessage": "Approval session expired or was denied"
        }
    }
}
```

Una chiave di `CREATE_FAILED` status non può essere utilizzata per operazioni crittografiche. Per riprovare l'importazione, è necessario inviare una nuova `ImportKey` richiesta, che creerà una nuova sessione di approvazione.

Risoluzione dei problemi AWS Identità e accesso alla crittografia dei pagamenti

Gli argomenti verranno aggiunti a questa sezione man mano che verranno identificati IAM-related problemi specifici della crittografia dei AWS pagamenti. Per contenuti generali sulla risoluzione dei problemi sugli argomenti IAM, consulta la sezione sulla risoluzione dei problemi della Guida per l'utente IAM.

Monitoraggio della crittografia AWS dei pagamenti

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni della crittografia dei AWS pagamenti e delle altre soluzioni AWS. AWS fornisce i seguenti strumenti di monitoraggio per monitorare la crittografia dei AWS pagamenti, segnalare quando qualcosa non va e intraprendere azioni automatiche quando necessario:

- Amazon CloudWatch monitora AWS le tue risorse e le applicazioni su cui esegui AWS in tempo reale. Puoi raccogliere i parametri e tenerne traccia, creare pannelli di controllo personalizzati e impostare allarmi per inviare una notifica o intraprendere azioni quando un parametro specificato raggiunge una determinata soglia. Ad esempio, puoi tenere CloudWatch traccia dell'utilizzo di alcuni APIs o avvisarti se ti stai avvicinando alle quote di crittografia dei AWS pagamenti. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.
- Amazon CloudWatch Logs ti consente di monitorare, archiviare e accedere ai tuoi file di registro da EC2 istanze Amazon e altre fonti. CloudTrail CloudWatch I log possono monitorare le informazioni nei file di registro e avvisarti quando vengono raggiunte determinate soglie. Puoi inoltre archiviare i dati del log in storage estremamente durevole. Per ulteriori informazioni, consulta la Amazon CloudWatch Logs User Guide.
- AWS CloudTrailacquisisce le chiamate API e gli eventi correlati effettuati da o per conto del tuo AWS account e invia i file di log a un bucket Amazon S3 da te specificato. Puoi identificare gli utenti e gli account chiamati AWS, l'endpoint chiamato, le risorse (chiavi) utilizzate, l'indirizzo IP di origine da cui sono state effettuate le chiamate e quando sono avvenute le chiamate. Per ulteriori informazioni, consulta la Guida per l'utente AWS CloudTrail.

Argomenti

- Registrazione delle chiamate API AWS di crittografia dei pagamenti tramite AWS CloudTrail

Registrazione delle chiamate API AWS di crittografia dei pagamenti tramite AWS CloudTrail

AWS La crittografia dei pagamenti è integrata con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o AWS servizio in AWS Payment Cryptography. CloudTrail acquisisce tutte le chiamate API per AWS Payment Cryptography come eventi. Le chiamate acquisite includono le chiamate dalla console di e le chiamate di codice alle

operazioni delle API . Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per AWS la crittografia dei pagamenti. Se non configuri un trail, puoi comunque visualizzare gli eventi di gestione più recenti (Control Plane) nella CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata a AWS Payment Cryptography, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la Guida per l'AWS CloudTrail utente.

Argomenti

- AWS Informazioni sulla crittografia dei pagamenti in CloudTrail
- Eventi del piano di controllo in CloudTrail
- Eventi relativi ai dati in CloudTrail
- Comprensione delle AWS voci dei file di registro di Payment Cryptography Control Plane
- Comprensione delle voci AWS del file di registro del piano dati della crittografia dei pagamenti

AWS Informazioni sulla crittografia dei pagamenti in CloudTrail

CloudTrail è abilitato sul tuo AWS account al momento della creazione dell'account. Quando si verifica un'attività in AWS Payment Cryptography, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi di AWS servizio nella cronologia degli eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS . Per ulteriori informazioni, consulta Visualizzazione degli eventi con la cronologia degli CloudTrail eventi.

Per una registrazione continua degli eventi del tuo AWS account, inclusi gli eventi per la crittografia dei AWS pagamenti, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando si crea un trail nella console, il trail sarà valido in tutte le Regioni AWS . Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail Per ulteriori informazioni, consulta gli argomenti seguenti:

- Panoramica della creazione di un percorso
- CloudTrail servizi e integrazioni supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni

- Ricezione di file di CloudTrail registro da più account

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali utente root o AWS Identity and Access Management (IAM).
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro AWS servizio.

Per ulteriori informazioni, consulta Elemento CloudTrail userIdentity.

Eventi del piano di controllo in CloudTrail

CloudTrail registra le operazioni AWS di crittografia dei pagamenti, come,,, e tutte le CreateKeyaltre ImportKeyoperazioni del DeleteKeypiano di ListKeyscontrollo TagResource.

Eventi relativi ai dati in CloudTrail

Gli eventi relativi ai dati forniscono informazioni sulle operazioni eseguite sulle risorse su o all'interno di una risorsa, ad esempio la crittografia di un payload o la traduzione di un pin. Gli eventi relativi ai dati sono attività ad alto volume che CloudTrail non vengono registrate per impostazione predefinita. È possibile abilitare la registrazione delle azioni dell'API per gli eventi del piano dati AWS di crittografia dei pagamenti utilizzando CloudTrail APIs la nostra console. Per ulteriori informazioni, consultare Registrazione di eventi di dati nella Guida per l'utente di AWS CloudTrail .

Con CloudTrail, è necessario utilizzare selettori di eventi avanzati per decidere quali attività dell'API AWS Payment Cryptography vengono registrate e registrate. Per registrare gli eventi del piano dati AWS di Payment Cryptography, è necessario includere il tipo di risorsa e. `AWS Payment Cryptography key` `AWS Payment Cryptography alias` Una volta impostato il tipo di risorsa, puoi affinare ulteriormente le tue preferenze di logging selezionando eventi di dati specifici da registrare, ad esempio utilizzando il filtro `eventName` per tenere traccia degli eventi `EncryptData`. Per ulteriori informazioni, consulta `AdvancedEventSelector` nella documentazione di riferimento dell'API AWS CloudTrail .

> **Note**
>
> Per sottoscrivere gli eventi relativi ai dati di AWS Payment Cryptography, è necessario utilizzare selettori di eventi avanzati. Ti consigliamo di iscriverti a eventi chiave e alias per assicurarti di ricevere tutti gli eventi.

AWS Eventi relativi ai dati di crittografia dei pagamenti:

- `DecryptData`
- `EncryptData`
- `GenerateCardValidationData`
- `GenerateMac`
- `GeneratePinData`
- `ReEncryptData`
- `TranslatePinData`
- `VerifyAuthRequestCryptogram`
- `VerifyCardValidationData`
- `VerifyMac`
- `VerifyPinData`

Per gli eventi di dati sono previsti costi aggiuntivi. Per ulteriori informazioni, consultare AWS CloudTrail Prezzi.

Comprensione delle AWS voci dei file di registro di Payment Cryptography Control Plane

Un trail è una configurazione che consente la distribuzione di eventi come file di log in un bucket Amazon S3 specificato dall'utente. CloudTrail i file di registro contengono una o più voci di registro. Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'azione richiesta, la data e l'ora dell'azione, i parametri della richiesta e così via. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche, quindi non vengono visualizzati in un ordine specifico.

L'esempio seguente mostra una voce di CloudTrail registro che illustra l'azione AWS Payment `CreateKey` Cryptography.

