Guida per l'utente

AWS CloudHSM



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

AWS CloudHSM: 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

Che cos'è AWS CloudHSM? 1
Casi d'uso 2
Come funziona 4
Cluster 5
Utenti in AWS CloudHSM 6
Chiavi in AWS CloudHSM 6
Cliente SDKs 7
backup 8
Regioni supportate per AWS CloudHSM 9
Prezzi per AWS CloudHSM 9
Nozioni di base 10
Creazione di amministratori IAM; 10
Creazione di un gruppo di amministratori e di un utente IAM; 11
Crea un VPC 11
Creazione di un cluster 12
Rivedi gruppo di sicurezza del cluster 16
Avviare un client EC2 17
Configura i gruppi di sicurezza dell'istanza EC2 20
Passaggio 1. Modifica il gruppo di sicurezza di default. 20
Passaggio 2. Connect l'istanza Amazon EC2 al cluster AWS CloudHSM 21
Crea un HSM 22
Verifica dell'identità dell'HSM (facoltativo) 24
Passaggio 1. Ottieni i certificati da HSM 26
Passaggio 2. Ottenere i certificati root 29
Fase 3. Verifica le catene di certificato 29
Passaggio 4. Estrai e confronta chiavi pubbliche 31
Inizializzazione del cluster 31
Panoramica di 32
Passaggio 1. Ottenere la CSR del cluster 32
Passaggio 2. Crea una chiave privata per la tua Root CA 34
Fase 3. Firma la CSR 35
Passaggio 4. Inizializzazione del cluster 45
Installazione della CLI di CloudHSM 47
Attivazione del cluster 52

Configurazione mTLS (consigliata) 55
Passaggio 1. Crea e registra un trust anchor sull'HSM 56
Passaggio 2. Abilita MTL per AWS CloudHSM 59
Fase 3. Imposta l'applicazione MTLS per AWS CloudHSM 65
Crea e usa le chiavi in AWS CloudHSM 66
Best practice 68
Gestione dei cluster 68
Scala il tuo cluster per gestire i picchi di traffico 68
Progetta il tuo cluster per garantire un'elevata disponibilità 68
Disponi di almeno tre HSM per garantire la durata delle chiavi appena generate 69
Accesso sicuro al tuo cluster 69
Riduci i costi adattandoli alle tue esigenze 69
Gestione degli utenti 70
Proteggi le credenziali dei tuoi utenti HSM 70
Disponi di almeno due amministratori per evitare il blocco 70
Abilita il quorum per tutte le operazioni di gestione degli utenti 70
Crea più utenti crittografici, ciascuno con autorizzazioni limitate 71
Gestione delle chiavi 71
Scegli il tipo di chiave giusto 71
Gestisci i limiti di archiviazione delle chiavi 71
Gestione e protezione del key wrapping 72
integrazione delle applicazioni 73
Avvia il tuo Client SDK 73
Effettua l'autenticazione per eseguire le operazioni 73
Gestisci in modo efficace le chiavi nell'applicazione 74
Usa il multithreading 75
Gestisci gli errori di limitazione 75
Integra i tentativi sulle operazioni del cluster 76
Implementa strategie di disaster recovery 76
Distribuzione graduale delle applicazioni 77
Monitoraggio 77
Monitora i log dei client 77
Monitora i registri di controllo 77
Monitoraggio AWS CloudTrail 78
Monitora le CloudWatch metriche di Amazon 78
Cluster 79

Architettura cluster 79
Sincronizzazione del cluster 81
Cluster a elevata disponibilità e sistema di bilanciamento del carico 82
modalità cluster 83
Tipi HSM 84
Connessione al cluster 86
Colloca il certificato di emissione su ogni istanza EC2 86
Specifica la posizione del certificato di emissione 86
Esegui il bootstrap di Client SDK 88
Ridimensionamento degli HSM 92
Aggiunta di un modulo HSM 93
Rimozione di un modulo HSM 95
Eliminazione di un cluster 96
Creazione di cluster dai backup 97
Crea cluster dai backup (console) 97
Crea cluster dai backup (AWS CLI) 98
Crea cluster dai backup (API)AWS CloudHSM 99
Migrazione dei tipi di cluster HSM 99
Migrazione da hsm1.medium a hsm2m.medium 100
Utenti HSM 109
Gestione degli utenti con CloudHSM CLI 109
Prerequisiti 110
Tipi di utente 111
Tabella delle autorizzazioni 113
Creare un amministratore 114
Creare un CU 115
Elencare tutti gli utenti 116
Modifica delle password 117
Eliminare gli utenti 119
Gestione dell'MFA degli utenti 120
Gestisci l'autenticazione del quorum (M of N) 135
Gestione degli utenti con CMU 160
Prerequisiti 161
Tipi di utente 165
Tabella delle autorizzazioni 166
Creazione di utenti 168

Elencare tutti gli utenti ... 171
Modifica delle password ... 172
Eliminare gli utenti ... 175
Gestisci l'autenticazione 2FA degli utenti ... 177
Utilizzo di CMU per gestire l'autenticazione del quorum ... 186
Chiavi ... 206
Sincronizzazione e durabilità delle chiavi ... 206
Concetti ... 207
Informazioni sulla sincronizzazione delle chiavi ... 208
Modifica le impostazioni di durabilità delle chiavi client ... 209
Sincronizzare le chiavi in cluster clonati ... 215
Wrapping della chiave AES ... 215
Algoritmi supportati ... 216
Utilizzo del key wrap in AES AWS CloudHSM ... 218
Chiavi attendibili ... 219
Informazioni sulle chiavi attendibili ... 219
Attributi delle chiavi attendibili ... 220
Come utilizzare le chiavi attendibili per eseguire il wrapping delle chiavi di dati ... 221
Come annullare il wrapping di una chiave di dati con una chiave attendibile ... 224
Gestione delle chiavi con CloudHSM CLI ... 224
Genera chiavi ... 225
Eliminazione delle chiavi ... 233
Condivisione e annullamento della condivisione delle chiavi ... 235
Filtra per chiavi ... 243
Contrassegna una chiave come affidabile ... 250
Gestisci l'autenticazione del quorum (M of N) ... 252
Gestione delle chiavi con KMU ... 272
Genera chiavi ... 273
Chiavi di importazione ... 275
Chiavi di esportazione ... 277
Eliminazione delle chiavi ... 279
Condivisione e annullamento della condivisione delle chiavi ... 279
Contrassegna una chiave come affidabile ... 280
Backup del cluster ... 282
Utilizzo dei backup ... 282
Rimozione delle chiavi scadute o degli utenti inattivi ... 283

Considerazioni sul ripristino di emergenza ... 283
Eliminare i backup ... 283
Ripristinare i backup ... 285
Configura la conservazione dei backup ... 286
Conservazione gestita dei backup ... 287
Copiare un backup tra regioni ... 290
Copia i backup in diverse regioni (console) ... 291
Copia i backup in diverse regioni (AWS CLI) ... 291
Copia i backup in diverse regioni (AWS CloudHSM API) ... 292
Utilizzo dei backup condivisi ... 292
Prerequisiti per la condivisione dei backup ... 293
Condivisione di un backup ... 293
Annullamento della condivisione di un backup condiviso ... 297
Identificazione di un backup condiviso ... 297
Autorizzazioni per i backup condivisi ... 298
Fatturazione e misurazione ... 298
Cluster clonati ... 299
Argomenti correlati ... 299
Aggiunta di tag alle risorse ... 300
Aggiungere o aggiornare i tag ... 300
Elenco dei tag ... 302
Rimuovere i tag ... 302
Strumenti a riga di comando ... 304
Strumento di Configurazione ... 305
Strumento di configurazione Client SDK 5 ... 306
Strumento di configurazione Client SDK 3 ... 339
CLI CloudHSM ... 348
Piattaforme supportate ... 349
Nozioni di base ... 350
Modalità di comando ... 358
Attributi chiave ... 360
Configurazioni avanzate. ... 368
Documentazione di riferimento ... 374
AWS CloudHSM Utilità di gestione ... 662
Piattaforme supportate ... 663
Nozioni di base ... 664

Installazione del client (Linux) 668
Installare il client (Windows) 671
Documentazione di riferimento 673
Utility di gestione delle chiavi 734
Nozioni di base 734
Installazione del client (Linux) 738
Installare il client (Windows) 741
Documentazione di riferimento 742
SDK del client 871
Verifica la tua versione 872
Confronta il supporto dei componenti 874
Libreria PKCS #11 874
Utility di gestione CloudHSM (CMU) 875
Utility di gestione delle chiavi (KMU) 875
Provider JCE 875
OpenSSL Dynamic Engine 875
Key Storage Provider (KSP) 876
Migrazione all'SDK più recente 876
Migra la libreria PKCS #11 877
Migrazione di OpenSSL Dynamic Engine 881
Migra Key Storage Provider (KSP) 885
Esegui la migrazione del provider JCE 892
Client SDK 5 905
Vantaggi dell'SDK più recente 906
Piattaforme supportate 907
Libreria PKCS #11 910
OpenSSL Dynamic Engine 967
Provider OpenSSL 975
Provider di archiviazione delle chiavi (KSP) 983
Provider JCE 1014
Versione precedente 1058
Aggiorna Client SDK 3 1059
Piattaforme supportate 1068
Libreria PKCS #11 1070
OpenSSL Dynamic Engine 1112
Provider JCE 1116

Provider KSP e CNG 1147
Integrazione di applicazioni di terze parti 1160
Offload SSL/TLS 1160
Come funziona 1161
Offload su Linux con OpenSSL Engine 1163
Offload su Linux con OpenSSL Provider 1211
Offload su Linux con JSSE 1247
Offload su Windows 1258
Aggiunta di un sistema di bilanciamento del carico (facoltativo) 1273
CA Windows Server 1280
Client SDK 5 con Windows Server CA 1281
Client SDK 3 con Windows Server CA 1286
Oracle Database Encryption 1292
Configurazione dei prerequisiti 1294
Fase 2: Aggiornare la configurazione del database Oracle 1295
Fase 3: Generazione della chiave di crittografia principale Oracle TDE 1296
Microsoft SignTool 1298
Client SDK 5 con Microsoft SignTool 1298
Client SDK 3 con Microsoft SignTool 1302
Java Keytool e Jarsigner 1307
Client SDK 5 con Java Keytool e Jarsigner 1307
Client SDK 3 con Java Keytool e Jarsigner 1322
Strumento di generazione e modifica di Microsoft Manifest 1338
Fase 1: configurazione dei prerequisiti 1338
Fase 2: Creare un certificato di firma 1339
Fase 3: Firma un file 1341
Altre integrazioni di fornitori di terze parti 1342
Monitoraggio 1344
Log del Client SDK 1344
Logging con Client SDK 5 1345
Logging con Client SDK 3 1346
AWS CloudTrail 1348
AWS CloudHSMinformazioni in CloudTrail 1349
ComprensioneAWS CloudHSMvoci del file di log 1350
AWS CloudHSMeventi di gestione in CloudTrail 1351
AWS CloudHSMeventi di servizio in CloudTrail 1351

Audit logs 1352
Funzionamento dei log 1353
Visualizzazione dei log di 1353
Interpretazione dei log 1356
Riferimento dei log 1371
CloudWatch metriche 1375
Performance 1377
Dati di prestazioni 1377
Limitazione HSM 1378
Sicurezza 1379
Controlla l'accesso alle API con le policy IAM 1380
Aggiornamento delle policy IAM a IPv6 1380
Protezione dei dati 1383
Crittografia dei dati a riposo 1384
Crittografia dei dati in transito 1385
End-to-end crittografia 1385
Backup del cluster 1386
Gestione dell'identità e degli accessi 1387
Concessione di autorizzazioni tramite le policy IAM 1388
Azioni API per AWS CloudHSM 1389
Chiavi di condizione per AWS CloudHSM 1390
Policy gestite da AWS predefinite per AWS CloudHSM 1390
Policy gestite dal cliente per AWS CloudHSM 1390
Service-linked ruoli 1394
Convalida della conformità 1396
PCI-PIN Domande frequenti 1398
Raggiunta obsolescenza 1399
Resilienza 1401
Sicurezza dell'infrastruttura 1401
Isolamento della rete 1402
Autorizzazione degli utenti 1402
Endpoint VPC (AWS PrivateLink) 1402
Considerazioni per AWS CloudHSM Endpoint VPC 1402
Creazione di un endpoint VPC di interfaccia per AWS CloudHSM 1403
Creazione di una policy sugli endpoint VPC per AWS CloudHSM 1403
Gestione degli aggiornamenti 1404

Risoluzione dei problemi 1405
AWS CloudHSM problemi noti 1405
Problemi noti per tutte le istanze HSM 1406
Problemi noti per hsm1.medium 1413
Problemi noti per hsm2m.medium 1413
Problemi noti per la libreria PKCS #11 1417
Problemi noti per l'SDK JCE 1425
Problemi noti per OpenSSL Dynamic Engine 1430
Problemi noti per il Key Storage Provider (KSP) 1433
Problemi noti per il provider OpenSSL 1435
Problemi noti per l'interfaccia a riga di comando di CloudHSM 1437
Problemi noti per le istanze Amazon EC2 che eseguono Amazon Linux 2 1438
Problemi noti per l'integrazione di applicazioni di terze parti 1438
Problemi noti relativi alla modifica dei cluster 1439
Problemi noti di errore operativo utilizzando AWS CloudHSM la versione client 5.12.0 su hsm2.medium 1440
Errori di sincronizzazione delle chiavi in Client SDK 3 1441
Client SDK 3 verifica le prestazioni 1442
Consigli sui test 1443
Opzioni configurabili per lo strumento pkpspeed 1444
Test che possono essere eseguiti con lo strumento pkpspeed 1444
Esempi 1445
L'utente o la policy di Client SDK 5 contengono valori non coerenti 1448
Risolvi i valori utente incoerenti 1449
Risolvi i valori delle policy MTL non coerenti 1456
Errori di replica degli utenti di Client SDK 5 1458
Problema: l'utente selezionato non è sincronizzato in tutto il cluster 1458
Problema: Nel cluster di destinazione esiste un utente con attributi diversi 1459
Problema: la replica utente da hsm2m.medium a hsm1.medium non riesce 1460
Errori di replica chiave di Client SDK 5 1460
Problema: la chiave selezionata non è sincronizzata in tutto il cluster 1460
Problema: una chiave con lo stesso riferimento esiste nel cluster di destinazione con informazioni o attributi diversi 1462
AWS CloudHSM errore rilevato durante il controllo della disponibilità delle chiavi 1463
Estrazione di chiavi tramite JCE 1464
getEncoded o getS getPrivateExponent restituisce null 1464

getEncoded getPrivateExponent o GETS restituiscono byte della chiave al di fuori dell'HSM 1464
Limitazione HSM 1464
Risoluzione 1465
Connessione persa 1466
Log AWS CloudHSM di controllo mancanti CloudWatch 1469
Non-compliant Avvolgimenti chiave AES 1470
Determina se il codice genera chiavi con wrapping irrecuperabili 1470
Azioni da intraprendere se il codice genera chiavi con wrapping irrecuperabili 1472
Risoluzione degli errori di creazione dei AWS CloudHSM cluster 1473
Aggiungere l'autorizzazione mancante 1473
Creare manualmente il ruolo legato al servizio 1474
Utilizza un utente non federato 1474
Recupero dei log di configurazione AWS CloudHSM del client 1475
Strumento di supporto per Client SDK 5 1475
Strumento di supporto per Client SDK 3 1477
Quote 1479
Download 1481
Ultima versione 1481
Versione Client SDK 5: versione 5.18.0 1482
Versione precedente 1490
Versioni deprecate 1543
Versioni obsolete di Client SDK 5 1543
Versioni obsolete di Client SDK 3 1558
End-of-life rilasci 1568
Cronologia dei documenti 1569
Aggiornamenti recenti 1569
Aggiornamenti precedenti 1576
..... mdlxxvii

Che cos'è AWS CloudHSM?

AWS CloudHSM combina i vantaggi del AWS cloud con la sicurezza dei moduli di sicurezza hardware (HSMs). Un modulo di sicurezza hardware (HSM) è un dispositivo che elabora le operazioni di crittografia e offre uno storage sicuro per le chiavi di crittografia. Con AWS CloudHSM, hai il controllo completo sull'alta disponibilità presente HSMs nel cloud AWS, hai accesso a bassa latenza e un root of trust sicuro che automatizza la gestione HSM (inclusi backup, provisioning, configurazione e manutenzione).

AWS CloudHSM offre ai clienti una serie di vantaggi:

Accesso a cluster FIPS e non FIPS

> AWS CloudHSM offre cluster in due modalità: FIPS e non FIPS. In modalità FIPS, è possibile utilizzare solo chiavi e algoritmi convalidati dal Federal Information Processing Standard (FIPS). La modalità non FIPS offre tutte le chiavi e gli algoritmi supportati da, indipendentemente dall'approvazione FIPS. AWS CloudHSM Per ulteriori informazioni, consulta AWS CloudHSM modalità cluster.

HSMs sono generici, a tenant singolo e sono convalidati FIPS 140-2 livello 3 o FIPS 140-3 livello 3 per i cluster in modalità FIPS

> AWS CloudHSM utilizza scopi generici HSMs che offrono maggiore flessibilità rispetto ai servizi AWS completamente gestiti che hanno algoritmi e lunghezze chiave predeterminati per l'applicazione. Offriamo HSMs prodotti conformi agli standard, single-tenant e convalidati FIPS 140-2 livello 3 o FIPS 140-3 livello 3 per i cluster in modalità FIPS. Per i clienti con casi d'uso che non rientrano nelle restrizioni della convalida FIPS 140-2 o FIPS 140-3 livello 3, offre anche cluster in modalità non FIPS. AWS CloudHSM Per ulteriori informazioni, consulta AWS CloudHSM grappoli.

La crittografia E2E non è visibile per AWS

> Poiché il piano dati è crittografato end-to-end (E2E) e non è visibile ad AWS, puoi controllare la tua gestione degli utenti (al di fuori dei ruoli IAM). Il compromesso per il controllo è una maggiore responsabilità rispetto a quando si utilizza un servizio AWS gestito.

Controllo completo delle chiavi, degli algoritmi e dello sviluppo di applicazioni

> AWS CloudHSM ti dà il pieno controllo degli algoritmi e delle chiavi che usi. È possibile creare, archiviare, importare, esportare, gestire e utilizzare le chiavi crittografiche, tra cui chiavi di

sessione, chiavi token, chiavi simmetriche e coppie di chiavi asimmetriche. Inoltre, AWS CloudHSM SDKs ti offre il pieno controllo sullo sviluppo delle applicazioni, sul linguaggio dell'applicazione, sul threading e sulla posizione fisica delle applicazioni.

Esegui la migrazione dei carichi di lavoro di crittografia nel cloud

I clienti che migrano un'infrastruttura a chiave pubblica che utilizzano Public Key Cryptography Standards #11 (PKCS #11), Java Cryptographic Extension (JCE), Cryptography API: Next Generation (CNG) o Key Storage Provider (KSP) possono effettuare la migrazione con meno modifiche alla propria applicazione. AWS CloudHSM

Per ulteriori informazioni su cosa è possibile utilizzare, consulta i seguenti argomenti. AWS CloudHSM Quando sei pronto per iniziare AWS CloudHSM, consultaNozioni di base.

> **Note**
>
> Se desideri un servizio gestito per la creazione e il controllo delle chiavi di crittografia ma non vuoi o non devi gestirlo da solo HSMs, prendi in considerazione l'utilizzo AWS Key Management Service.
> Se stai cercando un servizio elastico che gestisca pagamenti HSMs e chiavi per le applicazioni di elaborazione dei pagamenti nel cloud, prendi in considerazione l'utilizzo di AWS Payment Cryptography.

Indice

- AWS CloudHSM casi d'uso
- Come AWS CloudHSM funziona
- Prezzi per AWS CloudHSM

AWS CloudHSM casi d'uso

AWS CloudHSM può essere usato per raggiungere una varietà di obiettivi. Il contenuto di questo argomento fornisce una panoramica di ciò che è possibile utilizzare. AWS CloudHSM

Raggiungi la conformità alle normative

Le aziende che devono allinearsi agli standard di sicurezza aziendali possono AWS CloudHSM utilizzare la gestione di chiavi private che proteggono dati altamente riservati. I prodotti HSMs

forniti da AWS CloudHSM sono certificati FIPS 140-2 livello 3 e sono conformi allo standard PCI DSS. Inoltre, AWS CloudHSM è conforme a PCI PIN e PCI-3DS. Per ulteriori informazioni, consulta Convalida della conformità per AWS CloudHSM.

Crittografia e decrittografia dei dati

Viene utilizzato AWS CloudHSM per gestire le chiavi private che proteggono i dati altamente riservati, la crittografia in transito e la crittografia inattiva. Inoltre, AWS CloudHSM offre un'integrazione conforme agli standard con più sistemi crittografici. SDKs

Firma e verifica di documenti con chiavi private e pubbliche

Nella crittografia, l'utilizzo di una chiave privata per firmare un documento consente ai destinatari di utilizzare una chiave pubblica per verificare che il documento sia stato effettivamente inviato dalla persona in questione (e non da qualcun altro). Viene utilizzato AWS CloudHSM per creare coppie di chiavi pubbliche e private asimmetriche progettate specificamente per questo scopo.

Autentica i messaggi utilizzando e HMACs CMACs

Nella crittografia, i codici di autenticazione dei messaggi cipher (CMACs) e i codici di autenticazione dei messaggi basati su hash (HMACs) vengono utilizzati per autenticare e garantire l'integrità dei messaggi inviati su reti non sicure. Con AWS CloudHSM, puoi creare e gestire in modo sicuro chiavi simmetriche che supportano e. HMACs CMACs

Sfrutta i vantaggi di e AWS CloudHSM AWS Key Management Service

I clienti possono combinare AWS CloudHSM e AWS KMSarchiviare il materiale chiave in un ambiente single-tenant, ottenendo al contempo i principali vantaggi di gestione, scalabilità e integrazione cloud di. AWS KMS Per ulteriori informazioni al riguardo, consulta gli archivi delle chiavi AWS CloudHSM nella Guida per gli sviluppatori di AWS Key Management Service .

Elaborazione dell'offload SSL/TLS per server Web

Per inviare dati in modo sicuro su Internet, i server Web utilizzano coppie di chiavi pubbliche-private e certificati a chiave SSL/TLS pubblica per stabilire sessioni HTTPS. Questo processo comporta molte operazioni di calcolo per i server Web, ma è possibile ridurre il carico di calcolo fornendo al contempo una maggiore sicurezza trasferendo parte di questa operazione al cluster.

AWS CloudHSM Per informazioni sulla configurazione di SSL/TLS offload con, vedere. AWS CloudHSMOffload SSL/TLS

Abilitazione di Transparent Data Encryption (TDE)

Transparent Data Encryption (TDE) viene utilizzato per crittografare i file di database. Utilizzando TDE, il software del database crittografa i dati prima di archiviarli su disco. È possibile ottenere una maggiore sicurezza memorizzando la chiave di crittografia principale TDE HSMs nel proprio. AWS CloudHSM Per informazioni sulla configurazione di Oracle TDE con AWS CloudHSM, vedere. Oracle Database Encryption

Gestire le chiavi private di un'autorità di certificazione emittente (CA)

Un'autorità di certificazione (CA) è un'entità attendibile che emette certificati digitali che associano una chiave pubblica a un'identità (una persona o un'organizzazione). Per gestire una CA, è necessario mantenere l'attendibilità proteggendo la chiave privata che firma i certificati emessi dalla CA. È possibile memorizzare tali chiavi private nel AWS CloudHSM cluster e quindi utilizzarle HSMs per eseguire operazioni di firma crittografica.

Generare numeri casuali

La generazione di numeri casuali per creare chiavi di crittografia è fondamentale per la sicurezza online. AWS CloudHSM possono essere utilizzati per generare in modo sicuro numeri casuali sotto il HSMs tuo controllo e sono visibili solo a te.

Come AWS CloudHSM funziona

Questo argomento fornisce una panoramica dei concetti e dell'architettura di base utilizzati per crittografare in modo sicuro i dati ed eseguire operazioni crittografiche. HSMs AWS CloudHSM opera nel tuo Amazon Virtual Private Cloud (VPC). Prima di poterlo utilizzare AWS CloudHSM, devi prima creare un cluster, HSMs aggiungerlo, creare utenti e chiavi e quindi utilizzare Client SDKs per integrarlo HSMs con la tua applicazione. Fatto ciò, usi i log di Client SDK AWS CloudTrail, i log di controllo e Amazon CloudWatch per il monitoraggio. AWS CloudHSM

Scopri i concetti AWS CloudHSM di base e come interagiscono per proteggere i tuoi dati.

Argomenti

- AWS CloudHSM grappoli
- Utenti in AWS CloudHSM
- Chiavi in AWS CloudHSM
- Cliente SDKs per AWS CloudHSM
- AWS CloudHSM backup dei cluster
- Regioni supportate per AWS CloudHSM

AWS CloudHSM grappoli

Far HSMs lavorare insieme le persone in modo sincronizzato, ridondante e ad alta disponibilità può essere difficile, ma AWS CloudHSM fornisce moduli di sicurezza hardware () in cluster per voi rappresenta un grosso problema. HSMs Un cluster è un insieme di individui che si mantengono sincronizzati. HSMs AWS CloudHSM Quando si esegue un'attività o un'operazione su un HSM in un cluster, gli altri HSMs componenti del cluster vengono aggiornati automaticamente.

AWS CloudHSM offre cluster in due modalità: FIPS e non FIPS. In modalità FIPS, è possibile utilizzare solo chiavi e algoritmi convalidati dal Federal Information Processing Standard (FIPS). La modalità non FIPS offre tutte le chiavi e gli algoritmi supportati da, indipendentemente dall'approvazione FIPS. AWS CloudHSM AWS CloudHSM offre anche due tipi di HSMs: hsm1.medium e hsm2m.medium. Per i dettagli sulle differenze tra ogni tipo di HSM e la modalità cluster, vedere. AWS CloudHSM modalità cluster Il tipo HSM hsm1.medium sta raggiungendo la fine del supporto, quindi non è possibile creare nuovi cluster con questo tipo. Per ulteriori informazioni, consulta Notifiche di deprecazione per i dettagli.

Per raggiungere gli obiettivi di disponibilità, durabilità e scalabilità, è necessario impostare il numero di componenti del cluster HSMs in più zone di disponibilità. Puoi creare un cluster da 1 a 28 HSMs (il limite predefinito è 6 HSMs per AWS account per AWS regione). Puoi posizionarlo HSMs in diverse zone di disponibilità di una AWS regione. L'aggiunta di altri HSMs elementi a un cluster offre prestazioni più elevate. La diffusione di cluster tra le zone di disponibilità fornisce ridondanza e disponibilità elevate.

Per ulteriori informazioni sui cluster, consulta Cluster in AWS CloudHSM.

Per creare un cluster, consulta Nozioni di base.

Utenti in AWS CloudHSM

A differenza della maggior parte dei AWS servizi e delle risorse, non si utilizzano utenti AWS Identity and Access Management (IAM) o policy IAM per accedere alle risorse all'interno AWS CloudHSM del cluster. Al contrario, si utilizzano gli utenti HSM direttamente HSMs nel AWS CloudHSM cluster.

Gli utenti HSM sono diversi dagli utenti IAM. Gli utenti IAM che dispongono delle credenziali corrette possono creare HSMs interagendo con le risorse tramite l'API AWS. Poiché la crittografia E2E non è visibile per AWS, è necessario utilizzare credenziali utente HSM per autenticare le operazioni nell'HSM poiché le credenziali vengono impiegate direttamente nell'HSM. L'HSM autentica ogni utente HSM tramite credenziali definite e gestite da te. Ogni utente HSM dispone di un tipo che stabilisce quali operazioni può eseguire nell'HSM. Ogni HSM autentica ogni utente HSM tramite credenziali definite utilizzando la CLI di CloudHSM.

Se utilizzi la serie di versioni SDK precedente, utilizzerai CloudHSM Management Utility (CMU).

Chiavi in AWS CloudHSM

AWS CloudHSM consente di generare, archiviare e gestire in modo sicuro le chiavi di crittografia in single-tenant che si trovano HSMs nel cluster. AWS CloudHSM Le chiavi possono essere simmetriche o asimmetriche, possono essere chiavi di sessione (chiavi temporanee) per sessioni singole, chiavi token (chiavi persistenti) per uso a lungo termine e possono essere esportate e importate in AWS CloudHSM . Le chiavi possono essere utilizzate anche per completare attività e funzioni crittografiche comuni:

- Firmare dati crittografici ed eseguire la verifica della firma con algoritmi di crittografia simmetrici e asimmetrici.
- Utilizza le funzioni hash per calcolare i digest dei messaggi e i codici di autenticazione dei messaggi basati su hash (). HMACs
- Eseguire il wrapping di altre chiavi e proteggerle.
- Accedere a dati casuali protetti da crittografia.

Il numero massimo di chiavi che un cluster può avere dipende dal tipo di chiavi HSMs presenti nel cluster. Ad esempio, hsm2m.medium memorizza più chiavi di hsm1, medium. Per un confronto, vedi. AWS CloudHSM quote

Inoltre, AWS CloudHSM segue alcuni principi fondamentali per l'utilizzo e la gestione delle chiavi:

Molti tipi di chiavi e algoritmi tra cui scegliere

Per consentire la personalizzazione delle proprie soluzioni, AWS CloudHSM offre molti tipi di chiavi e algoritmi tra cui scegliere. Gli algoritmi supportano una vasta gamma di dimensioni di chiavi. Per ulteriori informazioni, consulta le pagine relative agli attributi e ai meccanismi di ogni Offload delle operazioni con AWS CloudHSM SDK del client.

Come gestire le chiavi

AWS CloudHSM le chiavi sono gestite tramite strumenti SDKs a riga di comando. Per ulteriori informazioni su come utilizzare questi strumenti per gestire le chiavi, consulta le pagine Chiavi in AWS CloudHSM e Le migliori pratiche per AWS CloudHSM.

Chi possiede le chiavi

Nel AWS CloudHSM, l'utente crittografico (CU) che crea la chiave la possiede. Il proprietario può utilizzare i key unshare comandi key share and per condividere e annullare la condivisione della chiave con altri. CUs Per ulteriori informazioni, consulta Condividi e annulla la condivisione delle chiavi utilizzando la CLI di CloudhSM.

L'accesso e l'utilizzo possono essere controllati con la crittografia basata su attributi

AWS CloudHSM consente di utilizzare la crittografia basata sugli attributi, una forma di crittografia che consente di utilizzare gli attributi chiave per controllare chi può decrittografare i dati in base alle politiche.

Cliente SDKs per AWS CloudHSM

Durante l'utilizzo AWS CloudHSM, si eseguono operazioni crittografiche con AWS CloudHSM Client Software Development Kits (). SDKs AWS CloudHSM Il client include SDKs :

- Public Key Cryptography Standard #11 (PKCS #11)
- Provider JCE
- OpenSSL Dynamic Engine
- Key Storage Provider (KSP) per Microsoft Windows

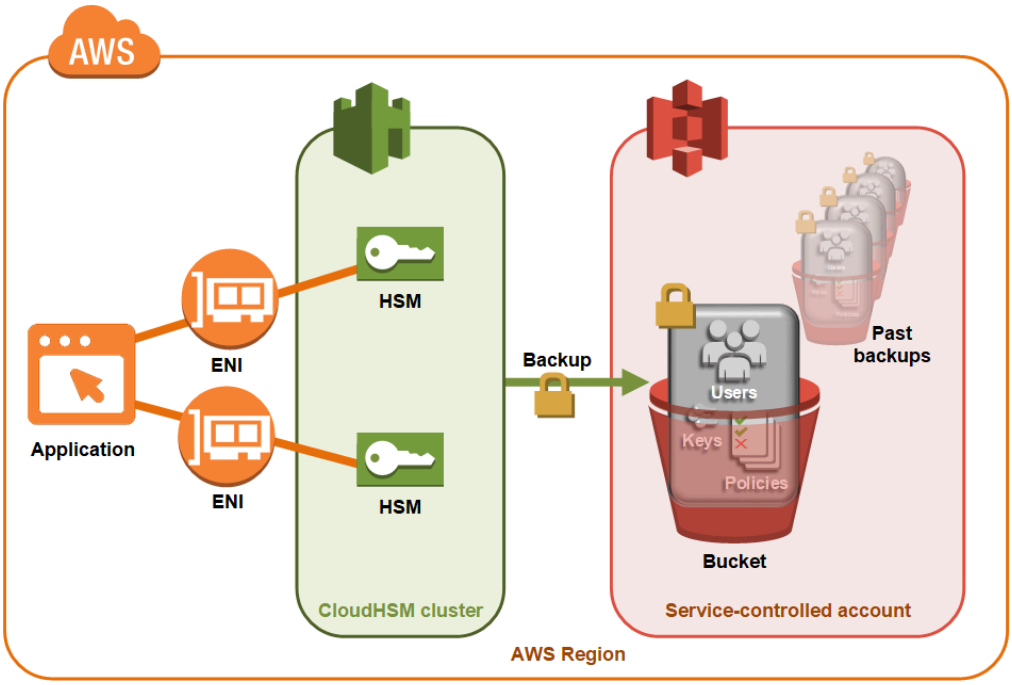
Puoi utilizzare uno o tutti questi SDK nel tuo AWS CloudHSM cluster. Scrivi il codice dell'applicazione per utilizzarli SDKs per eseguire operazioni crittografiche nel tuo. HSMs Per scoprire quali piattaforme e tipi di HSM supportano ciascun SDK, consulta AWS CloudHSM Piattaforme supportate da Client SDK 5

Gli strumenti di utilità e da riga di comando sono necessari non solo per l'uso, SDKs ma anche per configurare le credenziali, le politiche e le impostazioni dell'applicazione. Per ulteriori informazioni, vedi AWS CloudHSM strumenti da riga di comando.

Per ulteriori informazioni sull'installazione e sull'utilizzo dell'SDK del client o sulla sicurezza della connessione del client, consulta le pagine SDK del client e End-to-end crittografia.

AWS CloudHSM backup dei cluster

AWS CloudHSM esegue backup periodici degli utenti, delle chiavi e delle politiche del cluster. I backup sono sicuri, durevoli e aggiornati secondo una pianificazione prevedibile. La figura seguente mostra la relazione tra i backup e il cluster.



Per ulteriori informazioni sull'utilizzo dei backup, consulta Backup del cluster.

Sicurezza

> Quando AWS CloudHSM esegue un backup dall'HSM, quest'ultimo crittografa tutti i dati prima di inviarli a. AWS CloudHSM I dati non lasciano mai l'HSM sotto forma di testo normale. Inoltre,

i backup non possono essere decrittografati AWS perché AWS non ha accesso alla chiave utilizzata per decrittografare i backup. Per ulteriori informazioni, consulta Sicurezza dei backup dei cluster

Durabilità

AWS CloudHSM archivia i backup in un bucket Amazon Simple Storage Service (Amazon S3) controllato dal servizio nella stessa regione del cluster. I backup hanno un livello di durabilità del 99,999999999%, come qualsiasi oggetto archiviato in Amazon S3.

Regioni supportate per AWS CloudHSM

Per informazioni sulle regioni supportate per AWS CloudHSM, consulta AWS CloudHSM Regioni ed endpoint nella Riferimenti generali di AWSo nella tabella delle regioni.

AWS CloudHSM potrebbe non essere disponibile in tutte le zone di disponibilità di una determinata regione. Tuttavia, ciò non dovrebbe influire sulle prestazioni, poiché il carico AWS CloudHSM viene automaticamente bilanciato su tutti i componenti HSMs di un cluster.

Come la maggior parte AWS delle risorse, i cluster e HSMs sono risorse regionali. Non puoi riutilizzare o estendere un cluster tra più regioni. Per creare un cluster in una nuova regione, devi eseguire la procedura indicata in Guida introduttiva con AWS CloudHSM.

Ai fini del disaster recovery, AWS CloudHSM consente di copiare i backup del AWS CloudHSM cluster da una regione all'altra. Per ulteriori informazioni, consulta AWS CloudHSM backup dei cluster.

Prezzi per AWS CloudHSM

Con AWS CloudHSM, paghi a ore senza impegni a lungo termine o pagamenti anticipati. Per ulteriori informazioni, consulta la sezione AWS CloudHSM Prezzi sul AWS sito web.

Guida introduttiva con AWS CloudHSM

I seguenti argomenti aiutano a creare, inizializzare e attivare un cluster in. AWS CloudHSM Dopo aver completato queste procedure, sarai pronto a gestire gli utenti e i cluster, nonché a eseguire operazioni di crittografia utilizzando le librerie software in dotazione. Per un'esperienza ottimale, segui gli argomenti nell'ordine elencato.

Indice

- Crea gruppi amministrativi IAM per AWS CloudHSM
- Crea un cloud privato virtuale (VPC) per AWS CloudHSM
- Crea un cluster in AWS CloudHSM
- Esamina il gruppo di sicurezza per il tuo cluster in AWS CloudHSM
- Avvia un'istanza client Amazon EC2 con cui interagire AWS CloudHSM
- Configurazione dei gruppi di sicurezza delle istanze Client Amazon EC2 per AWS CloudHSM
- Crea un HSM in AWS CloudHSM
- Verifica l'identità e l'autenticità dell'HSM del tuo cluster in AWS CloudHSM (facoltativo)
- Inizializza il cluster in AWS CloudHSM
- Installa e configura la CLI di CloudHSM
- Attiva il cluster in AWS CloudHSM
- Configura TLS reciproco tra client e AWS CloudHSM (consigliato)
- Crea e usa le chiavi in AWS CloudHSM

Crea gruppi amministrativi IAM per AWS CloudHSM

Il primo passo per iniziare AWS CloudHSM è configurare le autorizzazioni IAM.

Come best practice, non utilizzare le tue Utente root dell'account AWS per interagire con AWS, ad esempio AWS CloudHSM. Utilizza invece AWS Identity and Access Management (IAM) per creare un utente IAM, un ruolo IAM o un utente federato. Segui i passaggi indicati nella sezione Creazione di un gruppo di amministratori e di un utente IAM; per creare un gruppo di amministratori e allegare ad esso la AdministratorAccess policy. Quindi, crea un nuovo utente amministratore e aggiungilo al gruppo. Aggiungi altri utenti al gruppo, se necessario. Ogni utente aggiunto eredita la AdministratorAccess policy dal gruppo.

Un'altra procedura consigliata consiste nel creare un gruppo di AWS CloudHSM amministratori che disponga solo delle autorizzazioni necessarie per l'esecuzione. AWS CloudHSM Aggiungi utenti singoli a questo gruppo, se necessario. Ogni utente eredita le autorizzazioni limitate collegate al gruppo anziché l'accesso completo ad AWS . La Policy gestite dal cliente per AWS CloudHSM sezione che segue contiene la politica da allegare al gruppo di AWS CloudHSM amministratori.

AWS CloudHSM definisce un ruolo collegato al servizio per il tuo account. AWS Il ruolo collegato al servizio attualmente definisce le autorizzazioni che consentono all'account di registrare gli eventi. AWS CloudHSM Il ruolo può essere creato automaticamente AWS CloudHSM o manualmente dall'utente. Non è possibile modificare il ruolo, ma è possibile eliminarlo. Per ulteriori informazioni, vedi Service-linked ruoli per AWS CloudHSM.

Creazione di un gruppo di amministratori e di un utente IAM;

Registrati per un Account AWS

Per iniziare AWS, hai bisogno di un Account AWS. Per informazioni sulla creazione di un Account AWS, vedere Guida introduttiva a Account AWS nella Guida Gestione dell'account AWS di riferimento.

Ad esempio, le policy AWS CloudHSM che puoi allegare al tuo gruppo di utenti IAM, vediGestione delle identità e degli accessi per AWS CloudHSM.

Crea un cloud privato virtuale (VPC) per AWS CloudHSM

È necessario un cloud privato virtuale (VPC) per il cluster. AWS CloudHSM Se non ne hai già uno, segui i passaggi descritti in questo argomento per creare un VPC.

> **Note**
>
> Seguendo questi passaggi creerai sottoreti pubbliche e private.

Per creare un VPC

1. Apri la console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Sulla barra di navigazione, utilizza il selettore di regione per scegliere una delle AWS regioni in cui AWS CloudHSM è attualmente supportata.

3. Seleziona il pulsante Crea VPC.
4. Per Risorse da creare, scegli VPC e altro.
5. Per la generazione automatica del tag nome, digita un nome identificabile, ad esempio **`CloudHSM`**.
6. Per il blocco IPv6 CIDR, seleziona il blocco IPv6 CIDR fornito da Amazon per utilizzare la IPv6 connettività per il tuo HSMs e fai AWS allocare un blocco IPv6 CIDR per il tuo cluster. Questa impostazione supporta il tipo di rete dual-stack. Mantieni l'impostazione predefinita se non hai bisogno di connettività. IPv6
7. Lascia tutte le altre opzioni impostate sui valori predefiniti.
8. Seleziona Crea VPC.
9. Dopo aver creato il VPC, seleziona Visualizza VPC per visualizzare il VPC appena creato.

Crea un cluster in AWS CloudHSM

Un cluster è una raccolta di singoli moduli di sicurezza hardware (HSMs). AWS CloudHSM li sincronizza HSMs in ogni cluster in modo che funzionino come unità logica. AWS CloudHSM offre due tipi di HSMs: hsm1.medium e hsm2m.medium. Quando crei un cluster, scegli quale dei due farà parte del tuo cluster. Per i dettagli sulle differenze tra ogni tipo di HSM e la modalità cluster, consultaAWS CloudHSM modalità cluster.

Quando crei un cluster, AWS CloudHSM crea un gruppo di sicurezza per il cluster per tuo conto. Questo gruppo di sicurezza controlla l'accesso alla rete all' HSMs interno del cluster. Consente connessioni in entrata solo da istanze Amazon Elastic Compute Cloud EC2 (Amazon) che fanno parte del gruppo di sicurezza. Per impostazione predefinita, il gruppo di sicurezza non contiene istanze. In seguito, è possibile avviare un'istanza del client e configurare il gruppo di sicurezza del cluster per consentire la comunicazione e le connessioni con HSM.

Considerazioni

- Di seguito sono riportate alcune considerazioni relative alla creazione di un cluster in: AWS CloudHSM
  - Quando si AWS CloudHSM crea un cluster, viene creato un ruolo collegato al servizio denominato HSM. AWSService RoleForCloud Se AWS CloudHSM non riesci a creare il ruolo o il ruolo non esiste già, potresti non essere in grado di creare un cluster. Per ulteriori informazioni, consulta Risoluzione degli errori di creazione dei AWS CloudHSM cluster. Per ulteriori informazioni sui ruoli legati al servizio, vedi Service-linked ruoli per AWS CloudHSM.

- Se stai usando l'endpoint AWS CloudHSM dual-stack (ovvero cloudhsmv2). *<region>*.api.aws), assicurati che le tue policy IAM siano aggiornate per essere gestite. IPv6 Per ulteriori informazioni, consulta la sezione Aggiornamento delle politiche IAM alla IPv6 sezione Sicurezza.

Puoi creare un cluster dalla console AWS CloudHSM, da AWS Command Line Interface (AWS CLI) o dall'API AWS CloudHSM .

Per i dettagli sugli argomenti del cluster e APIs, consulta create-clusterla sezione AWS CLI Command Reference.

Console

Per creare un cluster (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Sulla barra di navigazione, utilizza lo strumento di selezione della regione per scegliere una delle regioni AWS in cui AWS CloudHSM è attualmente supportato.
3. Scegli Crea cluster.
4. Nella sezione Configurazione del cluster, procedi come segue:
   a. In VPC, seleziona il VPC che hai creato in Crea un cloud privato virtuale (VPC) per AWS CloudHSM.
   b. In Zone di disponibilità (AZ), a fianco di ciascuna zona di disponibilità, scegli la sottorete privata creata.

      > **Note**
      >
      > Anche se non AWS CloudHSM è supportata in una determinata zona di disponibilità, le prestazioni non dovrebbero risentirne, HSMs in quanto il carico AWS CloudHSM viene automaticamente bilanciato tra tutti i componenti di un cluster. Vedi AWS CloudHSM Regioni ed endpoint nella sezione Riferimenti generali di AWSper vedere il supporto per le zone di disponibilità. AWS CloudHSM

   c. Per il tipo di HSM, seleziona il tipo di HSM che può essere creato nel cluster insieme alla modalità desiderata del cluster. Per scoprire quali tipi di HSM sono supportati in ogni regione, consulta il calcolatore dei AWS CloudHSM prezzi.

> **Important**
>
> Dopo la creazione del cluster, la modalità cluster non può essere modificata. Per informazioni sul tipo e sulla modalità più adatti al tuo caso d'uso, consultaAWS CloudHSM modalità cluster.

d. Per Tipo di rete, scegli i protocolli di indirizzi IP per accedere ai tuoi HSMs. IPv4 limita la comunicazione tra l'applicazione e IPv4 solo HSMs verso. Questa è l'opzione predefinita. Il dual-stack consente sia IPv4 la comunicazione che la comunicazione. IPv6 Per utilizzare il dual-stack, aggiungi entrambi IPv4 e alle configurazioni IPv6 CIDRs VPC e subnet. Il tipo di rete è difficile da modificare dopo la configurazione iniziale. Per modificarlo, crea un backup del cluster esistente e ripristina un nuovo cluster con il tipo di rete desiderato. Per ulteriori informazioni, consulta Creazione di AWS CloudHSM cluster dai backup

e. Per Cluster source, specifica se desideri creare un nuovo cluster o ripristinarne uno da un backup esistente.

- I backup di cluster in modalità non FIPS possono essere utilizzati solo per ripristinare cluster in modalità non FIPS.
- I backup dei cluster in modalità FIPS possono essere utilizzati solo per ripristinare i cluster in modalità FIPS.

5. Scegli Next (Successivo).
6. Specifica per quanto tempo il servizio deve conservare i backup.

   - Accetta il periodo di conservazione predefinito di 90 giorni o digita un nuovo valore tra 7 e 379 giorni. Il servizio eliminerà automaticamente i backup in questo cluster più vecchi del valore specificato qui. Puoi modificare questa impostazione in un secondo momento. Per ulteriori informazioni, vedi Configura la conservazione dei backup.

7. Scegli Next (Successivo).
8. (Facoltativo) Digita tag per una chiave di un valore di tag facoltativo. Per aggiungere più di un tag al cluster scegli Aggiungi tag.
9. Scegli Rivedi.
10. Rivedi la configurazione del cluster, quindi scegli Crea cluster.

Se i tentativi di creare un cluster falliscono, è possibile che ciò sia dovuto a problemi con i ruoli collegati ai servizi. AWS CloudHSM Per assistenza nella risoluzione del problema, vedi Risoluzione degli errori di creazione dei AWS CloudHSM cluster.

AWS CLI

Per creare un cluster (AWS CLI)

- Al prompt dei comandi, esegui il comando create-cluster. Specificate il tipo di istanza HSM, il periodo di conservazione dei backup e la sottorete IDs delle sottoreti da creare. HSMs Utilizzate la sottorete IDs delle sottoreti private che avete creato. Specifica una sola sottorete per ogni zona di disponibilità.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
                    --backup-retention-policy Type=DAYS,Value=<number of days> \
                    --subnet-ids <subnet ID> \
                    --mode <FIPS> \
                    --network-type <IPV4>

{
    "Cluster": {
        "BackupPolicy": "DEFAULT",
        "BackupRetentionPolicy": {
            "Type": "DAYS",
            "Value": 90
         },
        "VpcId": "vpc-50ae0636",
        "SubnetMapping": {
            "us-west-2b": "subnet-49a1bc00",
            "us-west-2c": "subnet-6f950334",
            "us-west-2a": "subnet-fd54af9b"
        },
        "SecurityGroup": "sg-6cb2c216",
        "HsmType": "hsm2m.medium",
        "NetworkType": "IPV4",
        "Certificates": {},
        "State": "CREATE_IN_PROGRESS",
        "Hsms": [],
        "ClusterId": "cluster-igklspoyj5v",
        "ClusterMode": "FIPS",
        "CreateTimestamp": 1502423370.069
    }
```

```
}
```

> **Note**
>
> `ClusterMode`è un parametro obbligatorio per tutti i tipi di hsm tranne hsm1.medium. `--mode`:
>
> ```
> $ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
>       --backup-retention-policy Type=DAYS,Value=<number of days> \
>       --subnet-ids <subnet ID> \
>     --mode NON_FIPS
> ```

Se i tentativi di creare un cluster falliscono, è possibile che ciò sia dovuto a problemi con i ruoli AWS CloudHSM collegati ai servizi. Per assistenza nella risoluzione del problema, vedi Risoluzione degli errori di creazione dei AWS CloudHSM cluster.

AWS CloudHSM API

Per creare un cluster (API)AWS CloudHSM

- Inviare una richiesta CreateCluster. Specificate il tipo di istanza HSM, la politica di conservazione dei backup e la sottorete IDs delle sottoreti da creare. HSMs Utilizzate la sottorete IDs delle sottoreti private che avete creato. Specifica una sola sottorete per ogni zona di disponibilità.

Se i tentativi di creare un cluster falliscono, è possibile che ciò sia dovuto a problemi con i ruoli collegati ai servizi. AWS CloudHSM Per assistenza nella risoluzione del problema, vedi Risoluzione degli errori di creazione dei AWS CloudHSM cluster.

Esamina il gruppo di sicurezza per il tuo cluster in AWS CloudHSM

Quando crei un cluster o aggiungi un HSM a un cluster, AWS CloudHSM crea un gruppo di sicurezza con lo stesso nome `cloudhsm-cluster-<clusterID>-sg` se non ne esiste già uno. Questo gruppo di sicurezza contiene una regola TCP preconfigurata che consente la comunicazione in entrata e in uscita all'interno del gruppo di sicurezza del cluster su porte 2223-2225. Questo SG consente alle istanze EC2 di utilizzare il VPC con cui comunicare all'interno del cluster. HSMs

⚠ Warning

- Non eliminare o modificare la regola TCP preconfigurata, che viene inserita in tale gruppo di sicurezza. Questa regola può prevenire problemi di connettività e accessi non autorizzati al tuo. HSMs
- Il gruppo di sicurezza del cluster impedisce l'accesso non autorizzato al tuo. HSMs Chiunque possa accedere alle istanze del gruppo di sicurezza può accedere alle tue. HSMs La maggior parte delle operazioni richiede un utente per l'accesso all'HSM. Tuttavia, è possibile effettuare l'azzeramento HSMs senza autenticazione, il che distrugge il materiale chiave, i certificati e gli altri dati. In questo caso, i dati creati o modificati dopo il backup più recente vengono persi e non possono essere più recuperati. Prevenire gli accessi non autorizzati, in modo che solo gli amministratori attendibili possano modificare o accedere alle istanze nel gruppo di sicurezza di default.
  - I cluster hsm2m.medium introducono la funzionalità mTLS per impedire agli utenti non autorizzati di connettersi al cluster. Gli utenti non autorizzati avranno bisogno di credenziali MTLS valide per connettersi correttamente al cluster prima di tentare la zeroizzazione.

Nella fase successiva, puoi avviare un'istanza Amazon EC2 e collegarla alla tua HSMs collegandovi il gruppo di sicurezza del cluster.

Avvia un'istanza client Amazon EC2 con cui interagire AWS CloudHSM

Per interagire e gestire il AWS CloudHSM cluster e le istanze HSM, devi essere in grado di comunicare con le interfacce di rete elastiche dei tuoi HSM. Il modo più semplice per farlo è utilizzare un'istanza EC2 nello stesso VPC del cluster. Puoi anche utilizzare le risorse AWS seguenti per connettersi al cluster:

- Peering Amazon VPC
- Direct Connect
- Connessioni VPN

> **Note**
>
> Questa guida fornisce un esempio semplificato di come connettere un'istanza EC2 al cluster. AWS CloudHSM Per le migliori pratiche relative alle configurazioni di rete sicure, consulta. Accesso sicuro al tuo cluster

La AWS CloudHSM documentazione in genere presuppone che si stia utilizzando un'istanza EC2 nello stesso VPC e nella stessa zona di disponibilità (AZ) in cui si crea il cluster.

Per creare un'istanza EC2

1. Apri la dashboard EC2 all'indirizzo. https://console.aws.amazon.com/ec2/
2. Seleziona Avvia istanza. Dal menu a tendina, seleziona Avvia istanza.
3. Nel campo Nome immetti un nome per l'istanza EC2.
4. Nella sezione Applicazioni e immagini del sistema operativo (Amazon Machine Image), seleziona un'Amazon Machine Image (AMI), che corrisponde a una piattaforma supportata da CloudHSM. Per ulteriori informazioni, vedi AWS CloudHSM Piattaforme supportate da Client SDK 5.
5. Nella sezione Tipo di istanza, seleziona il tipo di istanza.
6. Nella sezione Coppia di chiavi, usa una coppia di chiavi esistente o seleziona Crea nuova coppia di chiavi e completa i seguenti passaggi:

    a. In Nome coppia di chiavi, immetti un nome per la coppia di chiavi.

    b. Per Tipo coppia di chiavi, scegli una coppia di chiavi.

    c. Per Formato file chiave privata, scegli il formato in cui salvare la chiave privata.

    d. Seleziona Crea coppia di chiavi.

    e. Scarica e salva il file della chiave privata.

    > **Important**
    >
    > Questo è l'unica possibilità per salvare il file della chiave privata. Scarica e archivia il file in un luogo sicuro. Devi fornire il nome della tua coppia di chiavi quando avvii un'istanza. Inoltre, è necessario fornire la chiave privata corrispondente ogni volta che ci si connette all'istanza e scegliere la coppia di chiavi creata al momento della configurazione.

7. In Impostazioni di rete, seleziona Modifica.

8. Per VPC, scegli il VPC creato in precedenza per il cluster.
9. Per Sottorete, scegli la sottorete pubblica creata per il VPC.
10. Per Assegna automaticamente IP pubblico, scegli Abilita.
11. Per l'assegnazione automatica dell' IPv6 IP, scegli Abilita per utilizzare la connettività IPv6 con i cluster e il Dual-stack. NetworkType Se abiliti questa opzione, aggiorna le regole dei gruppi di sicurezza, le tabelle di routing VPC e subnet e la rete dell'istanza Amazon EC2 ACLs per consentire il traffico in IPv6 uscita dall'istanza a. HSMs
12. Scegli Seleziona un gruppo di sicurezza esistente.
13. In Gruppi di sicurezza comuni, seleziona il gruppo di sicurezza predefinito dal menu a tendina.
14. In Configurazione archiviazione, utilizza i menu a tendina per scegliere una configurazione di archiviazione.
15. Nella finestra Riepilogo, seleziona Avvia istanza.

> **Note**
>
> Il completamento di questo passaggio avvierà le operazioni necessarie per la creazione dell'istanza EC2.

Per ulteriori informazioni sulla creazione di un client Amazon EC2 per Linux, consulta Nozioni di base sulle istanze Amazon EC2 Linux. Per informazioni sulla connessione al client in esecuzione, vedi i seguenti argomenti:

- Connessione all'istanza Linux tramite SSH
- Connessione all'istanza Linux da Windows tramite PuTTY

La guida dell'utente Amazon EC2 contiene istruzioni dettagliate per la configurazione e l'utilizzo delle istanze Amazon EC2. L'elenco seguente fornisce una panoramica della documentazione disponibile per i client Amazon EC2 di Linux e Windows:

- Per creare un client Amazon EC2 Linux, vedi Iniziare a usare le istanze di Amazon EC2 Linux.

  Per informazioni sulla connessione al client in esecuzione, vedi i seguenti argomenti:

  - Connessione all'istanza Linux tramite SSH
  - Connessione all'istanza Linux da Windows tramite PuTTY

- Per creare un client Amazon EC2 di Windows, vedi Iniziare a usare le istanze Windows di Amazon EC2. Per ulteriori informazioni sulla connessione al client Windows, vedi Connessione all'istanza Windows.

> **Note**
>
> La tua istanza EC2 può eseguire tutti i AWS CLI comandi contenuti in questa guida. Se la AWS CLI non è installata, è possibile scaricarla da AWS Command Line Interface. Se utilizzi Windows, è possibile scaricare ed eseguire il programma di installazione a 64 bit o a 32 bit di Windows. Se si utilizza Linux o macOS, è possibile installare la CLI tramite pip.

Configurazione dei gruppi di sicurezza delle istanze Client Amazon EC2 per AWS CloudHSM

Quando hai lanciato un'istanza Amazon EC2 per il tuo cluster in AWS CloudHSM, l'hai associata a un gruppo di sicurezza Amazon VPC predefinito. Questo argomento spiega come associare il gruppo di sicurezza del cluster con l'istanza EC2. Questa associazione consente al AWS CloudHSM client in esecuzione sulla tua istanza EC2 di comunicare con i tuoi HSM. Per connettere l'istanza EC2 al AWS CloudHSM cluster, è necessario configurare correttamente il gruppo di sicurezza predefinito VPC e associare il gruppo di sicurezza del cluster all'istanza.

Utilizza i seguenti passaggi per completare le modifiche alla configurazione.

Argomenti

- Passaggio 1. Modifica il gruppo di sicurezza di default.
- Passaggio 2. Connect l'istanza Amazon EC2 al cluster AWS CloudHSM

Passaggio 1. Modifica il gruppo di sicurezza di default.

Devi modificare il gruppo di sicurezza di default per consentire la connessione SSH o RDP in modo che sia possibile scaricare e installare il software client, e interagire con il tuo HSM.

Modifica del gruppo di sicurezza predefinito

1. Apri il pannello di controllo EC2 all'indirizzo https://console.aws.amazon.com/ec2/.

2. Seleziona Istanze (in esecuzione), quindi seleziona la casella di controllo accanto all'istanza EC2 su cui desideri installare il client. AWS CloudHSM
3. Nella scheda Sicurezza, scegli il gruppo di sicurezza denominato Default.
4. Nella parte superiore della pagina, seleziona Azioni, Modifica regole in entrata.
5. Seleziona Aggiungi regola.
6. Per Tipo, procedere in uno dei seguenti modi:
   - Per un'istanza Windows Server Amazon EC2, scegli RDP. La porta 3389 viene completata automaticamente.
   - Per un'istanza Amazon EC2 Linux, scegli SSH. L'intervallo di porte 22 viene completato automaticamente.
7. Per entrambe le opzioni, imposta Fonte del Mio IP per consentirti di comunicare con la tua istanza Amazon EC2.

   > **Important**
   >
   > Non specificare 0.0.0.0/0 come intervallo CIDR per evitare di consentire a chiunque di accedere all'istanza.

8. Scegli Save (Salva).

Passaggio 2. Connect l'istanza Amazon EC2 al cluster AWS CloudHSM

È necessario collegare il gruppo di sicurezza del cluster all'istanza EC2 in modo che l'istanza EC2 possa comunicare con HSMs il cluster. Il gruppo di sicurezza del cluster contiene una regola preconfigurata che consente la comunicazione sulle porte 2223-2225 in entrata.

Per connettere l'istanza EC2 al cluster AWS CloudHSM

1. Apri il pannello di controllo EC2 all'indirizzo. https://console.aws.amazon.com/ec2/
2. Seleziona Istanze (in esecuzione), quindi seleziona la casella di controllo relativa all'istanza EC2 su cui desideri installare il client. AWS CloudHSM
3. Nella parte superiore della pagina, scegli Azioni, Sicurezza, quindi Cambia gruppi di sicurezza.
4. Seleziona il gruppo di sicurezza con il nome del gruppo che corrisponde all'ID del cluster, come `cloudhsm-cluster-<clusterID>-sg`.
5. Seleziona Applica i gruppi di sicurezza.

6. Seleziona Salva.

> **Note**
>
> È possibile assegnare un massimo di cinque gruppi di sicurezza a un'istanza Amazon EC2. Se hai raggiunto il limite massimo, è necessario modificare il gruppo di sicurezza predefinito dell'istanza Amazon EC2 e il gruppo di sicurezza del cluster:
>
> Nel gruppo di sicurezza di default, procedi nel seguente modo:
>
> - Aggiungi una regola in entrata per consentire il traffico utilizzando il protocollo TCP sulle porte `2223-2225` dal gruppo di sicurezza del cluster.
>
> Nel gruppo di sicurezza del cluster, procedi nel seguente modo:
>
> - Aggiungi una regola in entrata per consentire il traffico utilizzando il protocollo TCP sulle porte `2223-2225` dal gruppo di sicurezza di default.

Crea un HSM in AWS CloudHSM

Dopo aver creato un cluster in AWS CloudHSM, è possibile creare un modulo di sicurezza hardware (HSM). Tuttavia, prima di essere in grado di creare un HSM nel cluster, devi impostare quest'ultimo come non inizializzato. Per determinare lo stato del cluster, visualizza la pagina dei cluster nella AWS CloudHSM console, usa il comando AWS CLI per eseguire il describe-clusters comando o invia una DescribeClustersrichiesta nell' AWS CloudHSM API. È possibile creare un HSM dalla console dell'AWS CloudHSM, dalla AWS CLI o dall'API AWS CloudHSM .

> **Important**
>
> Crea un solo HSM mentre il cluster è nello stato non inizializzato.

Console

Per creare un HSM (console)

1. Apri la AWS CloudHSM console a casa. https://console.aws.amazon.com/cloudhsm/

2. Seleziona il pulsante di opzione accanto all'ID del cluster per il quale desideri creare un HSM.
3. Seleziona Azioni. Dal menu a tendina, scegli Inizializza.
4. Scegli una zona di disponibilità (AZ) per il modulo HSM in fase di creazione.
5. Seleziona Crea.

Dopo avere creato un cluster e un HSM, potrai verificare l'identità dell'HSM oppure procedere direttamente a Inizializzazione del cluster.

AWS CLI

Per creare un HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando create-hsm. Specifica l'ID del cluster creato in precedenza e una zona di disponibilità per l'HSM. Specifica la zona di disponibilità con il formato `us-west-2a`, `us-west-2b` e così via.