```
      {
    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={
    }
  }
}
```

L'esempio seguente mostra una voce di CloudTrail registro che dimostra che la crittografia dei AWS pagamenti consente la replica delle chiavi in più regioni.

```
{
    "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"
}
```

Comprensione delle voci AWS del file di registro del piano dati della crittografia dei pagamenti

Gli eventi del piano dati possono essere configurati opzionalmente e funzionare in modo simile ai log del piano di controllo, ma in genere sono volumi molto più elevati. Data la natura sensibile di alcuni input e output relativi alle operazioni del piano dati AWS di crittografia dei pagamenti, è possibile trovare alcuni campi con il messaggio «*** Dati sensibili redatti***». Non è configurabile e ha lo scopo di impedire la visualizzazione di dati sensibili nei registri o nei percorsi.

L'esempio seguente mostra una voce di CloudTrail registro che illustra l'azione Payment Cryptography. AWS `EncryptData`

```
      {
        "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": {
```

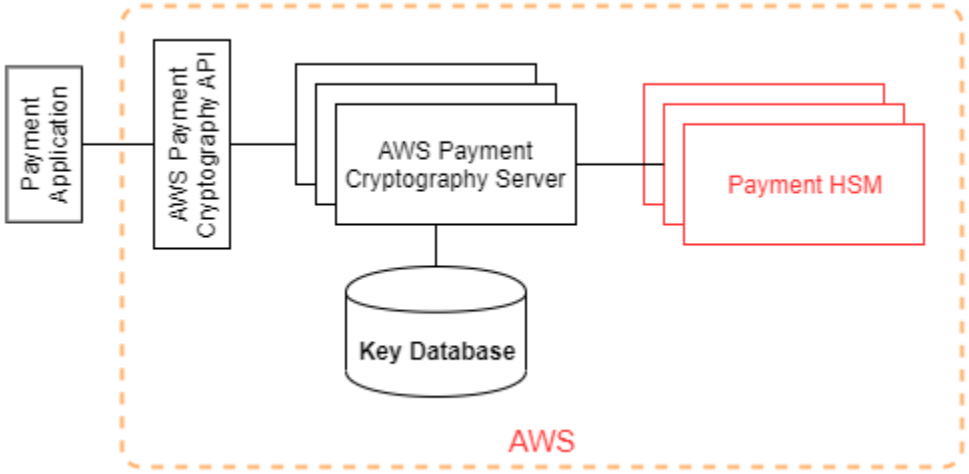
```
                            "type": "Role",
                            "principalId": "TESTXECZ5U9M4LGF2N6Y5",
                            "arn": "arn:aws:iam::111122223333:role/Admin",
                            "accountId": "111122223333",
                            "userName": "Admin"
                        },
                        "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"
                }
            }
        ]
    }
```

Dettagli crittografici

AWS Payment Cryptography fornisce un'interfaccia web per generare e gestire chiavi crittografiche per le transazioni di pagamento. AWS Payment Cryptography offre servizi standard di gestione delle chiavi e crittografia delle transazioni di pagamento e strumenti che è possibile utilizzare per la gestione e il controllo centralizzati. Questa documentazione fornisce una descrizione dettagliata delle operazioni crittografiche che è possibile utilizzare nella crittografia dei AWS pagamenti per aiutarvi a valutare le funzionalità offerte dal servizio.

AWS La crittografia dei pagamenti contiene più interfacce (inclusa un'API RESTful, tramite l'AWS CLI, l'SDK AWS e Console di gestione AWS) per richiedere le operazioni crittografiche di una flotta distribuita di moduli di sicurezza hardware PCI PTS. HSM-validated



AWS Payment Cryptography è un servizio a più livelli composto da host di crittografia dei AWS pagamenti rivolti al web e da un livello di HSM. Il raggruppamento di questi host a più livelli costituisce lo stack Payment Cryptography. AWS Tutte le richieste a AWS Payment Cryptography devono essere effettuate tramite il protocollo Transport Layer Security (TLS) e terminate su un host Payment Cryptography. AWS Gli host di servizi consentono il TLS solo con una suite di crittografia che offre una perfetta segretezza di inoltro. Il servizio autentica e autorizza le richieste utilizzando gli stessi meccanismi di credenziali e policy di IAM disponibili per tutte le altre operazioni API. AWS

AWS I server di crittografia dei pagamenti si connettono all'HSM sottostante tramite una rete privata non virtuale. Le connessioni tra i componenti del servizio e HSM sono protette con TLS reciproco (MTL) per l'autenticazione e la crittografia.

Argomenti

- Obiettivi di progettazione di
- Fondamenti
- Operazioni interne
- Operazioni con i clienti

Obiettivi di progettazione di

AWS La crittografia dei pagamenti è progettata per soddisfare i seguenti requisiti:

- Affidabile: l'uso delle chiavi è protetto da politiche di controllo degli accessi definite e gestite dall'utente. Non esiste alcun meccanismo per esportare chiavi di crittografia dei AWS pagamenti in testo non crittografato. La riservatezza delle chiavi di crittografia è fondamentale. Sono necessari più dipendenti Amazon con accesso specifico per ruolo ai controlli di accesso basati sul quorum per eseguire azioni amministrative su. HSMs Nessun dipendente Amazon ha accesso alle chiavi principali (o master) o ai backup HSM. Le chiavi principali non possono essere sincronizzate con chiavi HSMs che non fanno parte di una regione di crittografia dei AWS pagamenti. Tutte le altre chiavi sono protette dalle chiavi principali HSM. Pertanto, le chiavi AWS di crittografia dei pagamenti del cliente non sono utilizzabili al di fuori del servizio di crittografia dei AWS pagamenti che opera all'interno dell'account del cliente.
- Bassa latenza e velocità effettiva elevata: la crittografia dei AWS pagamenti fornisce operazioni crittografiche a livello di latenza e velocità effettiva adatte alla gestione delle chiavi crittografiche dei pagamenti e all'elaborazione delle transazioni di pagamento.
- Durabilità: la durabilità delle chiavi crittografiche è progettata per essere uguale a quella dei servizi di massima durabilità di AWS. Una singola chiave crittografica può essere condivisa con un terminale di pagamento, una chip card EMV o un altro dispositivo crittografico sicuro (SCD) in uso da molti anni.
- Regioni indipendenti: AWS fornisce regioni indipendenti per i clienti che devono limitare l'accesso ai dati in diverse regioni o devono rispettare i requisiti di residenza dei dati. L'utilizzo delle chiavi può essere isolato all'interno di una regione AWS.
- Fonte sicura di numeri casuali: poiché una crittografia avanzata dipende da una generazione di numeri casuali davvero imprevedibile, AWS Payment Cryptography fornisce una fonte di numeri casuali convalidata e di alta qualità. Tutta la generazione di chiavi per la crittografia dei AWS pagamenti utilizza un sistema HSM certificato PCI PTS HSM, che opera in modalità PCI.
- Audit: AWS Payment Cryptography registra l'uso e la gestione delle chiavi crittografiche nei log e nei CloudTrail log dei servizi disponibili tramite Amazon. CloudWatch Puoi utilizzare CloudTrail i log

per controllare l'uso delle tue chiavi crittografiche, incluso l'uso delle chiavi da parte degli account con cui hai condiviso le chiavi. AWS La crittografia dei pagamenti viene verificata da valutatori terzi in base agli standard PCI, al marchio delle carte e agli standard regionali di sicurezza dei pagamenti applicabili. Gli attestati e le guide sulla responsabilità condivisa sono disponibili su AWS Artifact.

- Elastic: AWS Payment Cryptography è scalabile orizzontalmente in base alle tue esigenze. Invece di prevedere e riservare la capacità HSM, Payment Cryptography fornisce la crittografia dei AWS pagamenti su richiesta. AWS Payment Cryptography si assume la responsabilità di mantenere la sicurezza e la conformità di HSM per fornire una capacità sufficiente a soddisfare i picchi di domanda dei clienti.

Fondamenti

Gli argomenti di questo capitolo descrivono le primitive crittografiche della crittografia dei AWS pagamenti e dove vengono utilizzate. Inoltre introducono gli elementi di base del servizio.

Argomenti

- Primitive di crittografia
- Entropia e generazione di numeri casuali
- Operazioni chiave simmetriche
- Operazioni chiave asimmetriche
- Archiviazione delle chiavi
- Importazione di chiavi tramite chiavi simmetriche
- Importazione di chiavi tramite chiavi asimmetriche
- Esportazione di chiavi
- Protocollo DUKPT (Derived Unique Key Per Transaction)
- Gerarchia delle chiavi

Primitive di crittografia

AWS La crittografia dei pagamenti utilizza algoritmi crittografici standard parametrizzabili in modo che le applicazioni possano implementare gli algoritmi necessari per il loro caso d'uso. L'insieme di algoritmi crittografici è definito dagli standard PCI, ANSI X9, EMVCo e ISO. Tutta la crittografia viene eseguita da moduli HSM PCI PTS HSM elencati negli standard in esecuzione in modalità PCI.

Entropia e generazione di numeri casuali

AWS La generazione di chiavi di crittografia dei pagamenti viene eseguita sugli HSM di crittografia dei pagamenti. AWS Gli HSM implementano un generatore di numeri casuali che soddisfa i requisiti PCI PTS HSM per tutti i tipi e i parametri di chiave supportati.

Operazioni chiave simmetriche

Sono supportati gli algoritmi a chiave simmetrica e i punti di forza chiave definiti in ANSI X9 TR 31, ANSI e PCI PIN X9.24 Annex C:

- Funzioni hash: algoritmi della famiglia SHA2 e SHA3 con dimensioni di output superiori a 2551. Fatta eccezione per la retrocompatibilità con i terminali POI PTS v3 pre-PCI.
- Crittografia e decrittografia: AES con dimensione della chiave maggiore o uguale a 128 bit o TDEA con dimensioni delle chiavi maggiori o uguali a 112 bit (2 o 3 chiavi).
- Codici di autenticazione dei messaggi (MAC) CMAC o GMAC con AES, nonché HMAC con una funzione hash approvata e una dimensione della chiave maggiore o uguale a 128.

AWS Payment Cryptography utilizza AES 256 per le chiavi principali HSM, le chiavi di protezione dei dati e le chiavi di sessione TLS.

Nota: alcune delle funzioni elencate vengono utilizzate internamente per supportare protocolli e strutture di dati standard. Consulta la documentazione dell'API per gli algoritmi supportati da azioni specifiche.

Operazioni chiave asimmetriche

Sono supportati gli algoritmi a chiave asimmetrica e i punti di forza chiave definiti in ANSI X9 TR 31, ANSI e PCI PIN Annex C: X9.24

- Schemi istituzionali fondamentali approvati, come descritti in NIST SP800-56A (accordo chiave), NIST (accordo ECC/FCC2-based chiave) e NIST SP800-56B (chiave). IFC-based SP800-38F AES-based encryption/wrapping

AWS Gli host di crittografia dei pagamenti consentono solo connessioni al servizio tramite TLS con una suite di crittografia che offre una perfetta segretezza di inoltro.

Nota: alcune delle funzioni elencate vengono utilizzate internamente per supportare protocolli e strutture dati standard. Consulta la documentazione dell'API per gli algoritmi supportati da azioni specifiche.

Archiviazione delle chiavi

AWS Le chiavi di crittografia dei pagamenti sono protette dalle chiavi principali HSM AES 256 e archiviate in blocchi di chiavi ANSI X9 TR 31 in un database crittografato. Il database viene replicato in un database in memoria sui server Payment Cryptography. AWS

Secondo l'allegato C della normativa sulla sicurezza dei PCI PIN, le chiavi AES 256 sono altrettanto potenti o più potenti di:

- TDEA a 3 tasti
- RSA a 15360 bit
- ECC a 512 bit
- DSA, DH e MQV 15360/512

Importazione di chiavi tramite chiavi simmetriche

AWS La crittografia dei pagamenti supporta l'importazione di crittogrammi e blocchi di chiavi con chiavi simmetriche o pubbliche con una chiave di crittografia a chiave simmetrica (KEK) che è altrettanto potente o più potente della chiave protetta per l'importazione.

Importazione di chiavi tramite chiavi asimmetriche

AWS La crittografia dei pagamenti supporta l'importazione di crittogrammi e blocchi di chiavi con chiavi simmetriche o pubbliche protette da una chiave di crittografia a chiave privata (KEK) che è altrettanto potente o più potente della chiave protetta per l'importazione. L'autenticità e l'integrità della chiave pubblica fornita per la decrittografia devono essere garantite da un certificato rilasciato da un'autorità di fiducia del cliente.

Le KEK pubbliche fornite da AWS Payment Cryptography hanno l'autenticazione e la protezione dell'integrità di un'autorità di certificazione (CA) con conformità attestata a PCI PIN Security e PCI P2PE Annex A.

Esportazione di chiavi

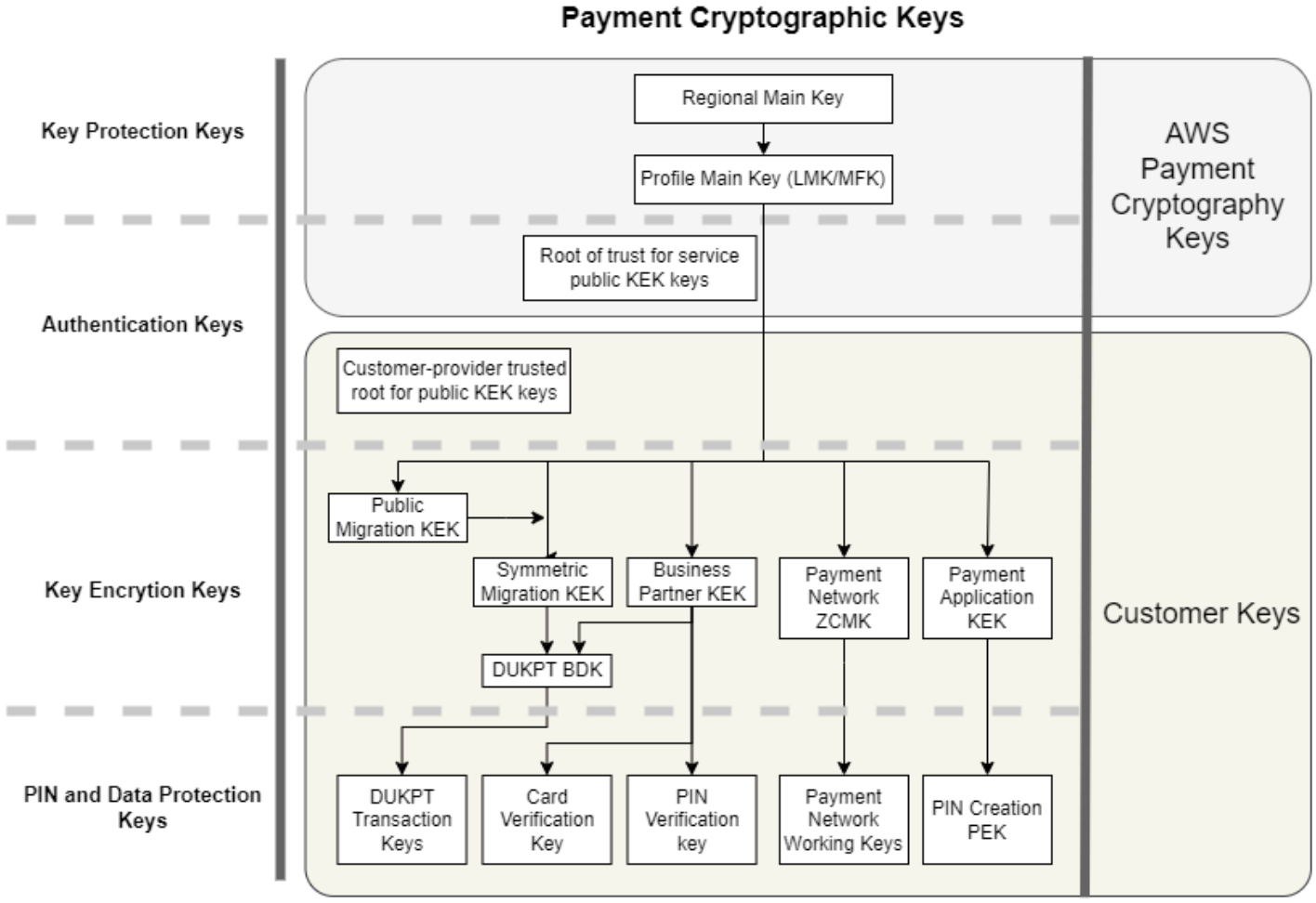
Le chiavi possono essere esportate e protette con chiavi appropriate KeyUsage e che siano altrettanto forti o più forti della chiave da esportare.