```
$ aws cloudhsmv2 create-hsm --cluster-id <cluster ID> --availability-
zone <Availability Zone>

{
    "Hsm": {
        "HsmId": "hsm-ted36yp5b2x",
        "EniIp": "10.0.1.12",
        "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
        "AvailabilityZone": "us-west-2a",
        "ClusterId": "cluster-igklspoyj5v",
        "EniId": "eni-5d7ade72",
        "SubnetId": "subnet-fd54af9b",
        "State": "CREATE_IN_PROGRESS"
    }
}
```

Dopo avere creato un cluster e un HSM, potrai verificare l'identità dell'HSM oppure procedere direttamente a Inizializzazione del cluster.

AWS CloudHSM API

Per creare un HSM (AWS CloudHSM API)

- Inviare una richiesta CreateHsm. Specifica l'ID del cluster creato in precedenza e una zona di disponibilità per l'HSM.

Dopo avere creato un cluster e un HSM, potrai verificare l'identità dell'HSM oppure procedere direttamente a Inizializzazione del cluster.

Verifica l'identità e l'autenticità dell'HSM del tuo cluster in AWS CloudHSM (facoltativo)

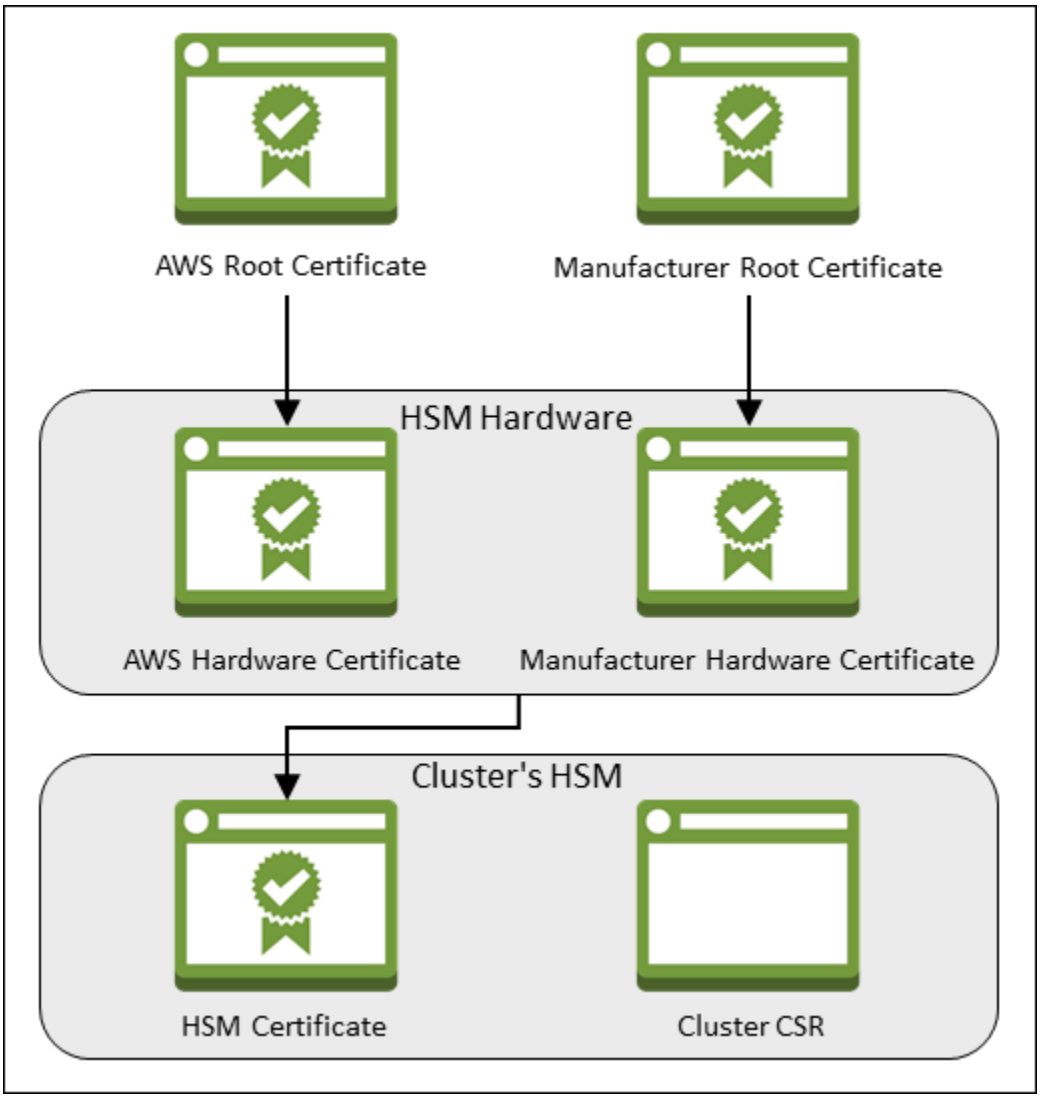
Per inizializzare il cluster in AWS CloudHSM, è necessario firmare una richiesta di firma del certificato (CSR) generata dal primo modulo di sicurezza hardware (HSM) del cluster. Prima di eseguire questa operazione, puoi verificare l'identità e l'autenticità dell'HSM.

> **Note**
>
> Questo processo è facoltativo. Tuttavia, funziona solo finché un cluster viene inizializzato. In seguito all'inizializzazione del cluster, non è possibile utilizzare questo processo per ottenere i certificati o verificare gli HSM.

Per verificare l'identità dell'HSM del primo cluster, segui i passi seguenti:

1. Ottieni i certificati e la CSR in questa fase, ottieni tre certificati e una CSR dall'HSM. Ottieni anche due certificati root, uno dal produttore dell'hardware HSM AWS CloudHSM e l'altro dal produttore dell'hardware HSM.
2. Verifica le catene di certificati: in questo passaggio, crei due catene di certificati, una relativa al certificato AWS CloudHSM radice e una al certificato radice del produttore. Quindi verificate il certificato HSM con queste catene di certificati per stabilirlo AWS CloudHSM e il produttore dell'hardware attesta l'identità e l'autenticità dell'HSM.
3. Confronta le chiavi pubbliche in questa fase, estrai e confronti le chiavi pubbliche nel certificato HSM e la CSR del cluster per accertarti che siano le stesse. Questo dovrebbe offrire la sicurezza che la CSR è stato generata da un HSM autentico e affidabile.

Il diagramma seguente mostra la CSR, i certificati e la loro correlazione. L'elenco successivo definisce ogni certificato.



AWS Certificato principale

Questo è AWS CloudHSM il certificato principale.

Certificato root del produttore

Questo è il certificato root del produttore dell'hardware.

AWS Certificato hardware

AWS CloudHSM ha creato questo certificato quando l'hardware HSM è stato aggiunto al parco macchine. Questo certificato afferma di essere il AWS CloudHSM proprietario dell'hardware.

Certificato hardware del produttore

Il produttore dell'hardware HSM ha creato questo certificato quando ha prodotto l'hardware HSM. Questo certificato attesta che il produttore ha creato l'hardware.

Certificato HSM

Il certificato HSM viene generato dall' FIPS-validated hardware quando si crea il primo HSM nel cluster. Questo certificato attesta che l'hardware HSM ha creato il modulo HSM.

CSR del cluster

Il primo HSM crea la CSR del cluster. Quando firmi la CSR del cluster, rivendichi il cluster. Quindi, puoi utilizzare la CSR firmata per inizializzare il cluster.

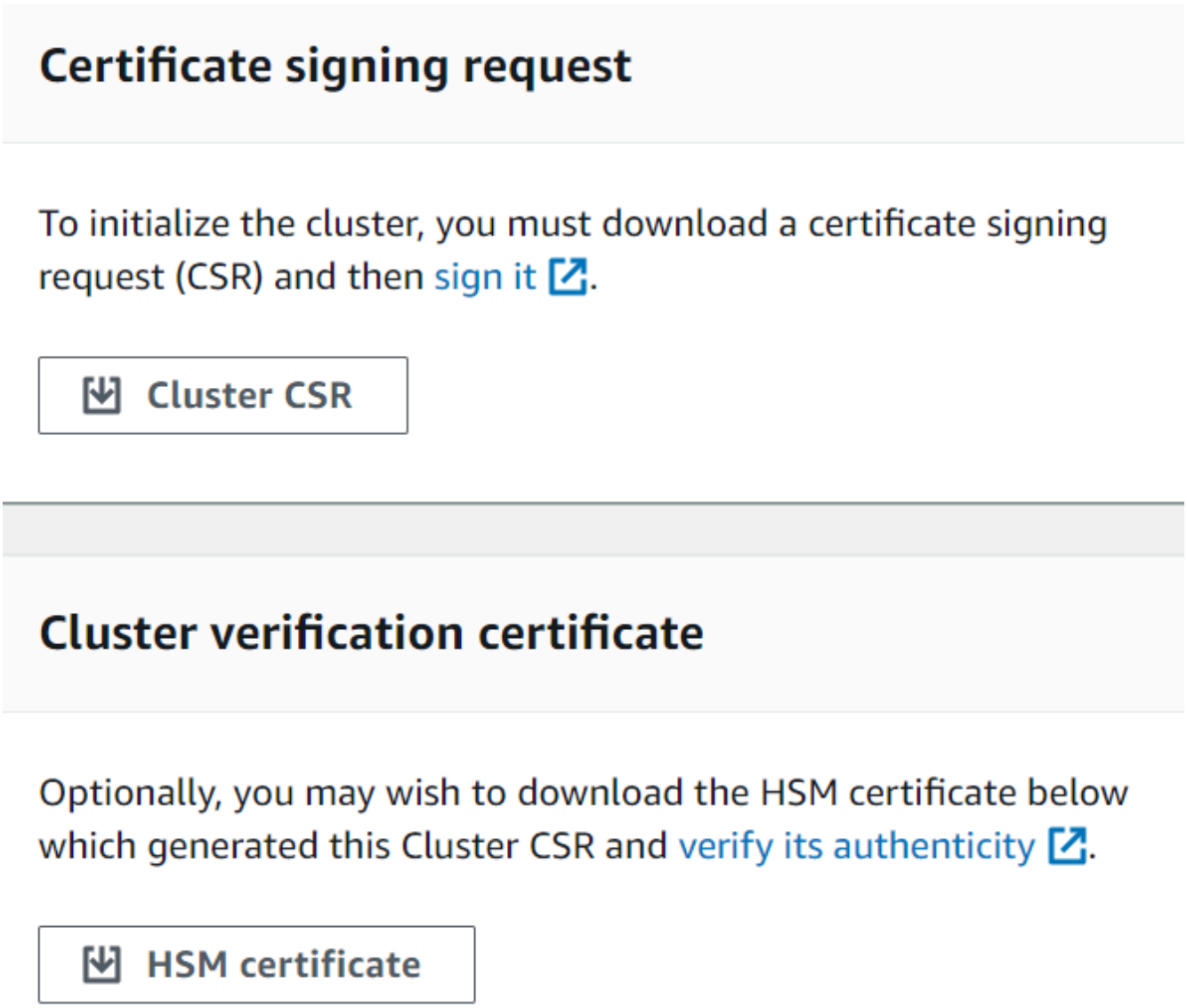
Passaggio 1. Ottieni i certificati da HSM

Per verificare l'identità e l'autenticità dell'HSM, inizia ottenendo una CSR e cinque certificati. Ottieni tre dei certificati dall'HSM, cosa che puoi fare con la AWS CloudHSM console, il AWS Command Line Interface (AWS CLI) o l'API. AWS CloudHSM

Console

Per ottenere i certificati CSR e HSM (della console)

1. Apri la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Seleziona il pulsante di opzione accanto all'ID del cluster con l'HSM che desideri verificare.
3. Seleziona Azioni. Dal menu a tendina, scegli Inizializza.
4. Se non hai completato il passaggio precedente per creare un HSM, scegli una zona di disponibilità (AZ) per l'HSM che stai creando. Quindi seleziona Crea.
5. Quando certificati e CSR sono pronti, vengono visualizzati i link per scaricarli.



6. Seleziona ogni link per scaricare e salvare CSR e certificati. Per semplificare le fasi successive, salva tutti i file nella stessa directory e utilizza i nomi di file predefiniti.

AWS CLI

Per ottenere i certificati CSR e HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando describe-clusters quattro volte, estraendo ogni volta la CSR e i diversi certificati e salvandoli sui file.

  a. Esegui il seguente comando per estrarre la CSR del cluster. Sostituisci `<cluster ID>` con l'ID del cluster creato in precedenza.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                   --output text \
                                   --query
 'Clusters[].Certificates.ClusterCsr' \
                                   > <cluster ID>_ClusterCsr.csr
```

b. Esegui il seguente comando per estrarre il certificato HSM. Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                   --output text \
                                   --query
 'Clusters[].Certificates.HsmCertificate' \
                                   > <cluster ID>_HsmCertificate.crt
```

c. Esegui il seguente comando per estrarre il certificato AWS hardware. Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                   --output text \
                                   --query
 'Clusters[].Certificates.AwsHardwareCertificate' \
                                   > <cluster ID>_AwsHardwareCertificate.crt
```

d. Esegui il seguente comando per estrarre il certificato hardware del produttore. Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                   --output text \
                                   --query
 'Clusters[].Certificates.ManufacturerHardwareCertificate' \
                                   > <cluster
 ID>_ManufacturerHardwareCertificate.crt
```

AWS CloudHSM API

Per ottenere i certificati CSR e HSM (AWS CloudHSM API)

- Invia una richiesta DescribeClusters, quindi estrai e salva CSR e certificati dalla risposta.

Passaggio 2. Ottenere i certificati root

Segui questi passaggi per ottenere i certificati root AWS CloudHSM e il produttore. Salva i file del certificato root nella directory che contiene i file dei certificati HSM e la CSR.

Per ottenere il AWS CloudHSM e certificati root del produttore

1. Scarica il certificato AWS CloudHSM principale: AWS_CloudHSM_ Root-G1.zip
2. Scarica il certificato root del produttore giusto per il tuo tipo di HSM:
   - certificato root del produttore hsm1.medium: liquid_security_certificate.zip
   - certificato principale del produttore hsm2m.medium: liquid_security_certificate.zip

   > **Note**
   >
   > Per scaricare ogni certificato dalla relativa pagina di destinazione, usa i seguenti link:
   >
   > - Pagina di destinazione per il certificato root del produttore di hsm1.medium
   > - Pagina iniziale del certificato root del produttore di hsm2m.medium
   >
   > Potrebbe essere necessario fare clic con il pulsante destro del mouse su Scarica certificato, quindi scegliere Salva collegamento con nome per salvare il file del certificato.
3. Dopo aver scaricato il file, estrai (decomprimi) il contenuto.

Fase 3. Verifica le catene di certificato

In questo passaggio, si creano due catene di certificati, una relativa al certificato radice e una al certificato AWS CloudHSM radice del produttore. Quindi utilizza OpenSSL per verificare il certificato HSM con ogni catena di certificati.

Per creare le catene di certificato, apri una shell Linux. È necessario disporre di OpenSSL, disponibile nella maggior parte delle shell Linux, del certificato root e dei file del certificato HSM che hai scaricato. Tuttavia, non ne avete bisogno AWS CLI per questo passaggio e non è necessario che la shell sia associata al vostro AWS account.

Per verificare il certificato HSM con il AWS CloudHSM certificato principale

1. Passare alla directory in cui è stato salvato il certificato certificato root e i file dei certificati HSM che hai scaricato. I comandi seguenti presuppongono che tutti i certificati siano nella directory corrente e utilizzino i nomi file predefiniti.

   Utilizzate il comando seguente per creare una catena di certificati che includa il certificato AWS hardware e il certificato AWS CloudHSM radice, in quest'ordine. Sostituiscilo *<cluster ID>* con l'ID del cluster creato in precedenza.

   ```
   $ cat <cluster ID>_AwsHardwareCertificate.crt \
         AWS_CloudHSM_Root-G1.crt \
         > <cluster ID>_AWS_chain.crt
   ```

2. Utilizza il seguente comando OpenSSL per verificare il certificato HSM con la catena di certificati AWS . Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

   ```
   $ openssl verify -CAfile <cluster ID>_AWS_chain.crt <cluster ID>_HsmCertificate.crt
   <cluster ID>_HsmCertificate.crt: OK
   ```

Per verificare il certificato HSM con il certificato root del produttore

1. Utilizza il comando seguente per creare una catena di certificati che includa il certificato hardware del produttore e il certificato root del produttore, in questo ordine. Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

   ```
   $ cat <cluster ID>_ManufacturerHardwareCertificate.crt \
         liquid_security_certificate.crt \
         > <cluster ID>_manufacturer_chain.crt
   ```

2. Utilizza il seguente comando OpenSSL per verificare il certificato HSM con la catena di certificati del produttore. Sostituisci *<cluster ID>* con l'ID del cluster creato in precedenza.

   ```
   $ openssl verify -CAfile <cluster ID>_manufacturer_chain.crt <cluster
    ID>_HsmCertificate.crt
   <cluster ID>_HsmCertificate.crt: OK
   ```

Passaggio 4. Estrai e confronta chiavi pubbliche

Utilizza OpenSSL per estrarre e confrontare le chiavi pubbliche nel certificato HSM e la CSR del cluster per accertarsi che siano le stesse.

Per confrontare le chiavi pubbliche, utilizzare la shell Linux. È necessario OpenSSL, disponibile nella maggior parte delle shell Linux, ma non è necessario AWS CLI per questo passaggio. Non è necessario che la shell sia associata al tuo account. AWS

Estrai e confronta chiavi pubbliche

1. Utilizza il comando seguente per estrarre la chiave pubblica dal certificato HSM.

```
$ openssl x509 -in <cluster ID>_HsmCertificate.crt -pubkey -noout > <cluster
 ID>_HsmCertificate.pub
```

2. Utilizzare il comando seguente per estrarre la chiave pubblica dalla CSR del cluster.

```
$ openssl req -in <cluster ID>_ClusterCsr.csr -pubkey -noout > <cluster
 ID>_ClusterCsr.pub
```

3. Utilizzare il comando seguente per confrontare le chiavi pubbliche. Se le chiavi pubbliche sono identiche, il comando seguente non produce alcun output.

```
$ diff <cluster ID>_HsmCertificate.pub <cluster ID>_ClusterCsr.pub
```

Dopo aver verificato l'identità e l'autenticità dell'HSM, passa a Inizializzazione del cluster.

Inizializza il cluster in AWS CloudHSM

Dopo aver creato il cluster e aggiunto il modulo di sicurezza hardware (HSM) AWS CloudHSM, è possibile inizializzare il cluster. Completa le fasi descritte negli argomenti seguenti per inizializzare il cluster .

> **Note**
>
> Prima di inizializzare il cluster, rivedi la procedura di verifica dell'identità e dell'autenticità dei moduli HSM. Questa procedura è facoltativa e puoi seguirla solo finché non inizializzi il

cluster. In seguito all'inizializzazione del cluster, non potrai utilizzare questa procedura per ottenere i certificati o verificare i moduli HSM.

Argomenti

- Panoramica di
- Passaggio 1. Ottenere la CSR del cluster
- Passaggio 2. Crea una chiave privata per la tua Root CA
- Fase 3. Firma la CSR
- Passaggio 4. Inizializzazione del cluster

Panoramica di

Il processo di inizializzazione del cluster stabilisce la proprietà e il controllo del cluster e degli HSM tramite un sistema di autenticazione basato su certificati. Questo processo dimostra crittograficamente che l'utente è l'unico proprietario degli HSM del cluster e crea le basi di fiducia necessarie per tutte le future connessioni ai moduli di protezione hardware.

Questa pagina ti mostrerà come fare quanto segue:

- Recupera la richiesta di firma del certificato (CSR) del tuo cluster.
- Genera e usa le chiavi private per creare un certificato radice autofirmato o una catena di certificati.
- Firma la CSR del tuo cluster per produrre un certificato HSM firmato.
- Inizializza il cluster utilizzando il certificato HSM firmato e il certificato o la catena di certificati autofirmati.

Quando sei pronto per iniziare, vai a Passaggio 1. Ottenere la CSR del cluster.

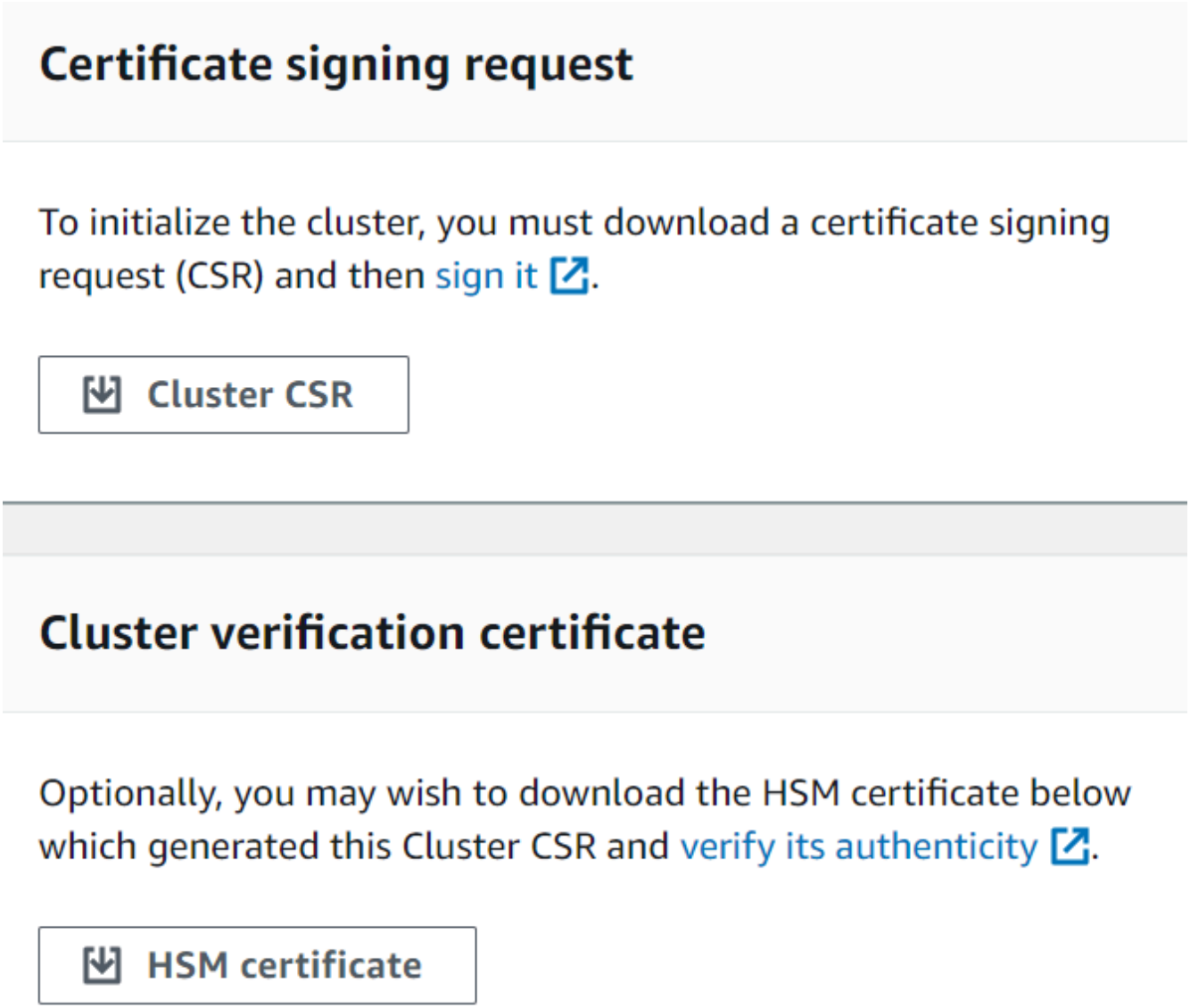
Passaggio 1. Ottenere la CSR del cluster

Prima di inizializzare il cluster, occorre scaricare e firmare una richiesta di firma del certificato (CSR) generata dal primo HSM del cluster. Se hai seguito le fasi per verificare l'identità dell'HSM del cluster, disponi già della CSR e puoi quindi firmarla. Altrimenti, ottieni subito la CSR utilizzando la AWS CloudHSM console, il AWS Command Line Interface (AWS CLI) o l'API. AWS CloudHSM

Console

Per ottenere la CSR (console)

1. Apri la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Seleziona il pulsante di opzione accanto all'ID del cluster con l'HSM che desideri verificare.
3. Seleziona Azioni. Dal menu a tendina, scegli Inizializza.
4. Se non hai completato il passaggio precedente per creare un HSM, scegli una zona di disponibilità (AZ) per l'HSM che stai creando. Quindi seleziona Crea.
5. Quando la CSR è pronta, verrà visualizzato un collegamento per scaricarla.



6. Sceglie CSR del cluster per scaricare e salvare la CSR.

AWS CLI

Per ottenere la CSR (AWS CLI)

- Al prompt dei comandi, esegui il seguente comando describe-clusters, che estrae la CSR e la salva in un file. Sostituiscilo *<cluster ID>* con l'ID del cluster creato in precedenza.

```
$ aws cloudhsmv2 describe-clusters --filters clusterIds=<cluster ID> \
                                   --output text \
                                   --query 'Clusters[].Certificates.ClusterCsr'
 \
                                   > <cluster ID>_ClusterCsr.csr
```

AWS CloudHSM API

Per ottenere il CSR (AWS CloudHSM API)

1. Inviare una richiesta DescribeClusters.
2. Estrai e salva la CSR dalla risposta.

Passaggio 2. Crea una chiave privata per la tua Root CA

> **Note**
>
> Per i cluster di produzione, la chiave deve essere creata in modo sicuro tramite una fonte attendibile di casualità. Ti consigliamo di utilizzare un HSM offline e fuori sede protetto o un equivalente. Archivia la chiave in modo sicuro. La chiave stabilisce l'identità del cluster e il controllo esclusivo dell'utente sui moduli HSM in esso contenuti.
> Durante le fasi di sviluppo e di testing, puoi utilizzare qualsiasi strumento adatto (come OpenSSL) per creare e firmare il certificato del cluster. Nell'esempio seguente viene illustrato come creare una chiave. Dopo aver creato un certificato autofirmato usando la chiave (vedi sotto), archivialo in modo sicuro. Per accedere all' AWS CloudHSM istanza, è necessario che il certificato sia presente, ma la chiave privata no.

La tabella seguente riporta gli algoritmi, le dimensioni delle chiavi e le curve supportati per la generazione dei certificati.

Algoritmi	Size/Curves
RSA PKCSv1.5	2048, 3072, 4096
RSA-PSS	2048, 3072, 4096
ECDSA	primo 256v1, secp 384r1, secp 521r1
Digest	SHA-224 SHA-256, SHA-384, e SHA-512

Utilizzate il seguente comando di esempio per creare una chiave privata per la vostra Root CA autofirmata.

```
$ openssl genrsa -aes256 -out customerRootCA.key 2048
Generating RSA private key, 2048 bit long modulus
........+++
............+++
e is 65537 (0x10001)
Enter pass phrase for customerRootCA.key:
Verifying - Enter pass phrase for customerRootCA.key:
```

Fase 3. Firma la CSR

Nei passaggi precedenti, hai recuperato la CSR del cluster e creato una chiave privata per la tua CA principale. In questo passaggio, utilizzerai la tua chiave privata per generare un certificato di firma per firmare la CSR del cluster. Gli argomenti seguenti ti guideranno attraverso il processo di creazione di un singolo certificato autofirmato, o di una catena di certificati, utilizzando OpenSSL. Non ne hai bisogno AWS CLI per questo passaggio e non è necessario che la shell sia associata al tuo account. AWS

> **Important**
>
> Per inizializzare il cluster, l'ancora di fiducia deve essere conforme alla RFC 5280 e soddisfare i seguenti requisiti:

- Se si utilizzano estensioni X509v3, deve essere presente l'estensione X509v3 Basic Constraints.
- L'ancora di fiducia deve essere un certificato autofirmato.
- I valori delle estensioni non devono essere in conflitto tra loro.

Scegli uno dei seguenti approcci per firmare la CSR del tuo cluster:

Scegli il tuo approccio basato sui certificati

È necessario scegliere uno dei due approcci seguenti. Non completare entrambi gli approcci.

Opzione A: certificato singolo autofirmato

Crea un unico certificato radice autofirmato per firmare la CSR del tuo cluster. Questo è il metodo più semplice e diretto per stabilire la fiducia.

Consigliato per:

- Ambienti in cui non è richiesta una PKI esterna
- Ambienti di test e sviluppo in cui è preferibile la semplicità

Vai a: Crea un unico certificato autofirmato

Opzione B: catena di certificati con CA intermedia

Crea una catena di certificati utilizzando un'autorità di certificazione intermedia. Una catena di certificati intermedia offre maggiore sicurezza, scalabilità e flessibilità, poiché consente alle autorità di certificazione (CA) principali di rimanere offline mentre delegano l'emissione dei certificati alle CA intermedie, riducendo così il rischio di compromettere la CA principale.

Consigliato per:

- Ambienti in cui è richiesta una PKI esterna
- Integrazione con AWS Private Certificate Authority (PCA)

Esempio di integrazione con AWS PCA: puoi usare AWS Private Certificate Authority per creare e gestire i tuoi certificati CA intermedi. Ciò fornisce una gestione automatizzata del ciclo di vita dei certificati, compresi il rinnovo e la revoca, mantenendo al contempo i vantaggi in termini di

sicurezza derivanti dal mantenere offline la CA root. Per ulteriori informazioni sulla AWS PCA, consulta la AWS Private Certificate Authority User Guide.

Vai a: Crea una catena di autorità di certificazione intermedia (ICA)

Crea un unico certificato autofirmato

L'hardware attendibile che usi per creare la chiave privata per il cluster di produzione deve fornire inoltre uno strumento software per la generazione di un certificato autofirmato tramite tale chiave. L'esempio seguente utilizza OpenSSL e la chiave privata creata nel passaggio precedente per creare un certificato di firma CA root autofirmato. Il certificato è valido per 10 anni (3652 giorni). Leggi le istruzioni a video e segui i prompt.

```
$ openssl req -new -x509 -days 3652 -key customerRootCA.key -out customerRootCA.crt
Enter pass phrase for customerRootCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Questo comando crea un file di certificato denominato `customerRootCA.crt`. Inserisci questo certificato su ogni host da cui ti connetterai al tuo cluster. AWS CloudHSM Se dai un nome diverso al file o se lo archivi in un percorso diverso dalla radice dell'host, devi modificare il file di configurazione del client di conseguenza. Utilizza il certificato e la chiave privata appena creati per firmare la richiesta di firma del certificato (CSR) del cluster nella fase successiva.

Firma la CSR del cluster con la tua Root CA autofirmata

L'hardware attendibile che hai utilizzato per creare la chiave privata per il cluster di produzione deve fornire inoltre uno strumento software per la firma della CSR tramite tale chiave. Nell'esempio

seguente viene utilizzato OpenSSL per la firma della CSR del cluster. Il comando di esempio seguente firma la CSR con la firma autofirmata `customerRootCA.crt`

```
$ openssl x509 -req -days 3652 -in <cluster ID>_ClusterCsr.csr \
  -CA <customerRootCA>.crt \
  -CAkey <customerRootCA>.key \
  -CAcreateserial \
  -out <cluster ID>_CustomerHsmCertificate.crt
Signature ok
subject=/C=US/ST=CA/O=Cavium/OU=N3FIPS/L=SanJose/CN=HSM:<HSM
 identifier>:PARTN:<partition number>, for FIPS mode
Getting CA Private Key
Enter pass phrase for <customerRootCA>.key:
```

Questo comando crea un file denominato *<cluster ID>*`_CustomerHsmCertificate.crt`. Utilizzalo come certificato firmato durante l'inizializzazione del cluster.

Verifica il certificato firmato confrontandolo con la CA principale (opzionale):

```
$ openssl verify -purpose sslserver -CAfile customerRootCA.crt <cluster
 ID>_CustomerHsmCertificate.crt
<cluster ID>_CustomerHsmCertificate.crt: OK
```

Dopo aver prodotto il certificato HSM firmato con la tua Root CA autofirmata, vai a. Passaggio 4. Inizializzazione del cluster

Crea una catena di autorità di certificazione intermedia (ICA)

Gli esempi seguenti illustreranno la creazione di una catena di certificati di lunghezza 2, composta da un'autorità di certificazione (CA) principale e una CA intermedia. Per prima cosa creerai un certificato CA radice autofirmato, quindi genererai una CA intermedia firmata dalla CA principale. Infine, utilizzerai la CA intermedia per firmare la CSR del cluster, creando una catena di fiducia completa dal certificato HSM alla CA principale. Questo approccio offre una maggiore sicurezza mantenendo offline la CA principale mentre utilizza la CA intermedia per le operazioni quotidiane dei certificati.

> **Important**
>
> Per inizializzare il cluster con una catena di certificati, la catena deve soddisfare i seguenti requisiti:

- La catena deve essere ordinata, a partire dalla CA intermedia che firma la CSR del cluster. In quest'ordine, la prima ICA dovrebbe avere un emittente che corrisponda all'oggetto dell'ICA successiva della catena, e così via.
- Solo la Root CA deve essere autofirmata, il che significa che l'emittente e l'oggetto devono essere identici.
- La catena non deve essere composta da più di 4 certificati (inclusa la Root CA alla fine) e la dimensione totale della catena non deve superare i 16 kb (kilobyte).
- Tutte le autorità di certificazione (CA) devono conformarsi alle linee guida RFC 5280.

Questa sezione fornisce esempi per creare una catena di autorità di certificazione intermedia utilizzando due approcci diversi: OpenSSL per la generazione di certificati locali e AWS Private Certificate Authority (PCA) per i servizi di certificazione gestiti. Scegli l'approccio più adatto al tuo ambiente e ai tuoi requisiti di sicurezza.

> **Note**
>
> Gli esempi seguenti sono casi d'uso generici e sono entrambi semplificati, utilizzando la configurazione di base. Per gli ambienti di produzione, esamina le opzioni di configurazione aggiuntive e i requisiti di sicurezza specifici per il tuo caso d'uso.

OpenSSL

Crea il file di configurazione OpenSSL con le estensioni v3 comuni per CA:

```
$ cat > ca-extensions.conf <<EOF
[req]
distinguished_name = req_distinguished_name
[req_distinguished_name]
C = Country Name (2 letter code)
ST = State or Province Name (full name)
L = Locality Name (eg, city)
O = Organization Name (eg, company)
OU = Organizational Unit Name (eg, section)
CN = Common Name (e.g. server FQDN or YOUR name)
[v3_ca]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
```

```
basicConstraints = critical,CA:true
keyUsage = critical, keyCertSign, cRLSign, digitalSignature
EOF
```

Genera una CA root autofirmata utilizzando OpenSSL:

```
$ openssl req -new -x509 -days 3652 -key customerRootCA.key -out customerRootCA.crt
 -extensions v3_ca -config ca-extensions.conf
Enter pass phrase for customerRootCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Genera una chiave CA intermedia:

```
$ openssl genrsa -aes256 -out intermediateCA.key 2048
Generating RSA private key, 2048 bit long modulus
........+++
............+++
e is 65537 (0x10001)
Enter pass phrase for intermediateCA.key:
Verifying - Enter pass phrase for intermediateCA.key:
```

Crea la richiesta di firma del certificato CA intermedia (CSR):

```
$ openssl req -new -key intermediateCA.key -out intermediateCA.csr
Enter pass phrase for intermediateCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
```

```
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Utilizzando la CA root autofirmata, crea il certificato CA intermedio:

```
$ openssl x509 -req -in intermediateCA.csr \
  -CA customerRootCA.crt \
  -CAkey customerRootCA.key \
  -CAcreateserial \
  -days 3652 \
  -extensions v3_ca \
  -extfile ca-extensions.conf \
  -out intermediateCA.crt

Certificate request self-signature ok
subject=C= , ST= , L= , O= , OU=
```

Combina i certificati in un file a catena:

```
$ cat intermediateCA.crt customerRootCA.crt > chainCA.crt

-----BEGIN CERTIFICATE-----
[Intermediate CA]
-----END CERTIFICATE-----
...
...
-----BEGIN CERTIFICATE-----
[Root CA]
-----END CERTIFICATE-----
```

Firma la CSR del cluster con la tua CA intermedia:

```
$ openssl x509 -req -days 3652 -in <cluster ID>_ClusterCsr.csr \
   -CA intermediateCA.crt \
   -CAkey intermediateCA.key \
   -CAcreateserial \
   -out <cluster ID>_CustomerHsmCertificate.crt
Signature ok
subject=/C=US/ST=CA/O=Cavium/OU=N3FIPS/L=SanJose/CN=HSM:<HSM
 identifier>:PARTN:<partition number>, for FIPS mode
Getting CA Private Key
Enter pass phrase for intermediateCA.key:
```

AWS PCA

Crea e attiva una CA root utilizzando AWS Private Certificate Authority:

```
$ # 1. Create Root CA
aws acm-pca create-certificate-authority \
    --certificate-authority-configuration \
        "KeyAlgorithm=RSA_4096,
        SigningAlgorithm=SHA256WITHRSA,

 Subject={Country=US,Organization=MyOrg,OrganizationalUnit=IT,CommonName=RootCA}" \
    --certificate-authority-type ROOT

# Store the Root CA Authority ARN from the previous output
ROOT_CA_AUTHORITY_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<ca-authority-id>"

# 2. Generate Root CA CSR
aws acm-pca get-certificate-authority-csr \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --output text > customerRootCA.csr

# 3. Self-sign Root CA Certificate
aws acm-pca issue-certificate \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --csr fileb://customerRootCA.csr \
    --signing-algorithm SHA256WITHRSA \
 --template-arn arn:aws:acm-pca:::template/RootCACertificate/V1 \
    --validity Value=3652,Type=DAYS
```

```
# Store the Root CA certificate ARN from the previous output
ROOT_CA_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-authority/<ca-
authority-id>/certificate/<cert-id>"

# 4. Retrieve the Root CA certificate
aws acm-pca get-certificate \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --certificate-arn $ROOT_CA_ARN \
    --output text > customerRootCA.crt

# 5. Import the Root CA Certificate
aws acm-pca import-certificate-authority-certificate \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --certificate fileb://customerRootCA.crt
```

Crea e attiva una CA subordinata (nota anche come CA intermedia):

```
$ # 6. Create Subordinate CA
aws acm-pca create-certificate-authority \
    --certificate-authority-configuration \
        "KeyAlgorithm=RSA_4096,
        SigningAlgorithm=SHA256WITHRSA,

 Subject={Country=US,Organization=MyOrg,OrganizationalUnit=IT,CommonName=SubordinateCA}"
 \
    --certificate-authority-type SUBORDINATE

# Store the Subordinate CA Authority ARN from the previous output
SUB_CA_AUTHORITY_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<sub-ca-authority-id>"

# 7. Generate Subordinate CA CSR
aws acm-pca get-certificate-authority-csr \
    --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
    --output text > intermediateCA.csr

# 8. Issue Subordinate CA Certificate using Root CA
aws acm-pca issue-certificate \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --csr fileb://intermediateCA.csr \
    --signing-algorithm SHA256WITHRSA \
    --template-arn arn:aws:acm-pca:::template/SubordinateCACertificate_PathLen0/V1 \
```

```
    --validity Value=3651,Type=DAYS

# Store the Subordinate CA certificate ARN from the previous output
SUB_CA_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-authority/<sub-ca-
authority-id>"

# 9. Retrieve Subordinate CA Certificate
aws acm-pca get-certificate \
    --certificate-authority-arn $ROOT_CA_AUTHORITY_ARN \
    --certificate-arn $SUB_CA_ARN \
    --query 'Certificate' \
    --output text > intermediateCA.crt

# 10. Import the Subordinate CA Certificate
aws acm-pca import-certificate-authority-certificate \
    --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
    --certificate fileb://intermediateCA.crt \
    --certificate-chain fileb://customerRootCA.crt
```

Combina i certificati in un file a catena:

```
$ cat intermediateCA.crt customerRootCA.crt > chainCA.crt

-----BEGIN CERTIFICATE-----
[Intermediate CA]
-----END CERTIFICATE-----
...
...
-----BEGIN CERTIFICATE-----
[Root CA]
-----END CERTIFICATE-----
```

Firma la CSR del cluster utilizzando AWS PCA:

```
$ aws acm-pca issue-certificate \
    --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
    --csr fileb://<cluster ID>_ClusterCsr.csr \
    --signing-algorithm SHA256WITHRSA \
    --template-arn arn:aws:acm-pca:::template/EndEntityCertificate/V1 \
    --validity Value=3650,Type=DAYS
```

```
# Store your cluster's cert ARN from the previous output
CLUSTER_CERT_ARN="arn:aws:acm-pca:<region>:<account-id>:certificate-
authority/<cluster-cert-arn>"
```

Scarica il certificato del cluster firmato:

```
$ aws acm-pca get-certificate \
    --certificate-authority-arn $SUB_CA_AUTHORITY_ARN \
    --certificate-arn $CLUSTER_CERT_ARN \
    --output text --query Certificate > <cluster ID>_CustomerHsmCertificate.crt
```

Questo comando crea un file denominato *<cluster ID>*_CustomerHsmCertificate.crt. Utilizzalo come certificato firmato durante l'inizializzazione del cluster.

Verifica il certificato firmato rispetto alla catena di certificati (opzionale):

```
$ openssl verify -purpose sslserver -CAfile chainCA.crt <cluster
 ID>_CustomerHsmCertificate.crt
<cluster ID>_CustomerHsmCertificate.crt: OK
```

Dopo aver prodotto il certificato HSM firmato con la tua CA intermedia, vai a. Passaggio 4. Inizializzazione del cluster

Passaggio 4. Inizializzazione del cluster

Utilizza il certificato firmato dell'HSM e il certificato di firma per inizializzare il cluster. Puoi usare la AWS CloudHSM console AWS CLI, l'o l' AWS CloudHSM API.

Console

Per inizializzare un cluster (console)

1. Apri la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Seleziona il pulsante di opzione accanto all'ID del cluster con l'HSM che desideri verificare.
3. Seleziona Azioni. Dal menu a tendina, scegli Inizializza.
4. Se non hai completato il passaggio precedente per creare un HSM, scegli una zona di disponibilità (AZ) per l'HSM che stai creando. Quindi seleziona Crea.

5. Nella pagina Scarica richiesta di firma del certificato, scegli Successivo. Se l'opzione Successivo non è disponibile, scegli innanzitutto uno dei collegamenti alla CSR o al certificato. Quindi scegli Successivo.
6. Nella pagina Firma richiesta di firma del certificato (CSR), scegli Successivo.
7. Nella pagina Carica certificati, procedi come segue:

   a. Accanto a Certificato cluster, scegli Carica file. Quindi, individua e seleziona il certificato dell'HSM firmato in precedenza. Se sono state effettuate le fasi riportate nella sezione precedente, seleziona il file denominato *<cluster ID>*_CustomerHsmCertificate.crt.

   b. Accanto a Certificazione, scegli Carica file. Quindi seleziona il certificato di firma in base all'approccio che hai scelto:

      - Se hai scelto l'opzione A (certificato autofirmato singolo): seleziona il file denominato *<customerRootCA>*.crt
      - Se hai scelto l'opzione B (catena di certificati): seleziona il file denominato *<chainCA>*.crt

   c. Scegli Carica e inizializza.

AWS CLI

Per inizializzare un cluster (AWS CLI)

- Al prompt dei comandi, esegui il comando initialize-cluster. Specifica quanto segue:
  - L'ID del cluster creato in precedenza.
  - Il certificato dell'HSM che hai firmato in precedenza. Se sono state effettuate le fasi riportate nella sezione precedente, quest'ultimo è salvato in un file denominato *<cluster ID>*_CustomerHsmCertificate.crt.
  - Il tuo certificato di firma in base all'approccio che hai scelto:
    - Se hai scelto l'opzione A (certificato autofirmato singolo): utilizza il file denominato *<customerRootCA>*.crt
    - Se hai scelto l'opzione B (catena di certificati): usa il file denominato *<chainCA>*.crt

```
$ aws cloudhsmv2 initialize-cluster --cluster-id <cluster ID> \
```

```
                                    --signed-cert file://<cluster
 ID>_CustomerHsmCertificate.crt \
                                    --trust-anchor file://<customerRootCA.crt OR
 chainCA.crt>
{
    "State": "INITIALIZE_IN_PROGRESS",
    "StateMessage": "Cluster is initializing. State will change to INITIALIZED
 upon completion."
}
```

AWS CloudHSM API

Per inizializzare un cluster (AWS CloudHSM API)

- Invia una richiesta InitializeCluster con quanto segue:
  - L'ID del cluster creato in precedenza.
  - Il certificato dell'HSM che hai firmato in precedenza. Se sono state effettuate le fasi riportate nella sezione precedente, quest'ultimo è salvato in un file denominato *<cluster ID>*_CustomerHsmCertificate.crt.
  - Il tuo certificato di firma basato sull'approccio che hai scelto:
    - Se hai scelto l'opzione A (certificato autofirmato singolo): utilizza il file denominato *<customerRootCA>*.crt
    - Se hai scelto l'opzione B (catena di certificati): usa il file denominato *<chainCA>*.crt

Installa e configura la CLI di CloudHSM

Per interagire con l'HSM nel tuo AWS CloudHSM cluster, hai bisogno della CLI CloudHSM.

Connettiti all'istanza del tuo client ed esegui i seguenti comandi per scaricare e installare gli strumenti da riga di comando. AWS CloudHSM Per ulteriori informazioni, consulta Avvia un'istanza client Amazon EC2 con cui interagire AWS CloudHSM.

Amazon Linux 2023

Amazon Linux 2023 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

Amazon Linux 2023 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Amazon Linux 2 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-
latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.x86_64.rpm
```

Amazon Linux 2 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-
latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

RHEL 10 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-
cli-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.x86_64.rpm
```

RHEL 10 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

RHEL 9 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.x86_64.rpm
```

RHEL 9 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

RHEL 8 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.x86_64.rpm
```

RHEL 8 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.aarch64.rpm
```

Ubuntu 26.04 LTS

Ubuntu 26.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-cli_latest_u26.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u26.04_amd64.deb
```

Ubuntu 26.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-cli_latest_u26.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u26.04_arm64.deb
```

Ubuntu 24.04 LTS

Ubuntu 24.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-
cli_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_amd64.deb
```

Ubuntu 24.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-
cli_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Ubuntu 22.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-
cli_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_amd64.deb
```

Ubuntu 22.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-cli_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_arm64.deb
```

Windows Server 2022

Per Windows Server 2022 su architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2019

Per Windows Server 2019 su architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2016

Per Windows Server 2016 su architettura x86_64, apri PowerShell come amministratore ed esegui il comando seguente:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /
quiet /norestart /log C:\client-install.txt' -Wait
```

Usa i seguenti comandi per configurare la CLI di CloudHSM.

Per eseguire il bootstrap di un'istanza EC2 Linux per Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the
 HSMs>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP dei moduli HSM presenti nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI
 IPv4 / IPv6 addresses of the HSMs>
```

Attiva il cluster in AWS CloudHSM

Quando si attiva un AWS CloudHSM cluster, lo stato del cluster cambia da inizializzato ad attivo. A quel punto puoi gestire gli utenti del modulo di sicurezza hardware (HSM) e utilizzare l'HSM.

Important

Prima di poter attivare il cluster, è necessario prima copiare il certificato di emissione nella posizione predefinita per la piattaforma su ogni EC2 istanza che si connette al cluster (il certificato di emissione viene creato quando si inizializza il cluster). Utilizza il file di certificato appropriato in base all'approccio scelto durante l'inizializzazione del cluster:

- Se hai scelto l'opzione A (certificato autofirmato singolo): Copia `customerRootCA.crt`
- Se hai scelto l'opzione B (catena di certificati): Copia `chainCA.crt`

Posizione Linux:

```
/opt/cloudhsm/etc/<customerRootCA.crt OR chainCA.crt>
```

Posizione Windows:

```
C:\ProgramData\Amazon\CloudHSM\<customerRootCA.crt OR chainCA.crt>
```

Dopo aver copiato il file del certificato, modificalo `/opt/cloudhsm/etc/cloudhsm-cli.cfg` per assicurarti che il nome del file del certificato corrisponda al nome del certificato CA che hai copiato.

Dopo aver inserito il certificato di emissione, installa la CLI di CloudHSM ed esegui il comando cluster activate sul tuo primo HSM. Noterai che l'account di amministratore sul primo HSM del cluster ha il ruolo di admin non attivato. Si tratta di un ruolo temporaneo che esiste solo prima dell'attivazione del cluster. Quando attivi il cluster, il ruolo di admin non attivato diventa admin.

Per attivare un cluster

1. Connettersi all'istanza del client avviata in precedenza. Per ulteriori informazioni, vedi Avvia un'istanza client Amazon EC2 con cui interagire AWS CloudHSM. È possibile avviare un'istanza Linux o Windows Server.
2. Esegui la CLI di CloudHSM in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

3. (Facoltativo) Usa il comando user list per visualizzare gli utenti esistenti.

   ```
   aws-cloudhsm > user list
   {
     "error_code": 0,
     "data": {
       "users": [
   ```

```
      {
        "username": "admin",
        "role": "unactivated-admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

4. Usa il comando cluster activate per impostare la password di admin iniziale.

```
aws-cloudhsm > cluster activate
Enter
 password:<NewPassword>
Confirm password:<NewPassword>
{
  "error_code": 0,
  "data": "Cluster activation successful"
}
```

È consigliabile annotare la nuova password su un foglio di lavoro delle password. Non perdere il foglio di lavoro. È consigliabile stampare una copia del foglio di lavoro delle password, utilizzarlo per registrare le password HSM critiche e quindi conservarlo in un luogo sicuro. Si consiglia inoltre di conservare una copia di questo foglio di lavoro in un posto sicuro fuori sede.

5. (Facoltativo) Usa il comando user list per verificare che il tipo di utente sia stato modificato in admin/CO.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
```

```
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
       {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

6. Usa il comando quit per chiudere la CLI di CloudHSM.

```
aws-cloudhsm > quit
```

Per ulteriori informazioni sull'utilizzo della CLI di CloudHSM o della CMU, vedi Capire gli utenti HSM e Capire la gestione degli utenti HSM con CMU.

Configura TLS reciproco tra client e AWS CloudHSM (consigliato)

I seguenti argomenti descrivono i passaggi da completare per abilitare il Mutual Transport Layer Security (MTL) tra client e. AWS CloudHSM

Considerazioni

- Attualmente questa funzionalità è disponibile esclusivamente su hsm2m.medium. Per ulteriori informazioni sul tipo di HSM, vedere. AWS CloudHSM modalità cluster
- mTLS non è supportato per i AWS CloudHSM key store utilizzati con. AWS Key Management Service

Argomenti

- Passaggio 1. Crea e registra un trust anchor sull'HSM
- Passaggio 2. Abilita MTL per AWS CloudHSM
- Fase 3. Imposta l'applicazione MTLS per AWS CloudHSM

Passaggio 1. Crea e registra un trust anchor sull'HSM

Un trust anchor deve essere creato e registrato nell'HSM prima di abilitare gli MTL. Si tratta di un processo in due fasi:

Argomenti

- Crea una chiave privata e un certificato radice autofirmato
- Registra il trust anchor sull'HSM

Crea una chiave privata e un certificato radice autofirmato

> **Note**
>
> Per i cluster di produzione, la chiave deve essere creata in modo sicuro tramite una fonte attendibile di casualità. Ti consigliamo di utilizzare un HSM offline e fuori sede protetto o un equivalente. Archivia la chiave in modo sicuro.
> Per lo sviluppo e il test, puoi utilizzare qualsiasi strumento utile (come OpenSSL) per creare la chiave e firmare automaticamente un certificato root. Avrai bisogno della chiave e del certificato root per firmare il certificato client nell'Enable MTLs for. AWS CloudHSM

Gli esempi seguenti mostrano come creare una chiave privata e un certificato root autofirmato con OpenSSL.

Example- Creazione di una chiave privata con OpenSSL

Usa il comando seguente per creare una chiave RSA a 4096 bit crittografata con l'algoritmo AES-256. Per utilizzare questo esempio, sostituiscilo *<mtls_ca_root_1.key>* con il nome del file in cui desideri memorizzare la chiave.

```
$ openssl genrsa -out <mtls_ca_root_1.key> -aes256 4096
Generating RSA private key, 4096 bit long modulus
```

```
.....................................+++
.+++
e is 65537 (0x10001)
Enter pass phrase for mtls_ca_root_1.key:
Verifying - Enter pass phrase for mtls_ca_root_1.key:
```

Example— Creare un certificato root autofirmato con OpenSSL

Usa il seguente comando per creare un certificato radice autofirmato denominato `mtls_ca_root_1.crt` dalla chiave privata che hai appena creato. Il certificato è valido per 25 anni (9130 giorni). Leggi le istruzioni a video e segui i prompt.

```
$ openssl req -new -x509 -days 9130 -key mtls_ca_root_1.key -out mtls_ca_root_1.crt
Enter pass phrase for mtls_ca_root_1.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Registra il trust anchor sull'HSM

Dopo aver creato un certificato radice autofirmato, l'amministratore deve registrarlo come riferimento di fiducia presso il cluster. AWS CloudHSM

Per registrare un trust anchor con l'HSM

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Usa il Registra un trust anchor con la CLI di CloudhSM comando per registrare il trust anchor. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help cluster mtls register-trust-anchor.

Example— Registra un trust anchor con cluster AWS CloudHSM

L'esempio seguente mostra come utilizzare il cluster mtls register-trust-anchor comando nella CLI di CloudHSM per registrare un trust anchor sull'HSM. Per utilizzare questo comando, l'amministratore deve aver eseguito l'accesso all'HSM. Sostituire questi valori con i propri valori:

```
aws-cloudhsm > cluster mtls register-trust-anchor --path </path/mtls_ca_root_1.crt>
{
  "error_code": 0,
  "data": {
    "trust_anchor": {
      "certificate-reference": "0x01",
      "certificate": "<PEM Encoded Certificate>",
      "cluster-coverage": "full"
    }
  }
}
```

> **Note**
>
> AWS CloudHSM supporta la registrazione di certificati intermedi come trust anchor. In questi casi, l'intero file della catena di certificati con codifica PEM deve essere registrato nell'HSM, con i certificati in ordine gerarchico.
> AWS CloudHSM supporta una catena di certificati di 6980 byte.

Dopo aver registrato con successo il trust anchor, puoi eseguire il cluster mtls list-trust-anchors comando per controllare gli attuali trust anchor registrati, come mostrato di seguito:

```
aws-cloudhsm > cluster mtls list-trust-anchors
{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

> **Note**
>
> Il numero massimo di trust anchors che è possibile registrare su hsm2m.medium è due (2).

Passaggio 2. Abilita MTL per AWS CloudHSM

Per abilitare gli MTL per AWS CloudHSM, è necessario creare una chiave privata e un certificato client firmato dal certificato radice che abbiamo generato in Create and register a trust anchor sull'HSM, quindi utilizzare uno qualsiasi degli strumenti di configurazione di Client SDK 5 per configurare il percorso della chiave privata e il percorso della catena di certificati client.

Argomenti

- Crea una chiave privata e una catena di certificati client

- Configura MTL per Client SDK 5

Crea una chiave privata e una catena di certificati client

Example- Creazione di una chiave privata con OpenSSL

Utilizzate il seguente comando per creare una chiave RSA a 4096 bit. Per utilizzare questo esempio, sostituitelo *<ssl-client.key>* con il nome del file in cui desiderate memorizzare la chiave.

```
$ openssl genrsa -out <ssl-client.key> 4096
Generating RSA private key, 4096 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
```

Example— Generazione di una richiesta di firma del certificato (CSR) con OpenSSL

Usa il seguente comando per generare una richiesta di firma del certificato (CSR) dalla chiave privata che hai appena creato. Leggi le istruzioni a video e segui i prompt.

```
$ openssl req -new -key <ssl-client.key> -out <ssl-client.csr>
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Example— Firma la CSR con il certificato principale

Utilizzate il seguente comando per firmare la CSR con il certificato root che abbiamo creato e registrato in Create e registrate un trust anchor sull'HSM e create un certificato client denominato. `ssl-client.crt` Il certificato è valido per 5 anni (1826 giorni).

```
$ openssl x509 -req -days 1826 -in <ssl-client.csr> -CA <mtls_ca_root_1.crt> -
CAkey <mtls_ca_root_1.key> -CAcreateserial -out <ssl-client.crt>
```

Example— Creare una catena di certificati client

Utilizzate il seguente comando per combinare il certificato client e il certificato radice che abbiamo creato e registrato in Create and register a trust anchor sull'HSM e creare una catena di certificati client denominata`ssl-client.pem`, che verrà utilizzata per la configurazione nel passaggio successivo.

```
$ cat <ssl-client.crt> <mtls_ca_root_1.crt> > <ssl-client.pem>
```

> **Note**
>
> Se hai registrato certificati intermedi in Create e hai registrato un trust anchor sull'HSM come trust anchor, assicurati di combinare il certificato client con l'intera catena di certificati per creare una catena di certificati client.

Configura MTL per Client SDK 5

Utilizza uno qualsiasi degli strumenti di configurazione di Client SDK 5 per abilitare il TLS reciproco fornendo il percorso chiave del client e il percorso della catena di certificati client corretti. Per ulteriori informazioni sullo strumento di configurazione per Client SDK 5, consulta. ???

PKCS #11 library

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

   ```
   $ sudo cp ssl-client.pem </opt/cloudhsm/etc>
   $ sudo cp ssl-client.key </opt/cloudhsm/etc>
   ```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

   ```
   $ sudo /opt/cloudhsm/bin/configure-pkcs11 \
   ```

```
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

OpenSSL Dynamic Engine

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Key Storage Provider (KSP)

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

JCE provider

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

CloudHSM CLI

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

Fase 3. Imposta l'applicazione MTLS per AWS CloudHSM

Dopo la configurazione con uno qualsiasi degli strumenti di configurazione di Client SDK 5, la connessione tra client e TLS AWS CloudHSM avverrà nel cluster. Tuttavia, la rimozione del percorso della chiave privata e del percorso della catena di certificati client dal file di configurazione trasformerà nuovamente la connessione in un normale TLS. Puoi utilizzare la CLI di CloudHSM per impostare l'applicazione mtls nel cluster completando i seguenti passaggi:

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
```

```
  }
}
```

> **Note**
>
> 1. Assicurati di aver configurato la CLI di CloudHSM e di avviare la CLI di CloudHSM con una connessione mTLS.
> 2. È necessario accedere come utente amministratore predefinito con nome utente come amministratore prima di impostare l'applicazione di mTLS.

3. Utilizzate il Imposta il livello di applicazione dell'MTLS con la CLI di CloudhSM comando per impostare l'applicazione. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help cluster mtls set-enforcement.

   Example— Imposta l'applicazione dell'MTLS con cluster AWS CloudHSM

   L'esempio seguente mostra come utilizzare il cluster mtls set-enforcement comando nella CLI di CloudHSM per impostare l'applicazione mTLS con l'HSM. Per utilizzare questo comando, l'amministratore con nome utente come amministratore deve accedere all'HSM.

```
aws-cloudhsm > cluster mtls set-enforcement --level cluster
{
  "error_code": 0,
  "data": {
    "message": "Mtls enforcement level set to Cluster successfully"
  }
}
```

> **Warning**
>
> Dopo aver imposto l'utilizzo di MTLS nel cluster, tutte le connessioni non MTLS esistenti verranno interrotte e sarà possibile connettersi al cluster solo con certificati MTLS.

Crea e usa le chiavi in AWS CloudHSM

Prima di poter creare e utilizzare le chiavi nel tuo nuovo cluster, crea un utente del modulo di sicurezza hardware (HSM) con la CLI di AWS CloudHSM. Per ulteriori informazioni, consulta

Understanding HSM User Management Tasks, Getting started with AWS CloudHSMCommand Line Interface (CLI) e Come gestire gli utenti HSM.

> **Note**
>
> Se si utilizza il Client SDK 3, utilizza CloudHSM Management Utility (CMU) anziché la CLI del CloudHSM.

Dopo aver creato gli utenti HSM, puoi accedere all'HSM e gestire le chiavi utilizzando una di queste opzioni:

- Usa l'utility di gestione delle chiavi, uno strumento da riga di comando
- Crea un'applicazione C usando la libreria PKCS #11
- Crea un'applicazione Java utilizzando il provider JCE
- Usa OpenSSL Dynamic Engine direttamente dalla riga di comando
- Usa OpenSSL Dynamic Engine per l'offload TLS con i server web NGINX e Apache
- Utilizza il Key Storage Provider (KSP) per AWS CloudHSM Microsoft Windows Server Certificate Authority (CA)
- Usa il Key Storage Provider (KSP) per AWS CloudHSM con Microsoft Sign Tool
- Utilizza il Key Storage Provider (KSP) per l'offload TLS con il server Web Internet Information Server (IIS)

Le migliori pratiche per AWS CloudHSM

Segui le best practice descritte in questo argomento per AWS CloudHSM utilizzarle in modo efficace.

Indice

- AWS CloudHSM best practice per la gestione dei cluster
- AWS CloudHSM best practice per la gestione degli utenti
- AWS CloudHSM best practice per la gestione delle chiavi
- AWS CloudHSM best practice per l'integrazione delle applicazioni
- AWS CloudHSM best practice di monitoraggio

AWS CloudHSM best practice per la gestione dei cluster

Segui le best practice in questa sezione per la creazione, l'accesso e la AWS CloudHSM gestione del cluster.

Scala il tuo cluster per gestire i picchi di traffico

Diversi fattori possono influenzare il throughput massimo che il cluster è in grado di gestire, tra cui la dimensione dell'istanza del client, la dimensione del cluster, la topografia della rete e le operazioni crittografiche necessarie per il caso d'uso.

Come punto di partenza, consulta l'argomento AWS CloudHSM informazioni sulle prestazioni per le stime delle prestazioni sulle dimensioni e le configurazioni comuni dei cluster. Ti consigliamo di eseguire un test di carico del cluster con il carico di picco previsto per determinare se l'architettura attuale è resiliente e sulla giusta scala.

Progetta il tuo cluster per garantire un'elevata disponibilità

Aggiungi ridondanza per tenere conto della manutenzione: AWS può sostituire il tuo HSM per la manutenzione programmata o se rileva un problema. Come regola generale, la dimensione del cluster deve avere una ridondanza di almeno +1. Ad esempio, se hai bisogno di due HSM per il funzionamento del servizio nelle ore di punta, la dimensione ideale del cluster sarà tre. Se seguite le migliori pratiche relative alla disponibilità, queste sostituzioni HSM non dovrebbero influire sul servizio. Tuttavia, le operazioni in corso sull'HSM sostituito potrebbero fallire e devono essere riprovate.

Distribuisci i tuoi HSM in molte zone di disponibilità: valuta in che modo il tuo servizio sarà in grado di funzionare durante un'interruzione della zona di disponibilità. AWS consiglia di distribuire gli HSM nel maggior numero possibile di zone di disponibilità. Per un cluster con tre HSM, è necessario distribuire gli HSM su tre zone di disponibilità. A seconda del sistema, potrebbe essere necessaria una ridondanza aggiuntiva.

Disponi di almeno tre HSM per garantire la durata delle chiavi appena generate

Per le applicazioni che richiedono la durabilità delle chiavi appena generate, consigliamo di disporre di almeno tre HSM distribuiti in diverse zone di disponibilità in una regione.

Accesso sicuro al tuo cluster

Usa le sottoreti private per limitare l'accesso alla tua istanza: avvia i tuoi HSM e le istanze client nelle sottoreti private del tuo VPC. Questo limita l'accesso ai tuoi HSM dal mondo esterno.

Usa gli endpoint VPC per accedere alle API: il piano AWS CloudHSM dati è stato progettato per funzionare senza bisogno di accedere a Internet o alle API AWS. Se l'istanza del cliente richiede l'accesso all' AWS CloudHSM API, puoi utilizzare gli endpoint VPC per accedere all'API senza richiedere l'accesso a Internet sull'istanza client. Per ulteriori informazioni, consulta AWS CloudHSM ed endpoint VPC.

Riduci i costi adattandoli alle tue esigenze

Non ci sono costi iniziali da utilizzare. AWS CloudHSM Paghi una tariffa oraria per ogni HSM avviato fino alla chiusura dell'HSM. Se il servizio non richiede un uso continuo AWS CloudHSM, è possibile ridurre i costi riducendo (eliminando) gli HSM a zero quando non sono necessari. Quando gli HSM sono nuovamente necessari, è possibile ripristinarli da un backup. Se, ad esempio, hai un carico di lavoro che richiede la firma del codice una volta al mese, in particolare l'ultimo giorno del mese, puoi prima scalare il cluster, ridimensionarlo eliminando gli HSM una volta completato il lavoro e quindi ripristinare il cluster per eseguire nuovamente le operazioni di firma alla fine del mese successivo.

AWS CloudHSM esegue automaticamente backup periodici degli HSM nel cluster. Quando si aggiunge un nuovo HSM in un secondo momento, AWS CloudHSM ripristinerà il backup più recente sul nuovo HSM in modo da poter riprendere l'utilizzo dallo stesso punto in cui l'hai lasciato. Per calcolare i costi AWS CloudHSM dell'architettura, consulta AWS CloudHSM Prezzi.

Risorse correlate:

- Panoramica generale dei backup
- Politica di conservazione dei backup
- Copia dei backup dei AWS CloudHSM cluster tra le regioni AWS

AWS CloudHSM best practice per la gestione degli utenti

Segui le best practice in questa sezione per gestire in modo efficace gli utenti del tuo AWS CloudHSM cluster. Gli utenti HSM sono diversi dagli utenti IAM. Gli utenti e le entità IAM che dispongono di una policy basata sull'identità con le autorizzazioni appropriate possono creare HSM interagendo con le risorse tramite l'API AWS. Dopo aver creato l'HSM, devi utilizzare le credenziali utente HSM per autenticare le operazioni sull'HSM. Per una guida dettagliata degli utenti HSM, consulta. Utenti HSM in AWS CloudHSM

Proteggi le credenziali dei tuoi utenti HSM

È fondamentale proteggere le credenziali degli utenti HSM in modo sicuro poiché gli utenti HSM sono le entità che possono accedere ed eseguire operazioni crittografiche e di gestione sull'HSM. AWS CloudHSM non ha accesso alle tue credenziali utente HSM e non sarà in grado di assisterti se perdi l'accesso ad esse.

Disponi di almeno due amministratori per evitare il blocco

Per evitare di rimanere esclusi dal cluster, ti consigliamo di avere almeno due amministratori nel caso in cui venga persa una password di amministratore. In tal caso, puoi utilizzare l'altro amministratore per reimpostare la password.

> **Note**
>
> Gli amministratori in Client SDK 5 sono sinonimo di responsabili delle criptovalute (CO) in Client SDK 3.

Abilita il quorum per tutte le operazioni di gestione degli utenti

Il quorum consente di impostare un numero minimo di amministratori che devono approvare un'operazione di gestione degli utenti prima che tale operazione possa aver luogo. A causa del privilegio di cui dispongono gli amministratori, ti consigliamo di abilitare il quorum per tutte

le operazioni di gestione degli utenti. Questo può limitare il potenziale impatto se una delle tue password di amministratore viene compromessa. Per ulteriori informazioni, consulta Managing Quorum.

Crea più utenti crittografici, ciascuno con autorizzazioni limitate

Separando le responsabilità degli utenti crittografici, nessun utente ha il controllo totale sull'intero sistema. Per questo motivo, ti consigliamo di creare più utenti crittografici e di limitare le autorizzazioni di ciascuno. In genere, ciò avviene assegnando ai diversi utenti crittografici responsabilità e azioni nettamente diverse da eseguire (ad esempio, avere un utente crittografico responsabile della generazione e della condivisione delle chiavi con altri utenti crittografici che poi le utilizzano nell'applicazione).

Risorse correlate:

- Condividi una chiave utilizzando la CLI di CloudHSM
- Annullare la condivisione di una chiave utilizzando la CLI di CloudhSM

AWS CloudHSM best practice per la gestione delle chiavi

Segui le best practice in questa sezione per la gestione delle chiavi. AWS CloudHSM

Scegli il tipo di chiave giusto

Quando si utilizza una chiave di sessione, le transazioni al secondo (TPS) saranno limitate a un HSM in cui esiste la chiave. Gli HSM aggiuntivi nel cluster non aumenteranno il throughput delle richieste per quella chiave. Se utilizzi una chiave token per la stessa applicazione, il carico delle richieste verrà bilanciato tra tutti gli HSM disponibili nel cluster. Per ulteriori informazioni, consulta Impostazioni di sincronizzazione e durabilità dei tasti in AWS CloudHSM.

Gestisci i limiti di archiviazione delle chiavi

Gli HSM hanno limiti al numero massimo di token e chiavi di sessione che possono essere archiviati su un HSM contemporaneamente. Per informazioni sui limiti di archiviazione delle chiavi, vedere. AWS CloudHSM quote Se la tua applicazione richiede più del limite, puoi utilizzare una o più delle seguenti strategie per gestire efficacemente le chiavi:

Utilizza Trusted Key Wrapping per archiviare le tue chiavi in un archivio dati esterno: utilizzando Trusted Key Wrapping, puoi superare il limite di archiviazione delle chiavi archiviando tutte le tue

chiavi in un archivio dati esterno. Quando è necessario utilizzare questa chiave, è possibile estrarla nell'HSM come chiave di sessione, utilizzarla per l'operazione richiesta e quindi scartare la chiave di sessione. I dati chiave originali rimangono archiviati in modo sicuro nel tuo archivio dati per essere utilizzati ogni volta che ne hai bisogno. L'utilizzo di chiavi affidabili a tale scopo massimizza la protezione.

Distribuisci le chiavi tra i cluster: un'altra strategia per superare il limite di archiviazione delle chiavi consiste nell'archiviare le chiavi in più cluster. In questo approccio, si mantiene una mappatura delle chiavi archiviate in ogni cluster. Usa questa mappatura per indirizzare le richieste dei tuoi clienti al cluster con la chiave richiesta. Per informazioni su come connettersi a più cluster dalla stessa applicazione client, consulta i seguenti argomenti:

- Connessione a più AWS CloudHSM cluster con il provider JCE
- Configurazione a più slot con libreria PKCS #11 per AWS CloudHSM

Gestione e protezione del key wrapping

Le chiavi possono essere contrassegnate come estraibili o non estraibili tramite l'attributo. `EXTRACTABLE` Per impostazione predefinita, le chiavi HSM sono contrassegnate come estraibili.