Protocollo DUKPT (Derived Unique Key Per Transaction)

AWS La crittografia dei pagamenti supporta le chiavi di derivazione di base (BDK) TDEA e AES, come descritto da ANSI. X9.24-3

Gerarchia delle chiavi

La gerarchia delle chiavi di AWS Payment Cryptography garantisce che le chiavi siano sempre protette da chiavi altrettanto potenti o più potenti delle chiavi che proteggono.



AWS Le chiavi di crittografia dei pagamenti vengono utilizzate per la protezione delle chiavi all'interno del servizio:

Chiave	Description
Chiave principale regionale	Protegge le immagini o i profili HSM virtuali utilizzati per l'elaborazione crittografica. Questa chiave esiste solo nei backup HSM e sicuri.
Chiave principale del profilo	Chiave di protezione delle chiavi del cliente di alto livello, tradizionalmente chiamata Local Master Key (LMK) o Master File Key (MFK) per le chiavi del cliente. Questa chiave esiste solo nei backup HSM e sicuri. I profili definisco no configurazioni HSM distinte, come richiesto dagli standard di sicurezza per i casi d'uso dei pagamenti.
Radice di fiducia per le chiavi di crittografia a chiave pubblica (KEK) di AWS Payment Cryptography	La chiave pubblica principale e il certificato affidabili per l'autenticazione e la convalida delle chiavi pubbliche forniti da AWS Payment Cryptography per l'importazione e l'esporta zione di chiavi utilizzando chiavi asimmetriche.

Le chiavi del cliente sono raggruppate in base alle chiavi utilizzate per proteggere altre chiavi e chiavi che proteggono i dati relativi ai pagamenti. Questi sono esempi di chiavi cliente di entrambi i tipi:

Chiave	Description
Customer-provided root affidabile per le chiavi KEK pubbliche	Chiave pubblica e certificato forniti dall'uten te come base di fiducia per l'autenticazione e la convalida delle chiavi pubbliche fornite per l'importazione e l'esportazione delle chiavi utilizzando chiavi asimmetriche.
Chiavi di crittografia a chiave (KEK)	Le KEK vengono utilizzate esclusivamente per crittografare altre chiavi per lo scambio tra archivi di chiavi esterni e AWS Payment Cryptography, partner commerciali, reti di

Chiave	Description
	pagamento o diverse applicazioni all'interno dell'organizzazione.
Chiave di derivazione base Derived Unique Key Per Transaction (DUKPT) (BDK)	I BDK vengono utilizzati per creare chiavi univoche per ogni terminale di pagamento e tradurre le transazioni da più terminali in un'unica chiave di funzionamento della banca o dell'acquirente. La best practice, richiesta dalla Point-to-Point crittografia PCI (P2PE), prevede l'utilizzo di BDK diversi per diversi modelli di terminale, servizi di iniezione o inizializzazione di chiavi o altra segmentazione per limitare l'impatto della compromissione di un BDK.
Chiave principale per il controllo della zona di rete di pagamento (ZCMK)	Le ZCMK, note anche come chiavi di zona o chiavi master di zona, vengono fornite dalle reti di pagamento per stabilire le chiavi di lavoro iniziali.
chiavi di transazione DUKPT	I terminali di pagamento configurati per DUKPT derivano una chiave unica per il terminale e la transazione. L'HSM che riceve la transazio ne può determinare la chiave dall'identificator e del terminale e dal numero di sequenza della transazione.

Chiave	Description
Chiavi per la preparazione dei dati delle carte	Le chiavi master dell'emittente EMV, le chiavi e i valori di verifica delle carte EMV e le chiavi di protezione dei file di dati per la personalizzazione delle carte vengono utilizzate per creare dati per singole carte utilizzabili da un fornitore di servizi di personalizzazione delle carte. Queste chiavi e i dati di convalida crittografica vengono utilizzati anche dalle banche emittenti, o dagli emittenti, per autenticare i dati delle carte nell'ambito dell'autorizzazione delle transazioni.
Chiavi per la preparazione dei dati delle carte	Le chiavi master dell'emittente EMV, le chiavi e i valori di verifica delle carte EMV e le chiavi di protezione dei file di dati per la personalizzazione delle carte vengono utilizzate per creare dati per singole carte utilizzabili da un fornitore di servizi di personalizzazione delle carte. Queste chiavi e i dati di convalida crittografica vengono utilizzati anche dalle banche emittenti, o dagli emittenti, per autenticare i dati delle carte nell'ambito dell'autorizzazione delle transazioni.
Chiavi funzionanti della rete di pagamento	Spesso denominate chiave di lavoro dell'emittente o chiave di lavoro dell'acquirente, sono le chiavi che crittografano le transazioni inviate o ricevute dalle reti di pagamento. Queste chiavi vengono ruotate frequentemente dalla rete, spesso ogni giorno o ogni ora. Si tratta di chiavi di crittografia PIN (PEK) per le transazioni. PIN/Debit

Chiave	Description
Chiavi di crittografia con numero di identific azione personale (PIN) (PEK)	Le applicazioni che creano o decrittografano blocchi PIN utilizzano il formato PEK per impedire l'archiviazione o la trasmissione di PIN in testo non crittografato.

Operazioni interne

Questo argomento descrive i requisiti interni implementati dal servizio per proteggere le chiavi dei clienti e le operazioni crittografiche per un servizio di crittografia dei pagamenti e gestione delle chiavi distribuito a livello globale e scalabile.

Argomenti

- Protezione HSM
- Gestione generale delle chiavi
- Gestione delle chiavi dei clienti
- Sicurezza delle comunicazioni
- Registrazione di log e monitoraggio

Protezione HSM

Specifiche e ciclo di vita HSM

AWS Payment Cryptography utilizza una flotta di strumenti disponibili in commercio. HSMs HSMs Sono conformi allo standard FIPS 140-2 di livello 3 e utilizzano anche le versioni del firmware e la politica di sicurezza elencate nell'elenco dei dispositivi PCI PTS approvati dal PCI Security Standards Council come conformi allo standard PCI HSM v3. Lo standard PCI PTS HSM include requisiti aggiuntivi per la produzione, la spedizione, l'implementazione, la gestione e la distruzione dell'hardware HSM, importanti per la sicurezza e la conformità dei pagamenti ma non soddisfatti da FIPS 140.