Le chiavi estraibili sono chiavi che possono essere esportate dall'HSM tramite key wrapping. Le chiavi racchiuse sono crittografate e devono essere scartate utilizzando la stessa chiave di wrapping prima di poter essere utilizzate. Non-extractable le chiavi non possono essere esportate dall'HSM in nessun caso. Non è possibile rendere estraibile una chiave non estraibile. Per questo motivo, è importante considerare se è necessario che le chiavi siano estraibili o meno e impostare di conseguenza l'attributo chiave corrispondente.

Se avete bisogno del key wrapping nella vostra applicazione, dovreste utilizzare il trusted key wrapping per limitare la capacità degli utenti HSM alle sole wrap/unwrap chiavi che sono state esplicitamente contrassegnate come attendibili da un amministratore. Per ulteriori informazioni, consulta gli argomenti relativi all'inserimento di chiavi affidabili. Chiavi in AWS CloudHSM

Risorse correlate

- Funzioni di wrapping e annullamento del wrapping
- Funzioni di cifratura per JCE
- Attributi chiave Java supportati per AWS CloudHSM Client SDK 5
- Attributi chiave per la CLI di CloudHSM

AWS CloudHSM best practice per l'integrazione delle applicazioni

Segui le best practice in questa sezione per ottimizzare il modo in cui l'applicazione si integra con il AWS CloudHSM cluster.

Avvia il tuo Client SDK

Prima che il client SDK possa connettersi al cluster, è necessario avviarlo. Quando esegui il bootstrap degli indirizzi IP nel cluster, ti consigliamo di utilizzare il parametro quando possibile. `--cluster-id` Questo metodo compila la configurazione con tutti gli indirizzi IP HSM presenti nel cluster senza dover tenere traccia di ogni singolo indirizzo. In questo modo si aumenta la resilienza dell'inizializzazione dell'applicazione nel caso in cui un HSM sia sottoposto a manutenzione o durante un'interruzione della zona di disponibilità. Per ulteriori dettagli, consultare Esegui il bootstrap di Client SDK.

Effettua l'autenticazione per eseguire le operazioni

In AWS CloudHSM, è necessario autenticarsi nel cluster prima di poter eseguire la maggior parte delle operazioni, ad esempio le operazioni crittografiche.

Autenticazione con CloudHSM CLI: puoi autenticarti con CloudHSM CLI utilizzando la modalità a comando singolo o la modalità interattiva. ??? ??? Usa il comando per autenticarti in modalità interattiva. Accedi a un HSM utilizzando CloudHSM CLI Per autenticarsi in modalità comando singolo, è necessario impostare le variabili `CLOUDHSM_ROLE` ambientali e. `CLOUDHSM_PIN` Per ulteriori informazioni su questa operazione, fare riferimento aModalità di comando singolo. AWS CloudHSM consiglia di archiviare in modo sicuro le credenziali HSM quando non vengono utilizzate dall'applicazione.

Autenticazione con PKCS #11: in PKCS #11, si effettua il login utilizzando l'API C_Login dopo aver aperto una sessione utilizzando C_. OpenSession È necessario eseguire solo un C_Login per slot (cluster). Dopo aver effettuato correttamente l'accesso, è possibile aprire sessioni aggiuntive utilizzando C_ OpenSession senza la necessità di eseguire ulteriori operazioni di accesso. Per esempi sull'autenticazione in PKCS #11, vedere. Esempi di codice per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

Autenticazione con JCE: il provider JCE supporta l' AWS CloudHSM accesso implicito ed esplicito. Il metodo più adatto a te dipende dal tuo caso d'uso. Quando possibile, consigliamo di utilizzare l'accesso implicito perché l'SDK gestirà automaticamente l'autenticazione se l'applicazione viene disconnessa dal cluster e deve essere nuovamente autenticata. L'utilizzo dell'accesso implicito

consente inoltre di fornire credenziali all'applicazione quando si utilizza un'integrazione che non consente di avere il controllo sul codice dell'applicazione. Per ulteriori informazioni sui metodi di accesso, vedere. Passaggio 2: fornire le credenziali al provider JCE

Autenticazione con OpenSSL: con OpenSSL Dynamic Engine, fornisci le credenziali tramite variabili di ambiente. AWS CloudHSM consiglia di archiviare in modo sicuro le credenziali HSM quando non vengono utilizzate dall'applicazione. Se possibile, è necessario configurare l'ambiente in modo da recuperare e impostare sistematicamente queste variabili di ambiente senza immetterle manualmente. Per dettagli sull'autenticazione con OpenSSL, vedere. Installa il motore dinamico OpenSSL per AWS CloudHSM Client SDK 5

Autenticazione con KSP: è possibile autenticarsi con Key Storage Provider (KSP) utilizzando il gestore delle credenziali di Windows o le variabili di ambiente, vedere. Installa il Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5

Gestisci in modo efficace le chiavi nell'applicazione

Usa gli attributi chiave per controllare cosa possono fare le chiavi: quando generi una chiave, utilizza gli attributi chiave per definire un set di autorizzazioni che consentiranno o negheranno tipi specifici di operazioni per quella chiave. Consigliamo di generare le chiavi con il minor numero di attributi necessari per completare l'attività. Ad esempio, una chiave AES utilizzata per la crittografia non dovrebbe essere autorizzata anche a estrarre le chiavi dall'HSM. Per ulteriori informazioni, consulta le nostre pagine degli attributi per i seguenti SDK client:

- Attributi chiave PKCS #11
- Attributi chiave JCE

Quando possibile, memorizza nella cache gli oggetti chiave per ridurre al minimo la latenza: le operazioni di key find interrogheranno tutti gli HSM del cluster. Questa operazione è costosa e non è scalabile in base al numero di HSM presenti nel cluster.

- Con PKCS #11, puoi trovare le chiavi utilizzando l'`C_FindObjects`API.
- Con JCE, trovi le chiavi usando il. KeyStore

Per prestazioni ottimali, AWS consiglia di utilizzare i comandi key find (ad esempio Cerca AWS CloudHSM le chiavi per attributi usando KMU eElenca le chiavi per un utente con CLI CloudhSM) una sola volta durante l'avvio dell'applicazione e di memorizzare nella cache l'oggetto chiave

restituito nella memoria dell'applicazione. Se richiedete questo oggetto chiave in un secondo momento, dovreste recuperarlo dalla cache anziché eseguire una query per questo oggetto per ogni operazione, il che comporterà un notevole sovraccarico in termini di prestazioni.

Leggi gli attributi dell'oggetto con il minor numero di chiamate possibile

La lettura degli attributi invia richieste al cluster e il numero di richieste dipende dall'SDK utilizzato:

- Con la libreria PKCS #11, ogni `C_GetAttributeValue` chiamata invia una richiesta, indipendentemente dal numero di attributi contenuti nel modello. Richiedi tutti gli attributi di cui hai bisogno in un modello. Per ulteriori informazioni, consulta Recupera gli attributi con la libreria PKCS #11 per AWS CloudHSM Client SDK 5.
- Con JCE, ogni attributo richiesto costa una richiesta. Richiedi solo gli attributi necessari alla tua applicazione.

In entrambi i casi, memorizzate nella cache i valori dell'applicazione e riutilizzateli finché qualcosa non modifica l'oggetto.

Usa il multithreading

AWS CloudHSM supporta le applicazioni multithread, ma ci sono alcuni aspetti da tenere a mente con le applicazioni multithread.

Con PKCS #11, è necessario inizializzare la libreria PKCS #11 (chiamata) una sola volta. `C_Initialize` A ogni thread deve essere assegnata una propria sessione (). `C_OpenSession` Non è consigliabile utilizzare la stessa sessione in più thread.

Con JCE, il AWS CloudHSM provider deve essere inizializzato una sola volta. Non condividere istanze di oggetti SPI tra thread. Ad esempio, Cipher, Signature, Digest, Mac KeyFactory o KeyGenerator gli oggetti devono essere utilizzati solo nel contesto del proprio thread.

Gestisci gli errori di limitazione

Potrebbero verificarsi errori di limitazione HSM nelle seguenti circostanze:

- Il cluster non è dimensionato correttamente per gestire i picchi di traffico.
- Il cluster non è dimensionato con una ridondanza di +1 durante gli eventi di manutenzione.
- Le interruzioni della zona di disponibilità comportano una riduzione del numero di HSM disponibili nel cluster.

Limitazione HSMPer informazioni su come gestire al meglio questo scenario, consulta.

Per garantire che il cluster sia di dimensioni adeguate e non subisca limitazioni, è consigliabile AWS eseguire il test di carico nel proprio ambiente con i picchi di traffico previsti.

Integra i tentativi sulle operazioni del cluster

AWS può sostituire il tuo HSM per motivi operativi o di manutenzione. Per rendere l'applicazione resiliente a tali situazioni, AWS consiglia di implementare la logica di ripetizione dei tentativi sul lato client in tutte le operazioni indirizzate al cluster. I tentativi successivi sulle operazioni non riuscite a causa di sostituzioni dovrebbero avere esito positivo.

Implementa strategie di disaster recovery

In risposta a un evento, potrebbe essere necessario spostare il traffico da un intero cluster o area geografica. Le sezioni seguenti descrivono diverse strategie per eseguire questa operazione.

Usa il peering VPC per accedere al tuo cluster da un altro account o regione: puoi utilizzare il peering VPC per accedere al tuo AWS CloudHSM cluster da un altro account o regione. Per informazioni su come configurarlo, vedi Cos'è il peering VPC? nella VPC Peering Guide. Dopo aver stabilito le connessioni peering e configurato i gruppi di sicurezza in modo appropriato, è possibile comunicare con gli indirizzi IP HSM nello stesso modo in cui si farebbe normalmente.

Connettiti a più cluster dalla stessa applicazione: il provider JCE, la libreria PKCS #11 e l'interfaccia a riga di comando CloudHSM in Client SDK 5 supportano la connessione a più cluster dalla stessa applicazione. Ad esempio, puoi avere due cluster attivi, ciascuno in regioni diverse, e l'applicazione può connettersi a entrambi contemporaneamente e bilanciare il carico tra i due nell'ambito delle normali operazioni. Se l'applicazione non utilizza Client SDK 5 (l'SDK più recente), non è possibile connettersi a più cluster dalla stessa applicazione. In alternativa, puoi mantenere attivo e funzionante un altro cluster e, in caso di interruzione regionale, spostare il traffico sull'altro cluster per ridurre al minimo i tempi di inattività. Per i dettagli, consulta le rispettive pagine:

- Configurazione a più slot con libreria PKCS #11 per AWS CloudHSM
- Connessione a più AWS CloudHSM cluster con il provider JCE
- Connessione a più cluster con la CLI CloudhSM

Ripristinare un cluster da un backup: è possibile creare un nuovo cluster da un backup di un cluster esistente. Per ulteriori informazioni, consulta Backup dei cluster in AWS CloudHSM.

Distribuzione graduale delle applicazioni

Segui strategie di distribuzione scaglionate, ad esempio distribuzioni basate su onde progressive, distribuzioni one-box e distribuzioni continue per le distribuzioni e i riavvii delle applicazioni client. Questo approccio riduce al minimo il potenziale impatto delle modifiche garantendo al contempo una capacità sufficiente per servire il traffico di produzione durante l'implementazione.

AWS CloudHSM best practice di monitoraggio

Questa sezione descrive diversi meccanismi che è possibile utilizzare per monitorare il cluster e l'applicazione. Per ulteriori dettagli sul monitoraggio, vedereMonitoraggio AWS CloudHSM.

Monitora i log dei client

Ogni Client SDK scrive dei log che puoi monitorare. Per informazioni sulla registrazione dei client, vedere. Utilizzo dei log SDK AWS CloudHSM del client

Su piattaforme progettate per essere temporanee, come Amazon ECS e AWS Lambda, la raccolta dei log dei clienti da un file può essere difficile. In queste situazioni, è consigliabile configurare i log di Client SDK per scrivere i log sulla console. La maggior parte dei servizi raccoglierà automaticamente questo output e lo pubblicherà CloudWatch nei log di Amazon affinché tu possa conservarli e visualizzarli.

Se utilizzi un'integrazione di terze parti oltre a AWS CloudHSM Client SDK, assicurati di configurare quel pacchetto software in modo che ne registri l'output anche sulla console. L'output di AWS CloudHSM Client SDK può essere acquisito da questo pacchetto e altrimenti scritto nel relativo file di registro.

Consulta il AWS CloudHSM Strumento di configurazione di Client SDK 5 per informazioni su come configurare le opzioni di registrazione nella tua applicazione.

Monitora i registri di controllo

AWS CloudHSM pubblica i log di controllo sul tuo CloudWatch account Amazon. I registri di controllo provengono dall'HSM e tengono traccia di determinate operazioni a fini di controllo.

È possibile utilizzare i log di controllo per tenere traccia di tutti i comandi di gestione richiamati sull'HSM. Ad esempio, è possibile attivare un allarme quando si nota l'esecuzione di un'operazione di gestione imprevista.

Per ulteriori dettagli, consulta Funzionamento del log degli audit dell'HSM.

Monitoraggio AWS CloudTrail

AWS CloudHSM è integrato con AWS CloudTrail un servizio che fornisce una registrazione delle azioni intraprese da un utente, da un ruolo o da un AWS servizio in AWS CloudHSM. AWS CloudTrail acquisisce tutte le chiamate API AWS CloudHSM come eventi. Le chiamate acquisite includono chiamate dalla AWS CloudHSM console e chiamate in codice alle operazioni AWS CloudHSM API.

Puoi utilizzarlo AWS CloudTrail per controllare qualsiasi chiamata API effettuata al piano di AWS CloudHSM controllo per assicurarti che non si verifichino attività indesiderate nel tuo account.

Per informazioni dettagliate, vedi Lavorare conAWS CloudTrailandAWS CloudHSM.

Monitora le CloudWatch metriche di Amazon

Puoi utilizzare le CloudWatch metriche di Amazon per monitorare il tuo AWS CloudHSM cluster in tempo reale. Le metriche possono essere raggruppate per regione, ID cluster o ID HSM e ID cluster.

Utilizzando Amazon CloudWatch Metrics, puoi configurare gli CloudWatch allarmi di Amazon per avvisarti di eventuali problemi che potrebbero verificarsi e che potrebbero influire sul tuo servizio. Ti consigliamo di configurare gli allarmi per monitorare quanto segue:

- Raggiungimento del limite chiave su un HSM
- Ci stiamo avvicinando al limite di numero di sessioni HSM su un HSM
- Ci stiamo avvicinando al limite di numero di utenti HSM su un HSM
- Differenze nel numero di utenti o chiavi HSM per identificare i problemi di sincronizzazione
- HSM non integri per scalare il cluster fino AWS CloudHSM a risolvere il problema

Per ulteriori dettagli, consultare Utilizzo di Amazon CloudWatch Logs e AWS CloudHSM Audit Logs.

Cluster in AWS CloudHSM

Un cluster è una raccolta di singoli moduli di sicurezza hardware (HSM) che vengono AWS CloudHSM mantenuti sincronizzati. Quando si esegue un'attività o operazione su un HSM in un cluster, gli altri HSM nel cluster vengono aggiornati automaticamente.

Puoi gestire i AWS CloudHSM cluster dalla AWS CloudHSM console o da uno degli AWS SDK o dagli strumenti a riga di comando. Per ulteriori informazioni, consulta i seguenti argomenti.

Per creare un cluster, consulta Nozioni di base.

I seguenti argomenti forniscono ulteriori informazioni sui cluster.

Argomenti

- AWS CloudHSM architettura dei cluster
- AWS CloudHSM sincronizzazione del cluster
- AWS CloudHSM elevata disponibilità e bilanciamento del carico del cluster
- AWS CloudHSM modalità cluster
- Tipi di HSM in AWS CloudHSM
- Connect l'SDK del client al AWS CloudHSM cluster
- Scalabilità degli HSM in un AWS CloudHSM cluster
- Eliminazione di un cluster AWS CloudHSM
- Creazione di AWS CloudHSM cluster dai backup
- Migrazione di tipo HSM del cluster

AWS CloudHSM architettura dei cluster

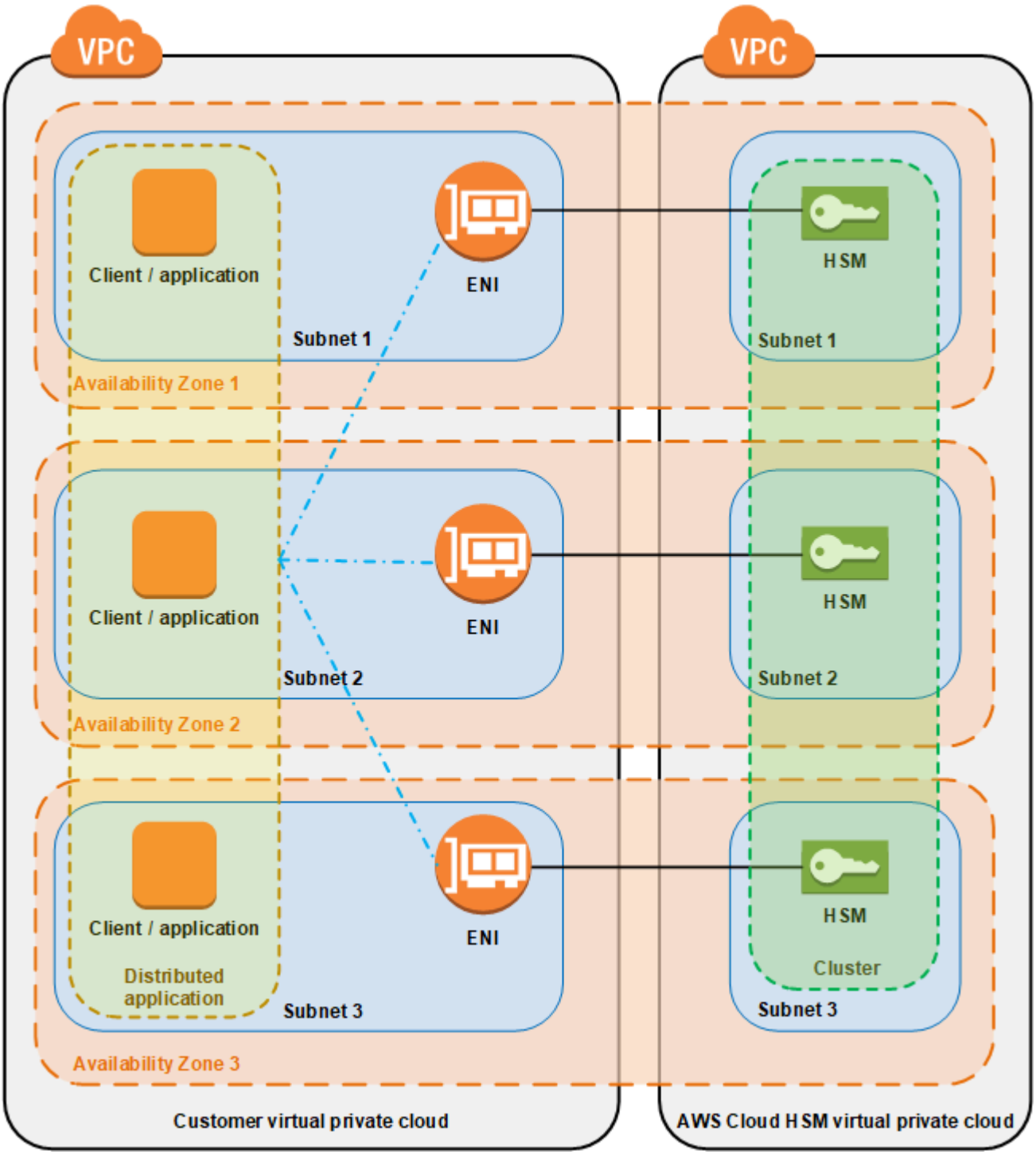
Quando crei un cluster, specifichi un Amazon Virtual Private Cloud (VPC) nel tuo AWS account e una o più sottoreti in quel VPC. Ti consigliamo di creare una sottorete in ogni zona di disponibilità (AZ) nella regione prescelta. AWS Quando crei un VPC, puoi creare le sottoreti private. Per ulteriori informazioni, consulta Crea un cloud privato virtuale (VPC) per AWS CloudHSM.

Ogni volta che si crea un HSM, è necessario specificare il cluster e la zona di disponibilità per garantire l'HSM. Inserendo gli HSM in diverse zone di disponibilità, è possibile raggiungere ridondanza ed elevata disponibilità nel caso in cui una zona di disponibilità non sia disponibile.

Quando crei un HSM, AWS CloudHSM inserisce un'elastic network interface (ENI) nella sottorete specificata del tuo AWS account. L'interfaccia di rete elastica è l'interfaccia per l'interazione con l'HSM. L'HSM risiede in un VPC separato in un AWS account di proprietà di. AWS CloudHSM L'HSM e l'interfaccia di rete corrispondente si trovano nella stessa zona di disponibilità.

Per interagire con gli HSM di un cluster, è necessario il software client. AWS CloudHSM In genere, si installa il client su istanze Amazon EC2, note come istanze client, che si trovano nello stesso VPC degli ENI dell'HSM, come illustrato nella figura che segue. Tuttavia, questa operazione non è tecnicamente necessaria. È possibile installare il client su qualsiasi computer compatibile, purché sia in grado di connettersi agli ENI dell'HSM. Il client comunica con gli HSM singoli nel cluster tramite i rispettivi ENI.

La figura seguente rappresenta un AWS CloudHSM cluster con tre moduli di protezione hardware, ciascuno in una zona di disponibilità diversa nel VPC.



AWS CloudHSM sincronizzazione del cluster

In un AWS CloudHSM cluster, AWS CloudHSM mantiene sincronizzate le chiavi sui singoli moduli di protezione hardware. Non è necessaria alcuna operazione per sincronizzare le chiavi sugli HSM. A differenza delle chiavi, non esiste un meccanismo lato server per sincronizzare gli utenti HSM nel

cluster. La AWS CloudHSM CLI esegue la sincronizzazione ottimale delle operazioni degli utenti tra gli HSM, ma possono verificarsi incongruenze se un'operazione fallisce parzialmente. Se gli utenti non sono sincronizzati, il comando mostrerà delle incongruenze. `user list` Per ulteriori informazioni, consulta L'utente o la policy di Client SDK 5 contengono valori non coerenti.

Quando si aggiunge un nuovo HSM a un cluster, AWS CloudHSM esegue un backup di tutte le chiavi, gli utenti e le politiche su un HSM esistente. Quindi, ripristina il backup nel nuovo HSM. In questo modo i due HSM sono sincronizzati.

Se le chiavi sugli HSM in un cluster non sono sincronizzate, AWS CloudHSM le risincronizza automaticamente. Per abilitare questa funzionalità, AWS CloudHSM utilizza le credenziali dell'utente dell'appliance. Questo utente esiste su tutti gli HSM forniti da AWS CloudHSM e dispone di autorizzazioni limitate. È possibile ottenere un hash di oggetti in HSM ed estrarre e inserire oggetti mascherati (crittografati). AWS non è in grado di visualizzare o modificare utenti o chiavi e non è in grado di eseguire tutte le operazioni di crittografia utilizzando tali chiavi.

Questa risincronizzazione automatica si applica solo alle chiavi. Gli utenti e le politiche (come le impostazioni mTLS) non vengono risincronizzati automaticamente. La CLI di CloudHSM esegue la sincronizzazione ottimale delle operazioni di utenti e policy tra gli HSM, ma possono comunque verificarsi incongruenze e potrebbe essere necessario risolverle manualmente. Per ulteriori informazioni, consulta L'utente o la policy di Client SDK 5 contengono valori non coerenti.

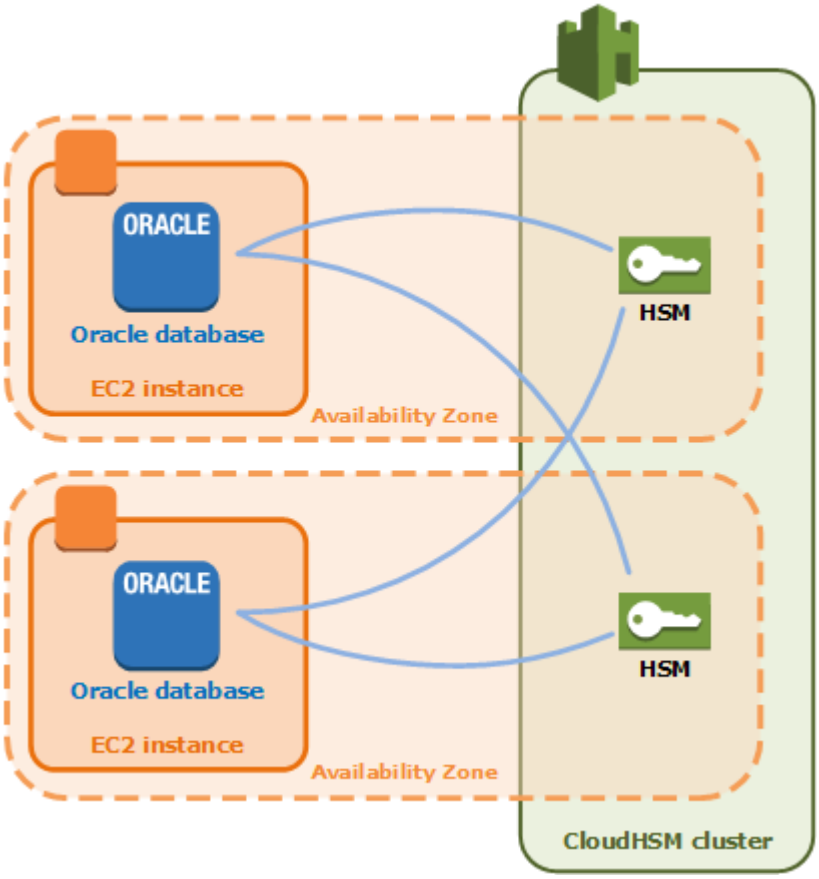
AWS CloudHSM elevata disponibilità e bilanciamento del carico del cluster

Quando si crea un AWS CloudHSM cluster con più di un HSM, si ottiene automaticamente il bilanciamento del carico. Bilanciamento del carico significa che il client AWS CloudHSM distribuisce le operazioni di crittografia su tutti gli HSM nel cluster in base alla capacità di ogni HSM di effettuare ulteriori elaborazioni.

Quando si creano gli HSM in diverse zone di AWS disponibilità, si ottiene automaticamente un'elevata disponibilità. Elevata disponibilità significa che è possibile ottenere maggiore affidabilità, perché nessun HSM singolo rappresenta un singolo punto di errore. Ti consigliamo di avere un minimo di due HSM in ogni cluster, con ogni HSM in zone di disponibilità diverse all'interno di una regione. AWS

Ad esempio, la figura riportata di seguito mostra un'applicazione del database Oracle che viene distribuito a due diverse zone di disponibilità. Le istanze del database archiviano le proprie chiavi

principali in un cluster che include un HSM in ogni zona di disponibilità. AWS CloudHSM sincronizza automaticamente le chiavi su entrambi gli HSM in modo che siano immediatamente accessibili e ridondanti.



AWS CloudHSM modalità cluster

AWS CloudHSM offre cluster in due modalità: FIPS e non FIPS. In modalità FIPS, è possibile utilizzare solo chiavi e algoritmi convalidati dal Federal Information Processing Standard (FIPS). La modalità non FIPS offre tutte le chiavi e gli algoritmi supportati da, indipendentemente dall'approvazione FIPS. AWS CloudHSM

Consulta i dettagli in questa pagina prima di decidere quale modalità cluster e tipo di HSM sono adatti alle tue esigenze.

> **Note**
>
> Tutti i cluster creati prima del 10 giugno 2024 sono in modalità FIPS e hanno un HSM di tipo hsm1.medium.

Per vedere la modalità e il tipo di HSM del cluster, usa il comando describe-clusters.

La tabella seguente elenca le principali differenze tra ciascuna modalità di cluster:

Funzione di differenziazione	modalità FIPS	Modalità non FIPS
Compatibilità di tipo HSM	Disponibile con hsm1.medium e hsm2m.medium.	Disponibile con hsm2m.medium.
Compatibilità di Backup	Può essere utilizzato solo per il backup dei cluster di ripristino in modalità FIPS.	Può essere utilizzato solo per il backup dei cluster di ripristino in modalità non FIPS.
Selezione dei tasti	Supporta la generazione e l'utilizzo di chiavi con meccanismi approvati [1] FIPS.	Supporta la generazione e l'utilizzo di chiavi con tutti i meccanismi convalidati FIPS, oltre ad altri meccanismi non convalidati.
Algoritmi	Supporta AWS CloudHSM algoritmi approvati dalla FIPS. [1]	Supporta AWS CloudHSM algoritmi approvati e non approvati FIPS.

[1] Per i dettagli, consulta Notifiche di obsolescenza.

Prima di scegliere una modalità cluster, tieni presente che la modalità di un cluster (FIPS o non FIPS) non può essere modificata dopo la sua creazione, quindi assicurati di selezionare la modalità giusta per le tue esigenze.

Tipi di HSM in AWS CloudHSM

AWS CloudHSM offre anche due tipi di moduli di sicurezza hardware (HSM): hsm1.medium e hsm2m.medium. Consulta i dettagli in questa pagina prima di decidere quale tipo di HSM è adatto alle tue esigenze.

Oltre alle modalità cluster, AWS CloudHSM offre due tipi di HSM: hsm1.medium e hsm2m.medium. Ogni tipo di HSM utilizza hardware diverso e ogni cluster può contenere solo un tipo di HSM. La tabella seguente elenca le principali differenze tra i due:

Caratteristica di differenziazione	hsm1. Medium	hsm2m. medio
Compatibilità con la modalità cluster	Disponibile per i cluster in modalità FIPS.	Disponibile per i cluster in modalità FIPS o non FIPS.
Compatibilità dei tipi di rete	Non disponibile	Disponibile per i cluster in modalità FIPS o non FIPS.
Compatibilità di Backup	Può essere utilizzato per il backup e il ripristino su cluster hsm1.medium e hsm2m.medium in modalità FIPS.	Può essere utilizzato solo per il backup e il ripristino dei cluster hsm2m.medium.
Capacità chiave	3.300 per cluster.	16.666 chiavi totali, con chiavi asimmetriche con un massimo di 3.333 per cluster.
Cliente SDKs	Supporta tutti i client SDKs.	Supporta tutti i client SDKs.
Versioni Client SDK	Compatibile con la versione SDK 3.1.0 e successive.	Compatibile con Client SDK versione 5.9.0 e successive.
Disponibilità regionale	CloudHSM non supporta più la creazione di nuovi cluster in nessuna regione. AWS Per ulteriori informazioni, consulta Notifiche di deprecazione per i dettagli.	Disponibile in quanto Regioni AWS CloudHSM è disponibile.
Prestazioni	Per vedere le prestazioni di ogni tipo di HSM, consulta. AWS CloudHSM informazioni sulle prestazioni	
Certificazione	Conforme a FIPS 140-2, PCI DSS, PCI PIN e PCI-3DS. SOC2	Conforme a FIPS 140-3, PCI DSS, PCI PIN e PCI-3DS. SOC2

Connect l'SDK del client al AWS CloudHSM cluster

Per connetterti al cluster con Client SDK 5 o Client SDK 3, devi prima svolgere due operazioni:

- Predisporre un certificato di emissione sull'istanza EC2
- Eseguire in bootstrap il Client SDK nel cluster

Colloca il certificato di emissione su ogni istanza EC2

Il certificato di emissione viene creato quando si inizializza il cluster. Copia il certificato di emissione nella posizione predefinita per la piattaforma su ogni istanza EC2 che si connette al cluster.

Linux

```
/opt/cloudhsm/etc/customerCA.crt
```

Windows

```
C:\ProgramData\Amazon\CloudHSM\customerCA.crt
```

Specifica la posizione del certificato di emissione

Con Client SDK 5, usi lo strumento di configurazione per specificare la posizione del certificato emittente.

PKCS #11 library

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --hsm-ca-cert <customerCA certificate file>
```

OpenSSL Dynamic Engine Provider

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --hsm-ca-cert <customerCA certificate file>
```

Key Storage Provider (KSP)

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --hsm-ca-cert <customerCA certificate file>
```

JCE provider

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-jce --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --hsm-ca-
cert <customerCA certificate file>
```

CloudHSM CLI

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-cli --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --hsm-ca-
cert <customerCA certificate file>
```

Per ulteriori informazioni, consulta Strumento Configure.

Per ulteriori informazioni sull'inizializzazione del cluster o sulla creazione e la firma del certificato, consulta Inizializzazione del cluster.

Esegui il bootstrap di Client SDK

Il processo di bootstrap è diverso a seconda della versione del Client SDK che stai usando, ma devi disporre dell'indirizzo IP di uno dei moduli di sicurezza hardware (HSM) del cluster. Puoi utilizzare l'indirizzo IP di qualsiasi HSM collegato al cluster. Dopo la connessione, il Client SDK

recupera gli indirizzi IP di eventuali HSM aggiuntivi ed esegue operazioni di bilanciamento del carico e sincronizzazione delle chiavi lato client.

Per ottenere un indirizzo IP per il cluster

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Per aprire la pagina dei dettagli del cluster, nella tabella dei cluster, scegli l'ID del cluster.
4. Per ottenere l'indirizzo IP, vai alla scheda HSM. Per i cluster IPv4, scegli un indirizzo elencato in Indirizzo ENI IPv4. Per i cluster dual-stack, utilizzare l'indirizzo ENI IPv4 o l'indirizzo ENI IPv6.

Per ottenere un indirizzo IP per un HSM (AWS CLI)

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters dalla AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp` e `EniIpV6` (se si tratta di un cluster dual-stack).

```
$ aws cloudhsmv2 describe-clusters
{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
                    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
```

Per ulteriori informazioni sulle operazioni di bootstrap, consulta la pagina Strumento Configure.

Per eseguire il bootstrap del Client SDK 5

PKCS #11 library

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 -a <HSM IP addresses>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" -a <HSM IP
 addresses>
```

OpenSSL Dynamic Engine

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn -a <HSM IP addresses>
```

OpenSSL Dynamic Engine Provider

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP addresses>
```

Key Storage Provider (KSP)

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" -a <HSM IP
 addresses>
```

JCE provider

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce -a <HSM IP addresses>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" -a <HSM IP
 addresses>
```

CloudHSM CLI

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the
 HSMs>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI
 IPv4 / IPv6 addresses of the HSMs>
```

> **Note**
>
> è possibile utilizzare il parametro `--cluster-id` al posto di `-a <HSM_IP_ADDRESSES>`. Per visualizzare i requisiti per l'utilizzo di `--cluster-id`, consulta AWS CloudHSM Strumento di configurazione di Client SDK 5.

Per eseguire il bootstrap del Client SDK 3

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 3

- Utilizzato configure per specificare l'indirizzo IP di un HSM nel cluster.

```
sudo /opt/cloudhsm/bin/configure -a <IP address>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 3

- Utilizzare configure per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

Per ulteriori informazioni relative alla configurazione, consulta ???.

Scalabilità degli HSM in un AWS CloudHSM cluster

Per aumentare o ridurre il AWS CloudHSM cluster, aggiungi o rimuovi HSM utilizzando la AWS CloudHSM console o uno degli AWS SDK o degli strumenti a riga di comando. Ti consigliamo di effettuare test di carico sul cluster per determinare il carico massimo da prevedere, quindi di aggiungere un altro modulo HSM per garantire un'elevata disponibilità.

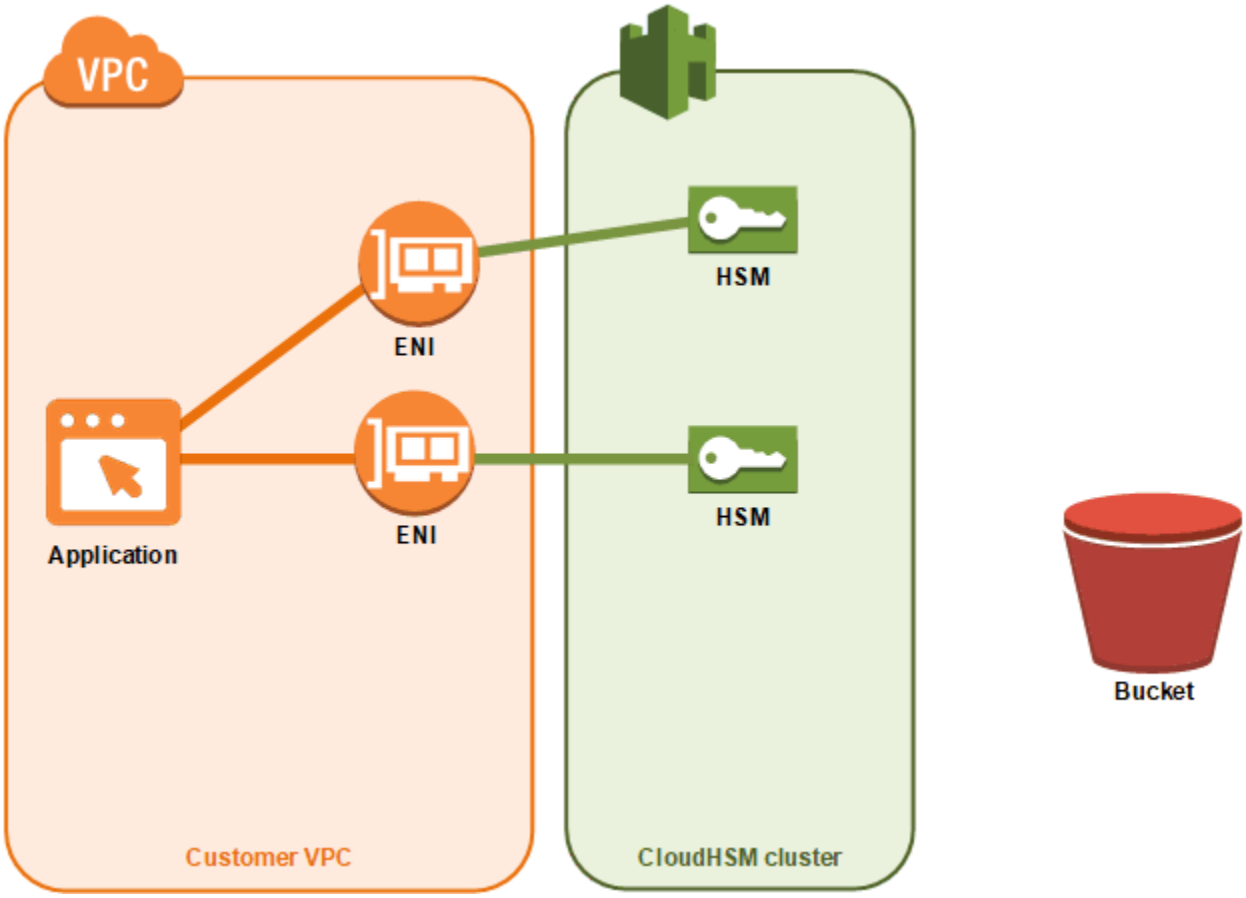
Argomenti

- Aggiungere un HSM a un AWS CloudHSM cluster

- Rimozione di un HSM da un AWS CloudHSM cluster

Aggiungere un HSM a un AWS CloudHSM cluster

La figura seguente mostra gli eventi che si verificano quando aggiungi un modulo HSM a un cluster.



1. Aggiungi un nuovo modulo HSM a un cluster. Le procedure riportate di seguito spiegano come fare ciò dalla console dell'AWS CloudHSM, dalAWS Command Line Interface (AWS CLI) e attraverso API AWS CloudHSM.

   Questa è l'unica operazione che devi effettuare. Gli altri eventi si verificano automaticamente.
2. AWS CloudHSM crea una copia di backup di un HSM esistente nel cluster. Per ulteriori informazioni, consulta backup.
3. AWS CloudHSM ripristina il backup sul nuovo HSM. in modo da garantirne la sincronizzazione con gli altri moduli HSM nel cluster.
4. Gli HSM esistenti nel cluster notificano al AWS CloudHSM client la presenza di un nuovo HSM nel cluster.

5. Il client stabilisce una connessione con il nuovo modulo HSM.

Per aggiungere un modulo HSM (console)

1. Apri la AWS CloudHSM console all'indirizzo. https://console.aws.amazon.com/cloudhsm/home
2. Scegli un cluster per il modulo HSM che stai aggiungendo.
3. Nella scheda HSM, scegli Crea modulo HSM.
4. Scegli una zona di disponibilità (AZ) per il modulo HSM in fase di creazione. Quindi, scegli Crea.

Per aggiungere un HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando create-hsm, specificando un ID per il cluster e una zona di disponibilità per il modulo HSM in fase di creazione. Se non conosci l'ID del cluster desiderato, esegui il comando describe-clusters. Specifica la zona di disponibilità con il formato `us-east-2a`, `us-east-2b` e così via.

```
$ aws cloudhsmv2 create-hsm --cluster-id <cluster ID> --availability-
zone <Availability Zone>
{
    "Hsm": {
        "State": "CREATE_IN_PROGRESS",
        "ClusterId": "cluster-5a73d5qzrdh",
        "HsmId": "hsm-lgavqitns2a",
        "SubnetId": "subnet-0e358c43",
        "AvailabilityZone": "us-east-2c",
        "EniId": "eni-bab18892",
        "EniIp": "10.0.3.10",
        "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733"
    }
}
```

Per aggiungere un HSM (AWS CloudHSM API)

- Invia una richiesta CreateHsm, specificando l'ID del cluster e una zona di disponibilità per il modulo HSM in fase di creazione.

Rimozione di un HSM da un AWS CloudHSM cluster

Puoi rimuovere un HSM utilizzando la AWS CloudHSM console, l'o l'AWS CLIAPI. AWS CloudHSM

Per rimuovere un modulo HSM (console)

1. Apri la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Scegli il cluster che contiene il modulo HSM da rimuovere.
3. Nella scheda HSM, scegli il modulo HSM che desideri rimuovere, quindi scegli Elimina modulo HSM.
4. Conferma l'eliminazione del modulo HSM. Scegli Elimina.

Per rimuovere un HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando delete-hsm. Inserire l'ID del cluster che contiene il modulo HSM che si sta eliminando e uno dei seguenti identificatori HSM:
  - ID del modulo HSM (`--hsm-id`)
  - Indirizzo IP del modulo HSM (`--eni-ip`)
  - ID dell'interfaccia di rete elastica del modulo HSM (`--eni-id`)

  Se non conosci i valori di questi identificatori, esegui il comando describe-clusters.

```
$ aws cloudhsmv2 delete-hsm --cluster-id <cluster ID> --eni-ip <HSM IP address>
{
    "HsmId": "hsm-lgavqitns2a"
}
```

Per rimuovere un HSM (AWS CloudHSM API)

- Invia una richiesta DeleteHsm, specificando l'ID del cluster e un identificatore per il modulo HSM in fase di eliminazione.

Eliminazione di un cluster AWS CloudHSM

Prima di poter eliminare un cluster, è necessario rimuovere tutto HSMs dal cluster. Per ulteriori informazioni, consulta Rimozione di un HSM da un AWS CloudHSM cluster.

Dopo aver rimosso tutto HSMs, puoi eliminare un cluster utilizzando la AWS CloudHSM console, il AWS Command Line Interface (AWS CLI) o l' AWS CloudHSM API.

Per eliminare un cluster (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Scegli il cluster da eliminare, quindi scegli Elimina cluster.
3. Conferma che intendi eliminare il cluster e scegli Elimina.

Per eliminare un cluster (AWS CLI)

- Al prompt dei comandi, eseguire il comando delete-cluster, passando l'ID del cluster da eliminare. Se non si conosce l'ID del cluster, eseguire il comando describe-clusters.

```
$ aws cloudhsmv2 delete-cluster --cluster-id <cluster ID>
{
    "Cluster": {
        "Certificates": {
            "ClusterCertificate": "<certificate string>"
        },
        "SourceBackupId": "backup-rtq2dwi2gq6",
        "SecurityGroup": "sg-40399d28",
        "CreateTimestamp": 1504903546.035,
        "SubnetMapping": {
            "us-east-2a": "subnet-f1d6e798",
            "us-east-2c": "subnet-0e358c43",
            "us-east-2b": "subnet-40ed9d3b"
        },
        "ClusterId": "cluster-kdmrayrc7gi",
        "VpcId": "vpc-641d3c0d",
        "State": "DELETE_IN_PROGRESS",
        "HsmType": "hsm1.medium",
        "StateMessage": "The cluster is being deleted.",
        "Hsms": [],
        "BackupPolicy": "DEFAULT"
    }
```

```
}
```

Per eliminare un cluster AWS CloudHSM (API)

- Invia una richiesta DeleteCluster, specificando l'ID del cluster da eliminare.

Creazione di AWS CloudHSM cluster dai backup

Per ripristinare un AWS CloudHSM cluster da un backup, segui i passaggi descritti in questo argomento. Il cluster conterrà gli stessi utenti, materiali delle chiavi, certificati, configurazioni e policy che si trovavano nel backup ripristinato. Per ulteriori informazioni sulla gestione dei backup, vedi Backup del cluster.

Crea cluster dai backup (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Scegli Crea cluster.
3. Nella sezione Configurazione del cluster, procedi come segue:
   a. Per VPC, scegli un VPC per il cluster che stai creando.
   b. Per AZ, scegli una sottorete privata per ogni zona di disponibilità che stai aggiungendo al cluster.
   c. Per Tipo di rete, scegli il protocollo IP che HSMs utilizzerai per le connessioni.
4. Nella sezione Origine cluster, procedi come segue:
   a. Scegli Ripristina cluster da un backup esistente.
   b. Scegli il backup da ripristinare.
5. Seleziona Successivo: Revisione.
6. Rivedi la configurazione del cluster, quindi scegli Crea cluster.
7. Specificare per quanto tempo il servizio deve conservare i backup.

   Accetta il periodo di conservazione predefinito di 90 giorni o digita un nuovo valore tra 7 e 379 giorni. Il servizio eliminerà automaticamente i backup in questo cluster più vecchi del valore specificato qui. Puoi modificare questa impostazione in un secondo momento. Per ulteriori informazioni, vedi Configura la conservazione dei backup.

8. Scegli Next (Successivo).
9. (Facoltativo) Digita tag per una chiave di un valore di tag facoltativo. Per aggiungere più di un tag al cluster scegli Aggiungi tag.
10. Scegli Rivedi.
11. Rivedi la configurazione del cluster, quindi scegli Crea cluster.

> **Tip**
>
> Per creare un HSM in questo cluster che contenga gli stessi utenti, materiale chiave, certificati, configurazione e policy presenti nel backup ripristinato, aggiungi un HSM al cluster.

Crea cluster dai backup (AWS CLI)

Per determinare l'ID del backup, esegui il comando describe-backups.

- Al prompt dei comandi, esegui il comando create-cluster. Specificate il tipo di istanza HSM, la sottorete IDs delle sottoreti in cui intendete creare HSMs e l'ID di backup del backup da ripristinare.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
                                --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID
 N> \
                                --source-backup-id <backup ID>
                                --mode <FIPS> \
                                --network-type <IPV4>
{
    "Cluster": {
        "HsmType": "hsm2m.medium",
        "VpcId": "vpc-641d3c0d",
        "Hsms": [],
        "State": "CREATE_IN_PROGRESS",
        "SourceBackupId": "backup-rtq2dwi2gq6",
        "BackupPolicy": "DEFAULT",
        "BackupRetentionPolicy": {
            "Type": "DAYS",
            "Value": 90
         },
        "NetworkType": "IPV4",
```

```
        "SecurityGroup": "sg-640fab0c",
        "CreateTimestamp": 1504907311.112,
        "SubnetMapping": {
            "us-east-2c": "subnet-0e358c43",
            "us-east-2a": "subnet-f1d6e798",
            "us-east-2b": "subnet-40ed9d3b"
        },
        "Certificates": {
            "ClusterCertificate": "<certificate string>"
        },
        "ClusterId": "cluster-jxhlf7644ne"
    }
}
```

Crea cluster dai backup (API)AWS CloudHSM

Fai riferimento al seguente argomento per scoprire come creare cluster dai backup utilizzando l'API.

- CreateCluster

Migrazione di tipo HSM del cluster

AWS CloudHSM offre la possibilità di modificare il tipo di HSM di un cluster esistente. Consulta la tabella in questa pagina per determinare se è consentita la modifica del tipo di HSM.

Per ulteriori informazioni sui tipi di dispositivi HSMs supportati e sulle relative funzionalità, fare riferimento aTipi di HSM in AWS CloudHSM.

> **Note**
>
> Non è possibile modificare la modalità FIPS di un cluster durante questa operazione.

Da	Per	Comment
hsm1.medium	hsm2m. medio	Consentito
hsm2m.medium	hsm1. medio	Condizionale. È possibile eseguire il rollback da

Da	Per	Comment
		hsm2m.medium a hsm1.medium entro 24 ore dall'inizio di una migrazione.

Argomenti

- Migrazione da hsm1.medium a hsm2m.medium

Migrazione da hsm1.medium a hsm2m.medium

Puoi migrare il tuo cluster da hsm1.medium a hsm2m.medium. AWS CloudHSM Questo argomento descrive i prerequisiti, il processo di migrazione e le procedure di rollback.

Prima di iniziare la migrazione, assicurati che l'applicazione segua i consigli in. Progetta il tuo cluster per garantire un'elevata disponibilità Questo aiuta a evitare tempi di inattività durante il processo.

> **Aggiornamento automatico della migrazione**
>
> A partire dal 20 gennaio 2026, sono iniziate le migrazioni automatiche verso hsm2m.medium.

Panoramica del processo di migrazione da hsm1.medium a hsm2m.medium

È possibile avviare la migrazione utilizzando la AWS CloudHSM Console, l'API o l'API. AWS CLI AWS CloudHSM Indipendentemente da dove viene avviata, la migrazione del AWS CloudHSM cluster utilizza l'endpoint `modify-cluster` API. In alternativa, AWS CloudHSM migrerà automaticamente il cluster per tuo conto. Una volta avviata la migrazione, l'intero cluster entra in modalità di scrittura limitata. Per ulteriori informazioni, consulta la modalità di scrittura limitata del cluster.

Per ridurre al minimo l'impatto, AWS CloudHSM modifica gli HSM da hsm1.medium a hsm2m.medium uno alla volta. Gli HSM sostitutivi mantengono gli stessi indirizzi IP, quindi non richiedono modifiche alla configurazione durante o dopo la migrazione.

Ecco come funziona la migrazione:

1. Prima di migrare il primo HSM, AWS CloudHSM crea un backup completo dell'intero cluster.

2. Utilizzando questo backup, AWS CloudHSM crea un nuovo HSM del tipo richiesto (hsm2m.medium) per sostituire il primo HSM.
3. Prima di migrare ogni HSM successivo, AWS CloudHSM crea un nuovo backup completo dell'intero cluster.
4. AWS CloudHSM ripete i passaggi 2 e 3 per ogni HSM del cluster, migrando un HSM alla volta.
5. Ogni singola migrazione HSM richiede circa 30 minuti.

AWS CloudHSM monitora lo stato del cluster ed esegue le convalide durante tutto il processo di migrazione. Se AWS CloudHSM rileva un aumento degli errori o un controllo di convalida fallisce, interromperà automaticamente la migrazione e ripristinerà il cluster al tipo HSM originale. È inoltre possibile eseguire il rollback manualmente per un massimo di 24 ore dopo l'avvio della migrazione. Prima di eseguire il rollback, consulta le considerazioni sul rollback del tipo HSM.

Prerequisiti per la migrazione a hsm2m.medium

Il AWS CloudHSM cluster esistente deve soddisfare questi requisiti per migrare a hsm2m.medium. Se una condizione non viene soddisfatta durante i controlli di convalida, ripristina AWS CloudHSM automaticamente il cluster al tipo HSM originale.

Per un elenco dei problemi di migrazione noti, consulta ???

- Negli ultimi 7 giorni:
  - Tutte le connessioni client hanno utilizzato SDK 5.9 o versioni successive.
    - Se si esegue ECDSA Verify, tutte le connessioni client hanno utilizzato SDK 5.13 o versioni successive.
  - AWS CloudHSM le istanze hanno utilizzato solo funzionalità supportate (e nessuna di quelle obsolete). Per i dettagli, consulta Notifiche di deprecazione.
  - Devi aver utilizzato un SDK per connetterti con almeno un HSM nel cluster negli ultimi 7 giorni.
- Il cluster è in uno stato ATTIVO.
- Il cluster ha 27 HSM o meno.
- Il tasso di errore per le operazioni HSM non aumenta durante la migrazione.

> **Note**
>
> La precedente restrizione che impediva la migrazione dei clienti con carichi di lavoro con token key è stata rimossa.

> **Esenzione dal requisito di connessione SDK**
>
> Se il cluster non ha avuto connessioni SDK o client negli ultimi 14 giorni, è esente dal requisito di connessione SDK di 7 giorni.

Modalità di scrittura limitata del cluster

Quando il cluster avvia la migrazione, entra in una modalità di scrittura limitata. Le operazioni che possono modificare lo stato HSM vengono rifiutate. Tutte le operazioni di lettura rimangono inalterate.

Durante la migrazione, l'applicazione riceve un errore dall'HSM quando tenta le seguenti operazioni:

- Generazione ed eliminazione delle chiavi token (i carichi di lavoro delle chiavi di sessione continuano a funzionare).
- Creazione, eliminazione o modifica di tutti gli utenti.
- Operazioni relative al quorum.
- Modifica delle chiavi all'interno dell'HSM, ad esempio modifica degli attributi delle chiavi.
- Registrazione MTL.

AWS CloudHSM inoltre, pone il cluster in uno `MODIFY_IN_PROGRESS` stato durante la migrazione. Durante questo periodo, non puoi aggiungere o rimuovere HSM dal cluster.

Avvio della migrazione

Il processo di migrazione del cluster sostituisce i singoli HSM del cluster uno alla volta. La durata dipende dal numero di HSM presenti nel cluster. In media, questo processo richiede circa 30 minuti per HSM. È possibile tenere traccia dei progressi monitorando il tipo HSM dei singoli HSM nel cluster per vedere quanti sono stati migrati al nuovo tipo.

Console

Per modificare il tipo di HSM (console)

1. Aprire la AWS CloudHSM console all'indirizzo. https://console.aws.amazon.com/cloudhsm/home
2. Seleziona il pulsante di opzione accanto all'ID del cluster che desideri modificare
3. Dal menu Azioni, scegli `Modify HSM Type` e seleziona il tipo di HSM desiderato

Questa procedura imposta lo `MODIFY_IN_PROGRESS` stato del cluster. Dopo la migrazione, il cluster ritorna allo `ACTIVE` stato.

AWS CLI

Per modificare il tipo di HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando modify-cluster. Specificare l'ID del cluster e il tipo di HSM desiderato.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> --hsm-type <HSM Type>


 {
     "Cluster": {
         "BackupPolicy": "DEFAULT",
         "BackupRetentionPolicy": {
             "Type": "DAYS",
             "Value": 90
          },
         "VpcId": "vpc-50ae0636",
         "SubnetMapping": {
             "us-west-2b": "subnet-49a1bc00",
             "us-west-2c": "subnet-6f950334",
             "us-west-2a": "subnet-fd54af9b"
         },
         "SecurityGroup": "sg-6cb2c216",
         "HsmType": "hsm2m.medium",
         "HsmTypeRollbackExpiration": 1730383180.000,
         "Certificates": {},
         "State": "MODIFY_IN_PROGRESS",
         "Hsms": [],
         "ClusterId": "cluster-igklspoyj5v",
```

```
        "ClusterMode": "FIPS",
        "CreateTimestamp": 1502423370.069
    }
}
```

Questa procedura inserisce il cluster nello `MODIFY_IN_PROGRESS` stato. Dopo la migrazione, il cluster ritorna allo `ACTIVE` stato.

AWS CloudHSM API

Per modificare il tipo di HSM (AWS CloudHSM API)

- Inviare una richiesta ModifyCluster. Specificare l'ID del cluster e il tipo di HSM desiderato per il cluster.

Questa procedura inserisce il cluster nello `MODIFY_IN_PROGRESS` stato. Dopo la migrazione, il cluster ritorna allo `ACTIVE` stato.

Ripristino della migrazione

AWS CloudHSM monitora i tassi di errore elevati ed esegue controlli di convalida continui durante tutta la migrazione. Se AWS CloudHSM rileva una diminuzione della qualità del servizio o eventuali errori di convalida, avvia automaticamente un rollback al tipo di HSM originale del cluster. Durante un rollback, per ogni HSM del cluster:

- AWS CloudHSM utilizza il backup eseguito all'inizio della migrazione di quell'HSM.
- Sostituisce un HSM alla volta fino a quando tutti gli HSM non vengono restituiti al tipo originale.
- Il cluster rimane in modalità di scrittura limitata per tutto il processo.

Puoi ripristinare la migrazione entro 24 ore dall'avvio. Per verificare la scadenza per il rollback:

1. Esegui il comando https://docs.aws.amazon.com/cli/latest/reference/cloudhsmv2/describe-clusters.html describe-clusters.
2. Cerca il valore. `HsmTypeRollbackExpiration` Questo timestamp è la tua scadenza per il rollback.

Se decidi di effettuare il rollback, fallo prima di questa scadenza. Il rollback utilizza il backup più recente del tipo di HSM originale.

> **Warning**
>
> Fai attenzione quando effettui il rollback dopo il completamento della migrazione. Se completi una migrazione e poi la usi AWS CloudHSM per creare nuove chiavi o utenti, il rollback può causare la perdita di dati. Ad esempio, una chiave creata in hsm2m.medium potrebbe non essere disponibile in hsm1.medium una volta completato il rollback. Vedi Sincronizzazione dei dati dopo un rollback per scoprire come mitigare la perdita di dati dopo un rollback.

Console

Per ripristinare il tipo di HSM (console)

1. Aprire la AWS CloudHSM console all'indirizzo. https://console.aws.amazon.com/cloudhsm/home
2. Seleziona l'ID del cluster di cui desideri ripristinare.
3. Dal menu Azioni, scegli `Modify HSM Type` e seleziona il tipo di HSM originale

Questa procedura imposta lo `ROLLBACK_IN_PROGRESS` stato del cluster. Dopo il rollback, il cluster ritorna allo `ACTIVE` stato.

AWS CLI

Per ripristinare il tipo HSM (AWS CLI)

- Al prompt dei comandi, esegui il comando modify-cluster. Specifica l'ID del cluster e il tipo di HSM originale.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> --hsm-type <HSM Type>

{
 "Cluster": {
     "BackupPolicy": "DEFAULT",
     "BackupRetentionPolicy": {
         "Type": "DAYS",
         "Value": 90
      },
```

```
    "VpcId": "vpc-50ae0636",
    "SubnetMapping": {
        "us-west-2b": "subnet-49a1bc00",
        "us-west-2c": "subnet-6f950334",
        "us-west-2a": "subnet-fd54af9b"
    },
    "SecurityGroup": "sg-6cb2c216",
    "HsmType": "hsm1.medium",
    "HsmTypeRollbackExpiration": 1730383180.000,
    "Certificates": {},
    "State": "ROLLBACK_IN_PROGRESS",
    "Hsms": [],
    "ClusterId": "cluster-igklspoyj5v",
    "ClusterMode": "FIPS",
    "CreateTimestamp": 1502423370.069
 }
}
```

Questa procedura inserisce il cluster nello `ROLLBACK_IN_PROGRESS` stato. Dopo il rollback, il cluster ritorna allo `ACTIVE` stato.

AWS CloudHSM API

Per ripristinare il tipo HSM (AWS CloudHSM API)

- Inviare una richiesta ModifyCluster. Specifica l'ID del cluster e il tipo HSM originale per il cluster.

Questa procedura inserisce il cluster nello `ROLLBACK_IN_PROGRESS` stato. Dopo il rollback, il cluster ritorna allo `ACTIVE` stato.

Sincronizzazione dei dati dopo un rollback

Durante la migrazione, gli HSM sono in modalità di scrittura limitata, impedendo modifiche allo stato HSM. Se si esegue il rollback durante questo periodo (mentre il cluster è attivo`MODIFY_IN_PROGRESS`), si ottiene un cluster con contenuti identici a quelli del cluster originale.

Una volta ripristinato `ACTIVE` lo stato del cluster, la modalità di scrittura limitata viene disattivata. Se crei una chiave o un utente mentre sei in `ACTIVE` stato e poi esegui il rollback, quella chiave o utente non sarà presente nel cluster di rollback.

Per gestire la sincronizzazione degli utenti, utilizza la gestione degli utenti con CloudHSM CLI per ricreare gli utenti mancanti nel cluster rollback. Gli utenti devono essere ricreati manualmente perché il `user replicate` comando non supporta la sincronizzazione degli utenti da hsm2m.medium a hsm1.medium. Vedi ??? Problemi noti relativi alla replica da parte degli utenti.

Per risolvere la sincronizzazione delle chiavi, usa il comando key replicate per replicare una chiave tra due cluster. Se non hai installato CloudHSM CLI, consulta le istruzioni in. ???

Per sincronizzare le chiavi dopo il rollback

Segui questi passaggi dopo aver completato il rollback. Utilizzeremo questi termini:

- «cluster-1": il tuo cluster ripristinato (ora hsm1.medium)
- «cluster-2": un nuovo cluster temporaneo hsm2m.medium che creerai

1. Crea un nuovo cluster hsm2m.medium (cluster-2) utilizzando il backup hsm2m.medium più recente di cluster-1:

```
aws cloudhsmv2 create-cluster --hsm-type hsm2m.medium \
                              --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID
 N> \
                              --source-backup-id <backup ID>
                              --mode <FIPS>
```

2. Crea un HSM in cluster-2:

```
aws cloudhsmv2 create-hsm --cluster-id <cluster-2 ID>
```

3. Elenca le chiavi nel cluster-2 che necessitano di replica:

```
cloudhsm-cli key list --cluster-id <cluster-2 ID>
```

4. Replica ogni chiave dal cluster-2 al cluster-1:

```
cloudhsm-cli key replicate --source-cluster-id <cluster-2 ID> \
```

```
                        --destination-cluster-id <cluster-1 ID> \
                        --filter attr.label=<key ID>
```

5. Ripetere il passaggio 4 per ogni chiave da copiare.
6. Elimina l'HSM nel cluster-2:

```
aws cloudhsmv2 delete-hsm --cluster-id <cluster-2 ID> --hsm-id <HSM ID>
```

7. Elimina cluster-2:

```
aws cloudhsmv2 delete-cluster --cluster-id <cluster-2 ID>
```

Utenti HSM in AWS CloudHSM

Prima di poter utilizzare il AWS CloudHSM cluster per l'elaborazione delle criptovalute, è necessario creare utenti e chiavi sui moduli di sicurezza hardware (HSM) del cluster.

> **Note**
>
> Gli utenti HSM sono diversi dagli utenti IAM. Gli utenti IAM che dispongono delle credenziali corrette possono creare HSMs interagendo con le risorse tramite l'API AWS. Dopo aver creato l'HSM, devi utilizzare le credenziali utente HSM per autenticare le operazioni sull'HSM.

In AWS CloudHSM, devi utilizzare gli strumenti a riga di comando CloudHSM CLI o CloudHSM Management Utility (CMU) per creare e gestire gli utenti sul tuo HSM. La CLI di CloudHSM è progettata per essere utilizzata con la serie di versioni SDK più recente, mentre la CMU è progettata per essere utilizzata con la serie di versioni SDK precedente.

Per ulteriori informazioni sulla gestione degli utenti HSM in, consulta i seguenti argomenti. AWS CloudHSMÈ inoltre possibile imparare a utilizzare l'autenticazione del quorum (anche detta controllo accesso "M of N").

Argomenti

- Gestione degli utenti HSM con CLI CloudHSM
- Gestione degli utenti HSM con CloudHSM Management Utility (CMU)

Gestione degli utenti HSM con CLI CloudHSM

Per gestire gli utenti del modulo di sicurezza hardware (HSM) AWS CloudHSM, devi accedere all'HSM con il nome utente e la password di un amministratore. Solo gli amministratori possono gestire gli utenti. L'HSM contiene un amministratore predefinito denominato admin. Hai impostato la password per admin quando hai attivato il cluster.

Questo argomento fornisce istruzioni dettagliate e dettagli sulla gestione degli utenti HSM con CloudHSM CLI.

Argomenti

- Prerequisiti per la gestione degli utenti nella CLI di CloudHSM

- Tipi di utente HSM per CloudHSM CLI
- Tabella delle autorizzazioni utente HSM per CloudHSM CLI
- Crea un amministratore utente HSM utilizzando la CLI CloudHSM
- Crea un utente crittografico HSM utilizzando la CLI di CloudHSM
- Elenca tutti gli utenti HSM nel cluster utilizzando la CLI di CloudHSM
- Modifica le password degli utenti HSM utilizzando la CLI di CloudHSM
- Eliminare gli utenti HSM utilizzando la CLI di CloudHSM
- Gestisci l'MFA per gli utenti HSM utilizzando la CLI di CloudHSM
- Gestisci l'autenticazione del quorum (controllo degli accessi M of N) utilizzando la CLI CloudhSM

Prerequisiti per la gestione degli utenti nella CLI di CloudHSM

Prima di utilizzare l'interfaccia della riga di comando di CloudHSM per gestire AWS CloudHSM gli utenti dei moduli di sicurezza hardware (HSM) in, è necessario completare questi prerequisiti. I seguenti argomenti descrivono come iniziare a usare la CLI CloudhSM.

Argomenti

- Ottieni l'indirizzo IP di un HSM in AWS CloudHSM
- Download della CLI di CloudHSM

Ottieni l'indirizzo IP di un HSM in AWS CloudHSM

Per utilizzare la CLI di CloudHSM, è necessario utilizzare lo strumento di configurazione per aggiornare la configurazione locale. Per istruzioni sull'esecuzione dello strumento di configurazione con la CLI di CloudHSM, consulta Nozioni di base su AWS CloudHSM Interfaccia a riga di comando (CLI). Il parametro `-a` richiede l'aggiunta dell'indirizzo IP di un HSM nel cluster. Se hai più HSM, puoi utilizzare qualsiasi indirizzo IP. Questo garantisce che la CLI di CloudHSM possa propagare le modifiche apportate all'intero cluster. Ricorda che la CLI di CloudHSM utilizza il suo file locale per tenere traccia delle informazioni sul cluster. Se il cluster è cambiato dall'ultima volta che hai usato la CLI di CloudHSM da un determinato host, devi aggiungere tali modifiche al file di configurazione locale memorizzato su quell'host. Non rimuovere mai un HSM mentre utilizzi la CLI di CloudHSM.

Per ottenere un indirizzo IP per un HSM (console)

1. Aprire la AWS CloudHSM console all'indirizzo https://console.aws.amazon.com/cloudhsm/home.

2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Per aprire la pagina dei dettagli del cluster, nella tabella dei cluster, scegli l'ID del cluster.
4. Per ottenere l'indirizzo IP, vai alla scheda HSM. Per i cluster IPv4, scegli un indirizzo elencato in Indirizzo ENI IPv4. Per i cluster dual-stack, utilizzare l'indirizzo ENI IPv4 o l'indirizzo ENI IPv6.

Per ottenere un indirizzo IP per un HSM (AWS CLI)

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters dalla AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp` e `EniIpV6` (se si tratta di un cluster dual-stack).

```
$ aws cloudhsmv2 describe-clusters
{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
                    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
```

Download della CLI di CloudHSM

L'ultima versione della CLI di CloudHSM è disponibile per le attività di gestione utenti HSM per il Client SDK 5. Per scaricare e installare la CLI di CloudHSM, segui le istruzioni in Installazione e configurazione della CLI di CloudHSM.

Tipi di utente HSM per CloudHSM CLI

La maggior parte delle operazioni eseguite sul modulo di sicurezza hardware (HSM) richiede le credenziali di un utente HSM. AWS CloudHSM La HSM autentica ogni utente HSM e ogni utente HSM dispone di un tipo che stabilisce quali operazioni può eseguire nell'HSM in qualità di utente.

> **Note**
>
> Gli utenti HSM sono diversi dagli utenti IAM. Gli utenti IAM che dispongono delle credenziali corrette possono creare HSM interagendo con le risorse tramite l'API AWS. Dopo aver creato l'HSM, devi utilizzare le credenziali utente HSM per autenticare le operazioni sull'HSM.

Tipi di utente

- Admin non attivato
- Admin
- Crypto user (CU)
- Utente dell'appliance (AU)

Admin non attivato

Nella CLI di CloudHSM, l'amministratore non attivato è un utente temporaneo che esiste solo sul primo HSM in un cluster AWS CloudHSM che non è mai stato attivato. Per attivare un cluster, esegui il comando cluster activate nella CLI di CloudHSM. Dopo aver eseguito il comando, all'amministratore non attivato viene richiesto di modificare la password. Dopo aver modificato la password, l'amministratore non attivato diventa amministratore.

Admin

Nella CLI di CloudHSM, l'amministratore può eseguire operazioni di gestione degli utenti. Ad esempio, può creare ed eliminare gli utenti e modificare le password degli utenti. Per ulteriori informazioni sugli amministratori, consulta Tabella delle autorizzazioni utente HSM per CloudHSM CLI.

Crypto user (CU)

Un utente di crittografia (CU) è in grado di eseguire le seguenti operazioni di crittografia e di gestione delle chiavi.

- Gestione chiavi: consente di creare, eliminare, condividere, importare ed esportare le chiavi di crittografia.
- Operazioni di crittografia: usa le chiavi di crittografia per la crittografia, la decrittografia, la firma, la verifica e altro ancora.

Per ulteriori informazioni, consulta Tabella delle autorizzazioni utente HSM per CloudHSM CLI.

Utente dell'appliance (AU)

L'utente dell'appliance (AU) può eseguire operazioni di clonazione e sincronizzazione sugli HSM del cluster. AWS CloudHSM utilizza l'AU per sincronizzare gli HSM in un cluster. AWS CloudHSM L'AU esiste su tutti gli HSM forniti da AWS CloudHSM e dispone di autorizzazioni limitate. Per ulteriori informazioni, consulta Tabella delle autorizzazioni utente HSM per CloudHSM CLI.

AWS non può eseguire alcuna operazione sui tuoi HSM. AWS non può visualizzare o modificare gli utenti o le chiavi e non può eseguire operazioni crittografiche utilizzando tali chiavi.

Tabella delle autorizzazioni utente HSM per CloudHSM CLI

La tabella seguente elenca le operazioni dei moduli di sicurezza hardware (HSM) ordinate in base al tipo di utente o sessione HSM in cui è possibile eseguire l'operazione. AWS CloudHSM

	Admin	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
Ottenimento info cluster di base[1]	✓ Sì	✓ Sì	✓ Sì	✓ Sì
Modifica della propria password	✓ Sì	✓ Sì	✓ Sì	Non applicabile
Modifica della password di qualsiasi utente	✓ Sì	✗ No	✗ No	✗ No
Aggiunta, rimozione di utenti	✓ Sì	✗ No	✗ No	✗ No
Ottenimento stato sincronizzazione[2]	✓ Sì	✓ Sì	✓ Sì	✗ No

	Admin	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
Estrazione, inserimento di oggetti nascosti[3]	✓ Sì	✓ Sì	✓ Sì	✗ No
Funzioni di gestione chiave[4]	✗ No	✓ Sì	✗ No	✗ No
Crittografia, decrittografia	✗ No	✓ Sì	✗ No	✗ No
Firma, verifica	✗ No	✓ Sì	✗ No	✗ No
Creazione di digest e HMAC	✗ No	✓ Sì	✗ No	✗ No

- [1] Le informazioni di base del cluster includono il numero di HSM, nonché l'indirizzo IP, il modello, il numero di serie, l'ID del dispositivo, l'ID del firmware e così via di ciascun HSM.
- [2] L'utente può ottenere un set di digest (hash) corrispondenti alle chiavi dell'HSM. Un'applicazione può confrontare tali set di digest per determinare lo stato della sincronizzazione dell'HSM in un cluster.
- [3] Gli oggetti mascherati sono chiavi crittografate prima di lasciare l'HSM. Non possono essere decrittografate esternamente all'HSM. Vengono decrittografate solo dopo essere state inserite in un HSM che si trova nello stesso cluster di quello da cui sono stati estratte. Un'applicazione è in grado di estrarre e inserire oggetti nascosti per sincronizzare gli HSM in un cluster.
- [4] Le funzioni di gestione chiave includono la creazione, l'eliminazione, il wrapping, l'annullamento del wrapping e la modifica degli attributi delle chiavi.

Crea un amministratore utente HSM utilizzando la CLI CloudHSM

Segui questi passaggi per creare un utente amministratore del modulo di sicurezza hardware (HSM) utilizzando la CLI di CloudHSM.

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizza il comando login ed esegui l'accesso al cluster come amministratore.

```
aws-cloudhsm > login --username <username> --role admin
```

3. Il sistema ti invita a inserire la password. Inserisci la password e l'output mostra che il comando ha avuto successo.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

4. Immetti il seguente comando per creare un amministratore:

```
aws-cloudhsm > user create --username <username> --role admin
```

5. Immetti la password per il nuovo utente.
6. Re-enter la password per confermare la password inserita è corretta.

Crea un utente crittografico HSM utilizzando la CLI di CloudHSM

Segui questi passaggi per creare un utente crittografico (CU) del modulo di sicurezza hardware (HSM) utilizzando la CLI di CloudHSM.

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizza il comando login ed esegui l'accesso al cluster come amministratore.

```
aws-cloudhsm > login --username <username> --role admin
```

3. Il sistema ti invita a inserire la password. Inserisci la password e l'output mostra che il comando ha avuto successo.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<USERNAME>",
    "role": "admin"
  }
}
```

4. Inserisci il seguente comando per creare un crypto user:

```
aws-cloudhsm > user create --username <username> --role crypto-user
```

5. Immetti la password per il nuovo crypto user.
6. Re-enter la password per confermare la password inserita è corretta.

Elenca tutti gli utenti HSM nel cluster utilizzando la CLI di CloudHSM

Utilizzate il user list comando nella CLI di CloudHSM per elencare tutti gli utenti del cluster. AWS CloudHSM Non è necessario effettuare l'accesso per eseguire user list. Tutti i tipi di utenti possono elencare utenti.

Attieniti alla seguente procedura per elencare tutti gli utenti sul cluster

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Immetti il seguente comando per elencare tutti gli utenti del cluster:

   ```
   aws-cloudhsm > user list
   ```

Per ulteriori informazioni su user list, consulta user list.

Modifica le password degli utenti HSM utilizzando la CLI di CloudHSM

Utilizzate il user change-password comando nella CLI di CloudHSM per modificare la password di un utente del modulo di sicurezza hardware (HSM).

Solo i tipi e le password prevedono la distinzione tra lettere maiuscole e minuscole, non i nomi utente.

Amministratore, crypto user (CU) e utente dell'applicazione (AU) possono modificare solo le proprie password. Per modificare la password di un altro utente, devi accedere come amministratore. Tuttavia, non potrai modificare la password di un utente che attualmente è connesso.

Per modificare la tua password

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizza il comando login e accedi come utente con la password da modificare.

```
aws-cloudhsm > login --username <username> --role <role>
```

3. Inserisci la password dell'utente.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "<role>"
  }
}
```

4. Immettere il comando user change-password.

```
aws-cloudhsm > user change-password --username <username> --role <role>
```

5. Immetti la nuova password.
6. Re-enter la nuova password.

Per modificare la password di un altro utente

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

```
aws-cloudhsm > login --username <admin> --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Immetti il comando user change-password insieme al nome utente dell'utente di cui desideri modificare la password.

```
aws-cloudhsm > user change-password --username <username> --role <role>
```

4. Immetti la nuova password.
5. Re-enter la nuova password.

Per ulteriori informazioni su user change-password, consulta user change-password.

Eliminare gli utenti HSM utilizzando la CLI di CloudHSM

Utilizzalo user delete nella CLI di CloudHSM per eliminare un utente del modulo di sicurezza hardware (HSM). Per eliminare un altro utente devi accedere come amministratore.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

Come eliminare un utente

1. Utilizza il seguente comando per avviare la CLI di CloudHSM in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizza il comando login ed esegui l'accesso al cluster come amministratore.

```
aws-cloudhsm > login --username <username> --role admin
```

3. Il sistema ti invita a inserire la password. Inserisci la password e l'output mostra che il comando ha avuto successo.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

4. Utilizza il comando user delete per eliminare l'utente.

```
aws-cloudhsm > user delete --username <username> --role <role>
```

Per ulteriori informazioni su user delete, consulta deleteUser.

Gestisci l'MFA per gli utenti HSM utilizzando la CLI di CloudHSM

Per una maggiore sicurezza, puoi configurare l'autenticazione a più fattori (MFA) per gli utenti per proteggere AWS CloudHSM il cluster.

Quando accedi a un cluster con un account utente HSM (Hardware Security Module) abilitato per MFA, fornisci alla CLI di CloudHSM la tua password, il primo fattore, quello che sai, e CloudHSM CLI ti fornisce un token e ti richiede di firmare il token.

Per fornire il secondo fattore (quello che possiedi) firmi il token con una chiave privata da una coppia di chiavi che hai già creato e associata all'utente HSM. Per accedere al cluster, fornisci il token firmato alla CLI di CloudHSM.

Per ulteriori informazioni sulla configurazione della tecnologia MFA per un utente, vedere Configurazione della MFA per la CLI di CloudHSM

I seguenti argomenti forniscono ulteriori informazioni sull'utilizzo dell'autenticazione quorum in. AWS CloudHSM

Argomenti

- Autenticazione quorum e MFA in AWS CloudHSM cluster che utilizzano CloudHSM CLI
- Requisiti per le coppie di chiavi MFA per AWS CloudHSM utilizzo della CLI CloudhSM
- Configurazione della MFA per la CLI di CloudHSM
- Crea utenti con MFA abilitata per CloudHSM CLI
- Accedi agli utenti con MFA abilitata per CloudHSM CLI
- Ruota le chiavi per gli utenti con MFA abilitata per CloudHSM CLI
- Annullare la registrazione di una chiave pubblica MFA utilizzando la CLI di CloudHSM
- Riferimento al file token per MFA con CLI CloudhSM

Autenticazione quorum e MFA in AWS CloudHSM cluster che utilizzano CloudHSM CLI

Il AWS CloudHSM cluster utilizza la stessa chiave per l'autenticazione quorum e per l'autenticazione a più fattori (MFA). Ciò significa che un utente con MFA abilitata è effettivamente registrato per il MoFN o il controllo degli accessi quorum. Per utilizzare correttamente l'autenticazione MFA e quorum per lo stesso utente HSM, tieni presenti i seguenti punti:

- Se oggi si utilizza l'autenticazione quorum per un utente, è necessario utilizzare la stessa coppia di chiavi creata per l'utente quorum per abilitare la MFA per l'utente.
- Se aggiungi il requisito MFA per un utente non MFA che non è un utente di autenticazione quorum, registri quell'utente come utente registrato al quorum (MoFN) con autenticazione MFA.
- Se rimuovi il requisito MFA o modifichi la password per un utente MFA che è anche un utente registrato per l'autenticazione quorum, rimuoverai anche la registrazione dell'utente come utente del quorum (MoFN).
- Se rimuovi il requisito MFA o modifichi la password per un utente MFA che è anche un utente con autenticazione quorum, ma desideri comunque che quell'utente partecipi all'autenticazione quorum, devi registrare nuovamente quell'utente come utente quorum (MoFN).

Per ulteriori informazioni sull'autenticazione quorum, consulta Gestisci l'autenticazione del quorum (M of N).

Requisiti per le coppie di chiavi MFA per AWS CloudHSM utilizzo della CLI CloudhSM

Per abilitare l'autenticazione a più fattori (MFA) per un utente del modulo di sicurezza hardware (HSM) AWS CloudHSM in, puoi creare una nuova coppia di chiavi o utilizzare una chiave esistente che soddisfi i seguenti requisiti:

- Tipo di chiave: asimmetrica
- Uso delle chiavi: firma e verifica
- Specifiche chiave: RSA_2048
- L'algoritmo di firma include: sha256WithRSAEncryption

> **Note**
>
> Se utilizzi l'autenticazione quorum o intendi utilizzare l'autenticazione quorum, consulta Autenticazione quorum e MFA in AWS CloudHSM cluster che utilizzano CloudHSM CLI

Puoi utilizzare la CLI di CloudHSM e la coppia di chiavi per creare un nuovo utente amministratore con autenticazione MFA abilitata.

Configurazione della MFA per la CLI di CloudHSM

Segui questi passaggi per configurare l'autenticazione a più fattori (MFA) per CloudHSM CLI.

1. Per configurare la MFA utilizzando la strategia di firma tramite token, devi prima generare una chiave privata RSA a 2048 bit e la chiave pubblica associata.

```
$ openssl genrsa -out officer1.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....................................................+++++
.....................................................................+++++
e is 65537 (0x010001)

$ openssl rsa -in officer1.key -outform PEM -pubout -out officer1.pub
writing RSA key
```

2. Immetti il seguente comando per avviare la CLI in modalità interattiva.

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

3. Utilizzando la CLI di CloudHSM, accedi al tuo account utente.

```
aws-cloudhsm > login --username <admin> --role <admin> --cluster-id <cluster ID>
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "<admin>"
  }
}
```

4. Esegui quindi il comando per modificare la tua strategia MFA. Devi fornire il parametro `--token`. Questo parametro specifica un file in cui verranno scritti token non firmati.

```
aws-cloudhsm > user change-mfa token-sign --token unsigned-tokens.json --
username <username> --role crypto-user --change-quorum
Enter password:
Confirm password:
```

5. Ora hai un file con token non firmati che devono essere firmati: `unsigned-tokens.json`. Il numero di token in questo file dipende dal numero di HSM nel cluster. Ogni token rappresenta un HSM. Questo file è in formato JSON e contiene token che devono essere firmati per dimostrare che disponi di una chiave privata.

```
$ cat unsigned-tokens.json
{
  "version": "2.0",
  "tokens": [
{
    {
      "unsigned": "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
      "signed": ""
```

```
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
      "signed": ""
    }
  ]
}
```

6. Il passaggio successivo consiste nel firmare questi token con la chiave privata creata nel passaggio 1. Inserisci nuovamente le firme nel file. Per prima cosa, devi estrarre e decodificare i token codificati in base64.

```
$ echo "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin
```

7. Ora hai token binari che puoi firmare utilizzando la chiave privata RSA creata nel passaggio 1.

```
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token1.bin \
      -out token1.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token2.bin \
      -out token2.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token3.bin \
```

```
      -out token3.sig.bin
```

8. Ora hai le firme binarie dei token. È necessario codificarli utilizzando base64 e quindi reinserirli nel file del token.

```
$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

9. Infine, puoi copiare e incollare nuovamente i valori base64 nel tuo file del token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

10. Ora che il tuo file del token ha tutte le firme richieste, puoi procedere. Inserisci il nome del file contenente i token firmati e premi il tasto INVIO. Inserisci poi il percorso della tua chiave pubblica.

```
Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:officer1.pub
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "crypto-user"
  }
}
```

A questo punto hai configurato il tuo utente con la MFA.

```
{
    "username": "<username>",
    "role": "crypto-user",
    "locked": "false",
    "mfa": [
      {
        "strategy": "token-sign",
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
},
```

Crea utenti con MFA abilitata per CloudHSM CLI

Segui questi passaggi per creare AWS CloudHSM utenti con l'autenticazione a più fattori (MFA) abilitata.

1. Utilizza la CLI di CloudHSM per accedere all'HSM come amministratore.
2. Usa il comando user create per creare un utente a tua scelta. Segui poi i passaggi Configurazione della MFA per la CLI di CloudHSM per configurare l'autenticazione MFA per l'utente.

Accedi agli utenti con MFA abilitata per CloudHSM CLI

Segui questi passaggi per accedere agli AWS CloudHSM utenti con l'autenticazione a più fattori (MFA) abilitata.

1. Utilizza il comando login mfa-token-sign nella CLI di CloudHSM per avviare il processo di accesso con MFA per un utente con autenticazione MFA abilitata.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --
token <unsigned-tokens.json>
Enter password:
```

2. Inserisci la password. Ti verrà quindi richiesto di inserire il percorso del file token che contiene le coppie di token, dove i unsigned/signed token firmati sono quelli generati utilizzando la tua chiave privata.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --
token <unsigned-tokens.json>
Enter password:
Enter signed token file path (press enter if same as the unsigned token file):
```

3. Quando ti viene richiesto di inserire il percorso del file del token firmato, puoi ispezionare il file del token non firmato in un terminale separato. Identifica il file con token non firmati che devono essere firmati: *<unsigned-tokens.json>*. Il numero di token in questo file dipende dal numero di HSM nel cluster. Ogni token rappresenta un HSM. Questo file è in formato JSON e contiene token che devono essere firmati per dimostrare che disponi di una chiave privata.

```
$ cat <unsigned-tokens.json>
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
      "signed": ""
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
```

```
      "signed": ""
    }
  ]
}
```

4. Firma i token non firmati con la chiave privata creata nel passaggio 2. Per prima cosa, devi estrarre e decodificare i token codificati in base64.

```
$ echo "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin
```

5. Ora hai dei token binari. Firmali utilizzando la chiave privata RSA creata in precedenza nel passaggio 1 della configurazione MFA.

```
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token1.bin \
      -out token1.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token2.bin \
      -out token2.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token3.bin \
      -out token3.sig.bin
```

6. Ora hai le firme binarie dei token. Codificali in base64 e inseriscili nuovamente nel tuo file token.

```
$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

7. Infine, copia e incolla nuovamente i valori base64 nel tuo file del token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

8. Ora che il tuo file del token ha tutte le firme richieste, puoi procedere. Inserisci il nome del file contenente i token firmati e premi il tasto INVIO. Ora dovresti accedere con successo.

```
aws-cloudhsm > login --username <username> --role <role> mfa-token-sign --
token <unsigned-tokens.json>
Enter password:
Enter signed token file path (press enter if same as the unsigned token file):
{
  "error_code": 0,
  "data": {
```

```
    "username": "<username>",
    "role": "<role>"
  }
}
```

Ruota le chiavi per gli utenti con MFA abilitata per CloudHSM CLI

Segui questi passaggi per ruotare le chiavi per AWS CloudHSM gli utenti con l'autenticazione a più fattori (MFA) abilitata.

<result>

Hai firmato il file token in formato JSON generato con la tua chiave privata e registrato una nuova chiave pubblica MFA.
</result>

1. Utilizza la CLI di CloudHSM per accedere all'HSM come amministratore o come utente specifico che ha abilitato la MFA (consulta Accesso degli utenti con MFA abilitata per maggiori dettagli).
2. Esegui quindi il comando per modificare la tua strategia MFA. Devi fornire il parametro --token. Questo parametro specifica un file in cui verranno scritti token non firmati.

```
aws-cloudhsm > user change-mfa token-sign --token unsigned-tokens.json --
username <username> --role crypto-user --change-quorum
Enter password:
Confirm password:
```

3. Identifica il file con token non firmati che devono essere firmati: `unsigned-tokens.json`. Il numero di token in questo file dipende dal numero di HSM nel cluster. Ogni token rappresenta un HSM. Questo file è in formato JSON e contiene token che devono essere firmati per dimostrare che disponi di una chiave privata. Questa sarà la nuova chiave privata della nuova coppia di public/private chiavi RSA che desideri utilizzare per ruotare la chiave pubblica attualmente registrata.

```
$ cat unsigned-tokens.json
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=",
      "signed": ""
```

```
    },
    {
      "unsigned": "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=",
      "signed": ""
    },
    {
      "unsigned": "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=",
      "signed": ""
    }
  ]
}
```

4. Firma questi token con la chiave privata creata in precedenza durante la configurazione. Per prima cosa, devi estrarre e decodificare i token codificati in base64.

```
$ echo "Vtf/9QOFY45v/E1osvpEMr59JsnP/hLDm4ItOO2vqL8=" > token1.b64
$ echo "wVbC0/5IKwjyZK2NBpdFLyI7BiayZ24YcdUdlcxLwZ4=" > token2.b64
$ echo "z6aW9RzErJBL5KqFG5h8lhTVt9oLbxppjod0Ebysydw=" > token3.b64
$ base64 -d token1.b64 > token1.bin
$ base64 -d token2.b64 > token2.bin
$ base64 -d token3.b64 > token3.bin
```

5. Ora hai dei token binari. Firmali utilizzando la chiave privata RSA creata in precedenza durante la configurazione.

```
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token1.bin \
      -out token1.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token2.bin \
      -out token2.sig.bin
$ openssl pkeyutl -sign \
      -inkey officer1.key \
      -pkeyopt digest:sha256 \
      -keyform PEM \
      -in token3.bin \
```

```
      -out token3.sig.bin
```

6. Ora hai le firme binarie dei token. Codificali in base64 e inseriscili nuovamente nel tuo file token.

```
$ base64 -w0 token1.sig.bin > token1.sig.b64
$ base64 -w0 token2.sig.bin > token2.sig.b64
$ base64 -w0 token3.sig.bin > token3.sig.b64
```

7. Infine, copia e incolla nuovamente i valori base64 nel tuo file del token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/
TK0PVaxLN42X+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/
mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQ
    }
  ]
}
```

8. Ora che il tuo file del token ha tutte le firme richieste, puoi procedere. Inserisci il nome del file contenente i token firmati e premi il tasto INVIO. Inserisci poi il percorso della tua nuova chiave pubblica. Ora vedrai quanto segue come parte dell'output dell'elenco degli utenti.

```
Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:officer1.pub
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "crypto-user"
  }
}
```

A questo punto hai configurato il tuo utente con la MFA.

```
{
    "username": "<username>",
    "role": "crypto-user",
    "locked": "false",
    "mfa": [
      {
        "strategy": "token-sign",
        "status": "enabled"
      }
    ],
    "cluster-coverage": "full"
},
```

Annullare la registrazione di una chiave pubblica MFA utilizzando la CLI di CloudHSM

Segui questi passaggi per annullare la registrazione di una chiave pubblica di autenticazione a più fattori (MFA) per AWS CloudHSM gli utenti amministratori quando la chiave pubblica MFA è registrata.

1. Utilizza la CLI di CloudHSM per accedere all'HSM come amministratore con MFA abilitata.
2. Utilizza il comando user change-mfa token-sign per rimuovere la MFA per un utente.

```
aws-cloudhsm > user change-mfa token-sign --username <username> --role admin --
deregister --change-quorum
```

```
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "<username>",
    "role": "admin"
  }
}
```

Riferimento al file token per MFA con CLI CloudhSM

Il file token generato durante la registrazione di una chiave pubblica di autenticazione a più fattori (MFA) o quando si tenta di accedere alla CLI di CloudhSM tramite MFA è costituito da quanto segue:

- Token: un array di coppie di token codificate in base64 sotto forma di oggetti letterali JSON. unsigned/signed
- Non firmati: un token con codifica base64 sottoposto ad hashing SHA256.
- Firmati: un token firmato con codifica base64 (firma) del token non firmato, che utilizza la chiave privata RSA a 2048 bit.

```
{
  "version": "2.0",
  "tokens": [
    {
      "unsigned": "1jqwxb9bJOUUQLiNb7mxXS1uBJsEXh0B9nj05BqnPsE=",
      "signed": "eiw3fZeCKIY50C4zPeg9Rt90M1Qlq3WlJh6Yw7xXm4nF6e9ETLE39+9M
+rUqDWMRZjaBfaMbg5d9yDkz5p13U7ch2tlF9LoYabsWutkT014KRq/rcYMvFsU9n/Ey/TK0PVaxLN42X
+pebV4juwMhN4mK4CzdFAJgM+UGBOj4yB9recpOBB9K8QFSpJZALSEdDgUc/mS1eDq3rU0int6+4NKuLQjpR
+LSEIWRZ6g6+MND2vXGskxHjadCQ09L7Tz8VcWjKDbxJcBiGKvkqyozl9zrGo8fA3WHBmwiAgS61Merx77ZGY4PFR37+j/
YMSC14prCN15DtMRv2xA1SGSb4w=="
    },
    {
      "unsigned": "LMMFc34ASPnvNPFzBbMbr9FProS/Zu2P8zF/xzk5hVQ=",
      "signed": "HBImKnHmw+6R2TpFEpfiAg4+hu2pFNwn43ClhKPkn2higbEhUD0JVi
+4MerSyvU/NN79iWVxDvJ9Ito+jpiRQjTfTGEoIteyuAr1v/Bzh+HjmrO53OQpZaJ/VXGIgApD0myuu/
ZGNKQTCSkkL7+V81FG7yR1Nm22jUeGa735zvm/E+cenvZdy0VVx6A7WeWrl3JEKKBweHbi+7BwbaW
+PTdCuIRd4Ug76Sy+cFhsvcG1k7cMwDh8MgXzIZ2m1f/hdy2j8qAxORTLlmwyUOYvPYOvUhc
+s83hx36QpGwGcD7RA0bPT5OrTx7PHd0N1CL+Wwy91We8yIOFBS6nxo1R7w=="
    },
```

```
    {
      "unsigned": "dzeHbwhiVXQqcUGj563z51/7sLUdxjL93SbOUyZRjH8=",
      "signed": "VgQPvrTsvGljVBFxHnswduq16x8ZrnxfcYVYGf/
N7gEzI4At3GDs2EVZWTRdvS0uGHdkFYp1apHgJZ7PDVmGcTkIXVD2lFYppcgNlSzkYlftr5EOjqS9ZjYEqgGuB4g//
MxaBaRbJai/6BlcE92NIdBusTtreIm3yTpjIXNAVoeRSnkfuw7wZcL96QoklNb1WUuSHw
+psUyeIVtIwFMHEfFoRC0t
+VhmnlnFnkjGPb9W3Aprw2dRRvFM3R2ZTDvMCiOYDzUCd43GftGq2LfxH3qSD51oFHglHQVOY0jyVzzlAvub5HQdtOQdErI
    }
  ]
}
```


Gestisci l'autenticazione del quorum (controllo degli accessi M of N) utilizzando la CLI CloudhSM

AWS CloudHSM i cluster supportano l'autenticazione quorum, nota anche come controllo degli accessi M of N. Questa funzionalità richiede la collaborazione degli utenti HSM per determinate operazioni, aggiungendo un ulteriore livello di protezione.

Con l'autenticazione quorum, nessun singolo utente dell'HSM può eseguire operazioni controllate dal quorum sull'HSM. Invece, un numero minimo di utenti HSM (almeno 2) deve cooperare per eseguire queste operazioni.

L'autenticazione del quorum consente di controllare le seguenti operazioni:

- Gestione degli utenti HSM da parte dell'amministratore: creazione ed eliminazione di utenti HSM o modifica della password di un altro utente HSM. Per ulteriori informazioni, consulta Gestione degli utenti con autenticazione quorum abilitata per AWS CloudHSM utilizzo della CLI CloudhSM.

Punti chiave sull'autenticazione del quorum in. AWS CloudHSM

- Un utente HSM può firmare il proprio token del quorum, ovvero fornire una delle approvazioni richieste per l'autenticazione del quorum.
- È possibile scegliere il numero minimo di approvatori del quorum, che varia da due (2) a otto (8).
- Gli HSM possono memorizzare fino a 1024 token di quorum. Quando viene raggiunto questo limite, l'HSM elimina un token scaduto per crearne uno nuovo.
- Per impostazione predefinita, i token scadono dieci minuti dopo la creazione.

- Per i cluster con MFA abilitata, viene utilizzata la stessa chiave per l'autenticazione quorum e l'autenticazione a più fattori (MFA). Per ulteriori informazioni, consulta Utilizzo della CLI di CloudHSM per gestire l'MFA.
- Può esistere un solo token di quorum per servizio di amministrazione su ogni HSM alla volta. È necessario completare o eliminare il token del quorum attivo per un servizio prima di poterne generare uno nuovo per lo stesso servizio. Per ulteriori informazioni, consulta Servizi e tipi supportati.

I seguenti argomenti forniscono ulteriori informazioni sull'autenticazione del quorum in AWS CloudHSM.

Argomenti

- Processo di autenticazione quorum per CloudHSM CLI
- Supportata AWS CloudHSM nomi e tipi di servizi per l'autenticazione del quorum con CloudhSM CLI
- Imposta l'autenticazione quorum per AWS CloudHSM amministratori che utilizzano CloudhSM CLI
- Gestione degli utenti con autenticazione quorum abilitata per AWS CloudHSM utilizzo della CLI CloudhSM
- Modifica il valore minimo del quorum per AWS CloudHSM utilizzo della CLI CloudhSM

Processo di autenticazione quorum per CloudHSM CLI

I passaggi seguenti riassumono i processi di autenticazione del quorum per CloudHSM CLI. Per le operazioni e gli strumenti specifici, consultare Gestione degli utenti con autenticazione quorum abilitata per AWS CloudHSM utilizzo della CLI CloudhSM.

1. Ogni utente del modulo di sicurezza hardware (HSM) crea una chiave asimmetrica per la firma. Gli utenti completano questa operazione al di fuori dell'HSM, assicurandosi di proteggere la chiave in modo appropriato.
2. Ciascun utente HSM effettua l'accesso all'HSM e registra la parte pubblica della propria chiave di firma (la chiave pubblica) nell'HSM.
3. Quando un utente HSM desidera effettuare un'operazione controllata dal quorum, tale utente effettua l'accesso all'HSM e ottiene un token del quorum.
4. L'utente HSM assegna il token del quorum a uno o più utenti HSM e richiede la loro approvazione.

5. Gli altri utenti HSM approvano utilizzando le loro chiavi per firmare crittograficamente il token del quorum. Ciò si verifica al di fuori dell'HSM.
6. Quando l'utente HSM dispone del numero di approvazioni necessario, tale utente effettua l'accesso all'HSM ed esegue l'operazione controllata dal quorum con l'argomento --approval, fornendo il file del token del quorum firmato contenente tutte le approvazioni (firme) necessarie.
7. L'HSM utilizza la chiavi pubbliche registrate di ciascun firmatario per verificare le firme. Se le firme sono valide, l'HSM approva il token e l'operazione controllata dal quorum viene eseguita.

Supportata AWS CloudHSM nomi e tipi di servizi per l'autenticazione del quorum con CloudhSM CLI

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admin come la creazione e l'eliminazione di utenti, la modifica delle password degli utenti, l'impostazione dei valori del quorum e la disattivazione delle funzionalità quorum e MFA.

Crypto User Services: l'autenticazione Quorum viene utilizzata per i servizi privilegiati degli utenti crittografici associati a una chiave specifica, come la firma con una chiave, una chiave, sharing/unsharing una chiave e l'impostazione dell'attributo di wrapping/unwrapping una chiave. Il valore quorum di una chiave associata viene configurato quando la chiave viene generata, importata o aperta. Il valore del quorum deve essere uguale o inferiore al numero di utenti a cui è associata la chiave, che include gli utenti con cui la chiave è condivisa e il proprietario della chiave.

Ogni tipo di servizio è ulteriormente suddiviso in un nome di servizio qualificante, che contiene un set specifico di operazioni di servizio supportate dal quorum che possono essere eseguite.

Nome del servizio	Tipo di servizio	Operazioni di servizio
Utente	Admin	• user create • user delete • user change-password • user change-mfa
quorum	Admin	• quorum token-sign set-quorum-value
gruppo 1	Admin	• cluster mtls register-trust-anchor

Nome del servizio	Tipo di servizio	Operazioni di servizio
		• cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gestione delle chiavi	Utente Crypto	• involucro per chiavi • scartare le chiavi • Condivisione chiave • Annulla condivisione chiave • key set-attribute
utilizzo delle chiavi	Utente Crypto	• segno chiave

[1] Il servizio cluster è disponibile esclusivamente su hsm2m.medium

Imposta l'autenticazione quorum per AWS CloudHSM amministratori che utilizzano CloudhSM CLI

Gli argomenti seguenti descrivono i passaggi da completare per configurare il modulo di sicurezza hardware (HSM) in modo che gli amministratori possano utilizzare l'autenticazione quorum. AWS CloudHSM È necessario eseguire questa procedura una sola volta quando si configura l'autenticazione del quorum per gli amministratori per la prima volta. Una volta completata questa procedura, consultare Gestione degli utenti con autenticazione quorum abilitata per AWS CloudHSM utilizzo della CLI CloudhSM.

Argomenti

- Prerequisiti
- Passaggio 1. Creazione e registrazione di una chiave per la firma
- Passaggio 2. Impostazione del valore minimo del quorum sull'HSM
- Valori minimi del quorum

Prerequisiti

Per comprendere questo esempio, è bene avere familiarità con la CLI di CloudHSM.

Passaggio 1. Creazione e registrazione di una chiave per la firma

Per utilizzare l'autenticazione del quorum, ogni amministratore deve completare tutti i seguenti passaggi:

Argomenti

- Creazione di una coppia di chiavi RSA
- Creazione e firma di un token di registrazione
- Registrazione della chiave pubblica con HSM

Creazione di una coppia di chiavi RSA

Esistono molti modi diversi per creare e proteggere una coppia di chiavi. Gli esempi a seguire mostrano come eseguire questa operazione con OpenSSL.

Example- Creazione di una chiave privata con OpenSSL

L'esempio seguente dimostra come utilizzare OpenSSL per creare una chiave RSA a 2048 bit. Per utilizzare questo esempio, sostituiscilo *<admin.key>* con il nome del file in cui desideri memorizzare la chiave.

```
$ openssl genrsa -out <admin.key>
Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
```

Successivamente, genera la chiave pubblica utilizzando la chiave privata appena creata.

Example- Creazione di una chiave pubblica con OpenSSL

L'esempio seguente dimostra come utilizzare OpenSSL per creare una chiave pubblica dalla chiave privata appena creata.

```
$ openssl rsa -in admin.key -outform PEM -pubout -out admin1.pub
writing RSA key
```

Creazione e firma di un token di registrazione

Crea un token e firmalo con la chiave privata appena generata nella fase precedente.

Example- Creazione di un token di registrazione

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Crea un token di registrazione eseguendo il comando quorum token-sign generate:

   ```
   aws-cloudhsm > quorum token-sign generate --service registration --token /path/
   tokenfile
   {
     "error_code": 0,
     "data": {
       "path": "/path/tokenfile"
     }
   }
   ```

3. Il comando quorum token-sign generate genera un token di registrazione nel percorso del file specificato. Ispeziona il file del token:

   ```
   $ cat /path/tokenfile
   {
     "version": "2.0",
     "tokens": [
       {
         "approval_data": <approval data in base64 encoding>,
         "unsigned": <unsigned token in base64 encoding>,
         "signed": ""
       }
     ]
   }
   ```

   Il file del token comprende:

- approval_data: un token di dati randomizzato con codifica base64 i cui dati non elaborati non superano il limite massimo di 245 byte.
- unsigned: un token con codifica base64 sottoposto ad hashing SHA-256 di approval_data.
- signed: un token firmato con codifica base64 (firma) del token non firmato, che utilizza la chiave privata RSA a 2048 bit generata precedentemente con OpenSSL.

Firma il token non firmato con la chiave privata per dimostrare di avere accesso alla chiave privata. Avrai bisogno del file del token di registrazione completamente compilato con una firma e la chiave pubblica per registrare l'amministratore come utente del quorum nel cluster. AWS CloudHSM

Example- Firma del token di registrazione non firmato

1. Decodifica il token non firmato con codifica base64 e inseriscilo in un file binario:

```
$ echo -n '6BMUj6mUjjko6ZLCEdzGlWpR5sILhFJfqhW1ej3Oq1g=' | base64 -d > admin.bin
```

2. Utilizza OpenSSL e la chiave privata per firmare il token di registrazione non firmato ora binario e crea un file di firma binario:

```
$ openssl pkeyutl -sign \
-inkey admin.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in admin.bin \
-out admin.sig.bin
```

3. Codifica la firma binaria in base64:

```
$ base64 -w0 admin.sig.bin > admin.sig.b64
```

4. Copia e incolla la firma con codifica base64 nel file token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "approval_data": <approval data in base64 encoding>,
```

```
      "unsigned": <unsigned token in base64 encoding>,
      "signed": <signed token in base64 encoding>
    }
  ]
}
```

Registrazione della chiave pubblica con HSM

Dopo aver creato una chiave, l'amministratore deve registrare la chiave pubblica nel cluster. AWS CloudHSM

Per registrare una chiave pubblica con l'HSM

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

   ```
   aws-cloudhsm > login --username <admin> --role admin
   Enter password:
   {
     "error_code": 0,
     "data": {
       "username": "<admin>",
       "role": "admin"
     }
   }
   ```

3. Utilizza il comando Registra la strategia del quorum di firma dei token di un utente utilizzando la CLI di CloudHSM per registrare una chiave pubblica. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help user change-quorum token-sign register.

Example— Registra una chiave pubblica con AWS CloudHSM cluster

L'esempio seguente mostra come usare il comando user change-quorum token-sign register nella CLI di CloudHSM per registrare una chiave pubblica di un amministratore nell'HSM. Per utilizzare questo comando, l'amministratore deve aver eseguito l'accesso all'HSM. Sostituire questi valori con i propri valori:

```
aws-cloudhsm > user change-quorum token-sign register --public-key </path/admin.pub> --
signed-token </path/tokenfile>
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

> **Note**
>
> /path/admin.pub: il percorso del file PEM a chiave pubblica
> Obbligatorio: sì
> /path/tokenfile: Il percorso del file con token firmato dalla chiave privata dell'utente
> Obbligatorio: sì

Una volta che tutti gli amministratori hanno registrato le proprie chiavi pubbliche, l'output del comando user list mostra l'avvenuta registrazione nel campo del quorum, indicando la strategia del quorum abilitata in uso, come illustrato di seguito:

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
```

```
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "admin2",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "admin3",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "admin4",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
```

```
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

In questo esempio, il AWS CloudHSM cluster ha due moduli di protezione hardware, ciascuno con gli stessi amministratori, come illustrato nel seguente output del comando. user list Per ulteriori informazioni sulla creazione di utenti, vedere Gestione degli utenti con CloudHSM CLI

Passaggio 2. Impostazione del valore minimo del quorum sull'HSM

Per utilizzare l'autenticazione del quorum, un amministratore deve effettuare l'accesso all'HSM e quindi impostare il valore minimo del quorum. Questo è il numero minimo di approvazioni dell'amministratore necessarie per l'esecuzione delle operazioni di gestione degli utenti HSM. Qualsiasi amministratore nell'HSM può impostare il valore minimo del quorum, compresi gli amministratori che non hanno registrato una chiave per la firma. Puoi modificare il valore minimo del quorum in qualsiasi momento. Per ulteriori informazioni, consulta Modifica del valore minimo.

Per impostare il valore minimo del quorum sull'HSM

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

   ```
   aws-cloudhsm > login --username <admin> --role admin
   ```

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

3. Utilizzare il comando Aggiornare un valore di quorum utilizzando la CLI di CloudhSM per impostare il valore minimo del quorum. Il `--service` flag identifica il servizio HSM per cui stai impostando i valori. Vedi l'esempio seguente o usa il help quorum token-sign set-quorum-value comando per ulteriori informazioni.

Example- Impostazione del valore minimo del quorum sull'HSM

Questo esempio utilizza un valore minimo del quorum pari a due (2). È possibile scegliere qualsiasi valore compreso tra due (2) e otto (8), fino al numero totale di amministratori sull'HSM. In questo esempio, l'HSM ha quattro (4) amministratori, quindi il valore massimo possibile è quattro (4).

Per utilizzare il comando di esempio seguente, sostituite il numero finale (*<2>*) con il valore minimo del quorum preferito.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service user --value <2>
{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

In questo esempio, il Mostra i valori del quorum utilizzando la CLI CloudhSM comando elenca i tipi, i nomi e le descrizioni dei servizi HSM inclusi nel servizio.

Valori minimi del quorum

Utilizza il comando quorum token-sign list-quorum-values per ottenere il valore minimo del quorum per un servizio:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
```

```
    "quorum": 1
  }
}
```

L'output del comando quorum token-sign list-quorum-values precedente mostra che il valore minimo del quorum per il servizio utente HSM, responsabile delle operazioni di gestione degli utenti, è ora due (2). Una volta completata questa procedura, consultare Gestione degli utenti con quorum (M of N).

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admin come la creazione e l'eliminazione di utenti, la modifica delle password degli utenti, l'impostazione dei valori del quorum e la disattivazione delle funzionalità quorum e MFA.

Crypto User Services: l'autenticazione Quorum viene utilizzata per i servizi privilegiati degli utenti crittografici associati a una chiave specifica, come la firma con una chiave, una chiave, sharing/unsharing una chiave e l'impostazione dell'attributo di wrapping/unwrapping una chiave. Il valore quorum di una chiave associata viene configurato quando la chiave viene generata, importata o aperta. Il valore del quorum deve essere uguale o inferiore al numero di utenti a cui è associata la chiave, che include gli utenti con cui la chiave è condivisa e il proprietario della chiave.

Ogni tipo di servizio è ulteriormente suddiviso in un nome di servizio qualificante, che contiene un set specifico di operazioni di servizio supportate dal quorum che possono essere eseguite.

Nome del servizio	Tipo di servizio	Operazioni di servizio
Utente	Admin	• user create • user delete • user change-password • user change-mfa
quorum	Admin	• quorum token-sign set-quorum-value
gruppo 1	Admin	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement

Nome del servizio	Tipo di servizio	Operazioni di servizio
gestione delle chiavi	Utente Crypto	• involucro per chiavi • scartare le chiavi • Condivisione chiave • Annulla condivisione chiave • key set-attribute
utilizzo delle chiavi	Utente Crypto	• segno chiave

[1] Il servizio cluster è disponibile esclusivamente su hsm2m.medium

Gestione degli utenti con autenticazione quorum abilitata per AWS CloudHSM utilizzo della CLI CloudhSM

Un AWS CloudHSM amministratore del modulo di sicurezza hardware (HSM) può configurare l'autenticazione quorum per le seguenti operazioni nel cluster: AWS CloudHSM

- Crea un AWS CloudHSM utente con CloudHSM CLI
- Eliminare un AWS CloudHSM utente con CloudHSM CLI
- Modifica della password di un utente con CloudhSM CLI
- La categoria user change-mfa nella CLI di CloudhSM

Dopo aver configurato il AWS CloudHSM cluster per l'autenticazione quorum, gli amministratori non possono eseguire autonomamente le operazioni di gestione degli utenti HSM. Nell'esempio seguente è mostrato l'output dopo che un amministratore ha tentato di creare un nuovo utente nell'HSM. Il comando ha esito negativo e viene restituito un errore che indica che è richiesta l'autenticazione del quorum.

```
aws-cloudhsm > user create --username user1 --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 1,
  "data": "Quorum approval is required for this operation"
}
```

Per svolgere un'operazione di gestione degli utenti HSM, un amministratore deve completare le seguenti attività:

Argomenti

- Passaggio 1. Ottenere un token del quorum
- Passaggio 2. Ottenere le firme dagli amministratori di approvazione
- Fase 3. Approva il token sul AWS CloudHSM raggruppa ed esegui un'operazione di gestione degli utenti

Passaggio 1. Ottenere un token del quorum

Innanzitutto, l'amministratore deve utilizzare la CLI di CloudHSM per richiedere un token del quorum.

Per ottenere un token del quorum

1. Utilizza il seguente comando per avviare la CLI di CloudHSM.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

   ```
   aws-cloudhsm > login --username <admin> --role admin
   Enter password:
   {
     "error_code": 0,
     "data": {
       "username": "<admin>",
       "role": "admin"
     }
   }
   ```

3. Utilizza il comando quorum token-sign generate per generare un token del quorum. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help quorum token-sign generate.

Example- Generare un token del quorum

In questo esempio si ottiene un token del quorum per l'amministratore con il nome utente `admin`, che viene salvato nel file `admin.token`. Per utilizzare il comando di esempio, sostituisci i valori i tuoi personali:

- *<admin>*— Il nome dell'amministratore che riceve il token. Deve essere lo stesso amministratore che ha eseguito l'accesso all'HSM e sta eseguendo il comando.
- *<admin.token>*— Il nome del file da utilizzare per archiviare il token del quorum.

Nel comando seguente, `user` identifica il nome del servizio per cui potrai utilizzare il token che stai generando. In questo caso, il token è destinato alle operazioni di gestione degli utenti HSM (servizio `user`).

```
aws-cloudhsm > login --username <admin> --role admin --password <password>
{
  "error_code": 0,
  "data": {
    "username": "<admin>",
    "role": "admin"
  }
}
```

```
aws-cloudhsm > quorum token-sign generate --service user --token </path/admin.token>
{
  "error_code": 0,
  "data": {
    "path": "/home/tfile"
  }
}
```

Il comando quorum token-sign generate genera un token del quorum per il servizio utente nel percorso del file specificato. Il file del token può essere ispezionato:

```
$ cat </path/admin.token>
{
  "version": "2.0",
  "service": "user-management",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJAlfK
+OWbNhZG1pbgAAAAAAAAAA
```

```
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAAAAAAAAAA
```

```
        "token-timeout-seconds": {
          "value": 597
        },
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Il tempo `token-timeout-seconds` indica il periodo di timeout in secondi per l'approvazione di un token generato prima della scadenza.

Passaggio 2. Ottenere le firme dagli amministratori di approvazione

Un amministratore che dispone di un token del quorum deve ottenerne l'approvazione da parte di altri amministratori. Per concedere l'approvazione, gli altri amministratori utilizzano la chiave di firma per firmare crittograficamente il token. Tale operazione viene svolta esternamente all'HSM.

Sono disponibili vari modi per firmare il token. L'esempio seguente mostra come eseguire questa operazione con OpenSSL. Per utilizzare uno strumento di firma diverso, assicurati che lo strumento utilizzi la chiave privata dell'amministratore (chiave di firma) per firmare un SHA-256 riassunto del token.

Example- Ottenere le firme dagli amministratori di approvazione

In questo esempio, l'amministratore che dispone del token (`admin`) necessita di almeno due (2) approvazioni. I seguenti comandi di esempio mostrano come due (2) amministratori possono utilizzare OpenSSL per firmare crittograficamente il token.

1. Decodifica il token non firmato con codifica base64 e inseriscilo in un file binario:

   ```
   $ echo -n '0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=' | base64 -d > admin.bin
   ```

2. Utilizza OpenSSL e la rispettiva chiave privata dell'approvatore (`admin3`) per firmare il token non firmato del quorum ora binario per il servizio utente e creare un file di firma binario:

   ```
   $ openssl pkeyutl -sign \
   -inkey admin3.key \
   -pkeyopt digest:sha256 \
   -keyform PEM \
   -in admin.bin \
   ```

```
-out admin.sig.bin
```

3. Codifica la firma binaria in base64:

```
$ base64 -w0 admin.sig.bin > admin.sig.b64
```

4. Infine, copia e incolla la firma con codifica base64 nel file token, seguendo il formato di valore letterale dell'oggetto JSON specificato in precedenza per la firma dell'approvatore:

```
{
  "version": "2.0",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJAlfK
+OWbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAAQADAAAAFQAAAAAAAAAW
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAABAAMAA
+b23gAAAAAAAAAA",
  "token": "0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": [
    {
      "username": "admin2",
      "role": "admin",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw=="
    },
    {
      "username": "admin3",
      "role": "admin",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw=="
    }
  ]
}
```

Fase 3. Approva il token sul AWS CloudHSM raggruppa ed esegui un'operazione di gestione degli utenti

Dopo che un amministratore ha le informazioni necessarie approvals/signatures, come descritto nella sezione precedente, può fornire quel token al AWS CloudHSM cluster insieme a una delle seguenti operazioni di gestione degli utenti:

- creazione
- Elimina
- Modifica-password
- user change-mfa

Per ulteriori informazioni sull'utilizzo di questi comandi, consulta Gestione degli utenti con CloudHSM CLI.

Durante la transazione, il token verrà approvato all'interno del AWS CloudHSM cluster ed eseguirà l'operazione di gestione degli utenti richiesta. La riuscita dell'operazione di gestione degli utenti dipende sia da un token del quorum approvato valido e sia da un'operazione di gestione degli utenti valida.

L'amministratore può utilizzare il token per un'unica operazione. Quando tale operazione va a buon fine, il token non è più valido. Per eseguire un'altra operazione di gestione degli utenti HSM, l'amministratore deve ripetere la procedura descritta sopra. Vale a dire che l'amministratore deve generare un nuovo token del quorum e nuove firme dagli approvatori, quindi approvare e utilizzare il nuovo token nell'HSM con l'operazione di gestione degli utenti richiesta.

> **Note**
>
> Il token del quorum è valido solo finché la sessione di accesso corrente è aperta. Se ti disconnetti dalla CLI di CloudHSM o se la rete si disconnette, il token non è più valido. Analogamente, un token autorizzato può essere utilizzato solo all'interno della CLI di CloudHSM. Non può essere utilizzato per l'autenticazione in un'applicazione diversa.

Example Creare un nuovo utente amministratore

Nel seguente esempio, un amministratore che ha effettuato l'accesso crea un nuovo utente nell'HSM:

```
aws-cloudhsm > user create --username user1 --role crypto-user --approval /path/
admin.token
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "user1",
    "role": "crypto-user"
  }
}
```

L'amministratore immette quindi il comando user list per confermare la creazione del nuovo utente:

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "admin2",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
```

```
      "cluster-coverage": "full"
    },
    {
      "username": "admin3",
      "role": "admin",
      "locked": "false",
      "mfa": [],
      "quorum": [
        {
          "strategy": "token-sign",
          "status": "enabled"
        }
      ],
      "cluster-coverage": "full"
    },
    {
      "username": "admin4",
      "role": "admin",
      "locked": "false",
      "mfa": [],
      "quorum": [
        {
          "strategy": "token-sign",
          "status": "enabled"
        }
      ],
      "cluster-coverage": "full"
    },
    {
      "username": "user1",
      "role": "crypto-user",
      "locked": "false",
      "mfa": [],
      "quorum": [],
      "cluster-coverage": "full"
    },
    {
      "username": "app_user",
      "role": "internal(APPLIANCE_USER)",
      "locked": "false",
      "mfa": [],
      "quorum": [],
      "cluster-coverage": "full"
    }
```

```
    ]
  }
}
```

Se l'amministratore tenta di eseguire un'altra operazione di gestione degli utenti HSM, questa avrà esito negativo con un errore di autenticazione del quorum:

```
aws-cloudhsm > user delete --username user1 --role crypto-user
{
  "error_code": 1,
  "data": "Quorum approval is required for this operation"
}
```

Come mostrato di seguito, il comando quorum token-sign list mostra che l'amministratore non ha token approvati. Per eseguire un'altra operazione di gestione degli utenti HSM, l'amministratore deve generare un nuovo token del quorum, ottenere nuove firme dagli approvatori ed eseguire l'operazione di gestione degli utenti desiderata con l'argomento --approval per fornire il token del quorum da approvare e utilizzato durante l'esecuzione dell'operazione di gestione degli utenti.

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": []
  }
}
```

Modifica il valore minimo del quorum per AWS CloudHSM utilizzo della CLI CloudhSM

Dopo aver impostato il valore minimo del quorum per gli amministratori di CloudhSM, potrebbe essere necessario modificare il valore minimo del quorum. L'HSM consente modifiche al valore minimo del quorum solo quando il numero di approvatori raggiunge o supera il valore corrente. Ad esempio, con un quorum minimo di due (2), almeno due (2) amministratori devono approvare eventuali modifiche.

> **Note**
>
> Il valore del quorum del servizio utente deve sempre essere inferiore o uguale al valore del quorum del servizio quorum. Per informazioni sui nomi dei servizi, vedere. Supportata AWS CloudHSM nomi e tipi di servizi per l'autenticazione del quorum con CloudhSM CLI

Per ottenere l'approvazione per modificare il valore minimo del quorum, occorre un token del quorum per il quorum service che utilizza il comando quorum token-sign set-quorum-value. Per generare un token del quorum per il quorum service che utilizza il comando quorum token-sign set-quorum-value, il servizio quorum deve essere maggiore di uno (1). Ciò significa che prima di poter modificare il valore minimo del quorum per il servizio utente, è possibile che occorra modificare il valore minimo del servizio quorum.

Passaggi per modificare il valore minimo del quorum per gli amministratori

1. Avvia la modalità interattiva CLI di CloudHSM.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizzando la CLI di CloudHSM, esegui l'accesso come amministratore.

   ```
   aws-cloudhsm > login --username <admin> --role admin
   Enter password:
   {
     "error_code": 0,
     "data": {
       "username": "<admin>",
       "role": "admin"
     }
   }
   ```

3. Controlla i valori minimi del quorum corrente:

```
aws-cloudhsm > quorum token-sign list-quorum-values
```

4. Se il valore minimo del quorum per il servizio quorum è inferiore al valore per il servizio utente, modifica il valore del servizio quorum:

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value <3>
```

5. Genera un token di quorum per il servizio quorum.
6. Ottieni le approvazioni (firme) dagli altri amministratori.
7. Approva il token sul cluster CloudHSM ed esegui un'operazione di gestione degli utenti. .
8. Modifica il valore minimo del quorum per il servizio utenti:

```
aws-cloudhsm > quorum token-sign set-quorum-value
```

Example Regolazione servizio per il quorum valori minimi

1. Controlla i valori correnti. L'esempio mostra che il valore minimo del quorum per il servizio utente è attualmente due (2).

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 1
  }
}
```

2. Modifica il valore del servizio quorum. Imposta il valore minimo del quorum per il servizio quorum su un valore uguale o superiore al valore per il servizio utente. Questo esempio imposta il valore minimo del quorum per il servizio quorum su due (2), lo stesso valore impostato per il servizio utente nell'esempio precedente.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2
{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

3. Verifica le modifiche. Questo esempio mostra che il valore minimo del quorum è ora due (2) per il servizio utente e il servizio quorum.

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 2
  }
}
```

Gestione degli utenti HSM con CloudHSM Management Utility (CMU)

Per gestire gli utenti del modulo di sicurezza hardware (HSM) AWS CloudHSM, è necessario accedere all'HSM con il nome utente e la password di un responsabile della crittografia (CO). Solo COs può gestire gli utenti. L'HSM contiene un CO predefinito denominato admin. Hai impostato la password per admin quando hai attivato il cluster.

Questo argomento fornisce step-by-step istruzioni e dettagli sulla gestione degli utenti HSM con AWS CloudHSM Management Utility (CMU).

Argomenti

- Prerequisiti per la gestione degli utenti in Management Utility AWS CloudHSM
- Tipi di utente HSM per Management Utility AWS CloudHSM
- tabella delle autorizzazioni utente HSM per Management Utility AWS CloudHSM
- Crea utenti HSM utilizzando Management Utility AWS CloudHSM
- Elenca tutti gli utenti HSM del cluster utilizzando Management Utility AWS CloudHSM
- Modificare le password degli utenti HSM utilizzando Management Utility AWS CloudHSM
- Eliminare gli utenti HSM utilizzando Management Utility AWS CloudHSM
- Gestisci la 2FA per gli utenti utilizzando AWS CloudHSM Management Utility
- Utilizzo di CloudHSM Management Utility (CMU) per gestire l'autenticazione del quorum (controllo dell'accesso "M of N")

Prerequisiti per la gestione degli utenti in Management Utility AWS CloudHSM

Prima di utilizzare AWS CloudHSM Management Utility (CMU) per gestire gli utenti dei moduli di sicurezza hardware (HSM) in AWS CloudHSM, è necessario completare questi prerequisiti. I seguenti argomenti descrivono come iniziare a usare la CMU.

Sections

- Ottieni l'indirizzo IP di un HSM in AWS CloudHSM
- Uso della CMU con i Client SDK 3.2.1 e versioni precedenti
- Download della CloudHSM Management Utility

Ottieni l'indirizzo IP di un HSM in AWS CloudHSM

Per utilizzare la CMU, è necessario utilizzare lo strumento di configurazione per aggiornare la configurazione locale. La CMU crea la propria connessione al cluster e tale connessione non riconosce il cluster. Per tenere traccia delle informazioni sul cluster, la CMU mantiene un file di configurazione locale. Questo significa che ogni volta che usi la CMU, devi innanzitutto aggiornare il file di configurazione eseguendo lo strumento da riga di comando configure con il parametro `--cmu`. Se usi il Client SDK 3.2.1 o versioni precedenti, devi adoperare un parametro diverso da `--cmu`. Per ulteriori informazioni, consulta the section called "Uso della CMU con i Client SDK 3.2.1 e versioni precedenti".

Il parametro `--cmu` richiede l'aggiunta dell'indirizzo IP di un HSM nel cluster. Se ne hai più HSMs, puoi usare qualsiasi indirizzo IP. Questo garantisce che la CMU possa propagare le modifiche apportate all'intero cluster. Ricorda che la CMU utilizza il suo file locale per tenere traccia delle informazioni sul cluster. Se il cluster è cambiato dall'ultima volta che hai usato la CMU da un determinato host, devi aggiungere tali modifiche al file di configurazione locale memorizzato su quell'host. Non aggiungere o rimuovere mai un HSM mentre usi la CMU.

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Per aprire la pagina dei dettagli del cluster, nella tabella dei cluster, scegli l'ID del cluster.

4. Per ottenere l'indirizzo IP, vai alla HSMs scheda. Per IPv4 i cluster, scegli un indirizzo elencato sotto l' IPv4 indirizzo ENI. Per i cluster dual-stack, utilizzare l'ENI o l'indirizzo ENI IPv4 . IPv6

Per ottenere un indirizzo IP per un HSM ()AWS CLI

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters dalla AWS CLI. Nell'output del comando, l'indirizzo IP di HSMs sono i valori di `EniIp` and `EniIpV6` (se si tratta di un cluster dual-stack).

```
$ aws cloudhsmv2 describe-clusters
{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
                    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
```

Uso della CMU con i Client SDK 3.2.1 e versioni precedenti

Con Client SDK 3.3.0, è AWS CloudHSM stato aggiunto il supporto per il `--cmu` parametro, che semplifica il processo di aggiornamento del file di configurazione per CMU. Se utilizzi una versione della CMU del Client SDK 3.2.1 o precedente, devi continuare a utilizzare i parametri `-a` and `-m` per aggiornare il file di configurazione. Per ulteriori informazioni sui parametri di configurazione, consulta Strumento Configure.

Download della CloudHSM Management Utility

L'ultima versione della CMU è disponibile per le attività di gestione degli utenti HSM indipendentemente dal fatto che si utilizzi il Client SDK 5 e il Client SDK 3.

Per scaricare e installare la CMU

- Scarica e installa la CMU.

  Amazon Linux

  ```
  $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-mgmt-util-latest.el6.x86_64.rpm
  ```

  ```
  $ sudo yum install ./cloudhsm-mgmt-util-latest.el6.x86_64.rpm
  ```

  Amazon Linux 2

  ```
  $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
  ```

  ```
  $ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
  ```

  CentOS 7.8+

  ```
  $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
  ```

  ```
  $ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
  ```

  CentOS 8.3+

  ```
  $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-mgmt-util-latest.el8.x86_64.rpm
  ```

  ```
  $ sudo yum install ./cloudhsm-mgmt-util-latest.el8.x86_64.rpm
  ```

  RHEL 7 (7.8+)

  ```
  $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-mgmt-util-latest.el7.x86_64.rpm
  ```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el7.x86_64.rpm
```

RHEL 8 (8.3+)

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-mgmt-util-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-mgmt-util_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-mgmt-util_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-mgmt-util_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-mgmt-util_latest_u18.04_amd64.deb
```

Windows Server 2012

1. Scarica la CloudHSM Management Utility.
2. Esegui il programma di installazione CMU (AWSCloudHSMManagementUtil-latest.msi) con privilegi amministrativi di Windows.

Windows Server 2012 R2

1. Scarica la CloudHSM Management Utility.
2. Esegui il programma di installazione CMU (AWSCloudHSMManagementUtil-latest.msi) con privilegi amministrativi di Windows.

Windows Server 2016

1. Scarica la CloudHSM Management Utility.
2. Esegui il programma di installazione CMU (AWSCloudHSMManagementUtil-latest.msi) con privilegi amministrativi di Windows.

Tipi di utente HSM per Management Utility AWS CloudHSM

La maggior parte delle operazioni eseguite sul modulo di sicurezza hardware (HSM) richiede le credenziali di un utente HSM. AWS CloudHSM La HSM autentica ogni utente HSM e ogni utente HSM dispone di un tipo che stabilisce quali operazioni può eseguire nell'HSM in qualità di utente.

> **Note**
>
> Gli utenti HSM sono diversi dagli utenti IAM. Gli utenti IAM che dispongono delle credenziali corrette possono creare HSMs interagendo con le risorse tramite l'API AWS. Dopo aver creato l'HSM, devi utilizzare le credenziali utente HSM per autenticare le operazioni sull'HSM.

Tipi di utente

- Precrypto officer (PRECO)
- Crypto officer (CO)
- Crypto user (CU)
- Utente dell'appliance (AU)

Precrypto officer (PRECO)

Sia sulla Cloud Management Utility (CMU) che sulla Key Management Utility (KMU), PRECO è un utente temporaneo che esiste solo sul primo HSM di un cluster AWS CloudHSM . Il primo HSM in un nuovo cluster contiene un utente PRECO che indica che questo cluster non è mai stato attivato. Per attivare un cluster, esegui cloudhsm-cli ed esegui il comando. cluster activate Accedere all'HSM e modificare la password di PRECO. Quando cambi la password, l'utente diventa un crypto officer (CO).

Crypto officer (CO)

Sia sulla Cloud Management Utility (CMU) che sulla Key Management Utility (KMU), un crypto officer (CO) può eseguire operazioni di gestione degli utenti. Ad esempio, può creare ed eliminare gli utenti e modificare le password degli utenti. Per ulteriori informazioni sugli utenti CO, consulta tabella delle autorizzazioni utente HSM per Management Utility AWS CloudHSM. Quando si attiva un nuovo cluster, l'utente passa da Precrypto Officer (PRECO) a Crypto Officer (CO).

Crypto user (CU)

Un utente di crittografia (CU) è in grado di eseguire le seguenti operazioni di crittografia e di gestione delle chiavi.

- Gestione chiavi: consente di creare, eliminare, condividere, importare ed esportare le chiavi di crittografia.
- Operazioni di crittografia: usa le chiavi di crittografia per la crittografia, la decrittografia, la firma, la verifica e altro ancora.

Per ulteriori informazioni, consulta tabella delle autorizzazioni utente HSM per Management Utility AWS CloudHSM.

Utente dell'appliance (AU)

L'utente dell'appliance (AU) può eseguire operazioni di clonazione e sincronizzazione sul cluster. HSMs AWS CloudHSM utilizza l'AU per sincronizzarli in un cluster. HSMs AWS CloudHSM L'AU esiste su tutti i HSMs servizi forniti da AWS CloudHSM e dispone di autorizzazioni limitate. Per ulteriori informazioni, consulta tabella delle autorizzazioni utente HSM per Management Utility AWS CloudHSM.

AWS non può eseguire alcuna operazione sul tuo HSMs . AWS non può visualizzare o modificare gli utenti o le chiavi e non può eseguire operazioni crittografiche utilizzando tali chiavi.

tabella delle autorizzazioni utente HSM per Management Utility AWS CloudHSM

Nella tabella seguente sono elencate le operazioni dei moduli di sicurezza hardware (HSM), ordinate in base al tipo di utente o sessione HSM in cui è possibile eseguire l'operazione. AWS CloudHSM

	Crypto officer (CO)	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
Ottenimento info cluster di base[1]	✓ Sì	✓ Sì	✓ Sì	✓ Sì
Modifica della propria password	✓ Sì	✓ Sì	✓ Sì	Non applicabile
Modifica della password di qualsiasi utente	✓ Sì	✗ No	✗ No	✗ No
Aggiunta, rimozione di utenti	✓ Sì	✗ No	✗ No	✗ No
Ottenimento stato sincronizzazione[2]	✓ Sì	✓ Sì	✓ Sì	✗ No
Estrazione, inserimento di oggetti nascosti[3]	✓ Sì	✓ Sì	✓ Sì	✗ No
Funzioni di gestione chiave[4]	✗ No	✓ Sì	✗ No	✗ No
Crittografia, decrittografia	✗ No	✓ Sì	✗ No	✗ No
Firma, verifica	✗ No	✓ Sì	✗ No	✗ No
Genera digest e HMACs	✗ No	✓ Sì	✗ No	✗ No

- [1] Le informazioni di base sul cluster includono il numero di componenti del HSMs cluster e l'indirizzo IP, il modello, il numero di serie, l'ID del dispositivo, l'ID del firmware di ciascun HSM, ecc.
- [2] L'utente può ottenere un set di digest (hash) corrispondenti alle chiavi dell'HSM. Un'applicazione può confrontare questi set di digest per comprendere lo stato di sincronizzazione di HSMs un cluster.
- [3] Gli oggetti mascherati sono chiavi crittografate prima di lasciare l'HSM. Non possono essere decrittografate esternamente all'HSM. Vengono decrittografate solo dopo essere state inserite in un HSM che si trova nello stesso cluster di quello da cui sono stati estratte. Un'applicazione può estrarre e inserire oggetti mascherati per sincronizzarli in un cluster. HSMs
- [4] Le funzioni di gestione chiave includono la creazione, l'eliminazione, il wrapping, l'annullamento del wrapping e la modifica degli attributi delle chiavi.