I valutatori di terze parti verificano il modello di fabbricazione, il firmware, la configurazione, la gestione fisica del ciclo di vita, il controllo delle modifiche, i controlli degli accessi degli operatori, la gestione delle chiavi principali e tutti i requisiti PCI PIN e P2PE relativi alle operazioni HSM. HSMs

Tutti HSMs sono gestiti in modalità PCI e configurati con la politica di sicurezza PCI PTS HSM. Sono abilitate solo le funzioni necessarie per supportare i casi d'uso della crittografia dei AWS pagamenti. AWS Payment Cryptography non prevede la stampa, la visualizzazione o la restituzione di testo non crittografato. PINs

Sicurezza fisica dei dispositivi HSM

Il servizio può utilizzare solo HSMs le chiavi del dispositivo firmate da un'autorità di certificazione (CA) di crittografia dei AWS pagamenti (CA) del produttore prima della consegna. La AWS Payment Cryptography è una CA secondaria della CA del produttore che è alla base della fiducia per i certificati dei produttori e dei dispositivi HSM. L'autorità di certificazione del produttore ha attestato la conformità agli standard PCI PIN Security Annex A e PCI P2PE Allegato A. Il produttore verifica che tutti gli HSM con chiavi del dispositivo firmate da AWS Payment Cryptography CA vengano spediti al destinatario designato da AWS.

Come richiesto da PCI PIN Security, il produttore fornisce un elenco di numeri seriali tramite un canale di comunicazione diverso da quello di spedizione HSM. Questi numeri di serie vengono controllati in ogni fase del processo di installazione di HSM nei data center AWS. Infine, gli operatori AWS di Payment Cryptography convalidano l'elenco degli HSM installati confrontandolo con l'elenco del produttore prima di aggiungere il numero di serie all'elenco degli HSM autorizzati a ricevere le chiavi di crittografia dei pagamenti. AWS

HSMs sono in un archivio sicuro o sotto doppio controllo in ogni momento, il che include:

- Spedizione dal produttore a un impianto di assemblaggio su rack AWS.
- Durante l'assemblaggio del rack.
- Spedizione dall'impianto di assemblaggio del rack a un data center.
- Ricezione e installazione in una sala di elaborazione sicura del data center. I rack HSM prevedono un doppio controllo con serrature con accesso tramite scheda, sensori allarmati sulle porte e telecamere.
- Durante le operazioni.
- Durante lo smantellamento e la distruzione.

Per ogni HSM viene mantenuto e monitorato un sistema completo chain-of-custody, con responsabilità individuale.

Inizializzazione HSM

Un HSM viene inizializzato come parte della flotta AWS Payment Cryptography solo dopo che la sua identità e integrità sono state convalidate mediante numeri di serie, chiavi del dispositivo installate dal produttore e checksum del firmware. Dopo la convalida dell'autenticità e dell'integrità di un HSM, questo viene configurato, inclusa l'attivazione della modalità PCI. Quindi vengono stabilite le chiavi principali della regione AWS Payment Cryptography e le chiavi principali del profilo e l'HSM è disponibile per il servizio.

Assistenza e riparazione HSM

HSM dispone di componenti riparabili che non richiedono la violazione del limite crittografico del dispositivo. Questi componenti includono ventole di raffreddamento, alimentatori e batterie. Se un HSM o un altro dispositivo all'interno del rack HSM necessita di assistenza, il doppio controllo viene mantenuto per tutto il periodo di apertura del rack.

Disattivazione HSM

La disattivazione avviene a causa end-of-life o a un guasto di un HSM. Gli HSM vengono azzerati logicamente prima di essere rimossi dal rack, se funzionanti, e poi distrutti all'interno delle sale di elaborazione sicure dei data center AWS. Non vengono mai restituiti al produttore per la riparazione, utilizzati per altri scopi o altrimenti rimossi da una sala di elaborazione sicura prima della distruzione.

Aggiornamento del firmware HSM

Gli aggiornamenti del firmware HSM vengono applicati quando necessario per mantenere l'allineamento con le versioni elencate di PCI PTS HSM e FIPS 140-2 (o FIPS 140-3), se un aggiornamento è legato alla sicurezza o se si stabilisce che i clienti possono beneficiare delle funzionalità di una nuova versione. AWS Payment Cryptography esegue il firmware, corrispondente alle versioni elencate in PCI HSMs PTS HSM. off-the-shelf L'integrità delle nuove versioni del firmware viene convalidata con le versioni del firmware certificate PCI o FIPS, quindi testate per verificarne la funzionalità prima dell'implementazione su tutte. HSMs

Accesso dell'operatore

Gli operatori possono accedere a HSM senza console per la risoluzione dei problemi nei rari casi in cui le informazioni raccolte da HSM durante le normali operazioni non siano sufficienti per identificare un problema o pianificare una modifica. Vengono eseguiti i seguenti passaggi:

- Le attività di risoluzione dei problemi vengono sviluppate e approvate e viene pianificata la sessione non basata sulla console.
- Un HSM viene rimosso dal servizio di elaborazione clienti.
- Le chiavi principali vengono eliminate, sotto doppio controllo.
- All'operatore è consentito l'accesso all'HSM senza console per eseguire attività di risoluzione dei problemi approvate, sotto doppio controllo.
  - Al termine della sessione non consolle, viene eseguito il processo di provisioning iniziale sull'HSM, restituendo il firmware e la configurazione standard, quindi sincronizzando la chiave principale, prima di restituire l'HSM ai clienti.
  - I record della sessione vengono registrati nel monitoraggio delle modifiche.
  - Le informazioni ottenute dalla sessione vengono utilizzate per pianificare le modifiche future.

Tutti i record di accesso non relativi alla console vengono esaminati per verificarne la conformità dei processi e le potenziali modifiche al monitoraggio HSM, al processo di non-console-access gestione o alla formazione degli operatori.

Gestione generale delle chiavi

Tutti gli HSM di una regione sono sincronizzati con una chiave principale regionale. Una chiave principale regionale protegge almeno una chiave principale del profilo. Una Profile Main Key protegge le chiavi del cliente.

Tutte le chiavi principali sono generate da un HSM e distribuite mediante distribuzione simmetrica delle chiavi utilizzando tecniche asimmetriche, in linea con ANSI X9 TR 34 e PCI PIN Annex A.

Generazione

Le chiavi principali AES a 256 bit vengono generate su uno degli HSM predisposti per il parco HSM del servizio, utilizzando il generatore di numeri casuali PCI PTS HSM.

Sincronizzazione delle chiavi principali della regione

Le chiavi principali della regione HSM sono sincronizzate dal servizio su tutta la flotta regionale con meccanismi definiti da ANSI X9 TR-34, che includono:

- Autenticazione reciproca tramite chiavi e certificati KDH (Key Distribution Host) e Key Receiver Device (KRD) per garantire l'autenticazione e l'integrità delle chiavi pubbliche.

- I certificati sono firmati da un'autorità di certificazione (CA) che soddisfa i requisiti del PIN PCI allegato A2, ad eccezione degli algoritmi asimmetrici e dei punti di forza chiave appropriati per proteggere le chiavi AES a 256 bit.
- L'identificazione e la protezione delle chiavi per le chiavi simmetriche distribuite sono coerenti con ANSI X9 TR-34 e PCI PIN Annex A1, ad eccezione degli algoritmi asimmetrici e dei punti di forza chiave appropriati per proteggere le chiavi AES a 256 bit.