Crea utenti HSM utilizzando Management Utility AWS CloudHSM

Utilizzare createUser in AWS CloudHSM Management Utility (CMU) per creare nuovi utenti sul modulo di sicurezza hardware (HSM). Devi accedere come CO per creare un utente.

Per creare un nuovo utente CO

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

3. Accedi all'HSM come utente CO.

```
aws-cloudhsm > loginHSM CO admin co12345
```

Assicurati che il numero di connessioni elencate dalla CMU corrisponda al numero di connessioni HSMs presenti nel cluster. In caso contrario, disconnettiti e ricomincia da capo.

4. Usa createUser per creare un utente CO denominato **example_officer** con una password di **password1**.

```
aws-cloudhsm > createUser CO example_officer password1
```

La CMU richiede informazioni sull'operazione di creazione utente.

```
**********CAUTION**********
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?
```

5. Tipo **y**.

Per creare un nuovo utente CU

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

Linux

```
$ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

3. Accedi all'HSM come utente CO.

   ```
   aws-cloudhsm > loginHSM CO admin co12345
   ```

   Assicurati che il numero di connessioni degli elenchi CMU corrisponda al numero di connessioni HSMs presenti nel cluster. In caso contrario, disconnettiti e ricomincia da capo.

4. Usa createUser per creare un utente CU denominato **`example_user`** con una password di **`password1`**.

   ```
   aws-cloudhsm > createUser CU example_user password1
   ```

   La CMU richiede informazioni sull'operazione di creazione utente.

   ```
   ************************CAUTION*******************************
   This is a CRITICAL operation, should be done on all nodes in the
   cluster. AWS does NOT synchronize these changes automatically with the
   nodes on which this operation is not executed or failed, please
   ensure this operation is executed on all nodes in the cluster.
   **************************************************************

   Do you want to continue(y/n)?
   ```

5. Tipo **`y`**.

Per ulteriori informazioni su createUser, consulta createUser.

Elenca tutti gli utenti HSM del cluster utilizzando Management Utility AWS CloudHSM

Utilizzare il listUsers comando nell'utilità di AWS CloudHSM gestione (CMU) per elencare tutti gli utenti del cluster. AWS CloudHSM Non è necessario accedere per eseguire listUsers; tutti i tipi di utenti possono elencare utenti.

Per elencare tutti gli utenti del cluster

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

3. Usa listUsers per elencare tutti gli utenti del cluster.

   ```
   aws-cloudhsm > listUsers
   ```

   La CMU elenca tutti gli utenti del cluster.

   ```
   Users on server 0(10.0.2.9):
   Number of users found:4
   ```

```
    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              AU              app_user                                NO
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               NO
Users on server 1(10.0.3.11):
Number of users found:4

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              AU              app_user                                NO
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               NO
Users on server 2(10.0.1.12):
Number of users found:4

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              AU              app_user                                NO
               0               NO
         2              CO              example_officer                         NO
               0               NO
         3              CU              example_user                            NO
               0               NO
```

Per ulteriori informazioni su listUsers, consulta listUsers.

Modificare le password degli utenti HSM utilizzando Management Utility AWS CloudHSM

Utilizzare changePswd nella AWS CloudHSM Management Utility (CMU) per modificare la password di un utente del modulo di sicurezza hardware (HSM).

Solo i tipi e le password prevedono la distinzione tra lettere maiuscole e minuscole, non i nomi utente.

CO, crypto user (CU) e utente dell'applicazione (AU) possono modificare solo le proprie password. Per modificare la password di un altro utente, è necessario accedere come CO. Tuttavia, non potrai modificare la password di un utente che attualmente è connesso.

Per modificare la tua password

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

3. Accedi all'HSM.

   ```
   aws-cloudhsm > loginHSM CO admin co12345
   ```

   Assicurati che il numero di connessioni elencate dalla CMU corrisponda al numero di connessioni HSMs presenti nel cluster. In caso contrario, disconnettiti e ricomincia da capo.

4. Usa changePswd per modificare la tua password.

   ```
   aws-cloudhsm > changePswd CO example_officer <new password>
   ```

   La CMU richiede informazioni sull'operazione di modifica della password.

```
************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?
```

5. Tipo **y**.

   La CMU richiede informazioni sull'operazione di modifica della password.

   ```
   Changing password for example_officer(CO) on 3 nodes
   ```

Per modificare la password di un altro utente

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

3. Accedi all'HSM come utente CO.

```
aws-cloudhsm > loginHSM CO admin co12345
```

Assicurati che il numero di connessioni degli elenchi CMU corrisponda al numero di connessioni HSMs presenti nel cluster. In caso contrario, disconnettiti e ricomincia da capo.

4. Usa changePswd per modificare la password di un altro utente.

```
aws-cloudhsm > changePswd CU example_user <new password>
```

La CMU richiede informazioni sull'operazione di modifica della password.

```
**********************CAUTION*********************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?
```

5. Tipo **y**.

La CMU richiede informazioni sull'operazione di modifica della password.

```
Changing password for example_user(CU) on 3 nodes
```

Per ulteriori informazioni su changePswd, consulta changePswd.

Eliminare gli utenti HSM utilizzando Management Utility AWS CloudHSM

Utilizzare deleteUser nella AWS CloudHSM Management Utility (CMU) per eliminare un utente del modulo di sicurezza hardware (HSM). Per eliminare un altro utente devi accedere come CO.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

Come eliminare un utente

1. Usa lo strumento di configurazione per aggiornare la configurazione della CMU.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

3. Accedi all'HSM come utente CO.

   ```
   aws-cloudhsm > loginHSM CO admin co12345
   ```

   Assicurati che il numero di connessioni elencate dalla CMU corrisponda al numero di connessioni HSMs presenti nel cluster. In caso contrario, disconnettiti e ricomincia da capo.

4. Usa deleteUser per eliminare un utente.

   ```
   aws-cloudhsm > deleteUser CO example_officer
   ```

   La CMU elimina l'utente.

   ```
   Deleting user example_officer(CO) on 3 nodes
   deleteUser success on server 0(10.0.2.9)
   deleteUser success on server 1(10.0.3.11)
   deleteUser success on server 2(10.0.1.12)
   ```

Per ulteriori informazioni su deleteUser, consulta deleteUser.

Gestisci la 2FA per gli utenti utilizzando AWS CloudHSM Management Utility

Per una maggiore sicurezza, puoi configurare l'autenticazione a due fattori (2FA) per proteggere il cluster. AWS CloudHSM Puoi abilitare la 2FA solo per i crypto officer (CO).

Quando accedi a un cluster con un account del modulo di servizio hardware (HSM) abilitato a 2FA, fornisci a cloudhsm_mgmt_util (CMU) la tua password, il primo fattore, quello che conosci, e CMU ti fornisce un token e ti chiede di firmare il token. Per fornire il secondo fattore, quello che possiedi, firmi il token con una chiave privata da una coppia di chiavi che hai già creato e che è associata all'utente HSM. Per accedere al cluster, fornisci il token firmato alla CMU.

> **Note**
>
> Non è possibile abilitare la 2FA per crypto user (CU) o utenti dell'applicazione. L'autenticazione a due fattori (2FA) è solo per gli utenti CO.

Argomenti

- Autenticazione quorum e 2FA nei cluster utilizzando Management Utility AWS CloudHSM AWS CloudHSM
- Requisiti della coppia di chiavi 2FA per l' AWS CloudHSM utilizzo AWS CloudHSM della Management Utility
- Crea utenti con 2FA abilitato per gli utenti della Management Utility AWS CloudHSM
- Gestisci la 2FA per gli utenti HSM utilizzando Management Utility AWS CloudHSM
- Disattiva la 2FA per gli utenti HSM utilizzando Management Utility AWS CloudHSM
- Riferimento di configurazione per 2FA con Management Utility AWS CloudHSM

Autenticazione quorum e 2FA nei cluster utilizzando Management Utility AWS CloudHSM AWS CloudHSM

Il cluster utilizza la stessa chiave per l'autenticazione quorum e per l'autenticazione a due fattori (2FA). Ciò significa che un utente con 2FA abilitato viene effettivamente registrato per M-of-N-access

-control (MoFN). Per utilizzare correttamente la 2FA e l'autenticazione del quorum per lo stesso utente HSM, considera i seguenti punti:

- Se oggi utilizzi l'autenticazione del quorum per un utente, dovresti usare la stessa coppia di chiavi che hai creato per l'utente del quorum per abilitare la 2FA per l'utente.
- Se aggiungi il requisito 2FA per un utente non 2FA che non è un utente di autenticazione del quorum, registri quell'utente come utente MoFN con autenticazione 2FA.
- Se rimuovi il requisito 2FA o modifichi la password per un utente 2FA che è anche un utente di autenticazione del quorum, rimuoverai anche la registrazione dell'utente del quorum come utente MoFN.
- Se rimuovi il requisito 2FA o modifichi la password per un utente 2FA che è anche un utente di autenticazione del quorum, ma desideri comunque che quell'utente partecipi all'autenticazione del quorum, devi registrare nuovamente quell'utente come utente MoFN.

Per ulteriori informazioni sull'autenticazione del quorum, vedi Utilizzo di CMU per gestire l'autenticazione del quorum.

Requisiti della coppia di chiavi 2FA per l' AWS CloudHSM utilizzo AWS CloudHSM della Management Utility

Per abilitare l'autenticazione a due fattori (2FA) per un utente del modulo di sicurezza AWS CloudHSM hardware (HSM), utilizzate una chiave che soddisfi i seguenti requisiti.

È possibile creare una nuova coppia di chiavi o utilizzare una chiave esistente che soddisfi i seguenti requisiti.

- Tipo di chiave: asimmetrica
- Utilizzo della chiave: firma e verifica
- Specifiche della chiave: RSA_2048
- L'algoritmo di firma include:
  - `sha256WithRSAEncryption`

> **Note**
>
> Se si utilizza l'autenticazione del quorum o si prevede di utilizzare l'autenticazione del quorum, vedi the section called "Autenticazione quorum".

Crea utenti con 2FA abilitato per gli utenti della Management Utility AWS CloudHSM

Usa AWS CloudHSM Management Utility CMU (CMU) e la key pair per creare un nuovo utente di crypto office (CO) con l'autenticazione a due fattori (2FA) abilitata.

Per creare utenti CO con 2FA abilitata

1. Su un terminale, esegui le seguenti operazioni:

   a. Accedi al tuo HSM e accedi all'utility CloudHSM Management:

   ```
   /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   b. Accedi come CO e utilizza il comando seguente per creare una nuova MFA utente con 2FA:

   ```
   aws-cloudhsm > createUser CO MFA <CO USER PASSWORD> -2fa /home/ec2-user/
   authdata
   **************************CAUTION*********************************
   This is a CRITICAL operation, should be done on all nodes in the
   cluster. AWS does NOT synchronize these changes automatically with the
   nodes on which this operation is not executed or failed, please
   ensure this operation is executed on all nodes in the cluster.
   ******************************************************************

   Do you want to continue(y/n)? y

   Creating User exampleuser3(CO) on 1 nodesAuthentication data written to: "/
   home/ec2-user/authdata"Generate Base64-encoded signatures for SHA256 digests in
    the authentication datafile.
   To generate the signatures, use the RSA private key, which is the second factor
    ofauthentication for this user. Paste the signatures and the corresponding
    public keyinto the authentication data file and provide
   the file path below.Leave this field blank to use the path initially
    provided.Enter filename:
   ```

c. Lascia il terminale di cui sopra in questo stato. Non premere invio né inserire alcun nome di file.

2. In un altro terminale, segui i passi descritti di seguito:

a. Accedi al tuo HSM e accedi all'utility CloudHSM Management:

```
/opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

b. Genera una coppia di chiavi pubblica-privata usando i seguenti comandi:

```
openssl genpkey -algorithm RSA -out private_key.pem -pkeyopt
 rsa_keygen_bits:2048
```

```
openssl rsa -pubout -in private_key.pem -out public_key.pem
```

c. Esegui il comando seguente per installare una funzionalità di interrogazione json per estrarre il Digest dal file authdata:

```
sudo yum install jq
```

d. Per estrarre il valore digest, trova innanzitutto i seguenti dati nel file authdata:

```
{
  "Version":"1.0",
  "PublicKey":"",
  "Data":[
    {
      "HsmId": <"HSM ID">,
      "Digest": <"DIGEST">,
      "Signature": ""
    }
  ]
}
```

> **Note**
>
> Il digest ottenuto è codificato in base64, tuttavia per firmare il digest è necessario che il file venga prima decodificato e poi firmato. Il comando seguente decodificherà il digest e memorizzerà il contenuto decodificato in 'digest1.bin'

```
cat authdata | jq '.Data[0].Digest' | cut -c2- | rev | cut -c2- | rev |
 base64 -d > digest1.bin
```

e. Converti il contenuto della chiave pubblica, aggiungendo "\n" e rimuovendo gli spazi come mostrato di seguito:

```
-----BEGIN PUBLIC KEY-----\n<PUBLIC KEY>\n-----END PUBLIC KEY-----
```

> **Important**
>
> Il comando precedente mostra come "\n" viene aggiunto subito dopo BEGIN PUBLIC KEY-----, gli spazi tra "\n" e il primo carattere della chiave pubblica vengono rimossi, "\n" viene aggiunto prima di -----END PUBLIC KEY e gli spazi vengono rimossi tra "\n" e la fine della chiave pubblica.

Questo è il formato PEM per la chiave pubblica accettato nel file authdata.

f. Incolla il contenuto della chiave pubblica in formato pem nella sezione della chiave pubblica del file authdata.

```
vi authdata
```

```
{
  "Version":"1.0",
  "PublicKey":"-----BEGIN PUBLIC KEY-----\n<"PUBLIC KEY">\n-----END PUBLIC
 KEY-----",
  "Data":[
    {
      "HsmId":<"HSM ID">,
      "Digest":<"DIGEST">,
      "Signature": ""
    }
  ]
}
```

g. firma il token del file utilizzando il seguente comando:

```
openssl pkeyutl -sign -in digest1.bin -inkey private_key.pem -pkeyopt
 digest:sha256 | base64
Output Expected:
<"THE SIGNATURE">
```

> **Note**
>
> Come mostrato nel comando precedente, usa openssl pkeyutl al posto di openssl dgst per la firma.

h. Aggiungi il digest firmato nel file Authdata nel campo "Firma".

```
vi authdata
```

```
{
    "Version": "1.0",
    "PublicKey": "-----BEGIN PUBLIC KEY----- ... -----END PUBLIC KEY-----",
    "Data": [
        {
            "HsmId": <"HSM ID">,
            "Digest": <"DIGEST">,
            "Signature": <"Kkdl ... rkrvJ6Q==">
        },
        {
            "HsmId": <"HSM ID">,
            "Digest": <"DIGEST">,
            "Signature": <"K1hxy ... Q261Q==">
        }
    ]
}
```

3. Torna al primo terminale e premi: **`Enter`**

```
Generate Base64-encoded signatures for SHA256 digests in the authentication
 datafile. To generate the signatures, use the RSA private key,
which is the second factor ofauthentication for this user. Paste the signatures and
 the corresponding public keyinto the authentication data file and provide the file
 path below. Leave this field blank to use the path initially provided.
Enter filename: >>>>> Press Enter here
```

```
createUser success on server 0(10.0.1.11)
```

Gestisci la 2FA per gli utenti HSM utilizzando Management Utility AWS CloudHSM

Utilizzalo changePswd in AWS CloudHSM Management Utility (CMU) per modificare l'autenticazione a due fattori (2FA) per un utente. Ogni volta che abiliti la 2FA, devi fornire una chiave pubblica per gli accessi 2FA.

changePswdesegue uno dei seguenti scenari:

- Modifica della password per un utente 2FA
- Cambia la password per un utente non 2FA
- Aggiungi la 2FA a un utente non 2FA
- Rimuovi la 2FA da un utente 2FA
- Ruota la chiave per un utente 2FA

Puoi anche combinare le attività. Ad esempio, puoi rimuovere la 2FA da un utente e modificare la password contemporaneamente, oppure puoi ruotare la chiave 2FA e modificare la password dell'utente.

Per modificare le password o ruotare le chiavi per gli utenti CO con 2FA abilitata

1. Usa CMU per accedere all'HSM come CO con 2FA abilitata.
2. Utilizza changePswd per modificare la password o ruotare la chiave tra gli utenti CO con 2FA abilitata. Utilizza il parametro `-2fa` e includi una posizione nel file system in cui il sistema possa scrivere il file `authdata`. Questo file include un digest per ogni HSM del cluster.

   ```
   aws-cloudhsm > changePswd CO example-user <new-password> -2fa /path/to/authdata
   ```

   CMU richiede di utilizzare la chiave privata per firmare i digest del file `authdata` e restituire le firme con la chiave pubblica.
3. Utilizza la chiave privata per firmare i digest del file `authdata`, aggiungi le firme e la chiave pubblica al file in formato JSON `authdata` e quindi fornisci a CMU la posizione del file `authdata`. Per ulteriori informazioni, vedi the section called "Informazioni di riferimento sulla configurazione".

> **Note**
>
> Il cluster utilizza la stessa chiave per l'autenticazione del quorum e la 2FA. Se utilizzi o prevedi di utilizzare l'autenticazione del quorum, vedi the section called "Autenticazione quorum".

Disattiva la 2FA per gli utenti HSM utilizzando Management Utility AWS CloudHSM

Utilizzate la AWS CloudHSM Management Utility (CMU) per disabilitare l'autenticazione a due fattori (2FA) per gli utenti del modulo di sicurezza hardware (HSM) in. AWS CloudHSM

Per disabilitare la 2FA per gli utenti CO con la 2FA abilitata

1. Usa CMU per accedere all'HSM come CO con 2FA abilitata.
2. Utilizza changePswd per rimuovere la 2FA dagli utenti CO con 2FA abilitata.

   ```
   aws-cloudhsm > changePswd CO example-user <new password>
   ```

   CMU richiede di confermare l'operazione di modifica della password.

   > **Note**
   >
   > Se rimuovi il requisito 2FA o modifichi la password per un utente 2FA che è anche un utente di autenticazione del quorum, rimuoverai anche la registrazione dell'utente del quorum come utente MoFN. Per ulteriori informazioni su utenti del quorum e 2FA, vedi the section called "Autenticazione quorum".

3. Tipo **y**.

   CMU conferma l'operazione di modifica della password.

Riferimento di configurazione per 2FA con Management Utility AWS CloudHSM

Di seguito è riportato un esempio delle proprietà di autenticazione a due fattori (2FA) presenti nel `authdata` file sia per la richiesta generata dalla AWS CloudHSM Management Utility (CMU) che per le risposte.

```
{
    "Version": "1.0",
    "PublicKey": "-----BEGIN PUBLIC KEY----- ... -----END PUBLIC KEY-----",
    "Data": [
        {
            "HsmId": "hsm-lgavqitns2a",
            "Digest": "k5O1p3f6foQRVQH7S8Rrjcau6h3TYqsSdr16A54+qG8=",
            "Signature": "Kkdl ... rkrvJ6Q=="
        },
        {
            "HsmId": "hsm-lgavqitns2a",
            "Digest": "IyBcx4I5Vyx1jztwvXinCBQd9lDx8oQe7iRrWjBAi1w=",
            "Signature": "K1hxy ... Q261Q=="
        }
    ]
}
```

Dati

Nodo di primo livello. Contiene un nodo subordinato per ogni modulo HSM del cluster. Viene visualizzato nelle richieste e nelle risposte per tutti i comandi della 2FA.

Digest

Questo è ciò che devi firmare per fornire il secondo fattore di autenticazione. Generato da CMU nelle richieste per tutti i comandi della 2FA.

HsmId

L'ID del tuo HSM. Viene visualizzato nelle richieste e nelle risposte per tutti i comandi della 2FA.

PublicKey

La parte della chiave pubblica della coppia di chiavi generata è stata inserita come stringa in formato PEM. Inseriscila nelle risposte per createUser e changePswd.

Firma

Il digest firmato codificato in Base 64. Inseriscilo nelle risposte per tutti i comandi della 2FA.

Versione

La versione del file in formato JSON dei dati di autenticazione. Viene visualizzato nelle richieste e nelle risposte per tutti i comandi della 2FA.

Utilizzo di CloudHSM Management Utility (CMU) per gestire l'autenticazione del quorum (controllo dell'accesso "M of N")

Il HSMs tuo AWS CloudHSM cluster supporta l'autenticazione quorum, nota anche come controllo degli accessi M of N. Con l'autenticazione del quorum, nessun utente singolo sull'HSM può eseguire le operazioni controllate dal quorum sull'HSM. Invece, un numero minimo di utenti HSM (almeno 2) deve cooperare per eseguire queste operazioni. L'autenticazione del quorum ti consente di aggiungere un ulteriore livello di protezione, in quanto richiede l'approvazione da parte di più utenti HSM.

L'autenticazione del quorum consente di controllare le seguenti operazioni:

- Gestione degli utenti HSM da parte di funzionari crittografici (COs): creazione ed eliminazione di utenti HSM e modifica della password di un altro utente HSM. Per ulteriori informazioni, consulta Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility.

Ricorda le seguenti informazioni aggiuntive sull'utilizzo dell'autenticazione del quorum in AWS CloudHSM.

- Un utente HSM può firmare il proprio token del quorum, ovvero, l'utente richiedente può fornire una delle approvazioni richieste per l'autenticazione del quorum.
- È possibile scegliere il numero minimo di approvatori del quorum per le operazioni controllate dal quorum. Il numero minore che si può scegliere è due (2) e il numero maggiore è otto (8).
- L'HSM può archiviare fino a 1.024 token del quorum. Se l'HSM dispone già di 1.024 token quando tenta di crearne uno nuovo, l'HSM elimina uno di quelli scaduti. Per impostazione predefinita, i token scadono dieci minuti dopo la loro creazione.
- Il cluster utilizza la stessa chiave per l'autenticazione del quorum e per l'autenticazione a due fattori (2FA). Per ulteriori informazioni sull'utilizzo dell'autenticazione del quorum e dell'autenticazione a due fattori, vedi Autenticazione del quorum e 2FA.

I seguenti argomenti forniscono ulteriori informazioni sull'autenticazione del quorum in AWS CloudHSM.

Argomenti

- Processo di autenticazione Quorum per Management Utility AWS CloudHSM

- Imposta l'autenticazione del quorum per AWS CloudHSM i funzionari crittografici
- Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility
- Modifica il valore minimo del quorum con Management Utility AWS CloudHSM

Processo di autenticazione Quorum per Management Utility AWS CloudHSM

Le seguenti operazioni riepilogano i processi di autenticazione del quorum. Per le operazioni e gli strumenti specifici, consultare Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility.

1. Ciascun utente HSM crea una chiave asimmetrica per la firma. Completa questa operazione al di fuori dell'HSM, assicurandosi di proteggere la chiave in modo appropriato.
2. Ciascun utente HSM effettua l'accesso all'HSM e registra la parte pubblica della propria chiave di firma (la chiave pubblica) nell'HSM.
3. Quando un utente HSM desidera effettuare un'operazione controllata dal quorum, ciascun utente accede all'HSM e ottiene un token del quorum.
4. L'utente HSM assegna il token del quorum a uno o più utenti HSM e richiede la loro approvazione.
5. Gli altri utenti HSM approvano utilizzando le loro chiavi per firmare crittograficamente il token del quorum. Ciò si verifica al di fuori dell'HSM.
6. Quando l'utente HSM raggiunge il numero richiesto di approvazioni, accede all'HSM e fornisce il token del quorum e le approvazioni (firme) all'HSM.
7. L'HSM utilizza la chiavi pubbliche registrate di ciascun firmatario per verificare le firme. Se le firme sono valide, l'HSM approva il token.
8. L'utente HSM può quindi eseguire un'operazione controllata dal quorum.

Imposta l'autenticazione del quorum per AWS CloudHSM i funzionari crittografici

I seguenti argomenti descrivono i passaggi da completare per configurare il modulo di sicurezza hardware (HSM) in modo che i responsabili AWS CloudHSM crittografici () possano utilizzare l'autenticazione quorum COs. È necessario eseguire questi passaggi solo una volta quando si configura per la prima volta l'autenticazione del quorum per. COs Una volta completata questa procedura, consultare Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility.

Argomenti

- Prerequisiti
- Passaggio 1. Creazione e registrazione di una chiave per la firma
- Passaggio 2. Impostazione del valore minimo del quorum sull'HSM

Prerequisiti

Per comprendere questo esempio, è bene avere familiarità con lo strumento a riga di comando cloudhsm_mgmt_util (CMU). In questo esempio, il AWS CloudHSM cluster ne ha due HSMs, ognuna con la stessa COs caratteristica, come illustrato nel seguente output del listUsers comando. Per ulteriori informazioni sulla creazione degli utenti, vedere Utenti HSM.

```
aws-cloudhsm > listUsers
Users on server 0(10.0.2.14):
Number of users found:7

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                               NO
           0               NO
         2              AU              app_user                            NO
           0               NO
         3              CO              officer1                            NO
           0               NO
         4              CO              officer2                            NO
           0               NO
         5              CO              officer3                            NO
           0               NO
         6              CO              officer4                            NO
           0               NO
         7              CO              officer5                            NO
           0               NO
Users on server 1(10.0.1.4):
Number of users found:7

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                               NO
           0               NO
         2              AU              app_user                            NO
           0               NO
```

```
        3               CO              officer1                        NO
          0               NO
        4               CO              officer2                        NO
          0               NO
        5               CO              officer3                        NO
          0               NO
        6               CO              officer4                        NO
          0               NO
        7               CO              officer5                        NO
          0               NO
```

Passaggio 1. Creazione e registrazione di una chiave per la firma

Per utilizzare l'autenticazione del quorum, ogni CO deve eseguire tutti i seguenti passaggi:

Argomenti

- Creazione di una coppia di chiavi RSA
- Creazione e firma di un token di registrazione
- Registrazione della chiave pubblica con HSM

Creazione di una coppia di chiavi RSA

Esistono molti modi diversi per creare e proteggere una coppia di chiavi. Gli esempi a seguire mostrano come eseguire questa operazione con OpenSSL.

Example- Creazione di una chiave privata con OpenSSL

L'esempio seguente spiega come utilizzare OpenSSL per creare una chiave RSA a 2.048 bit protetta da una passphrase. Per utilizzare questo esempio, *officer1.key* sostituitelo con il nome del file in cui desiderate memorizzare la chiave.

```
$ openssl genrsa -out <officer1.key> -aes256 2048
        Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
Enter pass phrase for officer1.key:
Verifying - Enter pass phrase for officer1.key:
```

Successivamente, genera la chiave pubblica utilizzando la chiave privata appena creata.

Example- Creazione di una chiave pubblica con OpenSSL

L'esempio seguente dimostra come utilizzare OpenSSL per creare una chiave pubblica dalla chiave privata appena creata.

```
$ openssl rsa -in officer1.key -outform PEM -pubout -out officer1.pub
Enter pass phrase for officer1.key:
writing RSA key
```

Creazione e firma di un token di registrazione

Crea un token e firmalo con la chiave privata appena generata nella fase precedente.

Example- Creazione di un token

Il token di registrazione è semplicemente un file con dati casuali che non supera la dimensione massima di 245 byte. Firma il token con la chiave privata per dimostrare di avere accesso alla chiave privata. Il comando seguente utilizza echo per reindirizzare una stringa in un file.

```
$ echo <token to be signed> > officer1.token
```

Firma il token e salvalo in un file di firma. Avrai bisogno del token firmato, del token non firmato e della chiave pubblica per registrare il CO come utente MofN nell'HSM.

Example- Firma del token

Utilizza OpenSSL e la chiave privata per firmare il token di registrazione e creare il file di firma.

```
$ openssl dgst -sha256 \
    -sign officer1.key \
    -out officer1.token.sig officer1.token
```

Registrazione della chiave pubblica con HSM

Dopo aver creato una chiave, il CO deve registrare la parte pubblica della chiave (chiave pubblica) con l'HSM.

Per registrare una chiave pubblica con l'HSM

1. Per avviare lo strumento a riga di comando cloudhsm_mgmt_util, utilizza il comando seguente.

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

2. Utilizza il comando loginHSM per effettuare l'accesso all'HSM come CO. Per ulteriori informazioni, consulta ???.
3. Utilizza il comando registerQuorumPubKey per registrare una chiave pubblica. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help registerQuorumPubKey.

Example- Registrazione di una chiave pubblica nell'HSM

L'esempio seguente mostra come usare il comando registerQuorumPubKey nello strumento a riga di comando cloudhsm_mgmt_util per registrare una chiave pubblica CO con HSM. Per utilizzare questo comando, il CO deve accedere all'HSM. Sostituire questi valori con i propri valori:

```
aws-cloudhsm > registerQuorumPubKey
 CO <officer1> <officer1.token> <officer1.token.sig> <officer1.pub>
*************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
***************************************************************

Do you want to continue(y/n)? y
registerQuorumPubKey success on server 0(10.0.2.14)
```

<officer1.token>

Il percorso a un file che contiene un token di registrazione non firmato. Può contenere qualsiasi dato casuale con dimensioni massime del file pari a 245 byte.

Obbligatorio: sì

<officer1.token.sig>

Il percorso di un file che contiene l'hash firmato dal meccanismo SHA256 _PKCS del token di registrazione.

Obbligatorio: sì

<officer1.pub>

Il percorso al file che contiene la chiave pubblica di una coppia di chiavi simmetriche RSA-2048. Utilizza la chiave privata per firmare il token di registrazione.

Obbligatorio: sì

Dopo aver COs registrato le proprie chiavi pubbliche, l'output del listUsers comando lo mostra nella `MofnPubKey` colonna, come mostrato nell'esempio seguente.

```
aws-cloudhsm > listUsers
Users on server 0(10.0.2.14):
Number of users found:7

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                           NO
           0               NO
         2              AU              app_user                        NO
           0               NO
         3              CO              officer1                       YES
           0               NO
         4              CO              officer2                       YES
           0               NO
         5              CO              officer3                       YES
           0               NO
         6              CO              officer4                       YES
           0               NO
         7              CO              officer5                       YES
           0               NO
Users on server 1(10.0.1.4):
Number of users found:7

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                           NO
           0               NO
         2              AU              app_user                        NO
           0               NO
         3              CO              officer1                       YES
           0               NO
         4              CO              officer2                       YES
           0               NO
         5              CO              officer3                       YES
           0               NO
         6              CO              officer4                       YES
           0               NO
```

```
        7             CO             officer5                                YES
          0              NO
```

Passaggio 2. Impostazione del valore minimo del quorum sull'HSM

Per utilizzare l'autenticazione quorum per COs, un CO deve accedere all'HSM e quindi impostare il valore minimo del quorum, noto anche come valore m. Questo è il numero minimo di approvazioni CO necessarie per l'esecuzione delle operazioni di gestione degli utenti HSM. Qualsiasi CO sull'HSM può impostare il valore minimo del quorum, compresi quelli COs che non hanno registrato una chiave per la firma. È possibile modificare il valore minimo del quorum in qualsiasi momento; per ulteriori informazioni, consultare Modifica del valore minimo.

Per impostare il valore minimo del quorum sull'HSM

1. Per avviare lo strumento a riga di comando cloudhsm_mgmt_util, utilizza il comando seguente.

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

2. Utilizza il comando loginHSM per effettuare l'accesso all'HSM come CO. Per ulteriori informazioni, consulta ???.
3. Utilizzare il comando setMVvalue per impostare il valore minimo del quorum. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help setMValue.

Example- Impostazione del valore minimo del quorum sull'HSM

Questo esempio utilizza un valore minimo del quorum pari a due. È possibile scegliere qualsiasi valore da due (2) a otto (8), fino al numero totale di COs sull'HSM. In questo esempio, l'HSM ne ha sei COs, quindi il valore massimo possibile è sei.

Per utilizzare il comando di esempio seguente, sostituite il numero finale (*2*) con il valore minimo del quorum preferito.

```
aws-cloudhsm > setMValue 3 <2>
*************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
***************************************************************
```

```
Do you want to continue(y/n)? y
Setting M Value(2) for 3 on 2 nodes
```

Nell'esempio precedente, il primo numero (3) identifica il servizio HSM di cui si sta impostando il valore minimo del quorum.

La tabella seguente elenca gli identificatori del servizio HSM con i relativi nomi, descrizioni e comandi inclusi nel servizio.

Identificatori servizio	Nome del servizio	Descrizione del servizio	Comandi HSM
3	USER_MGMT	Gestione degli utenti HSM	• createUser • deleteUser • changePswd (si applica solo quando si modifica la password di un utente HSM diverso)
4	MISC_CO	Servizio per CO vario	• setMValue

Per ottenere il valore minimo del quorum per un servizio, utilizzare il comando getMValue, come nell'esempio seguente.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

L'output del comando getMValue precedente mostra che il valore minimo del quorum per le operazioni di gestione degli utenti HSM (servizio 3) è ora due.

Una volta completata questa procedura, consultare Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility.

Gestione degli utenti con autenticazione del quorum abilitata per AWS CloudHSM Management Utility

Un ufficiale addetto alla AWS CloudHSM crittografia (CO) del modulo di sicurezza hardware (HSM) può configurare l'autenticazione quorum per le seguenti operazioni sull'HSM:

- Creazione di utenti HSM
- Eliminazione di utenti HSM
- Modifica della password di un altro utente HSM

Dopo aver configurato l'HSM per l'autenticazione del quorum, COs non può eseguire autonomamente le operazioni di gestione degli utenti HSM. Nell'esempio seguente è mostrato l'output dopo che un CO ha tentato di creare un nuovo utente nell'HSM. Il comando ha esito negativo con un errore `RET_MXN_AUTH_FAILED`, che indica che l'autenticazione del quorum non è stata effettuata correttamente.

```
aws-cloudhsm > createUser CU user1 password
**************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
****************************************************************

Do you want to continue(y/n)? y
Creating User user1(CU) on 2 nodes
createUser failed: RET_MXN_AUTH_FAILED
creating user on server 0(10.0.2.14) failed

Retry/Ignore/Abort?(R/I/A): A
```

Per svolgere un'operazione di gestione degli utenti HSM, i CO devono completare le seguenti attività:

1. Ottenere un token del quorum.
2. Ottieni approvazioni (firme) da altri. COs
3. Approvare il token sul modulo HSM.
4. Svolgere l'operazione di gestione degli utenti HSM.

Se non hai ancora configurato l'HSM per l'autenticazione quorum per, fallo ora. COs Per ulteriori informazioni, consulta Prima configurazione.

Passaggio 1. Ottenere un token del quorum

Per prima cosa il CO deve utilizzare lo strumento a riga di comando cloudhsm_mgmt_util per richiedere un token del quorum.

Per ottenere un token del quorum

1. Per avviare lo strumento a riga di comando cloudhsm_mgmt_util, utilizza il comando seguente.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

2. Utilizza il comando loginHSM per effettuare l'accesso all'HSM come CO. Per ulteriori informazioni, consulta ???.
3. Utilizza il comando getToken per ottenere un token del quorum. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help getToken.

Example- Ottenere un token del quorum

In questo esempio si ottiene un token del quorum per il CO con nome utente officer1, che viene salvato nel file `officer1.token`. Per utilizzare il comando di esempio, sostituisci i valori i tuoi personali:

- *`officer1`*— Il nome del CO che riceve il token. Deve essere lo stesso CO che ha eseguito l'accesso all'HSM e sta eseguendo il comando.
- *`officer1.token`*— Il nome del file da utilizzare per archiviare il token del quorum.

Nel comando seguente, `3` identifica il servizio per cui potrai utilizzare il token ottenuto. In questo caso, il token è destinato alle operazioni di gestione degli utenti HSM (servizio 3). Per ulteriori informazioni, consulta Passaggio 2. Impostazione del valore minimo del quorum sull'HSM.

```
aws-cloudhsm > getToken 3 officer1 officer1.token
getToken success on server 0(10.0.2.14)
Token:
Id:1
Service:3
Node:1
Key Handle:0
```

```
User:officer1
getToken success on server 1(10.0.1.4)
Token:
Id:1
Service:3
Node:0
Key Handle:0
User:officer1
```

Passaggio 2. Ottieni firme dopo l'approvazione COs

Un CO che ha un quorum token deve ottenere l'approvazione del token da altri. COs Per dare la propria approvazione, l'altro COs usa la propria chiave di firma per firmare crittograficamente il token. Tale operazione viene svolta esternamente all'HSM.

Sono disponibili vari modi per firmare il token. L'esempio seguente mostra come eseguire questa operazione con OpenSSL. Per utilizzare un altro strumento di firma, assicurati che lo strumento utilizzi la chiave privata del CO (chiave di firma) per firmare un digest SHA-256 del token.

Example— Ottieni firme dopo l'approvazione COs

In questo esempio, il CO che dispone del token (officer1) necessita di almeno due approvazioni. I seguenti comandi di esempio mostrano come due persone COs possono utilizzare OpenSSL per firmare crittograficamente il token.

Nel primo comando, officer1 firma il proprio token. Per utilizzare i seguenti comandi di esempio, sostituisci i valori con i tuoi personali:

- *officer1.key*e *officer2.key* — Il nome del file che contiene la chiave di firma del CO.
- *officer1.token.sig1*e *officer1.token.sig2* — Il nome del file da utilizzare per memorizzare la firma. Assicurati di salvare ogni firma in un file diverso.
- *officer1.token*— Il nome del file che contiene il token che il CO sta firmando.

```
$ openssl dgst -sha256 -sign officer1.key -out officer1.token.sig1 officer1.token
Enter pass phrase for officer1.key:
```

Nel comando seguente, officer2 firma lo stesso token.

```
$ openssl dgst -sha256 -sign officer2.key -out officer1.token.sig2 officer1.token
Enter pass phrase for officer2.key:
```

Fase 3. Approvazione del token firmato nell'HSM

Dopo che un CO ottiene il numero minimo di approvazioni (firme) da altri COs, deve approvare il token firmato sull'HSM.

Per approvare il token firmato nell'HSM.

1. Crea un file di approvazione del token. Per maggiori informazioni, consulta il seguente esempio:
2. Per avviare lo strumento a riga di comando cloudhsm_mgmt_util, utilizza il comando seguente.

```
$ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
```

3. Utilizza il comando loginHSM per effettuare l'accesso all'HSM come CO. Per ulteriori informazioni, consulta ???.
4. Utilizza il comando approveToken per approvare il token firmato, trasferendo il file di approvazione del token. Per maggiori informazioni, consulta il seguente esempio:

Example- Creazione di un file di approvazione del token e approvazione del token firmato nell'HSM

Il file di approvazione del token è un file di testo in un formato particolare richiesto dall'HSM. Il file contiene informazioni sui token, sui relativi approvatori e sulle firme degli approvatori. Di seguito è mostrato un esempio di file di approvazione del token.

```
# For "Multi Token File Path", type the path to the file that contains
# the token. You can type the same value for "Token File Path", but
# that's not required. The "Token File Path" line is required in any
# case, regardless of whether you type a value.
Multi Token File Path = officer1.token;
Token File Path = ;

# Total number of approvals
Number of Approvals = 2;

# Approver 1
# Type the approver's type, name, and the path to the file that
# contains the approver's signature.
Approver Type = 2; # 2 for CO, 1 for CU
Approver Name = officer1;
Approval File = officer1.token.sig1;

# Approver 2
```

```
# Type the approver's type, name, and the path to the file that
# contains the approver's signature.
Approver Type = 2; # 2 for CO, 1 for CU
Approver Name = officer2;
Approval File = officer1.token.sig2;
```

Dopo aver creato il file di approvazione token, il CO utilizza lo strumento a riga di comando di cloudhsm_mgmt_util per l'accesso all'HSM. Il CO utilizza quindi il comando approveToken per approvare il token, come mostrato nel seguente esempio. *approval.txt*Sostituiscilo con il nome del file di approvazione del token.

```
aws-cloudhsm > approveToken approval.txt
approveToken success on server 0(10.0.2.14)
approveToken success on server 1(10.0.1.4)
```

Se questo comando viene eseguito correttamente, l'HSM approva il token del quorum. Per controllare lo stato di un token, utilizza il comando listTokens, come mostrato nell'esempio di seguito. L'output del comando mostra che il token dispone del numero richiesto di approvazioni.

Il periodo di validità dei token indica per quanto tempo è garantita la persistenza del token nell'HSM. Potrai utilizzare il token anche dopo la scadenza del periodo di validità (zero secondi).

```
aws-cloudhsm > listTokens
====================
    Server 0(10.0.2.14)
====================
-------- Token - 0 ----------
Token:
Id:1
Service:3
Node:1
Key Handle:0
User:officer1
Token Validity: 506 sec
Required num of approvers : 2
Current num of approvals : 2
Approver-0: officer1
Approver-1: officer2
Num of tokens = 1

====================
    Server 1(10.0.1.4)
```

```
====================
-------- Token - 0 ----------
Token:
Id:1
Service:3
Node:0
Key Handle:0
User:officer1
Token Validity: 506 sec
Required num of approvers : 2
Current num of approvals : 2
Approver-0: officer1
Approver-1: officer2
Num of tokens = 1

listTokens success
```

Passaggio 4. Utilizza il token per operazioni di gestione degli utenti

Dopo avere ottenuto un token con il numero richiesto di approvazioni, come mostrato nella sezione precedente, il CO è in grado di eseguire una delle seguenti operazioni di gestione degli utenti HSM:

- Creare un utente HSM con il comando createUser
- Eliminare un utente HSM con il comando deleteUser
- Modificare la password di un altro utente HSM con il comando changePswd

Per ulteriori informazioni sull'utilizzo di questi comandi, consulta Utenti HSM.

Il CO può utilizzare il token per un'unica operazione. Quando tale operazione va a buon fine, il token non è più valido. Per eseguire un'altra operazione di gestione degli utenti HSM, il CO deve ottenere un nuovo token del quorum e nuove firme dagli approvatori, quindi approvare il nuovo token nell'HSM.

> **Note**
>
> Il token MofN è valido solo finché la sessione di accesso corrente è aperta. Se ti disconnetti da cloudhsm_mgmt_util o se la rete si disconnette, il token non è più valido. Analogamente, un token autorizzato può essere utilizzato solo all'interno di cloudhsm_mgmt_util e non può essere utilizzato per l'autenticazione in un'applicazione diversa.

Nel seguente comando di esempio, il CO crea un nuovo utente nell'HSM.

```
aws-cloudhsm > createUser CU user1 <password>
************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)? y
Creating User user1(CU) on 2 nodes
```

Dopo che quello precedente è stato eseguito correttamente, il comando listUsers successivo mostra il nuovo utente.

```
aws-cloudhsm > listUsers
Users on server 0(10.0.2.14):
Number of users found:8

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
        1               PCO             admin                            NO
          0               NO
        2               AU              app_user                         NO
          0               NO
        3               CO              officer1                        YES
          0               NO
        4               CO              officer2                        YES
          0               NO
        5               CO              officer3                        YES
          0               NO
        6               CO              officer4                        YES
          0               NO
        7               CO              officer5                        YES
          0               NO
        8               CU              user1                            NO
          0               NO
Users on server 1(10.0.1.4):
Number of users found:8

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
```

```
        1             PCO             admin                          NO
          0             NO
        2             AU              app_user                       NO
          0             NO
        3             CO              officer1                       YES
          0             NO
        4             CO              officer2                       YES
          0             NO
        5             CO              officer3                       YES
          0             NO
        6             CO              officer4                       YES
          0             NO
        7             CO              officer5                       YES
          0             NO
        8             CU              user1                          NO
          0             NO
```

Se il CO tenta di eseguire un'altra operazione di gestione degli utenti HSM, questa avrà esito negativo con un errore di autenticazione del quorum, come mostrato nel seguente esempio.

```
aws-cloudhsm > deleteUser CU user1
Deleting user user1(CU) on 2 nodes
deleteUser failed: RET_MXN_AUTH_FAILED
deleteUser failed on server 0(10.0.2.14)

Retry/rollBack/Ignore?(R/B/I): I
deleteUser failed: RET_MXN_AUTH_FAILED
deleteUser failed on server 1(10.0.1.4)

Retry/rollBack/Ignore?(R/B/I): I
```

Il comando listTokens mostra che il CO non dispone di token approvati, come illustrato nel seguente esempio. Per eseguire un'altra operazione di gestione degli utenti HSM, il CO deve ottenere un nuovo token del quorum e nuove firme dagli approvatori, quindi approvare il nuovo token nell'HSM.

```
aws-cloudhsm > listTokens
====================
    Server 0(10.0.2.14)
====================
Num of tokens = 0

====================
```

```
    Server 1(10.0.1.4)
====================
Num of tokens = 0

listTokens success
```

Modifica il valore minimo del quorum con Management Utility AWS CloudHSM

Dopo aver impostato il valore minimo del quorum in modo che i funzionari AWS CloudHSM crittografici (COs) possano utilizzare l'autenticazione del quorum, potresti voler modificare il valore minimo del quorum. Il modulo HSM ti consente di modificare il valore minimo del quorum solo se il numero di approvatori è uguale o superiore al valore minimo corrente. Ad esempio, se il valore minimo del quorum è due, almeno due COs devono approvare la modifica del valore minimo del quorum.

Per ottenere l'approvazione a modificare tale valore, occorre un token del quorum per il comando setMValue (servizio 4). Per ottenere un token del quorum per il comando setMValue setMValue (servizio 4), il valore minimo del quorum per il servizio 4 deve essere superiore a uno. Ciò significa che prima di poter modificare il valore minimo del quorum per COs (servizio 3), potrebbe essere necessario modificare il valore minimo del quorum per il servizio 4.

La tabella seguente elenca gli identificatori del servizio HSM con i relativi nomi, descrizioni e comandi inclusi nel servizio.

Identificatori servizio	Nome del servizio	Descrizione del servizio	Comandi HSM
3	`USER_MGMT`	Gestione degli utenti HSM	• createUser • deleteUser • changePswd (si applica solo quando si modifica la password di un utente HSM diverso)
4	`MISC_CO`	Servizio per CO vario	• setMValue

Per modificare il valore minimo del quorum per i responsabili della crittografia

1. Per avviare lo strumento a riga di comando cloudhsm_mgmt_util, utilizza il comando seguente.

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

2. Utilizza il comando loginHSM per effettuare l'accesso all'HSM come CO. Per ulteriori informazioni, consulta ???.
3. Utilizzare il comando getMVValue per ottenere il valore minimo del quorum per il servizio 3. Per maggiori informazioni, consulta il seguente esempio:
4. Utilizzare il comando getMVValue per ottenere il valore minimo del quorum per il servizio 4. Per maggiori informazioni, consulta il seguente esempio:
5. Se il valore minimo del quorum del servizio 4 è inferiore a quello del servizio 3, utilizzare il comando setMVValue per modificare il valore del servizio 4. Cambiare il valore del servizio 4 impostandolo su un valore uguale o superiore a quello del servizio 3. Per maggiori informazioni, consulta il seguente esempio:
6. Ottenere un token del quorum, accertandosi di specificare il servizio 4 come servizio per il quale è possibile utilizzare il token.
7. Ottieni approvazioni (firme) da altri. COs
8. Approvare il token sul modulo HSM.
9. Utilizzare il setMVValue comando per modificare il valore minimo del quorum per il servizio 3 (operazioni di gestione degli utenti eseguite da). COs

Example- Ottenimento dei valori minimi del quorum e modifica del valore per il servizio 4

Il seguente esempio di comando mostra che il valore minimo corrente del quorum per il servizio 3 è due.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

Il seguente esempio di comando mostra che il valore minimo corrente del quorum per il servizio 4 è uno.

```
aws-cloudhsm > getMValue 4
MValue of service 4[MISC_CO] on server 0 : [1]
```

```
MValue of service 4[MISC_CO] on server 1 : [1]
```

Per modificare il valore minimo del quorum del servizio 4, utilizzare il comando setMValue impostando un valore uguale o superiore al valore del servizio 3. L'esempio seguente imposta il valore minimo del quorum per il servizio 4 su due (2), lo stesso valore impostato per il servizio 3.

```
aws-cloudhsm > setMValue 4 2
**********CAUTION*********
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**********

Do you want to continue(y/n)? y
Setting M Value(2) for 4 on 2 nodes
```

I seguenti comandi mostrano che il valore minimo del quorum adesso è due per il servizio 3 e per il servizio 4.

```
aws-cloudhsm > getMValue 3
MValue of service 3[USER_MGMT] on server 0 : [2]
MValue of service 3[USER_MGMT] on server 1 : [2]
```

```
aws-cloudhsm > getMValue 4
MValue of service 4[MISC_CO] on server 0 : [2]
MValue of service 4[MISC_CO] on server 1 : [2]
```

Chiavi in AWS CloudHSM

Prima di poter utilizzare il AWS CloudHSM cluster per l'elaborazione delle criptovalute, è necessario creare utenti e chiavi sui moduli di sicurezza hardware (HSM) del cluster.

In AWS CloudHSM, utilizza uno dei seguenti strumenti per gestire le chiavi presenti HSMs nel cluster:

- Libreria PKCS #11
- Provider JCE
- Provider KSP e CNG
- CLI di CloudHSM

Prima di poter gestire le chiavi, è necessario accedere all'HSM con il nome utente e la password di un crypto user (CU). Solo un CU può creare una chiave. Il CU che crea una chiave ne diventa proprietario e gestore.

Per ulteriori informazioni sulla gestione delle chiavi in, consulta i seguenti argomenti AWS CloudHSM.

Argomenti

- Impostazioni di sincronizzazione e durabilità dei tasti in AWS CloudHSM
- Confezionamento delle chiavi AES AWS CloudHSM
- Utilizzo di chiavi affidabili in AWS CloudHSM
- Gestione delle chiavi con CloudHSM CLI
- Gestione delle chiavi con la AWS CloudHSM KMU

Impostazioni di sincronizzazione e durabilità dei tasti in AWS CloudHSM

AWS CloudHSM sincronizza ogni chiave token creata. La sincronizzazione delle chiavi è principalmente un processo automatico, ma è possibile utilizzare almeno due moduli di sicurezza hardware (HSM) nel cluster per rendere le chiavi più durevoli. Questo argomento descrive le impostazioni di sincronizzazione delle chiavi, i problemi comuni che i clienti riscontrano durante l'utilizzo delle chiavi in un cluster e le strategie per incrementare la durabilità delle chiavi.

Questo argomento descrive le impostazioni di sincronizzazione delle chiavi AWS CloudHSM, i problemi comuni che i clienti devono affrontare quando utilizzano le chiavi in un cluster e le strategie per rendere le chiavi più durevoli.

Argomenti

- AWS CloudHSM concetti chiave
- Comprensione AWS CloudHSM sincronizzazione dei tasti
- Modifica AWS CloudHSM impostazioni di durabilità delle chiavi del client
- Sincronizzazione delle chiavi tra clonate AWS CloudHSM cluster

AWS CloudHSM concetti chiave

Di seguito sono riportati i concetti da tenere a mente quando si lavora con le chiavi in AWS CloudHSM.

Chiavi token

: Chiavi persistenti create durante le operazioni di generazione, importazione o estrazione delle chiavi. AWS CloudHSM sincronizza le chiavi dei token in un cluster.

Chiavi di sessione

: Chiavi temporanee che esistono solo su un modulo di sicurezza hardware (HSM) del cluster. AWS CloudHSM non sincronizza le chiavi di sessione in un cluster.

Client-side sincronizzazione delle chiavi

: Un processo lato client che clona le chiavi token create durante le operazioni di generazione, importazione o annullamento del wrapping delle chiavi. È possibile incrementare la durabilità delle chiavi token eseguendo un cluster con un minimo di due HSM.

Server-side sincronizzazione dei tasti

: Clona periodicamente le chiavi su ogni HSM del cluster. Non richiede alcuna gestione.

Impostazioni di durabilità delle chiavi del client

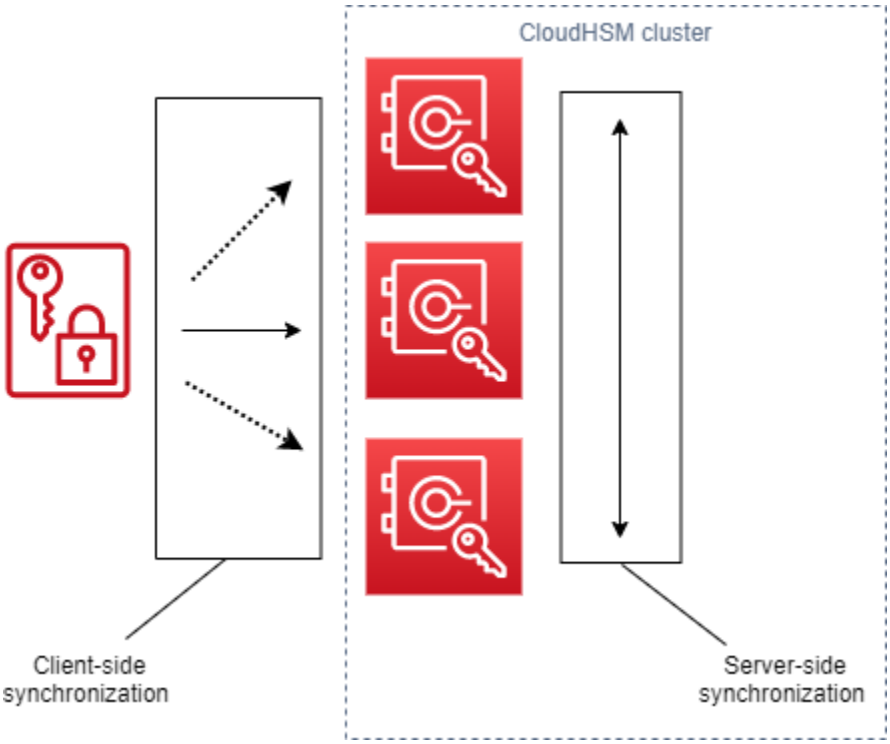
: Impostazioni che si configurano sul client che influiscono sulla durabilità delle chiavi. Queste impostazioni funzionano in modo diverso in Client SDK 5 e Client SDK 3.

  - In Client SDK 5, questa impostazione serve a eseguire un singolo cluster HSM.

- In Client SDK 3, questa impostazione serve a specificare il numero di HSM necessari per la riuscita delle operazioni di creazione delle chiavi.

Comprensione AWS CloudHSM sincronizzazione dei tasti

AWS CloudHSM utilizza la sincronizzazione delle chiavi per clonare le chiavi dei token su tutti i moduli di sicurezza hardware (HSM) di un cluster. Le chiavi token vengono create come chiavi persistenti durante le operazioni di generazione, importazione o annullamento del wrapping delle chiavi. Per distribuire le chiavi nel cluster, CloudHSM offre la sincronizzazione delle chiavi lato client e lato server.



L'obiettivo della sincronizzazione delle chiavi, sia lato server che lato client, è quello di distribuire nuove chiavi nel cluster il più rapidamente possibile dopo averle create. Questo è importante perché le successive chiamate effettuate per utilizzare nuove chiavi possono essere indirizzate a qualsiasi HSM disponibile nel cluster. Se la chiamata effettuata viene indirizzata a un HSM senza la chiave, la chiamata ha esito negativo. È possibile attenuare errori di questo tipo specificando che le applicazioni eseguiranno un nuovo tentativo per le chiamate successive effettuate dopo le operazioni di creazione delle chiavi. Il tempo necessario per la sincronizzazione può variare a seconda del carico di lavoro del cluster e di altri elementi indefiniti. Utilizzate le CloudWatch metriche per determinare la tempistica che l'applicazione deve impiegare in questo tipo di situazione. Per ulteriori informazioni, Parametri CloudWatch .

La difficoltà della sincronizzazione delle chiavi in un ambiente cloud è la durabilità delle chiavi. Le chiavi vengono create su un singolo HSM e spesso si inizia a utilizzarle immediatamente. Se l'HSM su cui vengono create le chiavi dovesse dare errore prima che le chiavi siano state clonate su un altro HSM del cluster, si perdono le chiavi e l'accesso a tutto ciò che è crittografato dalle chiavi. Per attenuare questo rischio, è disponibile la sincronizzazione lato client. La sincronizzazione lato client è un processo lato client che clona le chiavi create durante le operazioni di generazione, importazione o annullamento del wrapping delle chiavi. Clonare le chiavi man mano che vengono create ne migliora la durabilità. Chiaramente, non è possibile clonare le chiavi di un cluster con un singolo HSM. Per incrementare la durabilità delle chiavi, si consiglia inoltre di configurare il cluster per l'utilizzo di un minimo di due HSM. Con la sincronizzazione lato client e un cluster con due HSM, è possibile affrontare la difficoltà legata alla durabilità delle chiavi in un ambiente cloud.

Modifica AWS CloudHSM impostazioni di durabilità delle chiavi del client

La sincronizzazione delle chiavi è principalmente un processo automatico, ma è possibile gestire le impostazioni di durabilità delle chiavi sul lato client. Client-side le impostazioni di durabilità delle chiavi funzionano in modo diverso in Client SDK 5 e Client SDK 3.

- In Client SDK 5, viene introdotto il concetto di quorum di disponibilità delle chiavi che richiede l'esecuzione dei cluster con un minimo di due HSM. È possibile modificare le impostazioni di durabilità delle chiavi lato client per disattivare il requisito di utilizzo di due HSM. Per ulteriori informazioni sui quorum, consulta la pagina the section called “Concetti”.
- In Client SDK 3, si interviene sulle impostazioni di durabilità delle chiavi lato client per stabilire il numero di HSM su cui la creazione della chiave deve avere esito positivo per la riuscita dell'operazione complessiva.

Impostazioni di durabilità delle chiavi client di Client SDK 5

In Client SDK 5, la sincronizzazione delle chiavi è un processo completamente automatico. Con il quorum di disponibilità delle chiavi, le nuove chiavi create devono essere presenti su due HSM del cluster prima che l'applicazione possa utilizzare la chiave. Per utilizzare il quorum di disponibilità delle chiavi, il cluster deve avere almeno due HSM.

Se la configurazione del cluster non soddisfa i requisiti di durabilità delle chiavi, qualsiasi tentativo di creare o utilizzare una chiave token avrà esito negativo e nei log verrà visualizzato il seguente messaggio di errore:

```
Key <key handle> does not meet the availability requirements - The key must be
 available on at least 2 HSMs before being used.
```

È possibile utilizzare le impostazioni di configurazione del client per disattivare il quorum di disponibilità delle chiavi. Ad esempio, potresti volerlo disattivare per eseguire un cluster con un singolo HSM.

Concetti relativi a Client SDK 5

Quorum di disponibilità delle chiavi

AWS CloudHSM specifica il numero di moduli di protezione hardware in un cluster su cui devono esistere le chiavi prima che l'applicazione possa utilizzare la chiave. Richiede cluster con un minimo di due HSM.

Gestire le impostazioni di durabilità delle chiavi del client

Per gestire le impostazioni di durabilità delle chiavi del client è necessario utilizzare lo strumento di configurazione per Client SDK 5.

PKCS #11 library

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --disable-
key-availability-check
```

OpenSSL Dynamic Engine

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --disable-key-availability-check
```

OpenSSL Dynamic Engine Provider

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --disable-key-availability-
check
```

Key Storage Provider (KSP)

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-
key-availability-check
```

JCE provider

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-jce --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-
key-availability-check
```

CloudHSM CLI

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-cli --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --disable-
key-availability-check
```

Impostazioni di durabilità delle chiavi client di Client SDK 3

In Client SDK 3, la sincronizzazione delle chiavi è un processo prevalentemente automatico, ma è possibile incrementare la durabilità delle chiavi tramite le impostazioni di durabilità delle chiavi del client. È necessario specificare il numero di moduli di protezione hardware su cui la creazione della chiave deve avere esito positivo affinché l'intera operazione sia considerata riuscita. Client-side la sincronizzazione fa sempre del suo meglio per clonare le chiavi su ogni HSM del cluster, indipendentemente dall'impostazione scelta. L'impostazione impone la creazione delle chiavi sul numero di HSM specificato. Se viene specificato un valore e il sistema non è in grado di replicare la chiave su tale numero di HSM, il sistema pulisce automaticamente qualsiasi materiale della chiave indesiderato ed è quindi possibile riprovare.

> **Important**
>
> Se non vengono configurate le impostazioni di durabilità delle chiavi del client (o se si utilizza il valore predefinito 1), le chiavi sono vulnerabili alla perdita. Se l'HSM attuale dovesse dare errore prima che il servizio lato server abbia clonato la chiave su un altro HSM, il materiale della chiave andrà perso.

Per massimizzare la durabilità delle chiavi, valuta la possibilità di specificare almeno due HSM per la sincronizzazione lato client. Ricorda che, indipendentemente dal numero di HSM che specifichi, il carico di lavoro sul cluster rimane lo stesso. Client-side la sincronizzazione fa sempre del suo meglio per clonare le chiavi su ogni HSM del cluster.

Raccomandazioni

- Minimo: due HSM per cluster
- Massimo: un HSM in meno rispetto al numero totale di HSM del cluster

Se la sincronizzazione lato client dà errore, il servizio client pulisce tutte le chiavi indesiderate che potrebbero essere state create e che sono ora indesiderate. La pulizia viene eseguita al massimo delle capacità e potrebbe non sempre funzionare. Se pulizia non riesce, potrebbe essere necessario eliminare il materiale della chiave indesiderato. Per ulteriori informazioni, consulta la pagina Errori di sincronizzazione delle chiavi.

Impostare il file di configurazione per la durabilità delle chiavi del client

Per specificare le impostazioni di durabilità delle chiavi del client, è necessario modificare `cloudhsm_client.cfg`.

Per modificare il file di configurazione del client

1. Aprire `cloudhsm_client.cfg`.

   Linux:

   ```
   /opt/cloudhsm/etc/cloudhsm_client.cfg
   ```

   Windows:

```
C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

2. Nel `client` nodo del file, aggiungi `create_object_minimum_nodes` e specifica un valore per il numero minimo di HSM su cui AWS CloudHSM creare correttamente le chiavi affinché le operazioni di creazione delle chiavi abbiano esito positivo.

```
"create_object_minimum_nodes" : 2
```

> **Note**
>
> Lo strumento a riga di comando key_mgmt_util (KMU) presenta un'ulteriore impostazione per la durabilità delle chiavi del client. Per ulteriori informazioni, consulta the section called "Sincronizzazione lato client e KMU"

Informazioni di riferimento sulla configurazione

A seguire sono mostrate le proprietà di sincronizzazione lato client, in un estratto di `cloudhsm_client.cfg`:

```
{
    "client": {
        "create_object_minimum_nodes" : 2,
        ...
    },

    ...
}
```

create_object_minimum_nodes

Specifica il numero minimo di HSM necessari per garantire la riuscita delle operazioni di generazione, importazione o annullamento del wrapping delle chiavi. Se impostato, il valore predefinito è "1". Ciò significa che per ogni operazione di creazione di chiavi, il servizio lato client tenta di creare chiavi su ogni HSM del cluster, ma per la riuscita dell'operazione, il servizio deve creare solamente una singola chiave su un HSM del cluster.

Sincronizzazione lato client e KMU

Se si creano chiavi con lo strumento a riga di comando key_mgmt_util (KMU), si utilizza un parametro opzionale della riga di comando (`-min_srv`) per limitare il numero di HSM su cui clonare le chiavi. Se specificate il parametro della riga di comando e un valore nel file di configurazione, AWS CloudHSM rispetta il valore PIÙ GRANDE dei due valori.

Per ulteriori informazioni, consulta i seguenti argomenti:

- gen DSAKeyPair
- genere ECCKeyPair
- genere RSAKeyPair
- genere SymKey
- importare PrivateKey
- importare PubKey
- im SymKey
- insertMaskedObject
- uWrapKeyn

Sincronizzazione delle chiavi tra clonate AWS CloudHSM cluster

Client-side e la sincronizzazione lato server serve solo a sincronizzare le chiavi all'interno dello stesso cluster. AWS CloudHSM Se copi un backup di un cluster in un'altra regione, usa il comando key replicate per replicare una chiave tra due cluster. È possibile utilizzare cluster clonati per la ridondanza tra regioni o per semplificare il processo di ripristino di emergenza. Se non hai installato CloudHSM CLI, consulta le istruzioni in. ???

Confezionamento delle chiavi AES AWS CloudHSM

Questo argomento descrive le opzioni per il key wrapping in AWS CloudHSM AES. Il wrapping della chiave AES utilizza una chiave AES (la chiave di wrapping) per eseguire il wrapping di un'altra chiave di qualsiasi tipo (la chiave di destinazione). Il wrapping della chiave viene utilizzato per proteggere le chiavi archiviate o trasmettere le chiavi su reti non sicure.

Argomenti

- Algoritmi supportati
- Utilizzo del key wrap in AES AWS CloudHSM

Algoritmi supportati

AWS CloudHSM offre tre opzioni per il confezionamento delle chiavi AES, ognuna basata su come la chiave di destinazione viene riempita prima di essere inserita. Il padding viene eseguito automaticamente, in conformità con l'algoritmo in uso, quando si chiama il wrapping della chiave. Nella tabella seguente sono elencati gli algoritmi supportati e i dettagli associati che consentono di scegliere un meccanismo di wrapping appropriato per l'applicazione.

Algoritmo Wrapping Chiave AES	Specifiche	Tipi di chiavi di destinazione supportati	Schema di padding	AWS CloudHSM Disponibilità del cliente
Wrapping Chiavi AES con Zero Padding	RFC 5649 e SP 800—38F	Tutti	Aggiunge zeri dopo i bit chiave, se necessari o, per bloccare l'allineamento	SDK 3.1 e versioni successive
Wrapping Chiavi AES senza Padding	RFC 3394 e SP 800—38F	Tasti allineati a blocchi come AES e 3DES	Nessuno	SDK 3.1 e versioni successive
AES Wrapping Chiavi con Padding PKCS #5	Nessuno	Tutti	Almeno 8 byte vengono aggiunti come da schema di riempimento PKCS #5 per bloccare l'allinea mento	Tutti

Per informazioni su come utilizzare gli algoritmi Wrapping Chiavi AES della tabella precedente nell'applicazione, vedi la sezione Utilizzo di AES Key Wrap in AWS CloudHSM.

Comprensione dei vettori di inizializzazione in Wrapping Chiavi AES

Prima di eseguire il wrapping, CloudHSM aggiunge un vettore di inizializzazione (IV) alla chiave di destinazione per l'integrità dei dati. Ogni algoritmo di wrapping delle chiavi dispone di restrizioni specifiche sul tipo di IV consentito. Per impostare l'IV AWS CloudHSM, hai due opzioni:

- Implicito: impostare IV su NULLA e CloudHSM utilizza il valore predefinito di tale algoritmo per eseguire le operazioni di wrapping e annullamento del wrapping (scelta consigliata)
- Esplicito: impostare IV passando il valore IV predefinito alla funzione di wrapping delle chiavi

> **Important**
>
> È necessario capire quale IV è utilizzato nell'applicazione. Per annullare il wrapping delle chiavi, è necessario fornire lo stesso IV utilizzato per eseguire il wrapping delle chiavi. Se si utilizza un IV implicito per eseguire il wrapping, utilizzare un IV implicito per annullare il wrapping. Con un IV implicito, CloudHSM utilizzerà il valore predefinito per annullare il wrapping.

La tabella seguente descrive i valori consentiti per i quali IVs l'algoritmo di wrapping è specificato.

Algoritmo per Wrapping Chiavi AES	IV implicito	IV esplicito
Wrapping Chiavi AES con Zero Padding	Richiesto Valore predefinito: (IV calcolato internamente in base alle specifiche)	Non consentito
Wrapping Chiavi AES AES con Zero Padding	Consentito (scelta consigliata) Valore predefinito: `0xA6A6A6A6A6A6A6A6`	Consentito Unico valore accettato: `0xA6A6A6A6A6A6A6A6`
Wrapping Chiavi AES con Padding PKCS #5	Consentito (scelta consigliata)	Consentito

Algoritmo per Wrapping Chiavi AES	IV implicito	IV esplicito
	Valore predefinito: `0xA6A6A6A6A6A6A6A6`	Unico valore accettato: `0xA6A6A6A6A6A6A6A6`

Utilizzo del key wrap in AES AWS CloudHSM

Le operazioni di wrapping e annullamento del wrapping delle chiavi vengono eseguite come descritto di seguito:

- Nella libreria PCKS #11, seleziona il meccanismo appropriato per le funzioni `C_WrapKey` e `C_UnWrapKey`, come illustrato nella tabella seguente.
- Nel provider JCE, seleziona la combinazione di algoritmo, modalità e riempimento appropriata, implementando metodi di cifratura `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE` come mostrato nella tabella seguente.
- Nella CLI di CloudHSM, scegli l'algoritmo appropriato dall'elenco degli algoritmi e Il comando key wrap nella CLI di CloudHSM degli algoritmi Il comando key unwrap nella CLI di CloudHSM supportati, come mostrato nella tabella seguente.
- In key_mgmt_util (KMU), utilizza i comandi Esporta qualsiasi AWS CloudHSM chiave usando KMU e Scartare una AWS CloudHSM chiave usando KMU con valori m appropriati, come illustrato nella tabella seguente.

Algoritmo Wrapping Chiavi AES	Meccanismo PKCS #11	Metodo Java	Sottocomando CLI CloudHSM	Argomento della Key Management Utility (KMU)
Wrapping Chiavi AES con Zero Padding	• `CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD` (Meccanismo definito dal fornitore)	`AESWrap/ECB/ZeroPadding`	aes-zero-pad	m = 6

Algoritmo Wrapping Chiavi AES	Meccanismo PKCS #11	Metodo Java	Sottocomando CLI CloudHSM	Argomento della Key Management Utility (KMU)
Wrapping Chiavi AES senza Padding	• `CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD` (Meccanismo definito dal fornitore)	`AESWrap/ECB/NoPadding`	aes-no-pad	m = 5
Wrapping Chiavi AES con Padding PKCS #5	• `CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD` (Meccanismo definito dal fornitore)	`AESWrap/ECB/PKCS5Padding`	aes-pkcs5-pad	m = 4

Utilizzo di chiavi affidabili in AWS CloudHSM

AWS CloudHSM supporta il key wrapping affidabile per proteggere le chiavi di dati dalle minacce interne. Questo argomento descrive come creare chiavi attendibili per proteggere i dati.

Argomenti

- Comprendere le chiavi affidabili in AWS CloudHSM
- Attributi chiave affidabili in AWS CloudHSM
- Come utilizzare chiavi attendibili per racchiudere le chiavi dati AWS CloudHSM
- Come scartare una chiave dati con una chiave affidabile per AWS CloudHSM

Comprendere le chiavi affidabili in AWS CloudHSM

Una chiave affidabile è una chiave che viene utilizzata per racchiudere altre chiavi e che gli amministratori e i responsabili della crittografia (COs) identificano specificamente come attendibile

utilizzando l'attributo. CKA_TRUSTED Inoltre, gli amministratori e i responsabili della crittografia (COs) utilizzano CKA_UNWRAP_TEMPLATE gli attributi correlati per specificare quali azioni possono eseguire le chiavi dati una volta aperte da una chiave affidabile. Le chiavi di dati di cui viene annullato il wrapping mediante la chiave attendibile devono contenere a loro volta questi attributi affinché l'operazione di annullamento del wrapping abbia esito positivo, il che contribuisce a garantire che le chiavi di dati di cui viene annullato il wrapping siano ammesse solo per l'uso previsto.

Utilizza l'attributo CKA_WRAP_WITH_TRUSTED per identificare tutte le chiavi di dati di cui desideri eseguire il wrapping con chiavi attendibili. In questo modo è possibile applicare delle restrizioni alle chiavi di dati in modo che le applicazioni possano utilizzare solo chiavi attendibili per annullarne il wrapping. Una volta impostato questo attributo per le chiavi di dati, l'attributo diventa di sola lettura e non è possibile modificarlo. Con l'applicazione di tali attributi, le applicazioni possono annullare il wrapping delle chiavi di dati solo con le chiavi ritenute attendibili, e l'annullamento del wrapping restituisce sempre chiavi di dati con attributi che limitano la modalità di utilizzo di tali chiavi.

Attributi chiave affidabili in AWS CloudHSM

I seguenti attributi consentono di contrassegnare una AWS CloudHSM chiave come attendibile, specificare che una chiave dati può essere impacchettata e aperta solo con una chiave affidabile e controllare cosa può fare una chiave dati dopo essere stata aperta:

- CKA_TRUSTED: applica questo attributo (oltre a CKA_UNWRAP_TEMPLATE) alla chiave che eseguirà il wrapping delle chiavi di dati per specificare che un amministratore o un crypto officer (CO) reputa questa chiave attendibile con la dovuta diligenza. Solo un amministratore o un CO può impostare l'attributo CKA_TRUSTED. Il crypto user (CU) è il proprietario della chiave, ma solo un CO può impostare l'attributo CKA_TRUSTED per tale chiave.
- CKA_WRAP_WITH_TRUSTED: applica questo attributo a una chiave di dati esportabile per specificare che è possibile eseguire il wrapping della chiave solo con chiavi contrassegnate come CKA_TRUSTED. Una volta impostato l'attributo CKA_WRAP_WITH_TRUSTED su true, questo diventa di sola lettura e non è possibile modificarlo o rimuoverlo.
- CKA_UNWRAP_TEMPLATE: applica questo attributo alla chiave di wrapping (oltre a CKA_TRUSTED) per specificare quali nomi e valori degli attributi il servizio deve applicare automaticamente alle chiavi di dati di cui annulla il wrapping. Quando un'applicazione invia una chiave per l'annullamento del wrapping, può fornire anche il proprio modello di annullamento del wrapping. Se si specifica un modello di annullamento del wrapping e l'applicazione fornisce il proprio modello, l'HSM utilizza entrambi i modelli per applicare i nomi e i valori degli attributi alla chiave. Tuttavia, se un valore nel modello CKA_UNWRAP_TEMPLATE per la chiave di wrapping è in conflitto con un attributo fornito

dall'applicazione durante la richiesta di annullamento del wrapping, la richiesta di annullamento del wrapping dà esito negativo.

Per ulteriori informazioni sugli attributi, consulta i seguenti argomenti:

- Attributi chiave PKCS #11
- Attributi chiave JCE
- Attributi chiave della CLI di CloudHSM

Come utilizzare chiavi attendibili per racchiudere le chiavi dati AWS CloudHSM

Per utilizzare una chiave affidabile per racchiudere una chiave dati AWS CloudHSM, devi completare tre passaggi fondamentali:

1. Per la chiave di dati di cui intendi eseguire il wrapping con una chiave attendibile, imposta il relativo attributo `CKA_WRAP_WITH_TRUSTED` su true.
2. Per la chiave attendibile con cui intendi eseguire il wrapping della chiave di dati, imposta il relativo attributo `CKA_TRUSTED` su true.
3. Utilizza la chiave attendibile per eseguire il wrapping della chiave di dati.

Fase 1: imposta l'attributo **`CKA_WRAP_WITH_TRUSTED`** della chiave di dati su true

Per la chiave di dati di cui intendi annullare il wrapping, scegli una delle opzioni seguenti per impostare l'attributo `CKA_WRAP_WITH_TRUSTED` della chiave su true. In questo modo si applicano delle restrizioni alla chiave di dati in modo che le applicazioni possano utilizzare solo chiavi attendibili per eseguirne il wrapping.

Opzione 1: in caso di generazione di una nuova chiave, imposta **`CKA_WRAP_WITH_TRUSTED`** su true

Genera una chiave utilizzando PKCS #11, JCE o la CLI di CloudHSM. Per ulteriori dettagli, vedi gli esempi a seguire.

PKCS #11

Per generare una chiave con PKCS #11, è necessario impostare l'attributo `CKA_WRAP_WITH_TRUSTED` della chiave su true. Come mostrato nell'esempio seguente, esegui

questa operazione includendo questo attributo nel modello `CK_ATTRIBUTE template` della chiave e impostando l'attributo su true:

```
CK_BYTE_PTR label = "test_key";
CK_ATTRIBUTE template[] = {
        {CKA_WRAP_WITH_TRUSTED, &true_val,         sizeof(CK_BBOOL)},
        {CKA_LABEL,             label,             strlen(label)},
        ...
};
```

Per ulteriori informazioni, consulta i nostri esempi pubblici che illustrano la generazione di chiavi con PKCS #11.

JCE

Per generare una chiave con JCE, è necessario impostare l'attributo `WRAP_WITH_TRUSTED` della chiave su true. Come mostrato nell'esempio seguente, esegui questa operazione includendo questo attributo nel modello `KeyAttributesMap` della chiave e impostando l'attributo su true:

```
final String label = "test_key";
final KeyAttributesMap keySpec = new KeyAttributesMap();
keySpec.put(KeyAttribute.WRAP_WITH_TRUSTED, true);
keySpec.put(KeyAttribute.LABEL, label);
...
```

Per ulteriori informazioni, consulta i nostri esempi pubblici che illustrano la generazione di chiavi con JCE.

CloudHSM CLI

Per generare una chiave con la CLI di CloudHSM, è necessario impostare l'attributo `wrap-with-trusted` della chiave su true. Per farlo, includi `wrap-with-trusted=true` nell'argomento appropriato per il comando di generazione della chiave:

- Per le chiavi simmetriche, aggiungi `wrap-with-trusted` all'argomento `attributes`.
- Per le chiavi pubbliche, aggiungi `wrap-with-trusted` all'argomento `public-attributes`.
- Per le chiavi private, aggiungi `wrap-with-trusted` all'argomento `private-attributes`.

Per ulteriori informazioni sulla generazione di una coppia di chiavi, consulta la pagina La categoria di generazione di coppie asimmetriche in CloudHSM CLI.

Per ulteriori informazioni sulla generazione di chiavi simmetriche, consulta la pagina La categoria generate-symmetric nella CLI di CloudhSM.

Opzione 2: se utilizzi una chiave esistente, usa la CLI di CloudHSM per impostare l'attributo **`CKA_WRAP_WITH_TRUSTED`** su true

Per impostare l'attributo `CKA_WRAP_WITH_TRUSTED` di una chiave esistente su true, segui questa procedura:

1. Utilizza il comando Accedi a un HSM utilizzando CloudHSM CLI per accedere come crypto user (CU).
2. Utilizza il comando Imposta gli attributi delle chiavi con CloudhSM CLI per impostare l'attributo `wrap-with-trusted` della chiave su true.

```
aws-cloudhsm > key set-attribute --filter attr.label=test_key --name wrap-with-
trusted --value true
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
  }
}
```

Fase 2: imposta l'attributo **`CKA_TRUSTED`** della chiave attendibile su true

Per rendere una chiave attendibile, l'attributo `CKA_TRUSTED` deve essere impostato su true. Per farlo, è possibile utilizzare la CLI di CloudHSM o CloudHSM Management Utility (CMU).

- Se intendi utilizzare la CLI di CloudHSM per impostare l'attributo `CKA_TRUSTED` di una chiave, consulta la pagina Contrassegna una chiave come affidabile utilizzando la CLI di CloudhSM.
- Se intendi utilizzare CMU per impostare l'attributo `CKA_TRUSTED` di una chiave, consulta la pagina Come contrassegnare una chiave come attendibile con l'utilità AWS CloudHSM di gestione.

Fase 3. Utilizza la chiave attendibile per eseguire il wrapping della chiave di dati

Per eseguire il wrapping della chiave di dati a cui si fa riferimento nella fase 1 con la chiave attendibile impostata nella fase 2, consulta i link seguenti per vedere esempi di codice. Ciascun esempio mostra come eseguire il wrapping delle chiavi.

- AWS CloudHSM Esempi PKCS #11
- AWS CloudHSM Esempi JCE

Come scartare una chiave dati con una chiave affidabile per AWS CloudHSM

Per estrarre una chiave dati AWS CloudHSM, è necessaria una chiave affidabile `CKA_UNWRAP` impostata su true. Per essere attendibile, la chiave deve soddisfare inoltre i seguenti criteri:

- L'attributo `CKA_TRUSTED` della chiave deve essere impostato su true.
- La chiave deve utilizzare `CKA_UNWRAP_TEMPLATE` e gli attributi correlati per specificare le azioni che le chiavi di dati possono effettuare una volta annullato il wrapping. Se, ad esempio, desideri che una chiave di cui è stato annullato il wrapping sia non esportabile, imposta `CKA_EXPORTABLE = FALSE` nell'ambito del modello `CKA_UNWRAP_TEMPLATE`.

> **Note**
>
> `CKA_UNWRAP_TEMPLATE` è disponibile solo con PKCS #11.

Quando un'applicazione invia una chiave per l'annullamento del wrapping, può fornire anche il proprio modello di annullamento del wrapping. Se si specifica un modello di annullamento del wrapping e l'applicazione fornisce il proprio modello, l'HSM utilizza entrambi i modelli per applicare i nomi e i valori degli attributi alla chiave. Tuttavia, se durante una richiesta di annullamento del wrapping un valore nel modello `CKA_UNWRAP_TEMPLATE` della chiave attendibile è in conflitto con un attributo fornito dall'applicazione, la richiesta di annullamento del wrapping dà esito negativo.

Per vedere un esempio su come annullare il wrapping di una chiave di dati con una chiave attendibile, consulta questo esempio con PKCS #11.

Gestione delle chiavi con CloudHSM CLI

Se utilizzi la serie di versioni SDK più recente, utilizza la CLI di CloudHSM per gestire le chiavi nel cluster. AWS CloudHSM Per ulteriori informazioni, consulta gli argomenti riportati di seguito.

- L'uso di chiavi affidabili descrive come utilizzare la CLI di CloudhSM per creare chiavi affidabili per proteggere i dati.

- La pagina sulla generazione delle chiavi include istruzioni sulla creazione delle chiavi, tra cui chiavi simmetriche, chiavi RSA e chiavi EC.
- La pagina sull'eliminazione delle chiavi descrive in che modo i proprietari possono eliminare le chiavi.
- La pagina su condivisione e annullamento della condivisione delle chiavi illustra in che modo i proprietari possono condividere e annullare la condivisione delle chiavi.
- La pagina sul filtraggio delle chiavi offre delle linee guida su come utilizzare i filtri per trovare le chiavi.
- Manage key quorum authentication (M of N) offre linee guida su come configurare e utilizzare l'autenticazione quorum con le chiavi.

Genera chiavi con CloudHSM CLI

Prima di poter generare una chiave, è necessario avviare la CLI di CloudHSM e accedere come crypto user (CU). Per creare chiavi nell'HSM, utilizza il comando corrispondente al tipo di chiave da creare.

Argomenti

- Genera chiavi simmetriche con CloudHSM CLI
- Genera chiavi asimmetriche utilizzando CloudHSM CLI
- AWS CloudHSM argomenti correlati chiave

Genera chiavi simmetriche con CloudHSM CLI

Usa i comandi elencati in per generare chiavi simmetriche La categoria generate-symmetric nella CLI di CloudhSM per. AWS CloudHSM Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-symmetric.

Generazione di una chiave AES

Utilizza il comando key generate-symmetric aes per generare chiavi AES. Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-symmetric aes.

Example

L'esempio seguente genera una chiave AES a 32 byte.

```
aws-cloudhsm > key generate-symmetric aes \
    --label aes-example \
    --key-length-bytes 32
```

Arguments (Argomenti)

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave AES.

Obbligatorio: sì

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- 16, 24 e 32

Obbligatorio: sì

<KEY_ATTRIBUTES>

Specifica un elenco, separato da spazi, degli attributi delle chiavi da impostare per la chiave AES generata nel formato `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`)

Per un elenco degli attributi AWS CloudHSM chiave supportati, vedere. Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione. Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per convertire una chiave di sessione in una chiave (token) persistente, usa il comando key set-attribute.

Per impostazione predefinita, quando le chiavi vengono generate, sono persistent/token chiavi. Usando <SESSION>changes this, assicurandoti che una chiave generata con questo argomento sia session/ephemeral

Obbligatorio: no

Generazione di una chiave segreta generica

Usa il comando key generate-symmetric generic-secret per generare chiavi segrete generiche. Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-symmetric generic-secret.

Example

L'esempio seguente genera una chiave segreta generica a 32 byte.

```
aws-cloudhsm > key generate-symmetric generic-secret \
    --label generic-secret-example \
    --key-length-bytes 32
```

Arguments (Argomenti)

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave segreta generica.

Obbligatorio: sì

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- Da 1 a 800

Obbligatorio: sì

<KEY_ATTRIBUTES>

Specifica un elenco, separato da spazi, degli attributi delle chiavi da impostare per la chiave segreta generica generata nel formato `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`)

Per un elenco degli attributi AWS CloudHSM chiave supportati, vedereAttributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione. Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per convertire una chiave di sessione in una chiave (token) persistente, usa il comando key set-attribute.

Per impostazione predefinita, quando le chiavi vengono generate, sono persistent/token chiavi. Usando <SESSION>changes this, assicurandoti che una chiave generata con questo argomento sia session/ephemeral

Obbligatorio: no

Genera chiavi asimmetriche utilizzando CloudHSM CLI

Usa i comandi elencati in La categoria di generazione di coppie asimmetriche in CloudHSM CLI per generare coppie di chiavi asimmetriche per i cluster. AWS CloudHSM

Generazione di una chiave RSA

Utilizza il comando key generate-asymmetric-pair rsa per generare una coppia di chiavi RSA. Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-asymmetric-pair rsa.

Example

Nell'esempio di seguito viene creata una coppia di chiavi RSA a 2048 bit.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
    --public-exponent 65537 \
    --modulus-size-bits 2048 \
    --public-label rsa-public-example \
    --private-label rsa-private-example
```

Arguments (Argomenti)

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica.

Obbligatorio: sì

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata.

Obbligatorio: sì

<MODULUS_SIZE_BITS>

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Obbligatorio: sì

<PUBLIC_EXPONENT>

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Obbligatorio: sì

<PUBLIC_KEY_ATTRIBUTES>

Specifica un elenco, separato da spazi, degli attributi delle chiavi da impostare per la chiave pubblica RSA generata nel formato `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`).

Per un elenco degli attributi AWS CloudHSM chiave supportati, vedere. Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione. Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per convertire una chiave di sessione in una chiave (token) persistente, usa il comando key set-attribute.

Per impostazione predefinita, quando le chiavi vengono generate, sono persistent/token chiavi. Usando <SESSION>changes this, assicurandoti che una chiave generata con questo argomento sia session/ephemeral

Obbligatorio: no

Generazione di coppie di chiavi curve ellittiche (EC, Elliptic Curve)

Utilizza il comando key generate-asymmetric-pair ec per generare una coppia di chiavi EC. Per visualizzare tutte le opzioni disponibili, compreso un elenco delle curve ellittiche supportate, utilizza il comando help key generate-asymmetric-pair ec.

Example

L'esempio seguente genera una coppia di chiavi EC utilizzando la curva ellittica Secp384r1.

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve secp384r1 \
    --public-label ec-public-example \
    --private-label ec-private-example
```

Arguments (Argomenti)

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti ha un limite di 126 caratteri.

Obbligatorio: sì

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti ha un limite di 126 caratteri.

Obbligatorio: sì

<CURVE>

Specifica l'identificatore per la curva ellittica.

Valori validi:

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1
- ed25519 (supportato solo su istanze hsm2m.medium in modalità non FIPS)

Obbligatorio: sì

<PUBLIC_KEY_ATTRIBUTES>

Specifica un elenco, separato da spazi, degli attributi delle chiavi da impostare per la chiave pubblica EC generata nel formato `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `verify=true`).

Per un elenco degli attributi chiave supportati, consulta. AWS CloudHSM Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<PRIVATE_KEY_ATTRIBUTES>

Specifica un elenco, separato da spazi, degli attributi delle chiavi da impostare per la chiave privata EC generata nel formato `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`).

Per un elenco degli attributi AWS CloudHSM chiave supportati, vedereAttributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione. Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica

rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per convertire una chiave di sessione in una chiave (token) persistente, usa il comando key set-attribute.

Per impostazione predefinita, le chiavi vengono generate come chiavi (token) persistenti. L'inserimento dell'argomento <SESSIONE> modifica la situazione, assicurando che una chiave generata con questo argomento sia una chiave di sessione (effimera).

Obbligatorio: no

Genera coppie di ML-DSA chiavi

Usa il key generate-asymmetric-pair ml-dsa comando per generare una coppia di ML-DSA chiavi. Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-asymmetric-pair ml-dsa.

Example

L'esempio seguente genera una coppia di ML-DSA chiavi utilizzando l' ML-DSA-44 algoritmo.

```
aws-cloudhsm > key generate-asymmetric-pair ml-dsa \
    --mldsa-algorithm ML-DSA-44 \
    --public-label ml-dsa-public-example \
    --private-label ml-dsa-private-example
```

Arguments (Argomenti)

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica.

Obbligatorio: sì

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata.

Obbligatorio: sì

<MLDSA_ALGORITHM>

Specifica l' ML-DSA algoritmo.

Valori validi:

- ML-DSA-44
- ML-DSA-65
- ML-DSA-87

Obbligatorio: sì

AWS CloudHSM argomenti correlati chiave

Per ulteriori informazioni sulle immissioni, consulta le sezioni seguenti AWS CloudHSM.

- Attributi chiave per la CLI di CloudHSM
- La categoria di generazione di coppie asimmetriche in CloudHSM CLI
- La categoria generate-symmetric nella CLI di CloudhSM

Eliminare le chiavi utilizzando la CLI di CloudHSM

Segui l'esempio fornito in questo argomento per eliminare una chiave con la CLI di CloudHSM. Soltanto i proprietari possono eliminare le chiavi.

Argomenti

- Esempio: eliminare una chiave
- Argomenti correlati

Esempio: eliminare una chiave

1. Esegui il comando key list per identificare la chiave che desideri eliminare:

```
aws-cloudhsm > key list --filter attr.label="my_key_to_delete" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000540011",
        "key-info": {
          "key-owners": [
```

```
            {
              "username": "my_crypto_user",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "my_key_to_delete",
          "id": "",
          "check-value": "0x29bbd1",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1217,
          "public-exponent": "0x010001",
          "modulus":
"0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990
          "modulus-size-bits": 2048
        }
      }
    ],
```

```
    "total_key_count": 1,
    "returned_key_count": 1
  }
```

2. Dopo aver identificato la chiave, esegui il comando key delete con l'attributo `label` univoco della chiave per eliminare la chiave:

```
aws-cloudhsm > key delete --filter attr.label="my_key_to_delete"
{
  "error_code": 0,
  "data": {
    "message": "Key deleted successfully"
  }
}
```

3. Esegui il comando key list con l'attributo `label` univoco della chiave e verifica che la chiave sia stata eliminata. Come si può vedere nell'esempio seguente, nel cluster HSM non è presente nessuna chiave con l'etichetta `my_key_to_delete`:

```
aws-cloudhsm > key list --filter attr.label="my_key_to_delete"
{
  "error_code": 0,
  "data": {
    "matched_keys": [],
    "total_key_count": 0,
    "returned_key_count": 0
  }
}
```

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Eliminare una chiave con CloudHSM CLI

Condividi e annulla la condivisione delle chiavi utilizzando la CLI di CloudhSM

Utilizza i comandi indicati in questo argomento per condividere e annullare la condivisione delle chiavi nella CLI di CloudHSM. Nel AWS CloudHSM, l'utente crittografico (CU) che crea la chiave ne è il

proprietario. Il proprietario può utilizzare i key unshare comandi key share and per condividere e annullare la condivisione della chiave con altri. CUs Gli utenti con cui è condivisa la chiave possono utilizzare la chiave nelle operazioni crittografiche, ma non possono eliminare, esportare, condividere, annullare la condivisione, derivare o racchiudere la chiave.

Prima di poter condividere una chiave, è necessario accedere all'HSM come crypto user (CU) proprietario della chiave.

Argomenti

- Esempio: condividere e annullare la condivisione di una chiave
- Argomenti correlati

Esempio: condividere e annullare la condivisione di una chiave

Example

L'esempio seguente illustra come condividere e annullare la condivisione di una chiave con un crypto user (CU) `alice`. Oltre ai comandi key share e key unshare, i comandi di condivisione e annullamento della condivisione richiedono inoltre una chiave specifica che utilizzi i filtri chiave della CLI di CloudHSM e il nome utente specifico dell'utente con cui la chiave verrà condivisa o la cui condivisione verrà annullata.

1. Inizia eseguendo il comando key list con un filtro per restituire una chiave specifica e vedere con chi la chiave è già condivisa.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
```

```
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
            },
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
```

```
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

2. Visualizza l'output `shared-users` per individuare con chi la chiave è condivisa attualmente.
3. Per condividere questa chiave con il crypto user (CU) `alice`, inserisci il seguente comando:

```
aws-cloudhsm > key share --filter attr.label="rsa_key_to_share" attr.class=private-
key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key shared successfully"
  }
}
```

Si noti che, oltre al comando key share, questo comando utilizza l'etichetta univoca della chiave e il nome dell'utente con cui la chiave verrà condivisa.

4. Esegui il comando key list per verificare che la chiave sia stata condivisa con `alice`:

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{
```

```
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
            },
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
```

```
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```


5. Per annullare la condivisione della stessa chiave con `alice`, esegui il seguente comando unshare:

```
aws-cloudhsm > key unshare --filter attr.label="rsa_key_to_share"
 attr.class=private-key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key unshared successfully"
  }
}
```

Si noti che, oltre al comando key unshare, questo comando utilizza l'etichetta univoca della chiave e il nome dell'utente con cui la chiave verrà condivisa.

6. Esegui nuovamente il comando key list e verifica che la condivisione della chiave con il crypto user `alice` sia stata annullata:

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
```

```
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
            },
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
```

```
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```


Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Condividi una chiave utilizzando la CLI di CloudHSM
- Annullare la condivisione di una chiave utilizzando la CLI di CloudhSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Filtrare le chiavi utilizzando la CLI di CloudHSM

Utilizza i seguenti comandi delle chiavi per utilizzare i meccanismi di filtraggio delle chiavi standardizzati per la CLI di CloudHSM.

- key list
- key delete
- key share
- key unshare
- key set-attribute

Per selezionare le chiavi di and/or filtro con CloudHSM CLI, i comandi da tastiera utilizzano un meccanismo di filtraggio standardizzato basato su. Attributi chiave per la CLI di CloudHSM È possibile specificare una chiave o un set di chiavi nei comandi da tastiera utilizzando uno o più AWS CloudHSM attributi in grado di identificare una o più chiavi. Il meccanismo di filtraggio delle chiavi funziona solo sulle chiavi che l'utente attualmente connesso possiede e condivide, nonché su tutte le chiavi pubbliche del AWS CloudHSM cluster.

Argomenti

- Requisiti
- Filtrare per trovare una singola chiave
- Errori di filtraggio
- Argomenti correlati

Requisiti

Per filtrare le chiavi, devi aver effettuato l'accesso come utente crittografico (). CUs

Filtrare per trovare una singola chiave

Si noti che, negli esempi seguenti, ogni attributo utilizzato come filtro deve essere scritto nel formato `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE`. Ad esempio, se si desidera filtrare per attributo dell'etichetta, è necessario scrivere `attr.label=my_label`.

Example Utilizzare un singolo attributo per trovare una singola chiave

Questo esempio illustra come filtrare per individuare una singola chiave univoca utilizzando solamente un singolo attributo identificativo.

```
aws-cloudhsm > key list --filter attr.label="my_unique_key_label" --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ],
```

```
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "my_unique_key_label",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254c8f5
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Example Utilizzare vari attributi per trovare una singola chiave

L'esempio seguente illustra come trovare una singola chiave utilizzando vari attributi delle chiavi.

```
aws-cloudhsm > key list --filter attr.key-type=rsa attr.class=private-key attr.check-
value=0x29bbd1 --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000540011",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            }
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "my_crypto_user",
          "id": "",
          "check-value": "0x29bbd1",
          "class": "my_test_key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
```

```
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1217,
          "public-exponent": "0x010001",
          "modulus":
 "0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990c2a7
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Example Filtrare per trovare un set di chiavi

L'esempio seguente illustra come filtrare per trovare un set di chiavi rsa private.

```
aws-cloudhsm > key list --filter attr.key-type=rsa attr.class=private-key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "my_crypto_user",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
```

```
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu1",
            "key-coverage": "full"
          },
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa_key_to_share",
        "id": "",
        "check-value": "0xae8ff0",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": true,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1219,
        "public-exponent": "0x010001",
        "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254c8f5
        "modulus-size-bits": 2048
      }
```

```
    },
    {
      "key-reference": "0x0000000000540011",
      "key-info": {
        "key-owners": [
          {
            "username": "my_crypto_user",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          }
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "my_test_key",
        "id": "",
        "check-value": "0x29bbd1",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": true,
        "verify": false,
```

```
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1217,
          "public-exponent": "0x010001",
          "modulus":
 "0x8b3a7c20618e8be08220ed8ab2c8550b65fc1aad8d4cf04fbf2be685f97eeb78fcbbad9b02cd91a3b15e990c2a7
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 2,
    "returned_key_count": 2
  }
}
```

Errori di filtraggio

Alcune operazioni con le chiavi possono essere eseguite solo su una chiave alla volta. Per queste operazioni, la CLI di CloudHSM genererà un errore se i criteri di filtraggio non sono sufficientemente precisi e si trovano varie chiavi corrispondenti ai criteri. Di seguito viene illustrato un esempio di questo tipo con l'eliminazione della chiave.

Example Errore di filtraggio quando ci sono troppe chiavi corrispondenti

```
aws-cloudhsm > key delete --filter attr.key-type=rsa
{
  "error_code": 1,
  "data": "Key selection criteria matched 48 keys. Refine selection criteria to select
 a single key."
}
```

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM

Contrassegna una chiave come affidabile utilizzando la CLI di CloudhSM

Il contenuto di questa sezione fornisce istruzioni sull'utilizzo della CLI di CloudHSM per contrassegnare una chiave come attendibile.

1. Utilizzando il comando login della CLI di CloudHSM, accedi come crypto user (CU).

2. Utilizza il comando key list per individuare il riferimento alla chiave della chiave che desideri contrassegnare come attendibile. Il seguente esempio elenca la chiave con l'etichetta `test_aes_trusted`.

```
aws-cloudhsm > key list --filter attr.label=test_aes_trusted
        {
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000200333",
        "attributes": {
          "label": "test_aes_trusted"
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Utilizzando il comando Esci da un HSM utilizzando CloudHSM CLI, scollegati come crypto user (CU).

4. Utilizzando il comando Accedi a un HSM utilizzando CloudHSM CLI, accedi come amministratore.

5. Utilizzando il comando key set-attribute con il riferimento alla chiave individuato nella fase 2, imposta il valore "attendibile" della chiave su true:

```
aws-cloudhsm > key set-attribute --filter key-reference=<Key Reference> --name
 trusted --value true
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
  }
}
```

Gestisci l'autenticazione del quorum (controllo degli accessi M of N) utilizzando la CLI CloudhSM

I moduli di sicurezza hardware (HSM) del AWS CloudHSM cluster supportano l'autenticazione quorum, nota anche come controllo degli accessi M of N. Con l'autenticazione quorum, nessun singolo utente dell'HSM può eseguire operazioni controllate dal quorum. Invece, un numero minimo di utenti HSM (almeno 2) deve cooperare per eseguire queste operazioni. L'autenticazione quorum aggiunge un ulteriore livello di protezione richiedendo l'approvazione di più utenti HSM.

L'autenticazione del quorum consente di controllare le seguenti operazioni:

- Utilizzo e gestione delle chiavi HSM da parte di un utente crittografico: creazione di firme con una chiave o impacchettamento, apertura, condivisione, annullamento della condivisione e impostazione di un attributo di una chiave.

Considerazioni importanti

- Un utente HSM può firmare il proprio token del quorum, ovvero, l'utente richiedente può fornire una delle approvazioni richieste per l'autenticazione del quorum.
- È possibile scegliere il numero minimo di approvatori del quorum per le operazioni controllate dal quorum. Il numero minore che si può scegliere è due (2) e il numero maggiore è otto (8).
- L'HSM può memorizzare fino a 1.024 token di quorum. Se l'HSM ha già 1.024 token quando si tenta di crearne uno nuovo, l'HSM elimina uno dei token scaduti. Per impostazione predefinita, i token scadono dieci minuti dopo la loro creazione.
- Se l'autenticazione a più fattori (MFA) è abilitata, il cluster utilizza la stessa chiave per l'autenticazione quorum e per l'MFA. Per ulteriori informazioni sull'utilizzo dell'autenticazione quorum e dell'MFA, consulta Using CloudhSM CLI per gestire l'MFA.
- Può essere attiva una sola operazione di quorum alla volta per servizio. È necessario completare o eliminare il token del quorum attivo per un servizio prima di poterne generare uno nuovo per lo stesso servizio. Per ulteriori informazioni, consulta Servizi e tipi supportati.

I seguenti argomenti forniscono ulteriori informazioni sull'autenticazione del quorum in AWS CloudHSM.

Argomenti

- Processo di autenticazione quorum per CloudHSM CLI

- Nomi e tipi AWS CloudHSM di servizi supportati per l'autenticazione del quorum con CloudHSM CLI
- Configura l'autenticazione del quorum per AWS CloudHSM gli utenti crittografici utilizzando la CLI di CloudHSM
- Gestione e utilizzo delle chiavi con autenticazione quorum abilitata per l'utilizzo della CLI di AWS CloudHSM CloudHSM

Processo di autenticazione quorum per CloudHSM CLI

I passaggi seguenti riassumono i processi di autenticazione del quorum per CloudHSM CLI. Per le operazioni e gli strumenti specifici, consultare Gestione e utilizzo delle chiavi con autenticazione quorum abilitata per l'utilizzo della CLI di AWS CloudHSM CloudHSM.

1. Ogni utente del modulo di sicurezza hardware (HSM) crea una chiave asimmetrica per la firma. Gli utenti completano questa operazione al di fuori dell'HSM, assicurandosi di proteggere la chiave in modo appropriato.
2. Ciascun utente HSM effettua l'accesso all'HSM e registra la parte pubblica della propria chiave di firma (la chiave pubblica) nell'HSM.
3. Quando un utente HSM desidera effettuare un'operazione controllata dal quorum, tale utente effettua l'accesso all'HSM e ottiene un token del quorum.
4. L'utente HSM assegna il token del quorum a uno o più utenti HSM e richiede la loro approvazione.
5. Gli altri utenti HSM approvano utilizzando le loro chiavi per firmare crittograficamente il token del quorum. Ciò si verifica al di fuori dell'HSM.
6. Quando l'utente HSM dispone del numero di approvazioni necessario, tale utente effettua l'accesso all'HSM ed esegue l'operazione controllata dal quorum con l'argomento --approval, fornendo il file del token del quorum firmato contenente tutte le approvazioni (firme) necessarie.
7. L'HSM utilizza la chiavi pubbliche registrate di ciascun firmatario per verificare le firme. Se le firme sono valide, l'HSM approva il token e l'operazione controllata dal quorum viene eseguita.

Nomi e tipi AWS CloudHSM di servizi supportati per l'autenticazione del quorum con CloudHSM CLI

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admin come la creazione e l'eliminazione di utenti, la modifica delle password degli utenti, l'impostazione dei valori del quorum e la disattivazione delle funzionalità quorum e MFA.

Crypto User Services: l'autenticazione Quorum viene utilizzata per i servizi privilegiati degli utenti crittografici associati a una chiave specifica, come la firma con una chiave, una chiave, sharing/unsharing una chiave e l'impostazione dell'attributo di wrapping/unwrapping una chiave. Il valore quorum di una chiave associata viene configurato quando la chiave viene generata, importata o aperta. Il valore del quorum deve essere uguale o inferiore al numero di utenti a cui è associata la chiave, che include gli utenti con cui la chiave è condivisa e il proprietario della chiave.

Ogni tipo di servizio è ulteriormente suddiviso in un nome di servizio qualificante, che contiene un set specifico di operazioni di servizio supportate dal quorum che possono essere eseguite.

Nome del servizio	Tipo di servizio	Operazioni di servizio
Utente	Admin	• user create • user delete • user change-password • user change-mfa
quorum	Admin	• segno del token del quorum set-quorum-value
gruppo [1]	Admin	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gestione delle chiavi	Utente Crypto	• involucro per chiavi • scartare le chiavi • Condivisione chiave • Annulla condivisione chiave • key set-attribute
utilizzo delle chiavi	Utente Crypto	• segno chiave

[1] Il servizio cluster è disponibile esclusivamente su hsm2m.medium

Configura l'autenticazione del quorum per AWS CloudHSM gli utenti crittografici utilizzando la CLI di CloudHSM

Questi argomenti descrivono come configurare CloudHSM per l'autenticazione quorum da parte degli utenti crittografici. Esegui questi passaggi una volta durante la configurazione iniziale. Per la gestione e l'utilizzo delle chiavi successivi, fare riferimento aGestione e utilizzo delle chiavi con autenticazione quorum abilitata per l'utilizzo della CLI di AWS CloudHSM CloudHSM.

Argomenti

- Prerequisiti
- Passaggio 1. Creazione e registrazione di una chiave per la firma
- Passaggio 2. Imposta i valori del quorum chiave durante la generazione delle chiavi

Prerequisiti

- Familiarità con la CLI CloudHSM

Passaggio 1. Creazione e registrazione di una chiave per la firma

Per utilizzare l'autenticazione quorum, ogni cripto-utente deve completare tutti i seguenti passaggi:

Argomenti

- Creazione di una coppia di chiavi RSA
- Crea un token di registrazione
- Firma il token di registrazione non firmato
- Registrazione della chiave pubblica con HSM

Creazione di una coppia di chiavi RSA

Esistono molti modi diversi per creare e proteggere una coppia di chiavi. Gli esempi a seguire mostrano come eseguire questa operazione con OpenSSL.

Example- Creazione di una chiave privata con OpenSSL

L'esempio seguente dimostra come utilizzare OpenSSL per creare una chiave RSA a 2048 bit. Per utilizzare questo esempio, sostituiscilo *<crypto_user1.key>* con il nome del file in cui desideri memorizzare la chiave.

```
$ openssl genrsa -out <crypto_user1.key>
Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
```

Successivamente, genera la chiave pubblica utilizzando la chiave privata appena creata.

Example- Creazione di una chiave pubblica con OpenSSL

L'esempio seguente dimostra come utilizzare OpenSSL per creare una chiave pubblica dalla chiave privata appena creata.

```
$ openssl rsa -in crypto_user1.key -outform PEM -pubout -out crypto_user1.pub
writing RSA key
```

Crea un token di registrazione

Crea un token e firmalo con la chiave privata appena generata nella fase precedente.

Crea un token di registrazione

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Crea un token di registrazione eseguendo il comando quorum token-sign generate:

   ```
   aws-cloudhsm > quorum token-sign generate --service registration --token /path/
   tokenfile
   {
     "error_code": 0,
     "data": {
       "path": "/path/tokenfile"
     }
   ```

```
}
```

3. Il comando quorum token-sign generate genera un token di registrazione nel percorso del file specificato. Ispeziona il file del token:

```
$ cat /path/tokenfile
{
  "version": "2.0",
  "tokens": [
    {
      "approval_data": <approval data in base64 encoding>,
      "unsigned": <unsigned token in base64 encoding>,
      "signed": ""
    }
  ]
}
```

Il file del token comprende:

- approval_data: un token di dati randomizzato con codifica base64 i cui dati non elaborati non superano il limite massimo di 245 byte.
- unsigned: un token codificato e con SHA256 hash in base64 di approval_data.
- signed: un token firmato con codifica base64 (firma) del token non firmato, che utilizza la chiave privata RSA a 2048 bit generata precedentemente con OpenSSL.

Firma il token non firmato con la chiave privata per dimostrare di avere accesso alla chiave privata. Avrai bisogno del file del token di registrazione completamente compilato con una firma e la chiave pubblica per registrare il cripto-utente come utente quorum nel cluster. AWS CloudHSM

Firma il token di registrazione non firmato

1. Decodifica il token non firmato con codifica base64 e inseriscilo in un file binario:

```
$ echo -n '6BMUj6mUjjko6ZLCEdzGlWpR5sILhFJfqhW1ej3Oq1g=' | base64 -d >
 crypto_user.bin
```

2. Utilizza OpenSSL e la chiave privata per firmare il token di registrazione non firmato ora binario e crea un file di firma binario:

```
$ openssl pkeyutl -sign \
-inkey crypto_user1.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in crypto_user.bin \
-out crypto_user.sig.bin
```

3. Codifica la firma binaria in base64:

```
$ base64 -w0 crypto_user.sig.bin > crypto_user.sig.b64
```

4. Copia e incolla la firma con codifica base64 nel file token:

```
{
  "version": "2.0",
  "tokens": [
    {
      "approval_data": <approval data in base64 encoding>,
      "unsigned": <unsigned token in base64 encoding>,
      "signed": <signed token in base64 encoding>
    }
  ]
}
```

Registrazione della chiave pubblica con HSM

Dopo aver creato una chiave, l'utente crittografico deve registrare la chiave pubblica nel cluster. AWS CloudHSM

1. Avvia CloudhSM CLI:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Accedi come utente crittografico di cui desideri registrare la chiave pubblica.

```
aws-cloudhsm > login --username crypto_user1 --role crypto-user
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "crypto_user1",
    "role": "crypto-user"
  }
}
```

3. Registra la chiave pubblica con. Registra la strategia del quorum di firma dei token di un utente utilizzando la CLI di CloudHSM Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help user change-quorum token-sign register.

   Example— Registra una chiave pubblica con AWS CloudHSM cluster

   L'esempio seguente mostra come utilizzare il user change-quorum token-sign register comando nella CLI di CloudHSM per registrare una chiave pubblica di un utente crittografico con l'HSM. Per utilizzare questo comando, l'utente crittografico deve accedere all'HSM. Sostituire questi valori con i propri valori:

   ```
   aws-cloudhsm > user change-quorum token-sign register --public-key </path/
   crypto_user.pub> --signed-token </path/tokenfile>
   {
     "error_code": 0,
     "data": {
       "username": "crypto_user1",
       "role": "crypto-user"
     }
   }
   ```

   > **Note**
   >
   > /path/crypto_user.pub: il percorso del file PEM a chiave pubblica
   > Obbligatorio: sì
   > /path/token_file: il percorso del file con token firmato dalla chiave privata dell'utente
   > Obbligatorio: sì

4. Dopo che tutti i cripto-utenti hanno registrato le proprie chiavi pubbliche, l'output del user list comando lo mostra nel campo quorum, indicando la strategia di quorum abilitata in uso.

   In questo esempio, il AWS CloudHSM cluster ne ha due HSMs, ciascuno con gli stessi cripto-utenti, come mostrato nel seguente output del comando. user list Per ulteriori informazioni sulla creazione degli utenti, vedere Gestione degli utenti con CloudHSM CLI.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "crypto_user1",
        "role": "crypto-user",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "crypto_user2",
        "role": "crypto-user",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
```

```
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "crypto_user3",
        "role": "crypto-user",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Passaggio 2. Imposta i valori del quorum chiave durante la generazione delle chiavi

Per utilizzare l'autenticazione quorum, un cripto-utente deve accedere all'HSM e quindi impostare i valori del quorum chiave associati. Questo è il numero minimo di approvazioni da parte degli utenti crittografici necessarie per eseguire le operazioni con le chiavi HSM. management/usage Per ulteriori informazioni sui comandi da tastiera associati alla gestione o all'utilizzo delle chiavi, vedere. Servizi e tipi supportati

Genera una coppia di chiavi con i valori del quorum chiave impostati

1. Utilizza il seguente comando per avviare la CLI di CloudHSM:

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

2. Utilizzando la CLI di CloudhSM, accedi come utente crittografico.

```
aws-cloudhsm > login --username crypto_user1 --role crypto-user
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "crypto_user1",
    "role": "crypto-user"
  }
}
```

Questo esempio genera una coppia di chiavi RSA con valori di quorum chiave di due (2) set sia per le operazioni di gestione delle chiavi che per quelle di utilizzo delle chiavi. È possibile scegliere qualsiasi valore da zero (0) a otto (8), fino al numero totale di utenti crittografici sull'HSM. In questo esempio, l'HSM ha tre (3) utenti crittografici, quindi il valore massimo possibile è tre (3). Nota che in questo esempio condividiamo la chiave *<crypto_user2>* durante la generazione della chiave. Si noti inoltre che le chiavi pubbliche non hanno valori di quorum.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes verify=true \
--private-attributes sign=true
--share-crypto-users crypto_user2 \
--manage-private-key-quorum-value 2 \
--use-private-key-quorum-value 2
{
  "error_code": 0,
```

```
  "data": {
    "public_key": {
      "key-reference": "0x0000000000640006",
      "key-info": {
        "key-owners": [
          {
            "username": "crypto_user",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "0x",
        "check-value": "0x218f50",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
```

```
        "modulus":
"0xbdf471a3d2a869492f51c767bece8780730ae6479a9a75efffe7cea3594fb28ca518630e7b1d988b45d2fedc830
        "modulus-size-bits": 2048
      }
    },
    "private_key": {
      "key-reference": "0x0000000000640007",
      "key-info": {
        "key-owners": [
          {
            "username": "crypto_user",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "crypto_user2",
            "key-coverage": "full"
          }
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 2,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "0x",
        "check-value": "0x218f50",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
```

```
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1216,
        "public-exponent": "0x010001",
        "modulus":
 "0xbdf471a3d2a869492f51c767bece8780730ae6479a9a75efffe7cea3594fb28ca518630e7b1d988b45d2fedc830
        "modulus-size-bits": 2048
      }
    }
  }
}
```

Quando si genera una chiave con controlli del quorum, la chiave deve essere associata a un numero minimo di utenti pari al valore del quorum della chiave massimo. Gli utenti associati includono il proprietario della chiave e gli utenti Crypto con cui viene condivisa la chiave. Per determinare il numero minimo di utenti con cui condividere la chiave, ottieni il valore di quorum più alto tra il valore del quorum di utilizzo della chiave e il valore del quorum di gestione delle chiavi e sottrai 1 per tenere conto del proprietario della chiave, che per impostazione predefinita è associato alla chiave. Per condividere la chiave con più utenti, usa il comando. Condividi una chiave utilizzando la CLI di CloudHSM

La mancata condivisione della chiave con un numero sufficiente di utenti al momento della generazione della chiave comporterà un errore, come illustrato di seguito.

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes verify=true \
--private-attributes sign=true
--share-crypto-users crypto_user2 crypto_user3 \
--manage-private-key-quorum-value 3 \
--use-private-key-quorum-value 4
{
  "error_code": 1,
  "data": "Invalid quorum value provided."
```

```
}
```

Gestione e utilizzo delle chiavi con autenticazione quorum abilitata per l'utilizzo della CLI di AWS CloudHSM CloudHSM

Dopo aver configurato l'autenticazione quorum per il AWS CloudHSM cluster, gli utenti crittografici non possono eseguire autonomamente la gestione o l'utilizzo delle chiavi HSM se alla chiave sono associati valori di quorum. Questo argomento spiega come un utente crittografico può ottenere un token temporaneo per eseguire un'operazione di gestione o utilizzo delle chiavi HSM.

> **Note**
>
> Ogni token del quorum è valido per una sola operazione. Quando l'operazione ha esito positivo, il token non è più valido e l'utente crittografico deve ottenere un nuovo token. Un token quorum è valido solo durante la sessione di accesso corrente. Se esci dalla CLI di CloudHSM o se la rete si disconnette, il token non è più valido e devi procurartene uno nuovo. È possibile utilizzare solo un token CloudHSM all'interno della CLI CloudHSM. Non puoi usarlo per autenticarti in un'altra applicazione.

L'esempio seguente mostra l'output quando un utente crittografico tenta di creare una firma con una chiave associata al quorum sull'HSM dopo la configurazione dell'autenticazione quorum. Il comando fallisce con un errore, il che significa che l'autenticazione del quorum non è `Quorum Failed` riuscita:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --
hash-function sha256 --data YWJjMTIz
{
  "error_code": 1,
  "data": "Quorum Failed"
}
```

Un utente crittografico deve completare le seguenti attività per ottenere un token temporaneo per eseguire un'operazione di gestione o utilizzo delle chiavi sull'HSM:

Fasi

- Passaggio 1. Ottenere un token del quorum
- Passaggio 2. Ottieni firme dagli utenti che approvano le criptovalute
- Fase 3. Approvare il token sul CloudHSM; raggruppare ed eseguire un'operazione

Passaggio 1. Ottenere un token del quorum

1. Avvia CloudhSM CLI.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Accedi al cluster come utente crittografico.

   ```
   aws-cloudhsm > login --username <crypto_user1> --role crypto-user --
   password password123
   ```

   Questo esempio accede `crypto_user1` alla CLI di CloudHSM con il ruolo. `crypto-user` Sostituisci questi valori con i tuoi.

   ```
   {
     "error_code": 0,
     "data": {
       "username": "crypto_user1",
       "role": "crypto-user"
     }
   }
   ```

3. Genera un token di quorum utilizzando il quorum token-sign generate comando.

   Nel comando seguente, `key-usage` identifica il nome del servizio in cui utilizzerai il token che stai generando. In questo caso, il token serve per le operazioni di utilizzo delle chiavi (`key-usage`servizio). Questo esempio utilizza il `--filter` flag per associare il token a una chiave specifica.

   ```
   aws-cloudhsm > quorum token-sign generate --service key-usage --token </path/
   crypto_user1.token> --filter attr.label=rsa-private-key-example
   {
     "error_code": 0,
     "data": {
       "path": "/home/crypto_user1.token"
   ```

```
  }
}
```

Questo esempio ottiene un token quorum per l'utente crittografico con nome utente `crypto_user1` e lo salva in un file denominato. `crypto_user1.token` Per utilizzare il comando di esempio, sostituisci i valori i tuoi personali:

Il quorum token-sign generate comando genera un token quorum del servizio di utilizzo delle chiavi nel percorso del file specificato. È possibile ispezionare il file del token:

```
$ cat </path/crypto_user1.token>
{
  "version": "2.0",
  "service": "key-usage",
  "key_reference": "0x0000000000680006",
  "approval_data":
 "AAIABQAAABkAAAAAGgABi5CDa9x9VyyRIaFbkSrHgJjcnlwdG9fdXNlcgAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
+GJj8gXo9lKuANGNyeXB0b191c2VyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGc9AEsAAAAAAAAAA==",
  "token": "5GlgoWOlQU4fw4QIlbxkPGZVOVoDugFGuSKE/k67ncM=",
  "signatures": []
}
```

Il file del token comprende:

- service: un identificatore per il servizio quorum a cui è associato il token.
- key_reference: un identificatore per la chiave a cui è associato questo token del quorum.
- approval_data: un token di dati non elaborati con codifica base64 generato dall'HSM.
- token: un token con codifica base64 sottoposto ad hashing SHA-256 di approval_data
- firme: una matrice di token firmati codificati in base64 (firme) del token non firmato. Ogni firma dell'approvatore ha la forma di un oggetto letterale JSON:

  ```
  {
        "username": "<APPROVER_USERNAME>",
        "role": "<APPROVER_ROLE>",
        "signature": "<APPROVER_RSA2048_BIT_SIGNATURE>"
  }
  ```

  Ogni firma viene creata in base al risultato di un approvatore che utilizza la corrispondente chiave privata RSA a 2048 bit la cui chiave pubblica è stata registrata presso l'HSM.

4. Convalida il nuovo token del quorum del servizio utente. Il quorum token-sign list comando conferma che il token esiste su CloudHSM.

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": [
      {
        "username": "crypto_user",
        "service": "key-usage",
        "key-reference": "0x0000000000680006",
        "minimum-token-count": 2
      }
    ]
  }
}
```

`minimum-token-count`Presenta una visualizzazione aggregata del cluster del numero minimo utilizzabile di token chiave corrispondenti al nome utente, al servizio e al riferimento chiave recuperati da un singolo HSM nel cluster.

Ad esempio, supponendo un cluster 2-HSM, se riceviamo due (2) token di utilizzo delle chiavi generati dall'utente `crypto_user1` per la chiave con riferimento `0x0000000000680006` dal primo HSM del cluster e riceviamo un (1) token di utilizzo delle chiavi generati dall'utente `crypto_user1` per la chiave con riferimento `0x0000000000680006` dall'altro HSM del cluster, visualizzeremo`"minimum-token-count": 1`.

Passaggio 2. Ottieni firme dagli utenti che approvano le criptovalute

Un utente crittografico che ha un quorum token deve ottenere l'approvazione del token da altri cripto-utenti. Per dare la loro approvazione, gli altri utenti di crypto=-utilizzano la loro chiave di firma per firmare crittograficamente il token all'esterno dell'HSM.

Sono disponibili vari modi per firmare il token. L'esempio seguente mostra come firmare il token utilizzando OpenSSL. Per utilizzare uno strumento di firma diverso, assicurati che lo strumento utilizzi la chiave privata (chiave di firma) del cripto-utente per firmare un digest SHA-256 del token.

In questo esempio, l'utente crittografico che dispone del token (`crypto-user`) necessita di almeno due (2) approvazioni. I seguenti comandi di esempio mostrano come due (2) cripto-utenti possono utilizzare OpenSSL per firmare crittograficamente il token.

1. Decodifica il token non firmato con codifica base64 e inseriscilo in un file binario:

```
$echo -n '5GlgoWOlQU4fw4QIlbxkPGZVOVoDugFGuSKE/k67ncM=' | base64 -d >
 crypto_user1.bin
```

2. Usa OpenSSL e la chiave privata dell'approvatore per firmare il token unsigned del quorum binario per il servizio utente e creare un file di firma binario:

```
$openssl pkeyutl -sign \
-inkey crypto_user1.key \
-pkeyopt digest:sha256 \
-keyform PEM \
-in crypto_user1.bin \
-out crypto_user1.sig.bin
```

3. Codifica la firma binaria in base64:

```
$ base64 -w0 crypto_user1.sig.bin > crypto_user1.sig.b64
```

4. Copia e incolla la firma codificata base64 nel file token, utilizzando il formato letterale dell'oggetto JSON specificato in precedenza per la firma dell'approvatore:

```
{
  "version": "2.0",
  "service": "key-usage",
  "key_reference": "0x0000000000680006",
  "approval_data":
 "AAIABQAAABkAAAAAGgABi5CDa9x9VyyRIaFbkSrHgJjcnlwdG9fdXNlcgAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
+GJj8gXo9lKuANGNyeXB0b191c2VyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGc9AEsAAAAAAAAAA==",
  "token": "5GlgoWOlQU4fw4QIlbxkPGZVOVoDugFGuSKE/k67ncM=",
  "signatures": [
    {
      "username": "crypto_user1",
      "role": "crypto-user",
      "signature": "wa7aPzmGwBjcEoZ6jAzYASp841AfgOvcI27Y/
tGlCj1E9DawnFw5Uf0IJT2Ca7T5XD2ThVkUi0B+dhAomdqYNl6aUUFrJyH9GBJ
+E0PmA5jNVm25tzeRWBJzneTg4/
zTeE2reNqrHFHicWnttQLe9jS09J1znuDGWDe0HaBKWUaz2gUInJRqmeXDsZYdSvZksrqUH5dci/
```

```
RsaDE2+tGiS9g0RcIkFbsPW4HpGe2e5HVzGsqrV8O3PKlYQv6+fymfcNTTuoxKcHAkOjpl43QSuSIu2gVq7KI8mSmmW
+oiukaNfLJr+MoDKzAvCGDg4cDArg=="
    },
    {
      "username": "crypto_user2",
      "role": "crypto-user",
      "signature": "wa7aPzmGwBjcEoZ6jAzYASp841AfgOvcI27Y/
tGlCj1E9DawnFw5Uf0IJT2Ca7T5XD2ThVkUi0B+dhAomdqYNl6aUUFrJyH9GBJ
+E0PmA5jNVm25tzeRWBJzneTg4/
zTeE2reNqrHFHicWnttQLe9jS09J1znuDGWDe0HaBKWUaz2gUInJRqmeXDsZYdSvZksrqUH5dci/
RsaDE2+tGiS9g0RcIkFbsPW4HpGe2e5HVzGsqrV8O3PKlYQv6+fymfcNTTuoxKcHAkOjpl43QSuSIu2gVq7KI8mSmmW
+oiukaNfLJr+MoDKzAvCGDg4cDArg=="
    }
  ]
}
```

Fase 3. Approvare il token sul CloudHSM; raggruppare ed eseguire un'operazione

Dopo aver ottenuto le approvazioni e le firme necessarie, un utente crittografico può fornire quel token al cluster CloudHSM insieme a un'operazione di gestione o utilizzo delle chiavi.

Assicurati che l'operazione con la chiave corrisponda al servizio di quorum appropriato associato al token quorum. Per ulteriori informazioni, consulta Servizi e tipi supportati.

Durante la transazione, il token verrà approvato all'interno del AWS CloudHSM cluster ed eseguirà l'operazione chiave richiesta. Il successo dell'operazione chiave dipende sia da un token quorum approvato valido sia da un'operazione chiave valida.

Example Genera una firma con il meccanismo RSA-PKCS

Nell'esempio seguente, un cripto-utente registrato crea una firma con una chiave sull'HSM:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --
hash-function sha256 --data YWJjMTIz --approval /path/crypto_user1.token

{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000640007",
    "signature":
 "h6hMqXacBrT3x3MXV13RXHdQno0+IQ6iy0kVrGzo23+eoWT0ZZgrSpBCu5KcuP6IYYHw9goQ5CfPf4jI1nO5m/
IUJtF1A1lmcz0HjEy1CJ7ICXNReDRyeOU8m43dkJzt0OUdkbtkDJGAcxkbKHLZ02uWsGXaQ8bOKhoGwsRAHHF6nldTXquIC
```

```
+pZmUS38ythybney94Wj6fzYOER8v7VIY5ijQGa3LfxrjSG4aw6QijEEbno5LSf18ahEaVKmVEnDBL54tylCJBGvGsYSY9H
TDd2wfvP4PaxbFRyyHaw=="
  }
}
```

Se l'utente crittografico tenta di eseguire un'altra operazione di utilizzo della chiave HSM con lo stesso token, fallisce:

```
aws-cloudhsm > crypto sign rsa-pkcs --key-filter attr.label=rsa-private-key-example --
hash-function sha256 --data YWJjMTIz --approval /home/crypto_user1.token
{
  "error_code": 1,
  "data": "Quorum approval is required for this operation"
}
```

Per eseguire un'altra operazione con la chiave HSM, l'utente crittografico deve generare un nuovo token quorum, ottenere nuove firme dagli approvatori ed eseguire l'operazione chiave desiderata con l'argomento --approval per fornire il token quorum.

Usa il per verificare la disponibilità del token. quorum token-sign list Questo esempio mostra che l'utente crittografico non ha token approvati.

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": []
  }
}
```

Gestione delle chiavi con la AWS CloudHSM KMU

Se utilizzi la serie di versioni SDK più recente, utilizza la CLI di CloudHSM per gestire le chiavi nel cluster. AWS CloudHSM

Se utilizzi la serie di versioni SDK precedente, puoi gestire le chiavi sui moduli di sicurezza hardware (HSM) del AWS CloudHSM cluster utilizzando lo strumento da riga di comando key_mgmt_util (KMU). Prima di poter gestire le chiavi, è necessario avviare il AWS CloudHSM client, avviare key_mgmt_util e accedere a. HSMs Per ulteriori informazioni, consulta la pagina Getting Started with key_mgmt_util.

- La pagina sull'utilizzo di chiavi attendibili descrive come utilizzare gli attributi della libreria PKCS #11 e CMU per creare chiavi attendibili per proteggere i dati.
- La pagina sulla generazione delle chiavi presenta istruzioni sulla creazione delle chiavi, tra cui chiavi simmetriche, chiavi RSA e chiavi EC.
- La pagina sull'importazione delle chiavi fornisce i dettagli su come i proprietari possono importare le chiavi.
- La pagina sull'esportazione delle chiavi fornisce i dettagli su come i proprietari possono esportare le chiavi.
- La pagina sull'eliminazione delle chiavi fornisce i dettagli su come i proprietari possono eliminare le chiavi.
- La pagina su condivisione e annullamento della condivisione delle chiavi illustra in che modo i proprietari possono condividere e annullare la condivisione delle chiavi.

Genera AWS CloudHSM chiavi con la KMU

Per generare chiavi sul modulo di sicurezza hardware (HSM), utilizzate il comando in AWS CloudHSM key_mgmt_util (KMU) che corrisponde al tipo di chiave che desiderate generare.

Argomenti

- Genera chiavi simmetriche con la KMU AWS CloudHSM
- Genera coppie di chiavi RSA con la AWS CloudHSM KMU
- Genera coppie di chiavi ECC (crittografia a curva ellittica) utilizzando la KMU AWS CloudHSM

Genera chiavi simmetriche con la KMU AWS CloudHSM

Usa il genSymKeycomando in AWS CloudHSM key_mgmt_util (KMU) per generare AES e altri tipi di chiavi simmetriche per. AWS CloudHSM Per visualizzare tutte le opzioni disponibili, utilizza il comando genSymKey -h.

Nell'esempio seguente viene creata una chiave AES a 256 bit.

```
Command: genSymKey -t 31 -s 32 -l aes256
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Created.  Key Handle: 524295

Cluster Error Status
```

```
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Genera coppie di chiavi RSA con la AWS CloudHSM KMU

Per generare una coppia di chiavi RSA per AWS CloudHSM, usa il comando gen RSAKey Pair in AWS CloudHSM key_mgmt_util. Per visualizzare tutte le opzioni disponibili, utilizza il comando genRSAKeyPair -h.

Nell'esempio di seguito viene creata una coppia di chiavi RSA a 2048 bit.

```
Command: genRSAKeyPair -m 2048 -e 65537 -l rsa2048
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 524294    private key handle: 524296

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Genera coppie di chiavi ECC (crittografia a curva ellittica) utilizzando la KMU AWS CloudHSM

Per generare una coppia di chiavi ECC per AWS CloudHSM, usa il comando gen ECCKey Pair in AWS CloudHSM key_mgmt_util. Per visualizzare tutte le opzioni disponibili, tra cui un elenco delle curve ellittiche supportate, utilizza il comando genECCKeyPair -h.

Nell'esempio di seguito viene creata una coppia di chiavi ECC con la curva ellittica P-384 definita nella pubblicazione NIST FIPS 186-4.

```
Command: genECCKeyPair -i 14 -l ecc-p384
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 524297    private key handle: 524298

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

AWS CloudHSM Importa le chiavi con la KMU

Per importare chiavi segrete, ovvero chiavi simmetriche e chiavi private asimmetriche, nel modulo di sicurezza hardware (HSM) utilizzando AWS CloudHSM key_mgmt_util, devi prima creare una chiave di wrapping sull'HSM. Puoi importare le chiavi pubbliche direttamente senza una chiave di wrapping.

Argomenti

- Importa le chiavi segrete con la KMU AWS CloudHSM
- Importa le chiavi pubbliche con la AWS CloudHSM KMU

Importa le chiavi segrete con la KMU AWS CloudHSM

Completa i seguenti passaggi per importare una chiave segreta nell' AWS CloudHSM utilizzo di key_mgmt_util (KMU). Prima di importare una chiave segreta, salvala in un file. Salva le chiavi simmetriche come byte non elaborati e le chiavi private asimmetriche in formato PEM.