Vengono stabilite le chiavi principali della regione che sono state autenticate e fornite per una regione da: HSMs

- Una chiave principale viene generata su un HSM nella regione. Tale HSM è designato come host di distribuzione delle chiavi.
- Tutte le unità fornite HSMs nella regione generano un token di autenticazione KRD, che contiene la chiave pubblica dell'HSM e informazioni di autenticazione non rigiocabili.
- I token KRD vengono aggiunti all'elenco di autorizzazioni KDH dopo che il KDH ha convalidato l'identità e l'autorizzazione dell'HSM a ricevere le chiavi.
- Il KDH produce un token di chiave principale autenticabile per ogni HSM. I token contengono informazioni di autenticazione KDH e una chiave principale crittografata che può essere caricata solo su un HSM per cui sono stati creati.
- A ogni HSM viene inviato il token chiave principale creato appositamente. Dopo aver convalidato le informazioni di autenticazione proprie dell'HSM e le informazioni di autenticazione KDH, la chiave principale viene decrittografata dalla chiave privata KRD e caricata nella chiave principale.

Nel caso in cui un singolo HSM debba essere risincronizzato con una regione:

- Viene riconvalidato e dotato di firmware e configurazione.
- Se è nuovo nella regione:
  - L'HSM genera un token di autenticazione KRD.
  - Il KDH aggiunge il token all'elenco degli elementi consentiti.
  - Il KDH genera un token chiave principale per l'HSM.
  - L'HSM carica la chiave principale.
  - L'HSM viene messo a disposizione del servizio.

Ciò garantisce che:

- Solo gli HSM convalidati per l'elaborazione della crittografia dei AWS pagamenti all'interno di una regione possono ricevere la chiave principale di quella regione.
- Solo una chiave master di un AWS Payment Cryptography HSM può essere distribuita a un HSM del parco macchine.

Rotazione dei tasti principali della regione

Le chiavi principali della regione vengono ruotate alla scadenza del periodo crittografico, nell'improbabile eventualità che si sospetti una compromissione della chiave o dopo modifiche al servizio che si ritiene abbiano un impatto sulla sicurezza della chiave.

Una nuova chiave principale della regione viene generata e distribuita come durante il provisioning iniziale. Le chiavi principali del profilo salvate devono essere tradotte nella nuova chiave principale della regione.

La rotazione delle chiavi principali della regione non influisce sull'elaborazione dei clienti.

Sincronizzazione delle chiavi principali del profilo

Le chiavi principali del profilo sono protette dalle chiavi principali della regione. Ciò limita un profilo a una regione specifica.

Le chiavi principali del profilo vengono fornite di conseguenza:

- Una chiave principale del profilo viene generata su un HSM con la chiave principale della regione sincronizzata.
- La chiave principale del profilo viene archiviata e crittografata con la configurazione del profilo e altri contesti.
- Il profilo viene utilizzato per le funzioni crittografiche del cliente da qualsiasi HSM della regione con la chiave principale della regione.

Rotazione della chiave principale del profilo

Le chiavi principali del profilo vengono ruotate alla scadenza del periodo crittografico, dopo una sospetta compromissione della chiave o dopo modifiche al servizio che si ritiene abbiano un impatto sulla sicurezza della chiave.

Fasi di rotazione:

- Una nuova chiave principale del profilo viene generata e distribuita come chiave principale in sospeso, come nel caso del provisioning iniziale.
- Un processo in background traduce il materiale chiave del cliente dalla chiave principale del profilo stabilito alla chiave principale in sospeso.
- Quando tutte le chiavi del cliente sono state crittografate con la chiave in sospeso, la chiave in sospeso viene promossa alla chiave principale del profilo.
- Un processo in background elimina il materiale chiave del cliente protetto dalla chiave scaduta.

La rotazione delle chiavi principali del profilo non influisce sull'elaborazione dei clienti.

Protezione

Le chiavi dipendono solo dalla gerarchia delle chiavi per la protezione. La protezione delle chiavi principali è fondamentale per prevenire la perdita o la compromissione di tutte le chiavi del cliente.

Le chiavi principali della regione sono ripristinabili dal backup solo su sistemi HSM autenticati e forniti per il servizio. Queste chiavi possono essere archiviate solo come token di chiave principale crittografati e reciprocamente autenticabili da un KDH specifico per un HSM specifico.

Le chiavi master del profilo vengono archiviate con la configurazione del profilo e le informazioni di contesto crittografate per regione.

Le chiavi del cliente sono archiviate in blocchi chiave, protetti da una chiave master del profilo.

Tutte le chiavi esistono esclusivamente all'interno di un HSM o sono archiviate protette da un'altra chiave con una forza crittografica uguale o superiore.

Durabilità

Le chiavi del cliente per la crittografia delle transazioni e le funzioni aziendali devono essere disponibili anche in situazioni estreme che in genere causerebbero interruzioni. AWS La crittografia dei pagamenti utilizza un modello di ridondanza a più livelli tra zone e regioni di disponibilità. AWS Il cliente che richiede una disponibilità e una durabilità maggiori per le operazioni crittografiche di pagamento rispetto a quelle fornite dal servizio deve implementare architetture multiregionali.

L'autenticazione HSM e i token della chiave principale vengono salvati e possono essere utilizzati per ripristinare una chiave principale o sincronizzarsi con una nuova chiave principale, nel caso in cui sia necessario reimpostare un HSM. I token vengono archiviati e utilizzati solo sotto doppio controllo quando necessario.

Accesso dell'operatore alle chiavi principali HSM

Le chiavi principali esistono solo in HSM gestite dal servizio e protette in strutture AWS sicure. Le chiavi principali non possono essere esportate da alcun HSM o sincronizzate su un HSM non inizializzato dal produttore per essere utilizzate nel servizio. Gli operatori AWS non possono ottenere chiavi principali in alcuna forma che possano essere caricate in un HSM non gestito dal servizio.

Gestione delle chiavi dei clienti

At AWS, la fiducia dei clienti è la nostra massima priorità. Mantieni il pieno controllo delle chiavi che importi o crei nel servizio con il tuo account AWS. Sei responsabile della configurazione dell'accesso alle chiavi.

AWS Payment Cryptography è un fornitore di servizi che utilizza HSMs e gestisce le chiavi per conto dei clienti, in modo simile ai fornitori di servizi di pagamento di lunga data. Il servizio ha la completa responsabilità della sicurezza fisica e logica di HSM. La responsabilità della gestione delle chiavi è condivisa tra il servizio e i clienti perché il cliente deve fornire informazioni accurate sulle chiavi create o importate nel servizio, che il servizio utilizza per imporre l'uso e la gestione corretti delle chiavi. Le protezioni di segregazione dei dati di AWS vengono utilizzate per garantire che la compromissione delle chiavi appartenenti a un account AWS non possa compromettere le chiavi appartenenti a un altro.

AWS Payment Cryptography ha la piena responsabilità della conformità fisica HSM e della gestione delle chiavi per le chiavi gestite dal servizio. Ciò richiede la proprietà e la gestione delle chiavi principali HSM e la protezione delle chiavi dei clienti gestite dalla crittografia dei AWS pagamenti.

Separazione degli spazi delle chiavi del cliente

AWS Payment Cryptography applica politiche chiave per tutti gli usi delle chiavi, inclusa la limitazione dei principi all'account proprietario della chiave, a meno che una chiave non venga esplicitamente condivisa con un altro account.