In questo esempio viene mostrato come importare una chiave segreta con testo non crittografato da un file nell'HSM. Per importare una chiave crittografata da un file nell'HSM, usa il comando. unWrapKey

Per importare una chiave segreta

1. Utilizzate il genSymKeycomando per creare una chiave di wrapping. Il seguente comando crea una chiave di wrapping AES a 128 bit, valida solo per la sessione corrente. Puoi utilizzare una chiave di sessione o una persistente come chiave di wrapping.

```
Command: genSymKey -t 31 -s 16 -sess -l import-wrapping-key
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Created.  Key Handle: 524299

Cluster Error Status
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

2. Utilizza uno dei seguenti comandi, a seconda del tipo di chiave segreta che stai importando.

   - Per importare una chiave simmetrica, utilizza il comando imSymKey. Il seguente comando importa una chiave AES dal file `aes256.key` utilizzando la chiave di wrapping creata nella fase precedente. Per visualizzare tutte le opzioni disponibili, utilizza il comando imSymKey -h.

```
Command: imSymKey -f aes256.key -t 31 -l aes256-imported -w 524299
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Unwrapped.  Key Handle: 524300

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

- Per importare una chiave privata asimmetrica, utilizza il comando importPrivateKey. Il seguente comando importa una chiave privata dal file `rsa2048.key` utilizzando la chiave di wrapping creata nella fase precedente. Per visualizzare tutte le opzioni disponibili, utilizza il comando importPrivateKey -h.

```
Command: importPrivateKey -f rsa2048.key -l rsa2048-imported -w 524299
BER encoded key length is 1216

Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Private Key Unwrapped.  Key Handle: 524301

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Importa le chiavi pubbliche con la AWS CloudHSM KMU

Usa il importPubKeycomando contenuto in AWS CloudHSM key_mgmt_util (KMU) per importare una chiave pubblica. Per visualizzare tutte le opzioni disponibili, utilizza il comando importPubKey -h.

Nell'esempio seguente viene importata una chiave pubblica RSA dal file `rsa2048.pub`.

```
Command: importPubKey -f rsa2048.pub -l rsa2048-public-imported
Cfm3CreatePublicKey returned: 0x00 : HSM Return: SUCCESS

Public Key Handle: 524302

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Esporta AWS CloudHSM le chiavi con la KMU

Per esportare chiavi AWS CloudHSM segrete, ovvero chiavi simmetriche e chiavi private asimmetriche, dal modulo di sicurezza hardware (HSM) utilizzando AWS CloudHSM key_mgmt_util (KMU), devi prima creare una chiave di wrapping. Puoi esportare le chiavi pubbliche direttamente senza una chiave di wrapping.

Soltanto il proprietario della chiave può esportarla. Gli utenti con cui è condivisa la chiave possono utilizzarla in operazioni di crittografia, ma non possono esportarla. Durante l'esecuzione di questo esempio, assicurati di esportare una chiave creata.

> **Important**
>
> exSymKeyIl comando scrive una copia in testo semplice (non crittografata) della chiave segreta in un file. Il processo di esportazione richiede una chiave di wrapping, ma la chiave nel file non è di questo tipo. Per esportare una copia di una chiave di wrapping (crittografata), utilizza il comando wrapKey.

Argomenti

- Esporta le chiavi segrete con la KMU AWS CloudHSM
- Esporta le chiavi pubbliche con la KMU AWS CloudHSM

Esporta le chiavi segrete con la KMU AWS CloudHSM

Completa i seguenti passaggi per esportare una chiave segreta AWS CloudHSM utilizzando key_mgmt_util (KMU).

Per esportare una chiave segreta

1. Utilizzate il comando per creare una chiave di avvolgimento. genSymKey Il seguente comando crea una chiave di wrapping AES a 128 bit, valida solo per la sessione corrente.

```
Command: genSymKey -t 31 -s 16 -sess -l export-wrapping-key
Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

Symmetric Key Created.  Key Handle: 524304

Cluster Error Status
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

2. Utilizza uno dei seguenti comandi, a seconda del tipo di chiave segreta che stai esportando.

   - Per esportare una chiave simmetrica, usa il comando. exSymKey Il comando seguente esporta una chiave AES nel file `aes256.key.exp`. Per visualizzare tutte le opzioni disponibili, utilizza il comando exSymKey -h.

```
Command: exSymKey -k 524295 -out aes256.key.exp -w 524304
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS


Wrapped Symmetric Key written to file "aes256.key.exp"
```

> **Note**
>
> L'output del comando indica che una "Chiave simmetrica di wrapping" è stata scritta nel file di output. Tuttavia, il file di output contiene una chiave con testo non crittografato (non wrapping). Per esportare una chiave di wrapping (crittografata) in un file, utilizza il comando wrapKey.

   - Per esportare una chiave privata, utilizza il comando exportPrivateKey. Il comando seguente esporta una chiave privata nel file `rsa2048.key.exp`. Per visualizzare tutte le opzioni disponibili, utilizza il comando exportPrivateKey -h.

```
Command: exportPrivateKey -k 524296 -out rsa2048.key.exp -w 524304
Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS

PEM formatted private key is written to rsa2048.key.exp
```

Esporta le chiavi pubbliche con la KMU AWS CloudHSM

Usa il exportPubKey comando contenuto in AWS CloudHSM key_mgmt_util (KMU) per esportare una chiave pubblica. Per visualizzare tutte le opzioni disponibili, utilizza il comando exportPubKey -h.

Nell'esempio seguente viene esportata una chiave pubblica RSA nel file `rsa2048.pub.exp`.

```
Command: exportPubKey -k 524294 -out rsa2048.pub.exp
PEM formatted public key is written to rsa2048.pub.key

Cfm3ExportPubKey returned: 0x00 : HSM Return: SUCCESS
```

Eliminare le chiavi con KMU e CMU

Utilizza il comando deleteKey per eliminare una chiave, come mostrato nell'esempio seguente: Soltanto il proprietario della chiave può eliminarla.

```
Command: deleteKey -k 524300
Cfm3DeleteKey returned: 0x00 : HSM Return: SUCCESS

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Condividi e annulla la condivisione delle chiavi con KMU e CMU

Nel AWS CloudHSM, la CU che crea la chiave ne è proprietaria. Il proprietario gestisce la chiave, può esportarla ed eliminarla, nonché utilizzarla in operazioni di crittografia. Il proprietario può anche condividere la chiave con altri utenti CU. Gli utenti con cui è condivisa la chiave possono utilizzarla in operazioni di crittografia, ma non possono esportarla, eliminarla, né condividerla con altri utenti.

È possibile condividere le chiavi con altri utenti CU quando si crea la chiave, ad esempio utilizzando il `-u` parametro dei comandi genSymKeyo gen RSAKey Pair. Per condividere le chiavi esistenti con un altro utente HSM, utilizza lo strumento a riga di comando cloudhsm_mgmt_util. Questa operazione è

diversa dalla maggior parte delle attività illustrate in questa sezione, che utilizzano lo strumento a riga di comando key_mgmt_util.

Prima di poter condividere una chiave, devi avviare cloudhsm_mgmt_util, abilitare la crittografia e accedere a. end-to-end HSMs Per condividere una chiave, accedi all'HSM come crypto user (CU) che possiede la chiave. Solo i proprietari di chiavi possono condividere una chiave.

Usa il shareKey comando per condividere o annullare la condivisione di una chiave, specificando l'handle della chiave e l'utente o gli utenti. IDs Per condividere o annullare la condivisione con più di un utente, specifica un elenco di utenti separati da virgole. IDs Per condividere una chiave, utilizza il comando 1 come ultimo parametro, come nell'esempio seguente. Per annullare la condivisione, utilizza `0`.

```
aws-cloudhsm > shareKey 524295 4 1
************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)? y
shareKey success on server 0(10.0.2.9)
shareKey success on server 1(10.0.3.11)
shareKey success on server 2(10.0.1.12)
```

Di seguito è mostrata la sintassi del comando shareKey.

```
aws-cloudhsm > shareKey <key handle> <user ID> <Boolean: 1 for share, 0 for unshare>
```

Come contrassegnare una chiave come attendibile con l'utilità AWS CloudHSM di gestione

Il contenuto di questa sezione fornisce istruzioni sull'utilizzo dell'utilità di AWS CloudHSM gestione (CMU) per contrassegnare una chiave come attendibile.

1. Accedi come crypto officer (CO) utilizzando il comando loginHSM.
2. Usa il comando Imposta gli attributi delle AWS CloudHSM chiavi usando CMU con `OBJ_ATTR_TRUSTED` (valore `134`) impostato su true (1).

```
aws-cloudhsm > setAttribute <Key Handle> 134 1
```

Backup dei cluster in AWS CloudHSM

AWS CloudHSM esegue backup periodici del cluster almeno una volta ogni 24 ore. Un backup contiene copie crittografate dei seguenti dati:

- Utenti (COs CUs, e) AUs
- Materiale della chiave e certificati
- Configurazione e politiche per il modulo di sicurezza hardware (HSM)

Non è possibile indicare al servizio di eseguire backup, ma è possibile eseguire determinate azioni che costringono il servizio a creare un backup. Il servizio esegue un backup nel caso di una delle seguenti operazioni:

- Attivazione di un cluster
- Aggiunta di un HSM a un cluster attivo
- Rimozione di un HSM da un cluster attivo

AWS CloudHSM elimina i backup in base alla politica di conservazione dei backup impostata al momento della creazione dei cluster. Per informazioni sulla gestione della politica di conservazione dei backup, consulta Configura la conservazione dei backup.

Argomenti

- Utilizzo dei backup dei cluster AWS CloudHSM
- Eliminare i backup AWS CloudHSM dei cluster
- Ripristina i backup AWS CloudHSM
- Configura la AWS CloudHSM politica di conservazione dei backup
- Copia dei backup dei AWS CloudHSM cluster tra le regioni AWS
- Utilizzo dei backup condivisi in AWS CloudHSM

Utilizzo dei backup dei cluster AWS CloudHSM

Quando si aggiunge un modulo di sicurezza hardware (HSM) a un cluster AWS CloudHSM che in precedenza ne conteneva uno o più attivi HSMs, il servizio ripristina il backup più recente sul

nuovo HSM. Utilizza i backup per gestire HSMs l'utilizzo non frequente. Se non devi utilizzare l'HSM, puoi eliminarlo e questa operazione attiva un backup. Successivamente, quando avrai bisogno dell'HSM, creane uno nuovo sullo stesso cluster, ripristinando così il backup creato in precedenza con l'operazione di eliminazione dell'HSM.

Rimozione delle chiavi scadute o degli utenti inattivi

In alcuni casi, occorre rimuovere determinati materiali crittografici dall' ambiente, come ad esempio chiavi scadute o utenti inattivi. Si tratta di un processo suddiviso in due parti. Per prima cosa, elimina questi materiali dal tuo HSM. Poi elimina tutti i backup esistenti. La procedura garantisce di non ripristinare le informazioni eliminate durante l'inizializzazione di un nuovo cluster dal backup. Per ulteriori informazioni, consulta the section called "Eliminare i backup".

Considerazioni sul ripristino di emergenza

Si può creare un cluster da un backup. L'operazione può essere utile per impostare un punto di ripristino per il cluster. Rinomina un backup che contenga tutti gli utenti, il materiale della chiave e i certificati che desideri inserire nel punto di ripristino, quindi utilizza quel backup per creare un nuovo cluster. Per ulteriori informazioni sulla creazione di un cluster da un backup, consulta Creazione di cluster dai backup.

Puoi anche copiare un backup di un cluster in un'altra regione, dove puoi creare un nuovo cluster come clone dell'originale. È possibile eseguire questa operazione per una serie di motivi, tra cui la semplificazione del processo di disaster recovery. Per ulteriori informazioni sulla copia dei backup nelle regioni, consulta Copiare un backup tra regioni.

Eliminare i backup AWS CloudHSM dei cluster

Dopo aver eliminato un backup del AWS CloudHSM cluster, il servizio conserva il backup per sette giorni, durante i quali è possibile ripristinarlo. Dopo il periodo di sette giorni, non è più possibile ripristinare il backup. Per ulteriori informazioni sulla gestione dei backup, consulta Backup del cluster.

La tabella seguente descrive come eliminare un backup.

Console

Per eliminare un backup (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.

2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Nel pannello di navigazione, scegliere Backup.
4. Scegli il backup da eliminare.
5. Per eliminare il backup selezionato, scegli Actions (Operazioni), Delete (Elimina).

   Viene visualizzata la finestra di dialogo Delete backups (Elimina backup).
6. Scegli Elimina.

   Lo stato del backup cambia in`PENDING_DELETE`. Puoi ripristinare un backup in attesa di eliminazione per un massimo di 7 giorni dopo la richiesta di eliminazione.

Per elencare i backup (AWS CLI)

- Per un elenco di tutti i backup nello stato `PENDING_DELETION`, eseguire il comando describe-backups e includere `states=PENDING_DELETION` come filtro.

```
$ aws cloudhsmv2 describe-backups --filters states=PENDING_DELETION
{
    "Backups": [
        {
            "BackupId": "backup-ro5c4er4aac",
            "BackupState": "PENDING_DELETION",
            "ClusterId": "cluster-dygnwhmscg5",
            "CreateTimestamp": 1534461854.64,
            "DeleteTimestamp": 1536339805.522,
            "HsmType": "hsm2m.medium",
            "Mode": "NON_FIPS",
            "NeverExpires": false,
            "TagList": []
        }
}
```

AWS CLI

Controlla lo stato di un backup o individuane l'ID utilizzando il comando describe-backups da AWS CLI.

Per eliminare un backup (AWS CLI)

- Al prompt dei comandi eseguire il comando delete-backup specificando l'ID del backup da eliminare.

```
$ aws cloudhsmv2 delete-backup --backup-id <backup ID>
{
    "Backup": {
        "CreateTimestamp": 1534461854.64,
        "ClusterId": "cluster-dygnwhmscg5",
        "BackupId": "backup-ro5c4er4aac",
        "BackupState": "PENDING_DELETION",
        "DeleteTimestamp": 1536339805.522,
        "HsmType": "hsm1.medium",
        "Mode": "FIPS"
    }
}
```

AWS CloudHSM API

Fai riferimento DeleteBackupa come eliminare i backup utilizzando l'API.

Ripristina i backup AWS CloudHSM

AWS CloudHSM conserva i backup eliminati per sette giorni, durante i quali è possibile ripristinare il backup. Dopo il periodo di sette giorni, non è più possibile ripristinare il backup. Per ulteriori informazioni sulla gestione dei backup, consulta Backup del cluster.

La tabella seguente descrive come eliminare un backup.

Console

Per ripristinare un backup (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Nel pannello di navigazione, scegliere Backup.
4. Scegli un backup nello stato `PENDING_DELETE` da ripristinare.

5. Per ripristinare il backup selezionato, scegli Actions (Operazioni), Restore (Ripristina).

AWS CLI

Per ripristinare un backup (AWS CLI)

- Per ripristinare un backup, eseguire il comando restore-backup specificando l'ID di un backup nello stato `PENDING_DELETION`.

```
$ aws cloudhsmv2 restore-backup --backup-id <backup ID>
{
    "Backup": {
        "ClusterId": "cluster-dygnwhmscg5",
        "CreateTimestamp": 1534461854.64,
        "BackupState": "READY",
        "BackupId": "backup-ro5c4er4aac"
    }
}
```

AWS CloudHSM API

Fai riferimento RestoreBackupa come ripristinare i backup utilizzando l'API.

Configura la AWS CloudHSM politica di conservazione dei backup

AWS CloudHSM elimina i backup in base alla politica di conservazione dei backup impostata al momento della creazione di un cluster. La politica di conservazione del backup si applica ai cluster. Se si sposta un backup in un'altra regione, il backup non è più associato a un cluster e non ha alcuna politica di conservazione dei backup. È necessario eliminare manualmente tutti i backup non associati a un cluster. AWS CloudHSM non elimina l'ultimo backup di un cluster. Per i cluster migrati da hsm1.medium, AWS CloudHSM conserva il backup hsm1.medium più recente fino alla fine del ciclo di vita di hsm1.medium.

AWS CloudTrail segnala i backup contrassegnati per l'eliminazione. Puoi ripristinare i backup eliminati dal servizio esattamente come si ripristinano i backup eliminati manualmente. Per evitare situazioni di concorrenza, è necessario modificare la politica di conservazione dei backup per il cluster prima di ripristinare un backup eliminato dal servizio. Per mantenere invariata la politica di

conservazione e preservare determinati backup, è possibile specificare che il servizio escluda i backup dalla politica di conservazione dei backup del cluster.

Per ulteriori AWS CloudHSM informazioni Riduci i costi adattandoli alle tue esigenze sui prezzi, consulta.

Conservazione gestita dei backup

I cluster creati prima del 18 novembre 2020 hanno una politica di conservazione dei backup di 90 giorni più l'età del cluster. Ad esempio, se hai creato un cluster il 18 novembre 2019, il servizio assegnerebbe al cluster una politica di conservazione dei backup di un anno più 90 giorni (455 giorni). Puoi impostare questo periodo su qualsiasi numero compreso tra 7 e 379 giorni. AWS CloudHSM non elimina l'ultimo backup di un cluster. Per ulteriori informazioni sulla gestione dei backup, consulta Backup del cluster.

> **Note**
>
> È possibile disattivare completamente la conservazione dei backup gestiti contattando. Supporto AWS

La tabella seguente descrive come impostare la conservazione dei backup.

Console

Per configurare la politica di conservazione dei backup (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Fai clic sull'ID del cluster nello stato Active (Attivo) per gestire la politica di conservazione dei backup per quel cluster.
4. Per modificare la politica di conservazione dei backup, scegli Actions (Operazioni), Change backup retention period (Modifica periodo di conservazione dei backup).

   Viene visualizzata la finestra di dialogo Change backup retention period (Modifica del periodo di conservazione dei backup).
5. In Backup retention period (in days) (Periodo di conservazione del backup (in giorni)), digita un valore compreso tra 7 e 379 giorni.

6. Scegli Change backup retention period (Modifica periodo di conservazione dei backup).

Per escludere o includere un backup dalla politica di conservazione dei backup (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Per visualizzare i backup, scegli Backups (Backup) nel pannello di navigazione.
3. Fai clic sull'ID di backup di un backup nello stato Ready (Pronto) per escluderlo o includerlo.
4. Esegui una delle seguenti operazioni nella pagina dei Backup details (Dettagli del backup).
   - Per escludere un backup con una data in Expiration time (Ora di scadenza), scegli Actions (Operazioni), Disable expiration (Disabilita scadenza).
   - Per includere un backup che non ha scadenza, scegli Actions (Operazioni), Use cluster retention policy (Usa la politica di conservazione del cluster).

AWS CLI

Per configurare la politica di conservazione dei backup (AWS CLI)

- Al prompt dei comandi, esegui il comando modify-cluster. Specifica l'ID del cluster e la politica di conservazione dei backup.

```
$ aws cloudhsmv2 modify-cluster --cluster-id <cluster ID> \
                                --backup-retention-policy
 Type=DAYS,Value=<number of days to retain backups>
{
   "Cluster": {
      "BackupPolicy": "DEFAULT",
      "BackupRetentionPolicy": {
         "Type": "DAYS",
         "Value": 90
      },
      "Certificates": {},
      "ClusterId": "cluster-kdmrayrc7gi",
      "CreateTimestamp": 1504903546.035,
      "Hsms": [],
      "HsmType": "hsm1.medium",
      "SecurityGroup": "sg-40399d28",
      "State": "ACTIVE",
```

```
      "SubnetMapping": {
         "us-east-2a": "subnet-f1d6e798",
         "us-east-2c": "subnet-0e358c43",
         "us-east-2b": "subnet-40ed9d3b"
      },
      "TagList": [
         {
            "Key": "Cost Center",
            "Value": "12345"
         }
      ],
      "VpcId": "vpc-641d3c0d"
   }
}
```

Per escludere un backup dalla politica di conservazione dei backup (AWS CLI)

- Al prompt dei comandi, esegui il comando modify-backup-attributes. Specifica l'ID di backup e impostate il flag never-expires per conservare il backup.

```
$ aws cloudhsmv2 modify-backup-attributes --backup-id <backup ID> \
                                          --never-expires
{
   "Backup": {
      "BackupId": "backup-ro5c4er4aac",
      "BackupState": "READY",
      "ClusterId": "cluster-dygnwhmscg5",
      "NeverExpires": true
   }
}
```

Per includere un backup nella politica di conservazione dei backup (AWS CLI)

- Al prompt dei comandi, esegui il comando modify-backup-attributes. Specificate l'ID di backup e impostate il no-never-expires flag per includere il backup nella politica di conservazione dei backup, il che significa che il servizio alla fine eliminerà il backup.

```
$ aws cloudhsmv2 modify-backup-attributes --backup-id <backup ID> \
                                          --no-never-expires
{
```

```
    "Backup": {
        "BackupId": "backup-ro5c4er4aac",
        "BackupState": "READY",
        "ClusterId": "cluster-dygnwhmscg5",
        "NeverExpires": false
    }
}
```

AWS CloudHSM API

Consulta i seguenti argomenti su come gestire la conservazione dei backup utilizzando l'API.

- ModifyCluster
- ModifyBackupAttributes

Copia dei backup dei AWS CloudHSM cluster tra le regioni AWS

È possibile copiare i backup dei AWS CloudHSM cluster tra diverse regioni per molte ragioni, tra cui la resilienza interregionale, i carichi di lavoro globali e il disaster recovery. Dopo aver copiato i backup, questi vengono visualizzati nell'area di destinazione con stato `CREATE_IN_PROGRESS`. Una volta completata l'operazione, lo stato del backup copiato è `READY`. Se la copia non va a buon fine, lo stato del backup diventa `DELETED`. Controlla i parametri di input per gli errori e assicurati che l'origine di backup specificata non sia in stato `DELETED` prima di eseguire nuovamente l'operazione. Per informazioni su come creare un cluster da un backup vedi Backup del cluster o Creazione di cluster dai backup.

Tenere presente quanto segue:

- Per copiare il backup di un cluster in una regione di destinazione, l'account deve avere le autorizzazioni di policy IAM; corrette. Per copiare il backup in un'altra regione, la policy IAM ti deve consentire l'accesso alla regione di origine in cui si trova il backup. Una volta copiata in più regioni, la policy IAM deve consentire l'accesso alla regione di destinazione in modo da interagire con la copia di backup, che comprende l'utilizzo dell'operazione CreateCluster. Per ulteriori informazioni, vedi Creazione di amministratori IAM;.
- Il cluster originale e il cluster che può essere creato da un backup nella regione di destinazione non sono collegati. È necessario gestire ognuno di questi cluster in modo indipendente. Per ulteriori informazioni, consulta Cluster.

- I backup non possono essere copiati tra le partizioni. AWS I backup possono essere copiati tra aree con AWS restrizioni e aree standard all'interno della stessa partizione.

Copia i backup in diverse regioni (console)

Per copiare i backup in diverse regioni (console)

1. Apri la AWS CloudHSM console a casa. https://console.aws.amazon.com/cloudhsm/
2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Nel pannello di navigazione, scegliere Backup.
4. Scegli il backup da copiare in un'altra regione.
5. Per copiare il backup selezionato, scegli Azioni, Copia il backup in un'altra regione.

   Viene visualizzata la finestra di dialogo Copia il backup in un'altra regione.
6. In Regione di destinazione, scegli una regione da Seleziona una regione.
7. (Facoltativo) Digitare una tag di chiave e un valore di tag facoltativo. Per aggiungere più di un tag al cluster scegli Aggiungi tag.
8. Scegli Copia backup.

Copia i backup in diverse regioni (AWS CLI)

Per determinare l'ID di backup, esegui il describe-backups comando.

Per copiare i backup in aree diverse (AWS CLI)

- Al prompt dei comandi, esegui il comando copy-backup-to-region. Specifica la regione di destinazione e l'ID cluster del cluster di origine o l'ID backup del backup di origine. Se si specifica un ID di backup, il backup associato viene copiato.

```
$ aws cloudhsmv2 copy-backup-to-region --destination-region <destination region> \
                                       --backup-id <backup ID>
```

Copia i backup in diverse regioni (AWS CloudHSM API)

Fai riferimento al seguente argomento per scoprire come copiare i backup in diverse regioni utilizzando l'API.

- CopyBackupToRegion

Utilizzo dei backup condivisi in AWS CloudHSM

CloudHSM si integra AWS Resource Access Manager con AWS RAM() per consentire la condivisione delle risorse. AWS RAM è un servizio che consente di condividere alcune risorse CloudHSM con Account AWS altri o tramite. AWS Organizations Con AWS RAM, condividi le risorse di tua proprietà creando una condivisione di risorse. Una condivisione delle risorse specifica le risorse da condividere e gli utenti con cui condividerle. I consumatori includono:

- Specifico Account AWS all'interno o all'esterno della sua organizzazione in AWS Organizations
- Un'unità organizzativa all'interno della propria organizzazione in AWS Organizations
- Un'intera organizzazione in AWS Organizations

Per ulteriori informazioni in merito AWS RAM, consulta la Guida AWS RAM per l'utente.

Questo argomento spiega come condividere le tue risorse e come utilizzare le risorse condivise con te.

Indice

- Prerequisiti per la condivisione dei backup
- Condivisione di un backup
- Annullamento della condivisione di un backup condiviso
- Identificazione di un backup condiviso
- Autorizzazioni per i backup condivisi
- Fatturazione e misurazione

Prerequisiti per la condivisione dei backup

- Per condividere un backup, devi possederlo nel tuo. Account AWS Ciò significa che la risorsa deve essere allocata o assegnata all'account. Non puoi condividere un backup che è stato condiviso con te.
- Per condividere un backup, deve essere nello stato READY.
- Per condividere un backup con l'organizzazione o un'unità organizzativa in AWS Organizations, è necessario abilitare la condivisione con AWS Organizations. Per ulteriori informazioni, consulta Abilita la condivisione con AWS Organizations nella Guida per l'utente AWS RAM .

Condivisione di un backup

Quando condividi un backup con altri Account AWS, consenti loro di ripristinare i cluster dal backup che contengono le chiavi e gli utenti archiviati nel backup.

Per condividere un backup, è necessario aggiungerlo a una condivisione di risorse. Una condivisione di risorse è una risorsa AWS RAM che consente di condividere le risorse tra Account AWS. Una condivisione di risorse specifica le risorse da condividere e i consumatori con cui sono condivise. Quando condividi un backup utilizzando la console CloudhSM, lo aggiungi a una condivisione di risorse esistente. Per aggiungere il backup a una nuova condivisione di risorse, devi prima creare la condivisione di risorse utilizzando la AWS RAM console.

Se fai parte di un'organizzazione AWS Organizations e la condivisione all'interno dell'organizzazione è abilitata, ai consumatori dell'organizzazione viene automaticamente concesso l'accesso al backup condiviso. In caso contrario, i consumatori ricevono un invito a partecipare alla condivisione delle risorse e ottengono l'accesso al backup condiviso dopo aver accettato l'invito.

Puoi condividere un backup di tua proprietà utilizzando la AWS RAM console o AWS CLI.

Per condividere un backup di tua proprietà utilizzando la AWS RAM console

Consulta Creazione di una condivisione di risorse in Guida per l'utente di AWS RAM .

Per condividere un backup di tua proprietà (AWS RAM comando)

Utilizza il comando create-resource-share.

Per condividere un backup di tua proprietà (comando CloudHSM)

> **Important**
>
> Sebbene sia possibile condividere un backup utilizzando l'operazione PutResourcePolicy CloudHSM, consigliamo invece di AWS Resource Access Manager utilizzare AWS RAM(). L'utilizzo AWS RAM offre molteplici vantaggi in quanto crea la policy adatta all'utente, consente di condividere più risorse contemporaneamente e aumenta la reperibilità delle risorse condivise. Se utilizzi PutResourcePolicy e desideri che i consumatori siano in grado di descrivere i backup che hai condiviso con loro, devi promuovere il backup a una condivisione di AWS RAM risorse standard utilizzando l'operazione AWS RAM PromoteResourceShareCreatedFromPolicy API.

Utilizza il comando put-resource-policy.

1. Crea un file denominato `policy.json` e copia la seguente politica al suo interno.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Principal": {
      "AWS":"111122223333"
      },
      "Action":[
        "cloudhsm:CreateCluster",
        "cloudhsm:DescribeBackups"
       ],
       "Resource":"arn:aws:cloudhsm:us-west-2:111122223333:backup/backup-to-share"
    }
  ]
}
```

2. Aggiorna `policy.json` con l'ARN di backup e gli identificatori con cui condividerlo. L'esempio seguente concede l'accesso in sola lettura all'utente root per l'account identificato da 123456789012. AWS

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
     "Effect":"Allow",
     "Principal": {
        "AWS": [
          "123456789012"
      ]
    },
    "Action": [
      "cloudhsm:CreateCluster",
      "cloudhsm:DescribeBackups"
     ],
    "Resource":"arn:aws:cloudhsm:us-west-2:123456789012:backup/backup-123"
   }
 ]
}
```

> **Important**
>
> È possibile concedere le autorizzazioni solo a livello di account. DescribeBackups Quando condividi un backup con un altro cliente, qualsiasi responsabile che disponga DescribeBackups dell'autorizzazione per l'account può descrivere il backup.

3. Esegui il comando put-resource-policy.

```
$ aws cloudhsmv2 put-resource-policy --resource-arn <resource-arn> --policy file://
policy.json
```

> **Note**
>
> A questo punto, il consumatore può utilizzare il backup, ma questo non verrà visualizzato nella DescribeBackups risposta con il parametro condiviso. I passaggi successivi

descrivono come promuovere la condivisione AWS RAM delle risorse in modo che il backup venga incluso nella risposta.

4. Ottieni l'ARN per la condivisione AWS RAM delle risorse.

```
$ aws ram list-resources --resource-owner SELF --resource-arns <backup-arn>
```

Ciò restituisce una risposta simile a questa:

```
{
  "resources": [
    {
      "arn": "<project-arn>",
      "type": "<type>",
      "resourceShareArn": "<resource-share-arn>",
      "creationTime": "<creation-time>",
      "lastUpdatedTime": "<last-update-time>"
    }
  ]
}
```

Dalla risposta, copia il *<resource-share-arn>* valore da utilizzare nei passaggi successivi.

5. Esegui il comando AWS RAM promote-resource-share-created-from-policy.

```
$ aws ram promote-resource-share-created-from-policy --resource-share-
arn <resource-share-arn>
```

6. Per verificare che la condivisione delle risorse sia stata promossa, puoi eseguire il comando. AWS RAM get-resource-shares

```
$ aws ram get-resource-shares --resource-owner SELF --resource-share-
arns <resource-share-arn>
```

Quando la politica è stata promossa, quella `featureSet` elencata nella risposta è`STANDARD`. Ciò significa anche che il backup può essere descritto dai nuovi account inclusi nella politica.

Annullamento della condivisione di un backup condiviso

Quando annulli la condivisione di una risorsa, il consumatore non può più utilizzarla per ripristinare un cluster. I consumatori saranno comunque in grado di accedere a tutti i cluster ripristinati dal backup condiviso.

Per annullare la condivisione di un backup condiviso di cui sei proprietario, devi rimuoverlo dalla condivisione di risorse. È possibile eseguire questa operazione utilizzando la AWS RAM console o AWS CLI.

Per annullare la condivisione di un backup condiviso di tua proprietà utilizzando la console AWS RAM

Consulta Aggiornamento di una condivisione di risorse in Guida per l'utente di AWS RAM .

Per annullare la condivisione di un backup condiviso di tua proprietà (comando)AWS RAM

Utilizza il comando disassociate-resource-share.

Per annullare la condivisione di un backup condiviso di cui sei proprietario (comando CloudHSM)

Utilizza il comando delete-resource-policy.

```
$ aws cloudhsmv2 delete-resource-policy --resource-arn <resource-arn>
```

Identificazione di un backup condiviso

I consumatori possono identificare un backup condiviso con loro utilizzando la console CloudHSM e. AWS CLI

Per identificare i backup condivisi con te utilizzando la console CloudhSM

1. Apri la AWS CloudHSM console a casa. https://console.aws.amazon.com/cloudhsm/
2. Per modificare la Regione AWS, usa il selettore della regione nell'angolo in alto a destra della pagina.
3. Nel pannello di navigazione, scegliere Backup.
4. Nella tabella, scegli la scheda Backup condivisi.

Per identificare i backup condivisi con te utilizzando AWS CLI

Utilizzate il comando describe-backups con il `--shared` parametro per restituire i backup condivisi con voi.

Autorizzazioni per i backup condivisi

Autorizzazioni per i proprietari

I proprietari dei backup possono descrivere e gestire un backup condiviso e utilizzarlo per ripristinare un cluster.

Autorizzazioni per gli utenti

Gli utenti di backup non possono modificare un backup condiviso, ma possono descriverlo e utilizzarlo per ripristinare un cluster.

Fatturazione e misurazione

Non sono previsti costi aggiuntivi per la condivisione dei backup.

Cluster clonati in AWS CloudHSM

Utilizza la CLI di CloudHSM per sincronizzare un cluster in una regione remota, se il cluster in quella regione è stato originariamente creato dal backup di un cluster in un'altra regione. Supponiamo che tu abbia copiato un cluster in un'altra regione (destinazione) e successivamente desideri sincronizzare le modifiche dal cluster originale (origine). In scenari come questo, si utilizzano i comandi key replicate e user replicate per sincronizzare i cluster. Se non hai installato CloudHSM CLI, consulta le istruzioni in. ???

Argomenti correlati

- replica utente
- replica chiave
- Copiare un backup tra regioni

AWS CloudHSM Risorse per tag

Un tag è un'etichetta che assegni a una AWS risorsa. Puoi assegnare i tag ai cluster AWS CloudHSM . Ogni tag è formato da una chiave e un valore di tag, entrambi definiti da te. Ad esempio, la chiave del tag potrebbe essere Centro di costo e il valore potrebbe essere 12345. Le chiavi del tag devono essere univoche per ciascun cluster.

Puoi utilizzare i tag per scopi diversi. Un uso comune è la categorizzazione e il monitoraggio dei costi di AWS . Puoi applicare i tag che rappresentano categorie di business (come centri di costo, nomi di applicazioni o proprietari) per organizzare i costi tra più servizi. Quando aggiungi tag alle tue AWS risorse, AWS genera un rapporto sull'allocazione dei costi con utilizzo e costi aggregati per tag. È possibile utilizzare questo rapporto per visualizzare i AWS CloudHSM costi in termini di progetti o applicazioni, anziché visualizzare tutti i AWS CloudHSM costi come un'unica voce.

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 .

È possibile utilizzare la AWS CloudHSM console o uno degli AWS SDKs strumenti da riga di comando per aggiungere, aggiornare, elencare e rimuovere tag.

Argomenti

- Aggiungi o aggiorna i tag per AWS CloudHSM le risorse
- Elenca i tag per AWS CloudHSM le risorse
- Rimuovi i tag dalle AWS CloudHSM risorse

Aggiungi o aggiorna i tag per AWS CloudHSM le risorse

Puoi aggiungere o aggiornare i tag dalla console AWS CloudHSM, da AWS Command Line Interface (AWS CLI) o dall'API AWS CloudHSM .

Per aggiungere o aggiornare i tag (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Scegliere il cluster per cui effettuare il tagging.
3. Scegliere Tags (Tag).
4. Per aggiungere un tag, procedere come segue:

a. Scegliere Edit Tag (Modifica tag), quindi Add Tag (Aggiungi tag).

b. Per Key (Chiave), digitare una chiave per il tag.

c. (Facoltativo) Per Value (Valore), digitare un valore per il tag.

d. Scegli Save (Salva).

5. Per aggiornare un tag, procedere come segue:

a. Scegliere Edit Tag (Modifica tag).

> **Note**
>
> Se si aggiorna la chiave del tag di un tag esistente, la console elimina il tag esistente e ne crea uno nuovo.

b. Digitare il nuovo valore del tag.

c. Scegli Save (Salva).

Per aggiungere o aggiornare i tag (AWS CLI)

1. Al prompt dei comandi, eseguire il comando tag-resource specificando i tag e l'ID del cluster per cui si sta effettuando il tagging. Se non si conosce l'ID del cluster, eseguire il comando describe-clusters.

```
$ aws cloudhsmv2 tag-resource --resource-id <cluster ID> \
                              --tag-list Key="<tag key>",Value="<tag value>"
```

2. Per aggiornare i tag, utilizzare lo stesso comando, ma specificando una chiave del tag esistente. Se si specifica un nuovo valore del tag per un tag esistente, il tag viene sovrascritto con il nuovo valore.

Per aggiungere o aggiornare tag (AWS CloudHSM API)

- Inviare una richiesta TagResource. Specificare i tag e l'ID del cluster per cui si sta effettuando il tagging.

Elenca i tag per AWS CloudHSM le risorse

È possibile elencare i tag per un cluster dalla AWS CloudHSM console AWS CLI, dall'API o dall' AWS CloudHSM API.

Per elencare i tag (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.
2. Scegliere il cluster di cui elencare i tag.
3. Scegliere Tags (Tag).

Per elencare i tag (AWS CLI)

- Al prompt dei comandi, eseguire il comando list-tags specificando l'ID del cluster di cui elencare i tag. Se non si conosce l'ID del cluster, eseguire il comando describe-clusters.

```
$ aws cloudhsmv2 list-tags --resource-id <cluster ID>
{
    "TagList": [
        {
            "Key": "Cost Center",
            "Value": "12345"
        }
    ]
}
```

Per elencare i tag (AWS CloudHSM API)

- Invia una richiesta ListTags, specificando l'ID del cluster di cui elencare i tag.

Rimuovi i tag dalle AWS CloudHSM risorse

Puoi rimuovere i tag da un AWS CloudHSM cluster utilizzando la AWS CloudHSM console AWS CLI, l'o l' AWS CloudHSM API.

Per rimuovere i tag (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.

2. Scegliere il cluster di cui rimuovere i tag.
3. Scegliere Tags (Tag).
4. Scegliere Edit Tag (Modifica tag), quindi scegliere Remove tag (Rimuovi tag) per il tag da rimuovere.
5. Scegli Save (Salva).

Per rimuovere i tag (AWS CLI)

- Al prompt dei comandi, eseguire il comando untag-resource specificando le chiavi dei tag da rimuovere e l'ID del cluster di cui rimuovere i tag. Quando usi il AWS CLI per rimuovere i tag, specifica solo le chiavi dei tag, non i valori dei tag.

```
$ aws cloudhsmv2 untag-resource --resource-id <cluster ID> \
                                --tag-key-list "<tag key>"
```

Per rimuovere i tag (AWS CloudHSM API)

- Invia una UntagResourcerichiesta nell' AWS CloudHSM API, specificando l'ID del cluster e i tag che stai rimuovendo.

AWS CloudHSM strumenti da riga di comando

Oltre a AWS Command Line Interface (AWS CLI) che usi per gestire le tue risorse AWS, AWS CloudHSM offre strumenti da riga di comando per creare e gestire utenti e chiavi dei moduli di sicurezza hardware (HSM) sul tuo. HSMs In AWS CloudHSM, usi la familiare CLI per gestire il cluster e gli strumenti a riga di comando CloudHSM per gestire il tuo HSM.

Questi sono i vari strumenti a riga di comando:

Per gestire e raggruppare HSMs

HSMv2 Comandi cloud AWS CLI e HSM2 PowerShell cmdlet nel modulo Shell AWSPower

- Questi strumenti ottengono, creano, eliminano ed etichettano i AWS CloudHSM cluster e: HSMs
- Per utilizzare i comandi nei HSMv2 comandi Cloud nella CLI, devi installarli e configurarli. AWS CLI
- HSM2 PowerShell I cmdlet nel modulo AWSPower Shell sono disponibili in un modulo Windows e in un PowerShell modulo Core PowerShell multipiattaforma.

Gestire gli utenti HSM

CLI CloudhSM

- Utilizza la CLI di CloudhSM per creare utenti, eliminare utenti, elencare utenti, modificare le password degli utenti e aggiornare l'autenticazione a più fattori (MFA) degli utenti. Non è incluso nel software client AWS CloudHSM. Per indicazioni sull'installazione di questo strumento, vedi Installare e configurare la CLI di CloudHSM.

Strumenti d'aiuto

Due strumenti ti aiutano a utilizzare strumenti e librerie software: AWS CloudHSM

- Lo strumento di configurazione aggiorna i file di configurazione del client CloudHSM. Ciò consente AWS CloudHSM di sincronizzarli HSMs in un cluster.

  AWS CloudHSM offre due versioni principali e Client SDK 5 è l'ultima. Offre una serie di vantaggi rispetto a Client SDK 3 (la serie precedente).
- pkpspeed misura le prestazioni dell'hardware HSM indipendentemente dalle librerie di software.

Strumenti per le versioni precedenti SDKs

Utilizza lo strumento di gestione delle chiavi (KMU) per creare, eliminare, importare ed esportare chiavi simmetriche e coppie di chiavi asimmetriche:

- key_mgmt_util. Questo strumento è incluso nel software del client AWS CloudHSM .

  Utilizza lo strumento di gestione CloudHSM (CMU) per creare ed eliminare utenti HSM, inclusa l'implementazione dell'autenticazione del quorum delle attività di gestione degli utenti
- cloudhsm_mgmt_util. Questo strumento è incluso nel software del client AWS CloudHSM .

I seguenti argomenti descrivono ulteriormente gli strumenti da riga di comando disponibili per la gestione e l'utilizzo. AWS CloudHSM

Argomenti

- AWS CloudHSM strumento di configurazione
- AWS CloudHSM Interfaccia a riga di comando (CLI)
- AWS CloudHSM Utilità di gestione (CMU)
- AWS CloudHSM Utilità di gestione delle chiavi (KMU)

AWS CloudHSM strumento di configurazione

AWS CloudHSM sincronizza automaticamente i dati tra tutti i moduli di sicurezza hardware (HSM) di un cluster. Lo strumento configure aggiorna i dati dell'HSM nei file di configurazione utilizzati dal meccanismo di sincronizzazione. Consente configure di aggiornare i dati HSM prima di utilizzare gli strumenti della riga di comando, in particolare quando le impostazioni all' HSMs interno del cluster sono state modificate.

AWS CloudHSM include due versioni principali di Client SDK:

- Client SDK 5: questo è il nostro Client SDK più recente e quello predefinito. Per informazioni sui benefici e i vantaggi che offre, vedi Vantaggi di AWS CloudHSM Client SDK 5.
- Client SDK 3: Questo è il nostro vecchio Client SDK. Include un set completo di componenti per la compatibilità delle applicazioni basate su piattaforme e linguaggi e strumenti di gestione.

Per istruzioni sulla migrazione da Client SDK 3 a Client SDK 5, consulta. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Argomenti

- AWS CloudHSM Strumento di configurazione di Client SDK 5
- AWS CloudHSM Strumento di configurazione Client SDK 3

AWS CloudHSM Strumento di configurazione di Client SDK 5

Utilizza lo strumento di configurazione AWS CloudHSM Client SDK 5 per aggiornare i file di configurazione lato client.

Ogni componente di Client SDK 5 include uno strumento di configurazione con un designatore del componente nel nome del file dello strumento di configurazione. Ad esempio, la libreria PKCS #11 per Client SDK 5 include uno strumento di configurazione denominato `configure-pkcs11` su Linux o `configure-pkcs11.exe` Windows.

Argomenti

- AWS CloudHSM Sintassi di configurazione di Client SDK 5
- AWS CloudHSM Parametri di configurazione di Client SDK 5
- AWS CloudHSM Esempi di configurazione di Client SDK 5
- Provider OpenSSL Bootstrap
- Configurazioni avanzate per lo strumento di configurazione del Client SDK 5
- AWS CloudHSM Argomenti correlati a Client SDK 5

AWS CloudHSM Sintassi di configurazione di Client SDK 5

La tabella seguente illustra la sintassi dei file di AWS CloudHSM configurazione per Client SDK 5. Per ulteriori informazioni sui parametri, consulta the section called "Parameters".

PKCS #11

```
Usage: configure-pkcs11[ .exe ] [OPTIONS]

Options:
      --disable-certificate-storage
          Disables Certificate Storage
      --enable-certificate-storage
          Enables Certificate Storage
  -a <HSM ENI IP>...
          The address of the HSM instance
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
          Disables key availability check during key use
      --enable-key-availability-check
```

```
          Enables key availability check during key use
      --disable-validate-key-at-init
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
 [possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
  -h, --help
          Print help
```

OpenSSL

```
Usage: configure-dyn[ .exe ] [OPTIONS]

Options:
  -a <HSM ENI IP>...
          The address of the HSM instance
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
          Disables key availability check during key use
      --enable-key-availability-check
          Enables key availability check during key use
      --disable-validate-key-at-init
```

```
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
 [possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
  -h, --help
          Print help
```

KSP

```
Usage: configure-ksp.exe [OPTIONS]

Options:
  -a <HSM ENI IP>...
          The address of the HSM instance
      --server-client-cert-file <CLIENT CERTIFICATE FILE>
          The client certificate used for TLS client-server mutual authentication
      --server-client-key-file <CLIENT KEY FILE>
          The client private key used for TLS client-server mutual authentication
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
          Disables key availability check during key use
      --enable-key-availability-check
```

```
          Enables key availability check during key use
      --disable-validate-key-at-init
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
 [possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
      --enable-sdk3-compatibility-mode
          Enables key file usage for KSP
      --disable-sdk3-compatibility-mode
          Disables key file usage for KSP
  -h, --help
          Print help
```

JCE

```
Usage: configure-jce[ .exe ] [OPTIONS]

Options:
  -a <HSM ENI IP>...
          The address of the HSM instance
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
```

```
          Disables key availability check during key use
      --enable-key-availability-check
          Enables key availability check during key use
      --disable-validate-key-at-init
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
 [possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
  -h, --help
          Print help
```

CloudHSM CLI

```
Usage: configure-cli[ .exe ] [OPTIONS]

Options:
  -a <HSM ENI IP>...
          The address of the HSM instance
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
          Disables key availability check during key use
      --enable-key-availability-check
```

```
          Enables key availability check during key use
      --disable-validate-key-at-init
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
 [possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
  -h, --help
          Print help
```

OpenSSL Provider

```
Usage: configure-openssl-provider[ .exe ] [OPTIONS]

Options:
  -a <HSM ENI IP>...
          The address of the HSM instance
      --cluster-id <CLUSTER ID>
          The id of the cluster containing the HSM instance(s)
      --disable-key-availability-check
          Disables key availability check during key use
      --enable-key-availability-check
          Enables key availability check during key use
      --disable-validate-key-at-init
```

```
          Disables parameter validation during initialization of crypto operations
      --enable-validate-key-at-init
          Enables parameter validation during initialization of crypto operations
      --endpoint <ENDPOINT>
          Specify the AWS CloudHSM API Endpoint
      --region <REGION>
          The region of the cluster
      --hsm-ca-cert <HSM CA CERTIFICATE FILE>
          The HSM CA certificate file
      --log-type <LOG TYPE>
          The log type [possible values: term, file]
      --log-file <LOG FILE>
          The log file
      --log-level <LOG LEVEL>
          The logging level [possible values: error, warn, info, debug, trace]
      --log-rotation <LOG ROTATION>
          The log rotation interval [possible values: never, hourly, daily]
      --default-retry-mode <RETRY MODE>
          The default method of retry to use for certain non-terminal failures
[possible values: off, standard]
      --client-cert-hsm-tls-file <CLIENT CERTIFICATE HSM TLS FILE>
          The client certificate used for TLS client-hsm mutual authentication
      --client-key-hsm-tls-file <CLIENT KEY HSM TLS FILE>
          The client private key used for TLS client-hsm mutual authentication
  -h, --help
          Print help
```

AWS CloudHSM Parametri di configurazione di Client SDK 5

Di seguito è riportato un elenco di parametri per configurare AWS CloudHSM Client SDK 5.

-a ***<ENI IP address>***

Aggiunge l'indirizzo IP specificato ai file di configurazione di Client SDK 5. Inserire qualsiasi indirizzo IP ENI di un HSM dal cluster. Per ulteriori informazioni su come utilizzare questa opzione, vedi Bootstrap del Client SDK 5.

Obbligatorio: sì

--hsm-ca-cert ***<customerCA certificate file path>***

Percorso alla directory in cui è archiviato il certificato dell'autorità di certificazione (CA) utilizzato per connettere le istanze del client EC2 al cluster. Si tratta del file che crei quando inizializzi il cluster. Per impostazione predefinita, il sistema cerca questo file nella seguente posizione:

Linux

```
/opt/cloudhsm/etc/customerCA.crt
```

Windows

```
C:\ProgramData\Amazon\CloudHSM\customerCA.crt
```

Per ulteriori informazioni sull'inizializzazione del cluster o sull'inserimento del certificato, vedi ??? e ???.

Obbligatorio: no

--cluster-id ***<cluster ID>***

Effettua una chiamata `DescribeClusters` per trovare tutti gli indirizzi IP a interfaccia di rete elastica (ENI) dell'HSM nel cluster associati all'ID del cluster. Il sistema aggiunge gli indirizzi IP ENI ai file di configurazione. AWS CloudHSM

> **Note**
>
> Se utilizzi il `--cluster-id` parametro da un'istanza EC2 all'interno di un VPC che non ha accesso alla rete Internet pubblica, devi creare un endpoint VPC di interfaccia con cui connetterti. AWS CloudHSM Per ulteriori informazioni su endpoint VPC, vedi ???.

Obbligatorio: no

--endpoint ***<endpoint>***

Specificare l'endpoint AWS CloudHSM API utilizzato per effettuare la chiamata. `DescribeClusters` È necessario impostare questa opzione insieme a `--cluster-id`.

Obbligatorio: no

--region ***<region>***

Specifica la regione del cluster. È necessario impostare questa opzione insieme a `--cluster-id`.

Se non indichi il parametro `--region`, il sistema sceglie la regione tentando di leggere le variabili di ambiente `AWS_DEFAULT_REGION` o `AWS_REGION`. Se queste variabili non sono impostate, il sistema controlla la regione associata al tuo profilo indicata nel tuo file di AWS Config (generalmente `~/.aws/config`) a meno che non sia stato specificato un file diverso nella variabile di ambiente `AWS_CONFIG_FILE`. Se non è stata impostata nessuna delle variabili precedenti, il sistema utilizza la regione `us-east-1` per impostazione predefinita.

Obbligatorio: no

-- -archivio client-cert-hsm-tls ***<client certificate hsm tls path>***

Percorso del certificato client utilizzato per l'autenticazione reciproca TLS Client-HSM.

Utilizza questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-key-hsm-tls-file`.

Obbligatorio: no

-- -file client-key-hsm-tls ***<client key hsm tls path>***

Percorso della chiave client utilizzata per l'autenticazione reciproca TLS Client-HSM.

Utilizza questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-cert-hsm-tls-file`.

Obbligatorio: no

--log-level ***<error | warn | info | debug | trace>***

Specifica il livello di log minimo che il sistema deve scrivere nel file di log. Ogni livello include i livelli precedenti, con errore come livello minimo e traccia il livello massimo. Ciò significa che se si specificano errori, il sistema scrive solo gli errori nel log. Se si specifica tracciamento, il sistema scrive errori, avvisi, messaggi informativi (info) e di debug nel log. Per ulteriori informazioni, vedi Logging con Client SDK 5.

Obbligatorio: no

--log-rotation ***<daily | weekly>***

Specifica la frequenza con cui il sistema ruota i log. Per ulteriori informazioni, vedi Logging con Client SDK 5.

Obbligatorio: no

--file-log ***<file name with path>***

Specifica dove il sistema scriverà il file di log. Per ulteriori informazioni, vedi Logging con Client SDK 5.

Obbligatorio: no

--tipo-log ***<term | file>***

Specifica se il sistema scriverà il log su un file o su un terminale. Per ulteriori informazioni, vedi Logging con Client SDK 5.

Obbligatorio: no

-h | --aiuto

Visualizza aiuto.

Obbligatorio: no

--disable-key-availability-check

Contrassegna per disabilitare il quorum per la disponibilità delle chiavi. Usa questo flag per indicare che AWS CloudHSM devi disabilitare il quorum di disponibilità delle chiavi e puoi usare le chiavi che esistono solo su un HSM nel cluster. Per ulteriori informazioni sull'utilizzo di questo flag per impostare il quorum per la disponibilità delle chiavi, vedi ???.

Obbligatorio: no

--enable-key-availability-check

Contrassegna per abilitare il quorum della disponibilità delle chiavi. Utilizzate questo flag per indicare che AWS CloudHSM dovete utilizzare il quorum di disponibilità delle chiavi e non consentirvi di utilizzare le chiavi finché tali chiavi non sono presenti su due del cluster. HSMs Per ulteriori informazioni sull'utilizzo di questo flag per impostare il quorum per la disponibilità delle chiavi, vedi ???.

Abilitata come impostazione predefinita.

Obbligatorio: no

-- -init disable-validate-key-at

Migliora le prestazioni specificando che è possibile saltare una chiamata di inizializzazione per verificare le autorizzazioni su una chiave per le chiamate successive. Utilizza questa soluzione con cautela.

Background: alcuni meccanismi della libreria PKCS #11 supportano operazioni in più parti in cui una chiamata di inizializzazione verifica se è possibile utilizzare la chiave per chiamate successive. Ciò richiede una chiamata di verifica all'HSM, che aggiunge latenza all'operazione complessiva. Questa opzione consente di disabilitare la chiamata successiva e potenzialmente di migliorare le prestazioni.

Obbligatorio: no

-- -init enable-validate-key-at

Specifica che è necessario utilizzare una chiamata di inizializzazione per verificare le autorizzazioni su una chiave per le chiamate successive. Questa è l'opzione predefinita. Usa `enable-validate-key-at-init` per riprendere queste chiamate di inizializzazione dopo avere usato `disable-validate-key-at-init` per sospenderle.

Obbligatorio: no

AWS CloudHSM Esempi di configurazione di Client SDK 5

Questi esempi mostrano come utilizzare lo strumento di configurazione per AWS CloudHSM Client SDK 5.

Bootstrap del Client SDK 5

Example

In questo esempio viene utilizzato il parametro `-a` per aggiornare i dati HSM per il client SDK 5. Per utilizzare il `-a` parametro, è necessario disporre dell'indirizzo IP di uno dei componenti HSMs del cluster.

PKCS #11 library

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 -a <HSM IP addresses>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" -a <HSM IP
 addresses>
```

OpenSSL Dynamic Engine

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn -a <HSM IP addresses>
```

OpenSSL Dynamic Engine Provider

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP addresses>
```

Key Storage Provider (KSP)

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" -a <HSM IP
 addresses>
```

JCE provider

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce -a <HSM IP addresses>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Utilizzate lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" -a <HSM IP
 addresses>
```

CloudHSM CLI

Per effettuare il bootstrap di un'istanza EC2 Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the
 HSMs>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI
 IPv4 / IPv6 addresses of the HSMs>
```

> **Note**
>
> è possibile utilizzare il parametro `--cluster-id` al posto di `-a <HSM_IP_ADDRESSES>`. Per visualizzare i requisiti per l'utilizzo di `--cluster-id`, vedi AWS CloudHSM Strumento di configurazione di Client SDK 5.

Per ulteriori informazioni sul parametro `-a`, vedi the section called "Parameters".

Specifica cluster, regione ed endpoint per Client SDK 5

Example

Questo esempio utilizza il parametro `cluster-id` per avviare Client SDK 5 effettuando una chiamata `DescribeClusters`.

PKCS #11 library

Per effettuare il bootstrap di un'istanza EC2 Linux per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --cluster-id <cluster-1234567>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --cluster-
id <cluster-1234567>
```

OpenSSL Dynamic Engine

Per effettuare il bootstrap di un'istanza EC2 Linux per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --cluster-id <cluster-1234567>
```

Key Storage Provider (KSP)

Per effettuare il bootstrap di un'istanza EC2 Windows per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --cluster-
id <cluster-1234567>
```

JCE provider

Per effettuare il bootstrap di un'istanza EC2 Linux per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-jce --cluster-id <cluster-1234567>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --cluster-
id <cluster-1234567>
```

CloudHSM CLI

Per effettuare il bootstrap di un'istanza EC2 Linux per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli --cluster-id <cluster-1234567>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per Client SDK 5 con **`cluster-id`**

- Utilizzate l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --cluster-
id <cluster-1234567>
```

È possibile utilizzare i parametri `--region` e `--endpoint` in combinazione con il parametro `cluster-id` per specificare in che modo il sistema effettua la chiamata `DescribeClusters`. Ad esempio, se la regione del cluster è diversa da quella configurata come impostazione predefinita della CLI di AWS, è necessario utilizzare il parametro `--region` per utilizzare tale regione. Inoltre, hai la possibilità di specificare l'endpoint AWS CloudHSM API da utilizzare per la chiamata, cosa che potrebbe essere necessaria per varie configurazioni di rete, come l'utilizzo di endpoint di interfaccia VPC che non utilizzano il nome host DNS predefinito per. AWS CloudHSM

PKCS #11 library

Per effettuare il bootstrap di un'istanza EC2 Linux con un endpoint e una regione personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel cluster con una regione e un endpoint personalizzati.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --cluster-id <cluster-1234567> --
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Per effettuare il bootstrap di un'istanza EC2 Windows con un endpoint e una regione

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --cluster-
id <cluster-1234567>--region <us-east-1> --endpoint <https://cloudhsmv2.us-
east-1.amazonaws.com>
```

OpenSSL Dynamic Engine

Per effettuare il bootstrap di un'istanza EC2 Linux con un endpoint e una regione personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --cluster-id <cluster-1234567> --
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Key Storage Provider (KSP)

Per effettuare il bootstrap di un'istanza EC2 Windows con un endpoint e una regione

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --cluster-
id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-
east-1.amazonaws.com>
```

JCE provider

Per effettuare il bootstrap di un'istanza EC2 Linux con un endpoint e una regione personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
$ sudo /opt/cloudhsm/bin/configure-jce --cluster-id <cluster-1234567> --
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Per effettuare il bootstrap di un'istanza EC2 Windows con un endpoint e una regione

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --cluster-
id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-
east-1.amazonaws.com>
```

CloudHSM CLI

Per effettuare il bootstrap di un'istanza EC2 Linux con un endpoint e una regione personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
$ sudo /opt/cloudhsm/bin/configure-cli --cluster-id <cluster-1234567> --
region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Per effettuare il bootstrap di un'istanza EC2 Windows con un endpoint e una regione

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --cluster-
id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-
east-1.amazonaws.com>
```

Per ulteriori informazioni sui parametri `--cluster-id`, `--region` e `--endpoint`, vedi the section called “Parameters”.

Aggiorna il certificato e la chiave del client per l'autenticazione reciproca TLS Client-HSM

Example

Questo esempio mostra come utilizzare i `--client-key-hsm-tls-file` parametri `--client-cert-hsm-tls-file` and per riconfigurare SSL specificando una chiave personalizzata e un certificato SSL per AWS CloudHSM

PKCS #11 library

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
$ sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

OpenSSL Dynamic Engine

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Key Storage Provider (KSP)

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

JCE provider

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

CloudHSM CLI

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Linux

1. Copia la chiave e il certificato nella directory appropriata.

```
$ sudo cp ssl-client.pem </opt/cloudhsm/etc>
sudo cp ssl-client.key </opt/cloudhsm/etc>
```

2. Utilizza lo strumento di configurazione per specificare `ssl-client.pem` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --client-cert-hsm-tls-file </opt/cloudhsm/etc/ssl-client.pem> \
            --client-key-hsm-tls-file </opt/cloudhsm/etc/ssl-client.key>
```

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca TLS Client-HSM con Client SDK 5 su Windows

1. Copia la chiave e il certificato nella directory appropriata.

```
cp ssl-client.pem <C:\ProgramData\Amazon\CloudHSM\ssl-client.pem>
cp ssl-client.key <C:\ProgramData\Amazon\CloudHSM\ssl-client.key>
```

2. Con un PowerShell interprete, usa lo strumento di configurazione per specificare e. `ssl-client.pem` `ssl-client.key`

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --client-cert-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.pem> `
            --client-key-hsm-tls-file <C:\ProgramData\Amazon\CloudHSM\ssl-
client.key>
```

Per ulteriori informazioni sui parametri `--client-cert-hsm-tls-file` e `--client-key-hsm-tls-file`, vedi the section called “Parameters”.

Disattiva le impostazioni di durabilità delle chiavi del client

Example

Questo esempio utilizza il parametro `--disable-key-availability-check` per disabilitare le impostazioni di durabilità delle chiavi del client. Per eseguire un cluster con un singolo HSM, è necessario disabilitare le impostazioni di durabilità delle chiavi del client.

PKCS #11 library

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --disable-
key-availability-check
```

OpenSSL Dynamic Engine

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --disable-key-availability-check
```

OpenSSL Dynamic Engine Provider

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --disable-key-availability-check
```

Key Storage Provider (KSP)

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-key-availability-check
```

JCE provider

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-jce --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-key-availability-check
```

CloudHSM CLI

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Linux

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
$ sudo /opt/cloudhsm/bin/configure-cli --disable-key-availability-check
```

Per disabilitare la durabilità delle chiavi del client per Client SDK 5 su Windows

- Utilizza lo strumento di configurazione per disabilitare le impostazioni di durabilità delle chiavi del client.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --disable-
key-availability-check
```

Per ulteriori informazioni sul parametro `--disable-key-availability-check`, vedi the section called "Parameters".

Gestione delle opzioni di log

Example

Client SDK 5 utilizza i parametri `log-file`, `log-levellog-rotation`, e `log-type` per gestire i log.

> **Note**
>
> Per configurare il tuo SDK per ambienti serverless come AWS Fargate o AWS Lambda, ti consigliamo di configurare il tipo di log su. AWS CloudHSM `term` I log del client verranno inviati `stderr` e acquisiti nel gruppo di log CloudWatch Logs configurato per quell'ambiente.

PKCS #11 library

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Linux

  ```
  /opt/cloudhsm/run/cloudhsm-pkcs11.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-pkcs11.log
  ```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  $ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  $ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-type file --log-file <file name
   with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  $ sudo /opt/cloudhsm/bin/configure-pkcs11 --log-type term --log-level info
  ```

OpenSSL Dynamic Engine

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Linux
  ```

```
stderr
```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  $ sudo /opt/cloudhsm/bin/configure-dyn --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  $ sudo /opt/cloudhsm/bin/configure-dyn --log-type file --log-file <file name
   with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  $ sudo /opt/cloudhsm/bin/configure-dyn --log-type term --log-level info
  ```

OpenSSL Dynamic Engine Provider

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Linux

  ```
  stderr
  ```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  $ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  $ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-type file --log-file <file name with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  $ sudo /opt/cloudhsm/bin/configure-openssl-provider --log-type term --log-level info
  ```

Key Storage Provider (KSP)

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-ksp.log
  ```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-type file --log-file <file name with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --log-type
   term --log-level info
  ```

JCE provider

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Linux

  ```
  /opt/cloudhsm/run/cloudhsm-jce.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-jce.log
  ```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  $ sudo /opt/cloudhsm/bin/configure-jce --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  $ sudo /opt/cloudhsm/bin/configure-jce --log-type file --log-file <file name
   with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  $ sudo /opt/cloudhsm/bin/configure-jce --log-type term --log-level info
  ```

CloudHSM CLI

Posizione di log predefinita

- Se non si specifica una posizione per il file, il sistema scrive i log nella seguente posizione predefinita:

  Linux

  ```
  /opt/cloudhsm/run/cloudhsm-cli.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-cli.log
  ```

Per configurare il livello di log e lasciare le altre opzioni di log impostate sui valori predefiniti

- ```
  $ sudo /opt/cloudhsm/bin/configure-cli --log-level info
  ```

Per configurare le opzioni di log dei file

- ```
  $ sudo /opt/cloudhsm/bin/configure-cli --log-type file --log-file <file name
   with path> --log-rotation daily --log-level info
  ```

Per configurare le opzioni di log del terminale

- ```
  $ sudo /opt/cloudhsm/bin/configure-cli --log-type term --log-level info
  ```

Per ulteriori informazioni sui parametri `log-file`, `log-level`, `log-rotation` `elog-type`, vedi the section called "Parameters".
  ```

Inserisci il certificato di emissione per Client SDK 5

Example

Questo esempio utilizza il parametro `--hsm-ca-cert` per aggiornare la posizione del certificato di emissione per Client SDK 5.

PKCS #11 library

Per inserire il certificato di emissione su Linux per Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --hsm-ca-
cert <customerCA certificate file>
```

OpenSSL Dynamic Engine

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-dyn --hsm-ca-cert <customerCA certificate
 file>
```

OpenSSL Dynamic Engine Provider

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --hsm-ca-cert <customerCA
 certificate file>
```

Key Storage Provider (KSP)

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --hsm-ca-
cert <customerCA certificate file>
```

JCE provider

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-jce --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --hsm-ca-
cert <customerCA certificate file>
```

CloudHSM CLI

Per posizionare il certificato di emissione su Linux per il Client SDK 5

- Usa lo strumento di configurazione per specificare la posizione del certificato di emissione.

```
$ sudo /opt/cloudhsm/bin/configure-cli --hsm-ca-cert <customerCA certificate
 file>
```

Per posizionare il certificato di emissione su Windows per il Client SDK 5

- Utilizza lo strumento di configurazione per specificare una posizione per il certificato di emissione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" --hsm-ca-cert <customerCA certificate file>
```

Per ulteriori informazioni sul parametro `--hsm-ca-cert`, vedi the section called "Parameters".

Provider OpenSSL Bootstrap

Usa lo configure-openssl-provider strumento per avviare l'installazione del provider OpenSSL e collegarla al cluster. AWS CloudHSM

Per avviare il provider OpenSSL

1. Esegui il configure-openssl-provider comando con l'indirizzo IP di un HSM nel tuo cluster:

   ```
   $ sudo /opt/cloudhsm/bin/configure-openssl-provider -a <HSM IP address>
   ```

   Sostituiscilo *<HSM IP address>* con l'indirizzo IP di qualsiasi HSM del cluster.
2. Verifica la configurazione controllando che il provider OpenSSL possa connettersi al tuo cluster:

   ```
   $ openssl list -providers -provider-path /opt/cloudhsm/lib -provider cloudhsm
   ```

Per ulteriori informazioni sui parametri di configurazione, consulta. the section called "Parameters"

Configurazioni avanzate per lo strumento di configurazione del Client SDK 5

Lo strumento di configurazione AWS CloudHSM Client SDK 5 include configurazioni avanzate che non fanno parte delle funzionalità generali utilizzate dalla maggior parte dei clienti. Le configurazioni avanzate offrono funzionalità aggiuntive.