Gli account AWS forniscono una separazione completa dell'ambiente tra clienti o applicazioni, analogamente alle implementazioni non cloud in diversi data center. Ogni account fornisce controllo degli accessi isolati, reti, risorse di calcolo, archiviazione dei dati, chiavi crittografiche per la protezione dei dati e le transazioni di pagamento e tutte le risorse AWS. I servizi AWS come Organizations e Control Tower consentono la gestione aziendale di account applicativi separati, analogamente alle gabbie o alle stanze all'interno di un data center aziendale.

Accesso dell'operatore alle chiavi del cliente

Le chiavi cliente gestite dal servizio sono archiviate protette dalle chiavi principali di partizione e possono essere utilizzate solo dall'account cliente proprietario o dall'account che il proprietario ha configurato specificamente per la condivisione delle chiavi. Gli operatori AWS non possono esportare o eseguire operazioni di gestione delle chiavi o crittografiche con le chiavi dei clienti utilizzando l'accesso manuale al servizio, che è gestito dai meccanismi di accesso manuale dell'operatore di AWS.

Il codice di servizio che implementa la gestione e l'uso delle chiavi del cliente è soggetto alle pratiche di codice sicuro di AWS, secondo la valutazione PCI DSS di AWS.

Backup e ripristino

Le chiavi e le informazioni chiave archiviate internamente dal servizio per una regione vengono salvate in archivi crittografati da. AWS Gli archivi richiedono un doppio controllo AWS per il ripristino.

Blocchi chiave

Tutte le chiavi vengono archiviate ed elaborate in blocchi chiave in formato ANSI X9.143.

Le chiavi possono essere importate nel servizio da crittogrammi o altri formati di blocchi di chiavi supportati da. ImportKey Allo stesso modo, le chiavi possono essere esportate, se esportabili, in altri formati di blocchi di chiavi o crittogrammi supportati dai profili di esportazione delle chiavi.

Uso delle chiavi

L'uso delle chiavi è limitato a quello configurato KeyUsage dal servizio. Il servizio fallirà qualsiasi richiesta con utilizzo della chiave, modalità di utilizzo o algoritmo inappropriati per l'operazione di crittografia richiesta.

Relazioni di scambio chiave

PCI PIN Security e PCI P2PE richiedono che le organizzazioni che condividono chiavi di crittografia PINs o dati cartacei, incluse le chiavi di scambio di chiavi (KEK) utilizzate per condividere tali chiavi, non condividano le stesse chiavi con altre organizzazioni. È consigliabile condividere le chiavi simmetriche solo tra due parti per un unico scopo, anche all'interno della stessa organizzazione. Ciò riduce al minimo l'impatto dei sospetti compromessi chiave che impongono la sostituzione delle chiavi interessate.

Anche i casi aziendali che richiedono la condivisione delle chiavi tra più di 2 parti dovrebbero mantenere il numero di parti al minimo.

AWS Payment Cryptography fornisce tag chiave che possono essere utilizzati per tracciare e far rispettare l'utilizzo delle chiavi entro tali requisiti.

Ad esempio, KEK e BDK per diversi impianti di iniezione delle chiavi possono essere identificati impostando un «KIF» = «POSStation» per tutte le chiavi condivise con quel fornitore di servizi. Un altro esempio potrebbe essere quello di etichettare le chiavi condivise con le reti di pagamento con «Network» = «». PayCard L'etichettatura consente di creare controlli di accesso e creare report di audit per applicare e dimostrare le pratiche di gestione chiave.

Eliminazione delle chiavi

DeleteKey contrassegna le chiavi nel database per l'eliminazione dopo un periodo configurabile dal cliente. Dopo questo periodo la chiave viene eliminata irrimediabilmente. Si tratta di un meccanismo di sicurezza per impedire l'eliminazione accidentale o dolosa di una chiave. Le chiavi contrassegnate per l'eliminazione non sono disponibili per nessuna azione tranne RestoreKey.

Le chiavi eliminate rimangono nei backup dei servizi per 7 giorni dopo l'eliminazione. Non sono ripristinabili durante questo periodo.

Le chiavi che appartengono agli account AWS chiusi sono contrassegnate per l'eliminazione.
Se l'account viene riattivato prima del termine di eliminazione, tutte le chiavi contrassegnate per l'eliminazione vengono ripristinate, ma disattivate. È necessario riattivarle dall'utente per poterle utilizzare per operazioni crittografiche.

Sicurezza delle comunicazioni

Esterno

AWS Gli endpoint dell'API Payment Cryptography soddisfano gli standard AWS di sicurezza tra cui TLS versione 1.2 o superiore e Signature Version 4 per l'autenticazione e l'integrità delle richieste.

Le connessioni TLS in entrata vengono terminate sui sistemi di bilanciamento del carico di rete e inoltrate ai gestori API tramite connessioni TLS interne.

Interno

Le comunicazioni interne tra i componenti del servizio e tra i componenti del servizio e altri servizi AWS sono protette da TLS utilizzando una crittografia avanzata.

Gli HSM si trovano su una rete privata non virtuale raggiungibile solo dai componenti del servizio. Tutte le connessioni tra HSM e i componenti del servizio sono protette con TLS reciproco (mTLS), pari o superiore a TLS 1.2. I certificati interni per TLS e MTL sono gestiti da Amazon Certificate Manager utilizzando un'autorità di certificazione privata AWS. La rete interna VPCs e la rete HSM vengono monitorate per rilevare attività impreviste e modifiche alla configurazione.

Registrazione di log e monitoraggio

I registri di servizio interni includono:

- CloudTrail registri delle chiamate al servizio AWS effettuate dal servizio
- CloudWatch registri di entrambi gli eventi registrati direttamente nei log o CloudWatch negli eventi da HSM
- File di registro da HSM e dai sistemi di servizio
- Archivi di registro

Tutte le fonti di registro monitorano e filtrano le informazioni sensibili, incluse quelle relative alle chiavi. I log vengono esaminati sistematicamente per garantire che contengano informazioni riservate sui clienti e non contengano informazioni riservate.

L'accesso ai registri è limitato alle persone necessarie per completare i ruoli lavorativi.

Tutti i log vengono conservati in linea con le policy di conservazione dei log di AWS.

Operazioni con i clienti

AWS Payment Cryptography ha la piena responsabilità della conformità fisica dell'HSM agli standard PCI. Il servizio fornisce anche un archivio sicuro delle chiavi e garantisce che le chiavi possano essere utilizzate solo per gli scopi consentiti dagli standard PCI e specificati dall'utente durante la creazione o l'importazione. L'utente è responsabile della configurazione degli attributi chiave e dell'accesso per sfruttare le funzionalità di sicurezza e conformità del servizio.

Argomenti

- Generazione delle chiavi
- Importazione delle chiavi
- Esportazione delle chiavi
- Eliminazione delle chiavi

- Rotazione delle chiavi

Generazione delle chiavi

Quando si creano le chiavi, si impostano gli attributi utilizzati dal servizio per imporre l'uso conforme della chiave:

- Algoritmo e lunghezza della chiave
- Utilizzo
- Disponibilità e scadenza

I tag utilizzati per il controllo degli accessi basato sugli attributi (ABAC) vengono utilizzati per limitare l'utilizzo delle chiavi con partner o applicazioni specifici, inoltre è necessario impostare durante la creazione. Assicurati di includere politiche per limitare i ruoli autorizzati a eliminare o modificare i tag.

È necessario assicurarsi che le politiche che determinano i ruoli che possono utilizzare e gestire la chiave siano impostate prima della creazione della chiave.

> **Note**
>
> Le politiche IAM relative ai CreateKey comandi possono essere utilizzate per applicare e dimostrare il doppio controllo per la generazione delle chiavi.

Importazione delle chiavi

Quando si importano le chiavi, gli attributi per imporre un uso conforme della chiave vengono impostati dal servizio utilizzando le informazioni legate crittograficamente nel blocco chiave. Il meccanismo per impostare il contesto chiave fondamentale consiste nell'utilizzare blocchi chiave creati con l'HSM di origine e protetti da una KEK condivisa o asimmetrica. Ciò è in linea con i requisiti del PIN PCI e preserva l'utilizzo, l'algoritmo e la forza della chiave dell'applicazione di origine.

Oltre alle informazioni contenute nel blocco chiave, è necessario stabilire importanti attributi chiave, tag e politiche di controllo degli accessi al momento dell'importazione.

L'importazione di chiavi mediante crittogrammi non trasferisce gli attributi chiave dall'applicazione di origine. È necessario impostare gli attributi in modo appropriato utilizzando questo meccanismo.

Spesso le chiavi vengono scambiate utilizzando componenti di testo in chiaro, trasmesse dai custodi delle chiavi e quindi caricate con una cerimonia che prevede il doppio controllo in una stanza sicura. Questo non è supportato direttamente da AWS Payment Cryptography. L'API esporterà una chiave pubblica con un certificato che può essere importato dal proprio HSM per esportare un blocco chiave importabile dal servizio. Consente l'uso del proprio HSM per caricare componenti in testo non crittografato.

È necessario utilizzare Key check values (KCV) per verificare che le chiavi importate corrispondano alle chiavi di origine.

Le politiche IAM sull' ImportKey API possono essere utilizzate per applicare e dimostrare il doppio controllo per l'importazione delle chiavi.

Esportazione delle chiavi

La condivisione delle chiavi con partner o applicazioni locali può richiedere l'esportazione delle chiavi. L'utilizzo di blocchi chiave per le esportazioni mantiene un contesto chiave fondamentale con il materiale chiave crittografato.

I tag chiave possono essere utilizzati per limitare l'esportazione in KEK di chiavi che condividono lo stesso tag e lo stesso valore.

AWS La crittografia dei pagamenti non fornisce né visualizza componenti chiave in testo chiaro. Ciò richiede l'accesso diretto da parte dei custodi delle chiavi a dispositivi crittografici sicuri (SCD) testati PCI PTS HSM o ISO 13491 per la visualizzazione o la stampa. Puoi stabilire una KEK asimmetrica o una KEK simmetrica con il tuo SCD per condurre la cerimonia di creazione dei componenti chiave in testo chiaro sotto doppio controllo.

È necessario utilizzare i valori di controllo delle chiavi (KCV) per verificare che le chiavi importate dall'HSM di destinazione corrispondano alle chiavi di origine.

Eliminazione delle chiavi

È possibile utilizzare l'API di eliminazione della chiave per pianificare l'eliminazione delle chiavi dopo un periodo di tempo configurato. Prima di quel momento le chiavi erano recuperabili. Una volta eliminate, le chiavi vengono rimosse definitivamente dal servizio.

Le policy IAM sull' DeleteKey API possono essere utilizzate per applicare e dimostrare il doppio controllo per l'eliminazione delle chiavi.

Rotazione delle chiavi

L'effetto della rotazione delle chiavi può essere implementato utilizzando l'alias chiave creando o importando una nuova chiave, quindi modificando l'alias della chiave per fare riferimento alla nuova chiave. La vecchia chiave verrebbe eliminata o disabilitata, a seconda delle pratiche di gestione.

Quote per AWS Payment Cryptography

L'account AWS dispone delle seguenti quote predefinite, precedentemente definite limiti, per ogni servizio AWS. Salvo diversa indicazione, ogni quota è specifica per regione. Se per alcune quote è possibile richiedere aumenti, altre quote non possono essere modificate.

Nome	Predefinita	Adattabile	Description
Alias	Ogni regione supportata: 2.000	Sì	Il numero massimo di alias che puoi avere in questo account nella regione corrente.
Tasso combinato di richieste relative ai piani di controllo	Ogni regione supportata: 5 al secondo	Sì	Il numero massimo di richieste di piano di controllo al secondo che puoi effettuare in questo account nella regione corrente. Questa quota si applica a tutte le operazioni combinate del piano di controllo.
Tasso combinato di richieste sul piano dati (asimmetrico)	Ogni regione supportata: 20 al secondo	Sì	Il numero massimo di richieste al secondo per le operazioni sul piano dati con una chiave asimmetrica che è possibile effettuare in questo account nella regione corrente. Questa quota si applica a tutte le operazioni combinate sul piano dati.

Nome	Predefinita	Adatta e	Description
Tasso combinato di richieste sul piano dati (simmetrico)	Ogni regione supportata: 500 al secondo	Sì	Il numero massimo di richieste al secondo per le operazioni sul piano dati con una chiave simmetrica che è possibile effettuare in questo account nella regione corrente. Questa quota si applica a tutte le operazioni combinate sul piano dati.
Chiavi	Ogni regione supportata: 2.000	Sì	Il numero massimo di chiavi che puoi avere in questo account nella regione corrente, escluse le chiavi eliminate.

Cronologia dei documenti per AWS Guida per l'utente di Payment Cryptography

La tabella seguente descrive le versioni della documentazione per AWS Payment Cryptography.

Modifica	Descrizione	Data
Supporto asimmetrico per la replica delle chiavi Multi-Region	Multi-Region la replica delle chiavi ora supporta chiavi di crittografia di pagamento asimmetriche come RSA ed ECC, oltre alle chiavi simmetriche.	27 luglio 2026
Nuova funzionalità: AS2805	Supporto per algoritmi e flussi per supportare il supporto regionale AS2805	17 dicembre 2025
Nuova funzionalità: replica Multi-Region delle chiavi	Con Multi-Region la replica delle chiavi, puoi replicare le tue chiavi di crittografia dei AWS pagamenti su più chiavi. Regioni AWS	10 settembre 2025
Nuova funzionalità: ECDH	Con questa versione, l'ECDH può essere utilizzato per stabilire un KEK condiviso per un ulteriore scambio di chiavi.	30 marzo 2025
Nuove linee guida per Key Exchange	Sono state fornite nuove linee guida per gli scambi chiave. Sono state inoltre aggiunte informazioni sui comandi JCB comuni.	31 gennaio 2025
Lancio di una nuova regione	Aggiunti endpoint per il lancio di nuove regioni in Europa	31 luglio 2024

	(Francoforte), Europa (Irlanda) , Asia Pacifico (Singapore) e Asia Pacifico (Tokyo)	
CloudTrail per Data Plane e Dynamic Keys	Sono state aggiunte informazioni sull'utilizzo CloudTrail per le operazioni sul piano dati (crittografiche), inclusi esempi. Sono state inoltre aggiunte informazioni sull'utilizzo delle chiavi dinamiche per determinate funzioni per supportare meglio le chiavi monouso o a uso limitato che non devono essere importate in Payment Cryptography AWS	10 luglio 2024
Esempi aggiornati	Sono stati aggiunti nuovi esempi per l'emissione di carte	1° luglio 2024
Rilascio di funzionalità	Aggiunta di informazioni sugli endpoint VPC (PrivateLink) e sugli esempi iCVV.	30 maggio 2024
Rilascio di funzionalità	Sono state aggiunte informazioni sulle nuove funzionalità relative all' import/export utilizzo delle chiavi RSA e all'esportazione delle chiavi IPEK/IK DUKPT.	15 gennaio 2024
Versione iniziale	Versione iniziale della AWS Payment Cryptography User Guide	8 giugno 2023

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.