> **Important**
>
> Dopo avere apportato le modifiche alla configurazione, è necessario riavviare l'applicazione affinché le modifiche abbiano effetto.

- Configurazioni avanzate per PKCS #11
  - Configurazione a più slot con libreria PKCS #11 per AWS CloudHSM
  - Riprova i comandi per la libreria PKCS #11 per AWS CloudHSM
- Configurazioni avanzate per OpenSSL
  - Riprova i comandi per OpenSSL per AWS CloudHSM
- Configurazioni avanzate per KSP
  - SDK3 modalità di compatibilità per Key Storage Provider (KSP) per AWS CloudHSM
- Configurazioni avanzate per JCE
  - Connessione a più AWS CloudHSM cluster con il provider JCE
  - Riprova i comandi per JCE per AWS CloudHSM
  - Estrazione delle chiavi tramite JCE per AWS CloudHSM
- Configurazioni avanzate per AWS CloudHSM Command Line Interface (CLI)
  - Connessione a più cluster con la CLI CloudhSM

AWS CloudHSM Argomenti correlati a Client SDK 5

Per ulteriori informazioni su AWS CloudHSM Client SDK 5, consulta i seguenti argomenti correlati.

- Operazione API DescribeClusters
- Descrivi-Cluster CLI AWS;
- Get-HSM2Cluster PowerShell cmdlet
- Bootstrap del Client SDK 5
- Provider OpenSSL Bootstrap
- AWS CloudHSM Endpoint VPC
- Gestione delle impostazioni chiave di durabilità di Client SDK 5
- Logging con Client SDK 5
- Configurazione MTL (consigliata)

AWS CloudHSM Strumento di configurazione Client SDK 3

Utilizza lo strumento di configurazione AWS CloudHSM Client SDK 3 per avviare il daemon client e configurare CloudHSM Management Utility (CMU).

Argomenti

- AWS CloudHSM Sintassi di configurazione di Client SDK 3
- AWS CloudHSM Parametri di configurazione di Client SDK 3
- AWS CloudHSM Esempi di configurazione di Client SDK 3
- AWS CloudHSM Argomenti relativi alla configurazione di Client SDK 3

AWS CloudHSM Sintassi di configurazione di Client SDK 3

La tabella seguente illustra la sintassi dei file di AWS CloudHSM configurazione per Client SDK 3.

```
configure -h | --help
          -a <ENI IP address>
          -m [-i <daemon_id>]
          --ssl --pkey <private key file> --cert <certificate file>
          --cmu <ENI IP address>
```

AWS CloudHSM Parametri di configurazione di Client SDK 3

Di seguito è riportato un elenco di parametri per configurare AWS CloudHSM Client SDK 3.

-h | --aiuto

Visualizza la sintassi del comando.

Obbligatorio: sì

-a ***<ENI IP address>***

Aggiunge l'indirizzo IP dell'interfaccia di rete elastica (ENI) HSM specificata ai file di configurazione AWS CloudHSM . Immettere l'indirizzo IP ENI di uno qualsiasi dei componenti HSMs del cluster. Non importa quale selezioni.

Per ottenere gli indirizzi IP ENI del cluster, utilizzare l' HSMs DescribeClustersoperazione, il AWS CLI comando describe-clusters o il cmdlet. Get-HSM2Cluster PowerShell

> **Note**
>
> Prima di eseguire il comando, arrestate il client. `-a` configure AWS CloudHSM Quindi, al termine del `-a` comando, riavvia il AWS CloudHSM client. Per ulteriori informazioni, vedi gli esempi.

Questo parametro modifica i seguenti file di configurazione:

- `/opt/cloudhsm/etc/cloudhsm_client.cfg`: utilizzato dal AWS CloudHSM client e da key_mgmt_util.
- `/opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg`: utilizzato da cloudhsm_mgmt_util.

All'avvio, il AWS CloudHSM client utilizza l'indirizzo IP ENI nel suo file di configurazione per interrogare il cluster e aggiornare il `cluster.info` file (`/opt/cloudhsm/daemon/1/cluster.info`) con gli indirizzi IP ENI corretti per tutti gli utenti del cluster. HSMs

Obbligatorio: sì

-m

Aggiorna gli indirizzi IP ENI HSM nel file di configurazione utilizzato da CMU.

> **Note**
>
> Il parametro `-m` è destinato all'uso con CMU di Client SDK 3.2.1 e versioni precedenti. Per CMU dal Client SDK 3.3.0 e versioni successive, vedi parametro `--cmu`, che semplifica il processo di aggiornamento dei dati HSM per CMU.

Quando si aggiorna il `-a` parametro configure e quindi si avvia il AWS CloudHSM client, il daemon client interroga il cluster e aggiorna `cluster.info` i file con gli indirizzi IP HSM corretti per tutti HSMs gli utenti del cluster. Eseguendo il comando `-m` configure, l'aggiornamento viene completato copiando gli indirizzi IP HSM da `cluster.info` al file di configurazione `cloudhsm_mgmt_util.cfg` utilizzato da cloudhsm_mgmt_util uses.

Assicurati di eseguire il `-a` configure comando e riavviare il AWS CloudHSM client prima di eseguire il comando. `-m` In questo modo, i dati copiati nel file `cloudhsm_mgmt_util.cfg` dal file `cluster.info` saranno completi e accurati.

Obbligatorio: sì

-i

Specifica un client daemon alternativo. Il valore predefinito rappresenta il client AWS CloudHSM .

Impostazione predefinita: 1

Obbligatorio: no

--ssl

Sostituisce la chiave e il certificato SSL del cluster con la chiave privata e il certificato specificati. Quando utilizzi questo parametro, i parametri `--pkey` e `--cert` sono obbligatori.

Obbligatorio: no

--pkey

Specifica la nuova chiave privata. Inserisci il percorso e il nome del file contenente la chiave privata.

Obbligatorio: Sì, se --ssl specificato. Altrimenti non deve essere utilizzato.

--certificato

Specifica il nuovo certificato. Inserisci il percorso e il nome del file contenente il certificato. Il certificato deve essere collegato al certificato `customerCA.crt`, ossia il certificato autofirmato utilizzato per inizializzare il cluster. Per ulteriori informazioni, vedi Inizializzazione del cluster.

Obbligatorio: Sì, se --ssl specificato. Altrimenti non deve essere utilizzato.

--cmu ***<ENI IP address>***

Combina i parametri `-a` e `-m` in un unico parametro. Aggiunge l'indirizzo IP HSM elastic network interface (ENI) specificato ai file di AWS CloudHSM configurazione, quindi aggiorna il file di configurazione CMU. Immettere un indirizzo IP da un qualsiasi modulo HSM del cluster. Per Client SDK 3.2.1 e versioni precedenti, vedi Utilizzo di CMU con Client SDK 3.2.1 e versioni precedenti.

Obbligatorio: sì

AWS CloudHSM Esempi di configurazione di Client SDK 3

Questi esempi mostrano come utilizzare lo configure strumento per AWS CloudHSM Client SDK 3.

Example: Aggiorna i dati HSM per il AWS CloudHSM client e key_mgmt_util

Questo esempio utilizza il `-a` parametro di per aggiornare i dati HSM configure per il client e key_mgmt_util. AWS CloudHSM Per utilizzare il `-a` parametro, è necessario disporre dell'indirizzo IP di uno dei componenti del cluster. HSMs Usa la console o la CLI AWS per ottenere l'indirizzo IP.

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la AWS CloudHSM console a https://console.aws.amazon.com/cloudhsm/casa.

2. Per modificare la regione AWS, utilizza l'apposito selettore nell'angolo in alto a destra della pagina.
3. Per aprire la pagina dei dettagli del cluster, nella tabella dei cluster, scegli l'ID del cluster.
4. Per ottenere l'indirizzo IP, vai alla HSMs scheda. Per IPv4 i cluster, scegli un indirizzo elencato sotto l' IPv4 indirizzo ENI. Per i cluster dual-stack, utilizzare l'ENI o l'indirizzo ENI IPv4 . IPv6

Per ottenere un indirizzo IP per un HSM ()AWS CLI

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters dalla AWS CLI. Nell'output del comando, l'indirizzo IP di HSMs sono i valori di `EniIp` and `EniIpV6` (se si tratta di un cluster dual-stack).

```
$ aws cloudhsmv2 describe-clusters
{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
                    "EniIpV6": "2600:113f:404:be09:310e:ed34:3412:f733",
...
```

: aggiornare i dati HSM

1. Prima di aggiornare il `-a` parametro, arrestate il client. AWS CloudHSM In questo modo vengono evitati conflitti che potrebbero verificarsi mentre configure modifica il file di configurazione del client. Se il client è già stato arrestato, questo comando non ha alcun effetto, pertanto è possibile utilizzarlo in uno script.

   Amazon Linux

   ```
   $ sudo stop cloudhsm-client
   ```

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

CentOS 8

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Windows

- Per client Windows 1.1.2+:

  ```
  C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
  ```

- Per client Windows 1.1.1 e versioni precedenti:

  Usa Ctrl + C nella finestra di comando in cui hai avviato il AWS CloudHSM client.

2. Questa fase impiega il parametro `-a` di configure per aggiungere l'indirizzo IP ENI `10.0.0.9` ai file di configurazione.

   Amazon Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   Amazon Linux 2

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   CentOS 7

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   CentOS 8

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   RHEL 7

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   RHEL 8

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo /opt/cloudhsm/bin/configure -a 10.0.0.9
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a 10.0.0.9
   ```

3. Quindi, riavvia il AWS CloudHSM client. Quando viene avviato, il client utilizza l'indirizzo IP ENI nel proprio file di configurazione per eseguire query sul cluster. Quindi, scrive nel `cluster.info` file gli indirizzi IP ENI HSMs di tutti i componenti del cluster.

   Amazon Linux

   ```
   $ sudo start cloudhsm-client
   ```

   Amazon Linux 2

   ```
   $ sudo service cloudhsm-client start
   ```

   CentOS 7

   ```
   $ sudo service cloudhsm-client start
   ```

   CentOS 8

   ```
   $ sudo service cloudhsm-client start
   ```

   RHEL 7

   ```
   $ sudo service cloudhsm-client start
   ```

   RHEL 8

   ```
   $ sudo service cloudhsm-client start
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo service cloudhsm-client start
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo service cloudhsm-client start
   ```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe
 C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Al termine del comando, i dati HSM utilizzati dal AWS CloudHSM client e da key_mgmt_util sono completi e accurati.

Example: Aggiorna i dati HSM per CMU dal client SDK 3.2.1 e versioni precedenti

Questo esempio usa il comando `-m` configure per copiare i dati HSM aggiornati dal file `cluster.info` al file `cloudhsm_mgmt_util.cfg` utilizzato da cloudhsm_mgmt_util. Utilizzalo con CMU fornito con Client SDK 3.2.1 e versioni precedenti.

- Prima di eseguire `-m`, arrestate il AWS CloudHSM client, eseguite il `-a` comando e riavviate il AWS CloudHSM client, come illustrato nell'esempio precedente. In questo modo, i dati copiati nel file `cloudhsm_mgmt_util.cfg` dal file `cluster.info` saranno completi e accurati.

  Linux

  ```
  $ sudo /opt/cloudhsm/bin/configure -m
  ```

  Windows

  ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -m
  ```

Example: aggiorna i dati HSM per CMU dal client SDK 3.3.0 e versioni successive

In questo esempio viene utilizzato il parametro `--cmu` di configure per aggiornare i dati HSM per CMU. Utilizzalo con CMU fornito con Client SDK 3.3.0 e versioni successive. Per ulteriori informazioni

sull'utilizzo della CMU, vedi Usare CloudHSM Management Utility (CMU) per gestire gli utenti e Usare CMU con Client SDK 3.2.1 e versioni precedenti.

- Utilizza il `--cmu` parametro per passare l'indirizzo IP di un HSM nel cluster.

  Linux

  ```
  $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
  ```

  Windows

  ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
  ```

AWS CloudHSM Argomenti relativi alla configurazione di Client SDK 3

Per ulteriori informazioni su AWS CloudHSM Client SDK 3, consulta i seguenti argomenti correlati.

- Configura AWS CloudHSM key_mgmt_util

AWS CloudHSM Interfaccia a riga di comando (CLI)

La CLI di CloudhSM aiuta gli amministratori a gestire gli utenti e gli utenti di criptovalute a gestire le chiavi nel proprio cluster in. AWS CloudHSM La CLI include strumenti che possono essere utilizzati per creare, eliminare ed elencare utenti, modificare le password degli utenti, aggiornare l'autenticazione a più fattori (MFA) dell'utente. Include anche comandi che generano, eliminano, importano ed esportano chiavi, ottengono e impostano attributi, trovano chiavi ed eseguono operazioni crittografiche.

Per un elenco dettagliato di utenti della CLI di CloudHSM, vedi Gestione degli utenti HSM con CLI CloudHSM. Per un elenco definito di attributi chiave per la CLI di CloudHSM, vedere. Attributi chiave per la CLI di CloudHSM Per informazioni su come utilizzare la CLI di CloudHSM per gestire le chiavi, consulta. Gestione delle chiavi con CloudHSM CLI

Per una guida rapida, vedi Nozioni di base su AWS CloudHSM Interfaccia a riga di comando (CLI). Per informazioni dettagliate sui comandi della CLI di CloudHSM ed esempi di utilizzo dei comandi, vedi Riferimento per i comandi della CLI di CloudHSM.

Argomenti

- AWS CloudHSM Piattaforme supportate dall'interfaccia a riga di comando (CLI)
- Nozioni di base su AWS CloudHSM Interfaccia a riga di comando (CLI)
- Modalità di comando nella CLI di CloudHSM
- Attributi chiave per la CLI di CloudHSM
- Configurazioni avanzate per CloudHSM CLI
- Riferimento per i comandi della CLI di CloudHSM

AWS CloudHSM Piattaforme supportate dall'interfaccia a riga di comando (CLI)

Questo argomento descrive le piattaforme Linux e Windows supportate dalla AWS CloudHSM CLI.

Supporto Linux

Piattaforme supportate	Architettura: x86_64	Architettura ARM
Amazon Linux 2	Sì	Sì
Amazon Linux 2023	Sì	Sì
Red Hat Enterprise Linux 8 (8.3+)	Sì	Sì
Red Hat Enterprise Linux 9 (9.2+)	Sì	Sì
Red Hat Enterprise Linux 10 (versione 10.0+)	Sì	Sì
Ubuntu 22.04 LTS	Sì	Sì
Ubuntu 24.04 LTS	Sì	Sì
Ubuntu 26.04 LTS	Sì	Sì

- SDK 5.16 è stata l'ultima versione a fornire il supporto per la piattaforma Ubuntu 20.04 LTS. Per ulteriori informazioni, consulta il sito Web di Ubuntu. https://ubuntu.com/blog/ubuntu-20-04-lts-end-of-life-standard-support-is-coming-to-an-end-heres-how-to-prepare
- SDK 5.12 è stata l'ultima versione a fornire il supporto per la piattaforma CentOS 7 (7.8+). Per ulteriori informazioni, vedi il sito web CentOS.
- SDK 5.12 è stata l'ultima versione a fornire il supporto per la piattaforma Red Hat Enterprise Linux 7 (7.8+). Per ulteriori informazioni, consultate il sito Web di Red Hat.
- SDK 5.4.2 è stata l'ultima versione a fornire il supporto per la piattaforma CentOS 8. Per ulteriori informazioni, vedi il sito web CentOS.

Supporto Windows

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019
- Microsoft Windows Server 2022
- Microsoft Windows Server 2025

Nozioni di base su AWS CloudHSM Interfaccia a riga di comando (CLI)

Con l'interfaccia a riga di comando (CLI) CloudHSM, puoi gestire gli utenti nel tuo cluster. AWS CloudHSM Usa questo argomento per iniziare con le attività di gestione degli utenti di base del modulo di sicurezza hardware (HSM), come la creazione di utenti, l'elenco degli utenti e la connessione della CLI di CloudHSM al cluster.

Argomenti

- Installa l'interfaccia a riga di comando di CloudHSM
- Usa l'interfaccia a riga di comando di CloudHSM

Installa l'interfaccia a riga di comando di CloudHSM

Usa i seguenti comandi per scaricare e installare l'interfaccia a riga di comando di CloudHSM per. AWS CloudHSM

Amazon Linux 2023

Amazon Linux 2023 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.x86_64.rpm
```

Amazon Linux 2023 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Amazon Linux 2 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-
latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.x86_64.rpm
```

Amazon Linux 2 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-cli-
latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

RHEL 10 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-
cli-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.x86_64.rpm
```

RHEL 10 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-cli-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

RHEL 9 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.x86_64.rpm
```

RHEL 9 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-cli-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

RHEL 8 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.x86_64.rpm
```

RHEL 8 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-cli-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-cli-latest.el8.aarch64.rpm
```

Ubuntu 26.04 LTS

Ubuntu 26.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-cli_latest_u26.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u26.04_amd64.deb
```

Ubuntu 26.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-cli_latest_u26.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u26.04_arm64.deb
```

Ubuntu 24.04 LTS

Ubuntu 24.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-
cli_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_amd64.deb
```

Ubuntu 24.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/cloudhsm-
cli_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Ubuntu 22.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-
cli_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_amd64.deb
```

Ubuntu 22.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/cloudhsm-cli_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u22.04_arm64.deb
```

Windows Server 2022

Per Windows Server 2022 su architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2019

Per Windows Server 2019 su architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

Windows Server 2016

Per Windows Server 2016 su architettura x86_64, apri PowerShell come amministratore ed esegui il comando seguente:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/AWSCloudHSMCLI-latest.msi -Outfile C:\AWSCloudHSMCLI-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMCLI-latest.msi /
quiet /norestart /log C:\client-install.txt' -Wait
```

Usa i seguenti comandi per configurare la CLI di CloudHSM.

Per eseguire il bootstrap di un'istanza EC2 Linux per Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
$ sudo /opt/cloudhsm/bin/configure-cli -a <The ENI IPv4 / IPv6 addresses of the
 HSMs>
```

Per effettuare il bootstrap di un'istanza EC2 Windows per il Client SDK 5

- Usa lo strumento di configurazione per specificare l'indirizzo IP degli HSM sul cluster.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI
 IPv4 / IPv6 addresses of the HSMs>
```

Usa l'interfaccia a riga di comando di CloudHSM

Utilizza i seguenti comandi per avviare e utilizzare la CLI di CloudHSM.

1. Utilizza il seguente comando per avviare la CLI di CloudHSM.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
   ```

2. Utilizza il comando login per effettuare la connessione al cluster. Tutti gli utenti possono utilizzare questo comando.

   Il comando dell'esempio seguente consente di accedere ad admin, che è l'account admin predefinito. Imposta questa password dell'utente una volta che avrai attivato il cluster.

```
aws-cloudhsm > login --username admin --role admin
```

Il sistema ti invita a inserire la tua password. Immetti la password e l'output mostra che il comando è stato eseguito con successo.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin",
    "role": "admin"
  }
}
```

3. Esegui il comando user list per elencare tutti gli utenti del cluster.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

4. Utilizza user create per creare un utente CU denominato **example_user**.

Puoi creare CU perché in un passaggio precedente hai effettuato l'accesso come utente admin. Solo gli utenti admin possono eseguire attività di gestione degli utenti, come la creazione e l'eliminazione di utenti e la modifica delle password di altri utenti.

```
aws-cloudhsm > user create --username example_user --role crypto-user
Enter password:
Confirm password:
{
 "error_code": 0,
 "data": {
   "username": "example_user",
   "role": "crypto-user"
 }
}
```

5. Utilizza user list per elencare tutti gli utenti del cluster.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "example_user",
        "role": "crypto_user",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
```

```
      }
    ]
  }
}
```

6. Usa il logout comando per disconnetterti dal cluster. AWS CloudHSM

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

7. Utilizza il comando quit per chiudere la CLI.

```
aws-cloudhsm > quit
```

Modalità di comando nella CLI di CloudHSM

Nella CLI di CloudHSM, puoi eseguire i comandi in due modi diversi: in modalità comando singolo e in modalità interattiva. La modalità interattiva è progettata per gli utenti e la modalità a comando singolo è progettata per gli script.

> **Note**
>
> Tutti i comandi funzionano sia in modalità interattiva che in modalità a comando singolo.

Modalità interattiva

Usa i seguenti comandi per avviare la modalità interattiva nella CLI di CloudHSM

Linux

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive
```

Quando si utilizza la CLI in modalità interattiva, è possibile accedere a un account utente utilizzando il comando login.

```
aws-cloudhsm > login --username <USERNAME> --role ROLE>
```

Per visualizzare un elenco di tutti i comandi della CLI di CloudHSM, esegui il seguente comando:

```
aws-cloudhsm > help
```

Per ottenere la sintassi di un comando della CLI di CloudHSM, esegui il seguente comando:

```
aws-cloudhsm >  help <command-name>
```

Per ottenere un elenco di utenti su HSMs, digitareuser list.

```
aws-cloudhsm > user list
```

Per terminare la sessione della CLI di CloudHSM, esegui il seguente comando:

```
aws-cloudhsm > quit
```

Modalità di comando singolo

> **Note**
>
> Quando si utilizza la modalità a comando singolo, è necessario evitare eventuali caratteri speciali nelle variabili di ambiente e negli argomenti della riga di comando che possono essere interpretati dalla shell.

Se usi la CLI di CloudHSM CLI con la modalità a comando singolo, devi impostare due variabili di ambiente per fornire le credenziali: PIN_CLOUDHSM e RUOLO_CLOUDHSM:

```
$ export CLOUDHSM_ROLE=admin
```

```
$ export CLOUDHSM_PIN=admin_username:admin_password
```

Una volta fatto ciò, puoi eseguire i comandi utilizzando le credenziali memorizzate nel tuo ambiente.

```
$ cloudhsm-cli user change-password --username alice --role crypto-user
Enter password:
Confirm password:
{
    "error_code": 0,
    "data": {
      "username": "alice",
      "role": "crypto-user"
    }
}
```

Attributi chiave per la CLI di CloudHSM

Questo argomento descrive come utilizzare la CLI CloudHSM per impostare gli attributi di una chiave. L'attributo di una chiave nella CLI di CloudHSM può definire la tipologia di una chiave, come può funzionare una chiave o come viene etichettata una chiave. Alcuni attributi definiscono caratteristiche uniche (ad esempio il tipo di chiave). Altri attributi possono essere impostati su vero o falso: la loro modifica attiva o disattiva una funzionalità della chiave.

Per esempi che mostrano come utilizzare gli attributi delle chiavi, vedi i comandi elencati sotto il comando principale La categoria chiave dell'interfaccia CLI di CloudHSM.

I seguenti argomenti forniscono ulteriori dettagli sugli attributi chiave di CloudHSM CLI.

Argomenti

- Attributi supportati per CloudHSM CLI
- Verifica il valore nella CLI di CloudHSM
- Argomenti correlati per CloudHSM CLI

Attributi supportati per CloudHSM CLI

Come best practice, imposta i valori solo per gli attributi che desideri rendere restrittivi. Se non specifichi un valore, la CLI di CloudHSM utilizza il valore predefinito specificato nella tabella sottostante.

La tabella seguente elenca gli attributi chiave, i valori possibili, i valori predefiniti e le note correlate per CloudHSM CLI. Una cella vuota nella colonna Valore indica che non vi è alcun valore predefinito specifico assegnato all'attributo.

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`always-sensitive`	Il valore è `True` se `sensitive` è sempre stato impostato su `True` e non è mai stato cambiato.	No	No
`check-value`	Valore di controllo della chiave. Per ulteriori informazioni, vedi Ulteriori dettagli.	No	No
`class`	Valori possibili: `secret-key` , `public-key` e `private-key` .	No	Sì
`curve`	Curva ellittica usata per generare la coppia di chiavi EC. Valori validi:`secp224r1` ,,,`secp256r1` , e `prime256v1` `secp384r1` `secp256k1` `secp521r1` `ed25519` `ed25519`è supportato solo su istanze	No	Impostabile con EC, non impostabile con RSA

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
	hsm2m.medium in modalità non FIPS.		
`decrypt`	Impostazione predefinita: `False`	Sì	Sì
`derive`	Impostazione predefinita: `False`	Derive può essere impostato su istanze hsm2m.medium. Non può essere impostato per le chiavi RSA sulle istanze hsm1.medium.	Sì
`destroyable`	Impostazione predefinita: `True`	Sì	Sì
`ec-point`	Per le chiavi EC, DER-encoding con valore ANSI X9.62 ECpoint «Q» in formato esadecimale. Per altri tipi di chiavi, questo attributo non esiste.	No	No
`encrypt`	Impostazione predefinita: `False`	Sì	Sì

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`extractable`	Impostazione predefinita: `True`	Può essere modificato da vero a falso su istanze hsm2m.medium, ma non da falso a vero. Non può essere modificato sulle istanze hsm1.medium.	Sì
`id`	Impostazione predefinita: Vuoto	l'id può essere impostato sulle istanze hsm2m.medium. Non può essere impostato su istanze hsm1.medium.	Sì
`key-length-bytes`	Necessario per generare una chiave AES. Valori validi: `16`, `24`, e `32` byte.	No	No
`key-type`	Valori possibili:`aes`,, e `rsa` `ec` `ml-dsa`	No	Sì
`label`	Impostazione predefinita: Vuoto	Sì	Sì

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`local`	Impostazione predefinita: `True` per le chiavi generate nell'HSM, `False` per le chiavi importate nell'HSM.	No	No
`modifiable`	Impostazione predefinita: `True`	Può essere modificato da vero a falso, ma non da falso a vero.	Sì
`mldsa-algorithm`	L' ML-DSA algoritmo utilizzato per la generazione di coppie di ML-DSA chiavi. Valori validi: `ML-DSA-44` , `ML-DSA-65` e `ML-DSA-87`	No	Sì
`modulus`	Il modulo utilizzato per creare una coppia di chiavi RSA. Per altri tipi di chiavi, questo attributo non esiste.	No	No
`modulus-size-bits`	Necessario per generare una coppia di chiavi RSA. Il valore minimo è `2048`.	No	Da impostare con RSA, non può essere impostato con EC

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`never-extractable`	Il valore è `True` se la modalità Estraibile non è mai stata impostata su `False`. Il valore è `False` se la modalità Estraibile è stata impostata su `True`.	No	No
`private`	Impostazione predefinita: `True`	No	Sì
`public-exponent`	Necessario per generare una coppia di chiavi RSA. Valore valido: Il valore deve essere un numero dispari maggiore o uguale a `65537`.	No	Da impostare con RSA, non può essere impostato con EC
`sensitive`	Impostazione predefinita: • Il valore è `True` per le chiavi AES e le chiavi EC e RSA private. • Il valore è `False` per le chiavi EC e RSA pubbliche.	No	Si può impostare su chiavi private, non può essere impostato su chiavi pubbliche.

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`sign`	Impostazione predefinita: • Il valore è `True` per le chiavi AES. • Il valore è `False` per le chiavi RSA ed EC.	Sì	Sì
`token`	Impostazione predefinita: `True`	Può essere modificato da falso a vero, ma non da vero a falso.	Sì
`trusted`	Impostazione predefinita: `False`	Solo gli utenti amministratori possono impostare questo parametro.	No
`unwrap`	Impostazione predefinita: `False`	Sì	Sì, ad eccezione delle chiavi pubbliche.
`unwrap-template`	I valori devono utilizzare il modello di attributo applicato a qualsiasi chiave di cui è stato annullato il wrapping utilizzando questa chiave di wrapping.	Sì	No

Attributo della CLI del CloudHSM	Valore	Modificabile con key set-attribute	Da impostare al momento della creazione della chiave
`verify`	Impostazione predefinita: • Il valore è `True` per le chiavi AES. • Il valore è `False` per le chiavi RSA ed EC.	Sì	Sì
`wrap`	Impostazione predefinita: `False`	Sì	Sì, ad eccezione delle chiavi private.
`wrap-template`	I valori devono utilizzare il modello di attributo per abbinare la chiave sottoposta a wrapping usando questa chiave di wrapping.	Sì	No
`wrap-with-trusted`	Impostazione predefinita: `False`	Sì	Sì

Verifica il valore nella CLI di CloudHSM

Il valore di controllo in CloudHSM CLI è un hash o checksum a 3 byte di una chiave che viene generato quando l'HSM importa o genera una chiave. È possibile anche calcolare un valore di controllo al di fuori dell'HSM, ad esempio dopo aver esportato una chiave. È quindi possibile confrontare i valori del valore di controllo per confermare l'identità e l'integrità della chiave. Per ottenere il valore di controllo di una chiave, usa l'elenco delle chiavi con il flag output dettagliato.

AWS CloudHSM utilizza i seguenti metodi standard per generare un valore di controllo:

- Chiavi simmetriche: primi 3 byte del risultato della crittografia a blocchi zero con la chiave.

- Coppie di chiavi asimmetriche: primi 3 byte dell'SHA-1 hash della chiave pubblica.
- Chiavi HMAC: KCV per le chiavi HMAC attualmente non supportato.

Argomenti correlati per CloudHSM CLI

Per ulteriori informazioni sull'interfaccia CLI di CloudHSM, consulta i seguenti argomenti.

- La categoria chiave dell'interfaccia CLI di CloudHSM
- Riferimento per i comandi della CLI di CloudHSM

Configurazioni avanzate per CloudHSM CLI

L'interfaccia a riga di AWS CloudHSM comando (CLI) include la seguente configurazione avanzata, che non fa parte delle configurazioni generali utilizzate dalla maggior parte dei clienti. Queste configurazioni offrono funzionalità aggiuntive.

- Connessione a più cluster

Connessione a più cluster con la CLI CloudhSM

Con AWS CloudHSM Client SDK 5, puoi configurare CloudHSM CLI per consentire le connessioni a più cluster CloudHSM da una singola istanza CLI.

I seguenti argomenti descrivono come utilizzare la funzionalità multi-cluster della CLI di CloudHSM per connettersi a più cluster.

Argomenti

- Prerequisiti per più cluster per AWS CloudHSM
- Configura la CLI CloudHSM per la funzionalità multi-cluster
- Aggiungi un cluster alla tua configurazione AWS CloudHSM
- Rimuovi un cluster dalla tua configurazione AWS CloudHSM
- Interagisci con più cluster in AWS CloudHSM

Prerequisiti per più cluster per AWS CloudHSM

Prima di configurare il cluster per la connessione AWS CloudHSM a più cluster, è necessario soddisfare i seguenti prerequisiti:

- Due o più AWS CloudHSM cluster a cui desideri connetterti, insieme ai relativi certificati di cluster.
- Un' EC2 istanza con gruppi di sicurezza configurati correttamente per connettersi a tutti i cluster di cui sopra. Per ulteriori informazioni su come configurare un cluster e l'istanza del client, consulta la sezione Guida introduttiva AWS CloudHSM.
- Per configurare la funzionalità multi-cluster, è necessario aver già scaricato e installato la CLI ClouDHSM. Se non lo hai ancora fatto, consulta le istruzioni riportate in ???.
- Non sarai in grado di accedere a un cluster configurato con `./configure-cli[.exe] -a` poiché non sarà associato a un. `cluster-id` Puoi riconfigurarlo seguendo `config-cli add-cluster` come descritto in questa guida.

Configura la CLI CloudHSM per la funzionalità multi-cluster

Per configurare la CLI di CloudHSM per la funzionalità multi-cluster, procedi nel seguente modo:

1. Identifica i cluster a cui desideri connetterti.
2. Aggiungi questi cluster alla configurazione CLI di CloudHSM utilizzando il sottocomando configure-cli come descritto di seguito. `add-cluster`
3. Riavvia tutti i processi CLI di CloudHSM per rendere effettiva la nuova configurazione.

Aggiungi un cluster alla tua configurazione AWS CloudHSM

Quando ti connetti a più cluster, usa il `configure-cli add-cluster` comando per aggiungere un cluster alla tua configurazione.

Sintassi

```
configure-cli add-cluster [OPTIONS]
        --cluster-id <CLUSTER ID>
        [--region <REGION>]
        [--endpoint <ENDPOINT>]
        [--hsm-ca-cert <HSM CA CERTIFICATE FILE>]
        [--client-cert-hsm-tls-file <CLIENT CERTIFICATE FILE>]
        [--client-key-hsm-tls-file <CLIENT KEY FILE>]
```

```
        [-h, --help]
```

Esempi

Aggiungere un cluster utilizzando il parametro **`cluster-id`**

Example

Utilizza il parametro `configure-cli add-cluster` insieme a `cluster-id` per aggiungere un cluster (con l'ID del `cluster-1234567`) alla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli add-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" add-cluster --
cluster-id <cluster-1234567>
```

> **Tip**
>
> Se l'utilizzo del parametro `configure-cli add-cluster` con `cluster-id` non dà come risultato l'aggiunta del cluster, consulta l'esempio seguente per una versione più lunga del comando che richiede anche i parametri `--region` e `--endpoint` per identificare il cluster che si sta aggiungendo. Se, ad esempio, la regione del cluster è diversa da quella configurata come impostazione predefinita per la CLI di AWS, è necessario impiegare il parametro `--region` per utilizzare la regione corretta. Inoltre, hai la possibilità di specificare l'endpoint AWS CloudHSM API da utilizzare per la chiamata, che potrebbe essere necessario per varie configurazioni di rete, ad esempio l'utilizzo di endpoint di interfaccia VPC che non utilizzano il nome host DNS predefinito per. AWS CloudHSM

Aggiungere un cluster utilizzando i parametri **`cluster-id`**, **`endpoint`** e **`region`**

Example

Utilizza `configure-cli add-cluster` insieme ai parametri `cluster-id`, `endpoint` e `region` per aggiungere un cluster (con l'ID del `cluster-1234567`) alla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" add-cluster --cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Per ulteriori informazioni sui parametri `--cluster-id`, `--region` e `--endpoint`, vedi the section called "Parameters".

Parameters

--cluster-id ***<Cluster ID>***

Effettua una chiamata `DescribeClusters` per trovare tutti gli indirizzi IP a interfaccia di rete elastica (ENI) dell'HSM nel cluster associati all'ID del cluster. Il sistema aggiunge gli indirizzi IP ENI ai file di configurazione. AWS CloudHSM

> **Note**
>
> Se utilizzi il `--cluster-id` parametro da un' EC2 istanza all'interno di un VPC che non ha accesso alla rete Internet pubblica, devi creare un endpoint VPC di interfaccia con cui connetterti. AWS CloudHSM Per ulteriori informazioni sugli endpoint VPC, consulta la pagina ???.

Obbligatorio: sì

--endpoint ***<Endpoint>***

Specificare l'endpoint AWS CloudHSM API utilizzato per effettuare la chiamata. `DescribeClusters` È necessario impostare questa opzione insieme a `--cluster-id`.

Obbligatorio: no

--hsm-ca-cert ***<HsmCA Certificate Filepath>***

Speciifica il percorso del file del certificato CA HSM.

Obbligatorio: no

--region ***<Region>***

Specifica la regione del cluster. È necessario impostare questa opzione insieme a `--cluster-id`.

Se non indichi il parametro `--region`, il sistema sceglie la regione tentando di leggere le variabili di ambiente `AWS_DEFAULT_REGION` o `AWS_REGION`. Se queste variabili non sono impostate, il sistema controlla la regione associata al tuo profilo indicata nel tuo file di AWS Config (generalmente `~/.aws/config`) a meno che non sia stato specificato un file diverso nella variabile di ambiente `AWS_CONFIG_FILE`. Se non è stata impostata nessuna delle variabili precedenti, il sistema utilizza la regione `us-east-1` per impostazione predefinita.

Obbligatorio: no

-- -archivio client-cert-hsm-tls ***<client certificate hsm tls path>***

Percorso del certificato client utilizzato per l'autenticazione reciproca TLS Client-HSM.

Utilizza questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-key-hsm-tls-file`.

Obbligatorio: no

-- -file client-key-hsm-tls ***<client key hsm tls path>***

Percorso della chiave client utilizzata per l'autenticazione reciproca TLS Client-HSM.

Utilizza questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-cert-hsm-tls-file`.

Obbligatorio: no

Rimuovi un cluster dalla tua configurazione AWS CloudHSM

Quando ti connetti a più cluster con la CLI di CloudhSM, usa `configure-cli remove-cluster` il comando per rimuovere un cluster dalla tua configurazione.

Sintassi

```
configure-cli remove-cluster [OPTIONS]
        --cluster-id <CLUSTER ID>
        [-h, --help]
```

Esempi

Rimuovere un cluster utilizzando il parametro **`cluster-id`**

Example

Utilizza il parametro `configure-cli remove-cluster` insieme a `cluster-id` per rimuovere un cluster (con l'ID del `cluster-1234567`) dalla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-cli remove-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" remove-cluster --
cluster-id <cluster-1234567>
```

Per ulteriori informazioni sul parametro `--cluster-id`, vedi the section called "Parameters".

Parametro

--cluster-id ***<Cluster ID>***

L'ID del cluster da rimuovere dalla configurazione.

Obbligatorio: sì

Interagisci con più cluster in AWS CloudHSM

Dopo aver configurato più cluster con la CLI di CloudhSM, usa il comando per interagire con essi. `cloudhsm-cli`

Esempi

Impostazione di un valore predefinito quando si utilizza la modalità interattiva **`cluster-id`**

Example

Utilizza il parametro ??? insieme al `cluster-id` parametro per impostare un cluster predefinito (con l'ID di`cluster-1234567`) dalla tua configurazione.

Linux

```
$ cloudhsm-cli interactive --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" interactive --
cluster-id <cluster-1234567>
```

Impostazione di **`cluster-id`** quando si esegue un singolo comando

Example

Utilizzate il `cluster-id` parametro per impostare il cluster (con l'ID di`cluster-1234567`) da cui ??? partire.

Linux

```
$ cloudhsm-cli cluster hsm-info --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" cluster hsm-info
 --cluster-id <cluster-1234567>
```

Riferimento per i comandi della CLI di CloudHSM

L'interfaccia a riga di comando di CloudHSM aiuta gli amministratori a gestire gli utenti nel proprio cluster. AWS CloudHSM La CLI di CloudHSM può essere eseguita in due modalità: modalità interattiva e modalità a comando singolo. Per una guida rapida, vedi Nozioni di base su AWS CloudHSM Interfaccia a riga di comando (CLI).

Per eseguire la maggior parte dei comandi della CLI di CloudHSM, è necessario avviare la CLI di CloudHSM e accedere all'HSM. Se aggiungi o elimini moduli HSM, aggiorna i file di configurazione per la CLI di CloudHSM. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

I seguenti argomenti descrivono i comandi nella CLI di CloudHSM:

Comando	Description	Tipo di utente
attivazione	Attiva un cluster CloudHSM e conferma che il cluster è nuovo. Questa deve essere fatto prima di poter eseguire qualsiasi altra operazione.	Admin non attivato
hsm-info	Elenca gli HSM nel tuo cluster.	Tutti [1], compresi gli utenti non autenticati. L'accesso non è necessario.
ecdsa	Genera una firma utilizzando una chiave privata EC e il meccanismo di firma ECDSA.	Crypto user (CU)
ed25519	Genera una firma utilizzando una chiave privata Ed25519 e il PureEd meccanismo di firma DSA.	CU
ed25519ph	Genera una firma utilizzando una chiave privata Ed25519 e il meccanismo di firma DSA. HashEd	CU
ml-dsa	Genera una firma utilizzando una chiave ML-DSA [2] privata e il ML-DSA meccanismo di firma.	CU
rsa-pkcs	Genera una firma utilizzando una chiave privata RSA e il meccanismo di RSA-PKCS firma.	CU
rsa-pkcs-pss	Genera una firma utilizzando una chiave privata RSA e il	CU

Comando	Description	Tipo di utente
	meccanismo di RSA-PKCS-PSS firma.	
ecdsa	Conferma che un file è stato firmato nell'HSM con una determinata chiave pubblica. Verifica che la firma sia stata generata utilizzando il meccanismo di firma ECDSA. Confronta un file firmato con un file sorgente e determina se i due sono correlati crittografica mente in base a una determinata chiave pubblica ecdsa e a un determinato meccanismo di firma.	CU
ed25519	Verifica le firme PureEd DSA utilizzando una chiave pubblica Ed25519.	CU
ed25519ph	Verifica le firme HashEd DSA utilizzando una chiave pubblica Ed25519.	CU
ml-dsa	Verifica una firma utilizzando una chiave ML-DSA [2] pubblica e il meccanismo di verifica. ML-DSA	CU

Comando	Description	Tipo di utente
rsa-pkcs	Conferma che un file è stato firmato nell'HSM con una determinata chiave pubblica. Verifica che la firma sia stata generata utilizzan do il meccanismo di RSA-PKCS firma. Confronta un file firmato con un file sorgente e determina se i due sono correlati crittograficamente in base a una determinata chiave pubblica RSA e a un determinato meccanismo di firma.	CU
rsa-pkcs-pss	Conferma che un file è stato firmato nell'HSM con una determinata chiave pubblica. Verifica che la firma sia stata generata utilizzando il meccanismo di RSA-PKCS-PSS firma. Confronta un file firmato con un file sorgente e determina se i due sono correlati crittograficamente in base a una determinata chiave pubblica RSA e a un determinato meccanismo di firma.	CU
Elimina chiave	Elimina una chiave dal cluster. AWS CloudHSM	CU
Generazione chiavi-file	Genera un file chiave nel AWS CloudHSM cluster.	CU

Comando	Description	Tipo di utente
Generazione chiavi-asimmetriche-coppia rsa	Genera una coppia di chiavi RSA asimmetrica nel cluster. AWS CloudHSM	CU
Generazione chiavi-asimmetriche-coppia ec	Genera una coppia di chiavi asimmetrica Elliptic-curve (EC) nel cluster. AWS CloudHSM	CU
key generate-asimmetric-pair ml-dsa	Genera una coppia di chiavi asimmetrica nel cluster. ML-DSA AWS CloudHSM	CU
Generazione chiavi-simmetriche aes	Genera una chiave AES simmetrica nel cluster. AWS CloudHSM	CU
Generazione chiavi-simmetriche generiche-segrete	Genera una chiave segreta generica simmetrica nel cluster. AWS CloudHSM	CU
key import pem	Importa una chiave in formato PEM in un HSM. È possibile utilizzarlo per importare le chiavi pubbliche generate al di fuori del HSM.	CU
Elenco chiavi	Trova tutte le chiavi per l'utente corrente presente nel cluster. AWS CloudHSM	CU
replica della chiave	Replica una chiave da un cluster di origine a un cluster di destinazione clonato.	CU
key set-attribute	Imposta gli attributi delle chiavi nel tuo cluster. AWS CloudHSM	I CU possono eseguire questo comando, gli admin possono impostare l'attributo affidabile.

Comando	Description	Tipo di utente
Condivisione chiave	Condivide una chiave con altre CU del AWS CloudHSM cluster.	CU
Annulla condivisione chiave	Annulla la condivisione di una chiave con altre CU del AWS CloudHSM cluster.	CU
aes-gcm	Decomprime una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. AES-GCM	CU
aes-no-pad	Scompone una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. AES-NO-PAD	CU
aes-pkcs5-pad	Scompone una chiave di payload utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. AES-PKCS5-PAD	CU
aes-zero-pad	Scompone una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. AES-ZERO-PAD	CU

Comando	Description	Tipo di utente
cloudhsm-aes-gcm	Scompone una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. CLOUDHSM-AES-GCM	CU
rsa-aes	Scompone una chiave di payload utilizzando una chiave privata RSA e il meccanismo di unwrapping. RSA-AES	CU
rsa-oaep	Scompone una chiave di payload utilizzando la chiave privata RSA e il meccanismo di unwrapping. RSA-OAEP	CU
rsa-pkcs	Scompone una chiave di payload utilizzando la chiave privata RSA e il meccanism o di decomposizione. RSA-PKCS	CU
aes-gcm	Consente di avvolgere una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping. AES-GCM	CU
aes-no-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping. AES-NO-PAD	CU

Comando	Description	Tipo di utente
aes-pkcs5-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping. AES-PKCS5-PAD	CU
aes-zero-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping. AES-ZERO-PAD	CU
cloudhsm-aes-gcm	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping. CLOUDHSM-AES-GCM	CU
rsa-aes	Avvolge una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e sul meccanismo di wrapping. RSA-AES	CU
rsa-oaep	Consente di avvolgere una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e sul meccanismo di wrapping. RSA-OAEP	CU

Comando	Description	Tipo di utente
Utilizzate il key wrap rsa-pkcs comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `RSA-PKCS` L'attributo della chiave di payload deve essere impostato su. `extractable` `true` Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche. Per utilizzare il key wrap rsa-pkcs comando, è necessari o innanzitutto disporre di una chiave RSA nel cluster AWS CloudHSM . È possibile generare una coppia di chiavi RSA utilizzando il La categoria di generazione di coppie asimmetriche in CloudHSM CLI comando e l'`wrap`attributo `true` impostati su. Tipo di utente I seguenti tipi di utenti possono eseguire questo comando. • Utenti Crypto () CUs	Consente di avvolgere una chiave di payload utilizzan do una chiave pubblica RSA sull'HSM e sul meccanismo di wrapping. RSA-PKCS	CU

Comando	Description	Tipo di utente
Login	Accedi al tuo cluster. AWS CloudHSM	Admin, crypto user (CU) e utente dell'applicazione (AU)
Disconnessione	Esci dal tuo AWS CloudHSM cluster.	Admin, CU e utente dell'applicazione (AU)
Quorum token-firma elimina	Elimina uno o più token per un servizio autorizzato dal quorum.	Admin
quorum token-sign generate	Genera un token per un servizio autorizzato dal quorum.	Admin
quorum token-sign list	Elenca tutti i token-firma del quorum presenti nel cluster di CloudHSM.	Tutti [1], compresi gli utenti non autenticati. L'accesso non è necessario.
quorum token-sign list-quorum-values	Elenca i valori del quorum impostati nel cluster CloudHSM.	Tutti [1], compresi gli utenti non autenticati. L'accesso non è necessario.
quorum token-sign set-quorum-value	Imposta un nuovo valore di quorum per un servizio autorizzato dal quorum.	Admin
user change-mfa	Modifica la strategia di autenticazione a più fattori (MFA) di un utente.	Admin, CU
user change-password	Modifica la password degli utenti nei moduli HSM. Qualsiasi utente può modificare la propria password. Gli admin possono modificare la password di chiunque.	Admin, CU

Comando	Description	Tipo di utente
user create	Crea un utente nel tuo AWS CloudHSM cluster.	Admin
user delete	Elimina un utente nel AWS CloudHSM cluster.	Admin
user list	Elenca gli utenti del AWS CloudHSM cluster.	Tutti [1], compresi gli utenti non autenticati. L'accesso non è richiesto.
Modifica utente-quorum token-firma registra	Registra la strategia del quorum token-firma quorum per un utente.	Admin

Annotazioni

- [1] Tutti gli utenti includono tutti i ruoli elencati e gli utenti non connessi.
- [2] A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

La categoria di cluster nella CLI di CloudHSM

Nella cluster CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per i cluster. Attualmente, la categoria cluster è composta dai seguenti comandi:

Argomenti

- Attiva un cluster con CloudHSM CLI
- Elenco HSMs con CLI CloudHSM
- La categoria cluster mtls nella CLI di CloudhSM

Attiva un cluster con CloudHSM CLI

Utilizza il cluster activate comando nella CLI di CloudHSM per attivare un nuovo cluster in. AWS CloudHSMÈ necessario eseguire questo comando prima di poter utilizzare il cluster per eseguire operazioni di crittografia.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Admin non attivato

Sintassi

Questo comando non ha parametri.

```
aws-cloudhsm > help cluster activate
Activate a cluster

This command will set the initial Admin password. This process will cause your CloudHSM
 cluster to
move into the ACTIVE state.

USAGE:
    cloudhsm-cli cluster activate [OPTIONS] [--password <PASSWORD>]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --password <PASSWORD>
          Optional: Plaintext activation password If you do not include this argument
 you will be prompted for it

  -h, --help
          Print help (see a summary with '-h')
```

Esempio

Questo comando attiva il cluster impostando la password iniziale per l'utente amministratore.

```
aws-cloudhsm > cluster activate
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": "Cluster activation successful"
}
```

Argomenti correlati

- user create
- user delete
- user change-password

Elenco HSMs con CLI CloudHSM

Utilizzate il cluster hsm-info comando nella CLI di CloudHSM per elencare i HSMs moduli di sicurezza hardware () nel cluster. AWS CloudHSM Non è necessario che tu abbia eseguito l'accesso alla CLI di CloudHSM per eseguire questo comando.

> **Note**
>
> Se aggiungi o elimini HSMs, aggiorna i file di configurazione utilizzati dal AWS CloudHSM client e dagli strumenti della riga di comando. In caso contrario, le modifiche apportate potrebbero non essere valide per tutti HSMs gli utenti del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
aws-cloudhsm > help cluster hsm-info
List info about each HSM in the cluster
```

```
Usage: cloudhsm-cli cluster hsm-info [OPTIONS]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Questo comando elenca il HSMs presente nel AWS CloudHSM cluster.

```
aws-cloudhsm > cluster hsm-info
{
  "error_code": 0,
  "data": {
    "hsms": [
      {
        "vendor": "Marvell Semiconductors, Inc.",
        "model": "NITROX-III CNN35XX-NFBE",
        "serial-number": "5.3G1941-ICM000590",
        "hardware-version-major": "5",
        "hardware-version-minor": "3",
        "firmware-version-major": "2",
        "firmware-version-minor": "6",
        "firmware-build-number": "16",
        "firmware-id": "CNN35XX-NFBE-FW-2.06-16"
        "fips-state": "2 [FIPS mode with single factor authentication]"
      },
      {
        "vendor": "Marvell Semiconductors, Inc.",
        "model": "NITROX-III CNN35XX-NFBE",
        "serial-number": "5.3G1941-ICM000625",
        "hardware-version-major": "5",
        "hardware-version-minor": "3",
        "firmware-version-major": "2",
        "firmware-version-minor": "6",
        "firmware-build-number": "16",
        "firmware-id": "CNN35XX-NFBE-FW-2.06-16"
        "fips-state": "2 [FIPS mode with single factor authentication]"
      },
      {
```

```
        "vendor": "Marvell Semiconductors, Inc.",
        "model": "NITROX-III CNN35XX-NFBE",
        "serial-number": "5.3G1941-ICM000663",
        "hardware-version-major": "5",
        "hardware-version-minor": "3",
        "firmware-version-major": "2",
        "firmware-version-minor": "6",
        "firmware-build-number": "16",
        "firmware-id": "CNN35XX-NFBE-FW-2.06-16"
        "fips-state": "2 [FIPS mode with single factor authentication]"
      }
    ]
  }
}
```

L'output ha i seguenti attributi:

- Fornitore: il nome del fornitore dell'HSM.
- Modello: il numero di modello dell'HSM.
- Numero di serie: Il numero di serie dell'HSM. Questo potrebbe cambiare a causa di sostituzioni.
- H ardware-version-major: La versione hardware principale.
- H ardware-version-minor: La versione hardware secondaria.
- F irmware-version-major: La versione principale del firmware.
- F irmware-version-minor: La versione secondaria del firmware.
- F irmware-build-number: Il numero di build del firmware.
- Firmware-id: l'ID del firmware, che include le versioni principali e secondarie insieme al build.
- Stato FIPS: la modalità FIPS del cluster e il suo interno. HSMs In modalità FIPS, l'output è «2 [modalità FIPS con autenticazione a fattore singolo]». In modalità non FIPS, l'output è «0 [modalità non FIPS con autenticazione a fattore singolo]».

Argomenti correlati

- Attiva un cluster con CloudHSM CLI

La categoria cluster mtls nella CLI di CloudhSM

In CloudHSM cluster mtls CLI, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per i cluster. AWS CloudHSM Attualmente, questa categoria comprende i seguenti comandi:

Argomenti

- Annulla la registrazione di un trust anchor con la CLI di CloudhSM
- Ottieni il livello di applicazione dell'MTLS con la CLI di CloudhSM
- Elenca gli ancoraggi di fiducia con la CLI di CloudhSM
- Registra un trust anchor con la CLI di CloudhSM
- Imposta il livello di applicazione dell'MTLS con la CLI di CloudhSM

Annulla la registrazione di un trust anchor con la CLI di CloudhSM

Utilizza il cluster mtls deregister-trust-anchor comando nella CLI di CloudHSM per annullare la registrazione di un trust anchor per il TLS reciproco tra client e. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Requisiti

- Per eseguire questo comando, devi accedere come utente amministratore.

Sintassi

```
aws-cloudhsm > help cluster mtls deregister-trust-anchor

Deregister a trust anchor for mtls

Usage: cluster mtls deregister-trust-anchor [OPTIONS] --certificate-reference
 [<CERTIFICATE_REFERENCE>...]

Options:
```

```
      --certificate-reference <CERTIFICATE_REFERENCE>  A hexadecimal or decimal
 certificate reference
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --approval <APPROVAL>  Filepath of signed quorum token file to approve operation
  -h, --help                     Print help
```

Esempio

Example

Nell'esempio seguente, questo comando rimuove un trust anchor dall'HSM.

```
aws-cloudhsm > cluster mtls deregister-trust-anchor --certificate-reference 0x01

{
  "error_code": 0,
  "data": {
    "message": "Trust anchor with reference 0x01 deregistered successfully"
  }
}
```

È quindi possibile eseguire il list-trust-anchors comando per confermare che la registrazione di trust anchor è stata cancellata da: AWS CloudHSM

```
aws-cloudhsm > cluster mtls list-trust-anchors

{
  "error_code": 0,
  "data": {
    "trust_anchors": []
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<CERTIFICATE_REFERENCE>

Un riferimento al certificato in formato esadecimale o decimale.

Obbligatorio: sì

> ⚠ Warning
>
> Dopo aver annullato la registrazione di un trust anchor nel cluster, tutte le connessioni MTLS esistenti che utilizzano il certificato client firmato da quel trust anchor verranno eliminate.

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del quorum del servizio del cluster quorum è maggiore di 1.

Argomenti correlati

- cluster mtls reregister-trust-anchor
- cluster mtls list-trust-anchors
- Configurazione MTLs (consigliato)

Ottieni il livello di applicazione dell'MTLS con la CLI di CloudhSM

Utilizza il cluster mtls get-enforcement comando nella CLI di CloudHSM per ottenere il livello di applicazione dell'uso del TLS reciproco tra client e. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, devi accedere come utente amministratore o utente crittografico (CUs).

Sintassi

```
aws-cloudhsm > help cluster mtls get-enforcement

Get the status of mtls enforcement in the cluster

Usage: cluster mtls get-enforcement [OPTIONS]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Example

Nell'esempio seguente, questo comando elenca il livello di applicazione mtls di. AWS CloudHSM

```
aws-cloudhsm > cluster mtls get-enforcement

{
  "error_code": 0,
  "data": {
    "mtls-enforcement-level": "none"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

Argomenti correlati

- cluster mtls set-enforcement
- Configurazione MTLs (consigliato)

Elenca gli ancoraggi di fiducia con la CLI di CloudhSM

Utilizza il cluster mtls list-trust-anchors comando nella CLI di CloudHSM per elencare tutti gli ancoraggi di fiducia che possono essere utilizzati per il TLS reciproco tra client e. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
aws-cloudhsm > help cluster mtls list-trust-anchors

List all trust anchors for mtls

Usage: cluster mtls list-trust-anchors [OPTIONS]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Example

Nell'esempio seguente, questo comando elenca tutti gli ancoraggi di fiducia registrati da. AWS CloudHSM

```
aws-cloudhsm > cluster mtls list-trust-anchors
```

```
{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate 1>",
        "cluster-coverage": "full"
      },
      {
        "certificate-reference": "0x02",
        "certificate": "<PEM Encoded Certificate 2>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire l'operazione.

Obbligatorio: se sono stati configurati più cluster.

Argomenti correlati

- cluster: mtls reregister-trust-anchor
- cluster mtls deregister-trust-anchor
- Configurazione MTLs (consigliato)

Registra un trust anchor con la CLI di CloudhSM

Usa il cluster mtls register-trust-anchor comando nella CLI di CloudHSM per registrare un trust anchor per il TLS reciproco tra client e. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Requisiti

Accept Trust AWS CloudHSM Anchora con i seguenti tipi di chiavi:

Tipo di chiavi	Description
EC	curve secp256r1 (P-256), secp384r1 (P-384) e secp521r1 (P-521).
RSA	Chiavi RSA a 2048 bit, 3072 bit e 4096 bit.

Sintassi

```
aws-cloudhsm > help cluster mtls register-trust-anchor

Register a trust anchor for mtls

Usage: cluster mtls register-trust-anchor [OPTIONS] --path [<PATH>...]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --path <PATH>  Filepath of the trust anchor to register
      --approval <APPROVAL>  Filepath of signed quorum token file to approve operation
  -h, --help                     Print help
```

Esempio

Example

Nell'esempio seguente, questo comando registra un trust anchor sull'HSM. Il numero massimo di trust anchors che è possibile registrare è due (2).

```
aws-cloudhsm > cluster mtls register-trust-anchor --path /home/rootCA

{
```

```
  "error_code": 0,
  "data": {
    "trust_anchor": {
      "certificate-reference": "0x01",
      "certificate": "<PEM Encoded Certificate>",
      "cluster-coverage": "full"
    }
  }
}
```

È quindi possibile eseguire il list-trust-anchors comando per confermare che trust anchor è stato registrato su: AWS CloudHSM

```
aws-cloudhsm > cluster mtls list-trust-anchors

{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PATH>

Percorso del file del trust anchor da registrare.

Obbligatorio: sì

> **Note**
>
> AWS CloudHSM supporta la registrazione di certificati intermedi come trust anchor. In questi casi, l'intero file della catena di certificati con codifica PEM deve essere registrato nell'HSM, con i certificati in ordine gerarchico.
> AWS CloudHSM supporta una catena di certificati di 6980 byte.

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del quorum del servizio cluster quorum è maggiore di 1.

Argomenti correlati

- cluster mtls deregister-trust-anchor
- cluster mtls list-trust-anchors
- Configurazione MTLs (consigliato)

Imposta il livello di applicazione dell'MTLS con la CLI di CloudhSM

Utilizza il cluster mtls set-enforcement comando nella CLI di CloudHSM per impostare il livello di applicazione dell'uso del TLS reciproco tra client e. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Amministratore con nome utente come amministratore

Requisiti

Per eseguire questo comando:

- Almeno un trust anchor è stato registrato con successo su. AWS CloudHSM
- Configura la CLI di CloudHSM con la chiave privata e il certificato client corretti e avvia CloudHSM CLI con una connessione TLS reciproca.

- Devi accedere come amministratore predefinito con il nome utente «admin». Qualsiasi altro utente amministratore non sarà in grado di eseguire questo comando.

Sintassi

```
aws-cloudhsm > help cluster mtls set-enforcement

Set mtls enforcement policy in the cluster

Usage: cluster mtls set-enforcement [OPTIONS] --level [<LEVEL>...]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --level <LEVEL>  Level to be set for mtls in the cluster [possible values: none,
 cluster]
      --approval <APPROVAL>  Filepath of signed quorum token file to approve operation
  -h, --help                     Print help
```

Esempio

Example

Nell'esempio seguente, questo comando imposta il livello di applicazione mtls del cluster AWS CloudHSM to be. Il comando set-enforcement può essere eseguito solo in una connessione TLS reciproca e può essere effettuato l'accesso come utente amministratore con nome utente come amministratore, vedi impostare l'applicazione MTLS per. AWS CloudHSM

```
aws-cloudhsm > cluster mtls set-enforcement --level cluster

{
  "error_code": 0,
  "data": {
    "message": "Mtls enforcement level set to Cluster successfully"
  }
}
```

Puoi quindi eseguire il get-enforcement comando per confermare che il livello di applicazione è stato impostato su cluster:

```
aws-cloudhsm > cluster mtls get-enforcement

{
  "error_code": 0,
  "data": {
    "mtls-enforcement-level": "cluster"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<LEVEL>

Livello da impostare per mtls nel cluster.

Valori validi

- cluster: applica l'uso del TLS reciproco tra client e AWS CloudHSM nel cluster.
- nessuno: non imporre l'uso del TLS reciproco tra client e AWS CloudHSM nel cluster.

Obbligatorio: sì

> **Warning**
>
> Dopo aver imposto l'utilizzo di MTLS nel cluster, tutte le connessioni non MTLS esistenti verranno interrotte e sarà possibile connettersi al cluster solo con certificati MTLS.

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del quorum del servizio del cluster quorum è maggiore di 1.

Argomenti correlati

- cluster mtls get-enforcement
- Configurazione MTLs (consigliato)

La categoria delle criptovalute in CloudHSM CLI

Nella CLI di CloudHSM, crypto è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per le operazioni crittografiche. Attualmente, questa categoria comprende i seguenti comandi:

- Firma
  - ecdsa
  - ed25519
  - ed25519ph
  - ml-dsa
  - rsa-pkcs
  - rsa-pkcs-pss
- Verifica
  - ecdsa
  - ed25519
  - ed25519ph
  - ml-dsa
  - rsa-pkcs
  - rsa-pkcs-pss

La categoria delle firme crittografiche in CloudHSM CLI

Nella CLI di CloudHSM, crypto sign è una categoria principale per un gruppo di comandi che, se combinata con la categoria principale, utilizza una chiave privata scelta nel AWS CloudHSM cluster per generare una firma. crypto signha i seguenti sottocomandi:

- Genera una firma con il meccanismo ECDSA nell'interfaccia a riga di comando di CloudHSM
- Genera una firma con il meccanismo PureEd DSA nella CLI di CloudHSM

- Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM
- Genera una firma con il ML-DSA meccanismo della CLI di CloudHSM
- Genera una firma con il RSA-PKCS meccanismo della CLI di CloudHSM
- Genera una firma con il RSA-PKCS-PSS meccanismo della CLI di CloudHSM

Per utilizzarlocrypto sign, è necessario disporre di una chiave privata nell'HSM. È possibile generare una chiave privata con i seguenti comandi:

- Generazione chiavi-asimmetriche-coppia ec
- key generate-asimmetric-pair ml-dsa
- Generazione chiavi-asimmetriche-coppia rsa

Genera una firma con il meccanismo ECDSA nell'interfaccia a riga di comando di CloudHSM

Usa il crypto sign ecdsa comando in CloudHSM CLI per generare una firma utilizzando una chiave privata EC e il meccanismo di firma ECDSA.

Per utilizzare il crypto sign ecdsa comando, è necessario innanzitutto disporre di una chiave privata EC nel cluster. AWS CloudHSM È possibile generare una chiave privata EC utilizzando il Genera una coppia di key pair EC asimmetrica con CloudHSM CLI comando con l'`sign`attributo impostato `su``true`.

La firma ECDSA risultante viene generata nel formato`r | | s`, in cui i componenti r e s sono concatenati come dati binari non elaborati e restituiti in formato codificato base64.

> **Note**
>
> Le firme possono essere verificate con sottocomandi. AWS CloudHSM La categoria di verifica crittografica in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto sign ecdsa
Sign with the ECDSA mechanism

Usage: crypto sign ecdsa --key-filter [<KEY_FILTER>>...] --hash-
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --hash-function <HASH_FUNCTION>
          [possible values: sha1, sha224, sha256, sha384, sha512]
      --data-path <DATA_PATH>
          The path to the file containing the data to be signed
      --data <DATA>
          Base64 Encoded data to be signed
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare crypto sign ecdsa per generare una firma utilizzando il meccanismo di firma ECDSA e la funzione hash. SHA256 Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per i dati codificati in base 64

```
aws-cloudhsm > crypto sign ecdsa \
    --key-filter attr.label=ec-private \
    --hash-function sha256 \
    --data YWJjMTIz
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000007808dd",
    "signature": "4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw=="
  }
}
```

Example Esempio: generare una firma per un file di dati

```
aws-cloudhsm > crypto sign ecdsa \
    --key-filter attr.label=ec-private \
    --hash-function sha256 \
    --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000007808dd",
    "signature": "4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di ATTR.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE per selezionare una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Genera una firma con il meccanismo PureEd DSA nella CLI di CloudHSM

> **Important**
>
> PureEdLe operazioni di firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.

Usa il crypto sign ed25519 comando in CloudHSM CLI per generare una firma utilizzando una chiave privata Ed25519 e il meccanismo di firma DSA. PureEd PureEdDSA (chiamato anche Ed25519) firma il messaggio non elaborato direttamente senza prehashing, come definito nella RFC 8032, Sezione 5.1.6 sul sito Web IETF. https://www.rfc-editor.org/rfc/rfc8032#section-5.1.6

> **Note**
>
> La dimensione massima dei messaggi per DSA è 16.000 byte. PureEd Per messaggi di dimensioni maggiori, utilizzare. Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM

Per utilizzare il crypto sign ed25519 comando, devi prima avere una chiave privata Ed25519 nel tuo AWS CloudHSM cluster. È possibile generare una chiave privata Ed25519 utilizzando il Genera una coppia di key pair EC asimmetrica con CloudHSM CLI comando con il `curve` parametro impostato su `ed25519` e l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate con sottocomandi. AWS CloudHSM La categoria di verifica crittografica in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.
- PureEdLe operazioni di firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.
- I dati del messaggio non devono superare i 16.000 byte. Per messaggi più grandi, usa. Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM

Sintassi

```
aws-cloudhsm > help crypto sign ed25519
Sign with the Ed25519 mechanism

Usage: crypto sign ed25519 [OPTIONS] --key-filter [<KEY_FILTER>...] <--data-
path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --data-path <DATA_PATH>
          The path to the file containing the data to be signed
      --data <DATA>
          Base64 Encoded data to be signed
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto sign ed25519 generare una firma utilizzando il meccanismo di firma PureEd DSA. Questo comando firma il messaggio non elaborato direttamente senza prehashing. Questo comando utilizza una chiave privata Ed25519 nell'HSM.

Example Esempio: generare una firma per i dati codificati in base 64

```
aws-cloudhsm > crypto sign ed25519 \
    --key-filter attr.label=ed25519-private \
    --data YWJj
{
  "error_code": 0,
  "data": {
    "key-reference": "0x00000000006c1507",
    "signature": "NjWz797ntYSLFwg7nKYYdn
+On3cCMj4zKz059wadVVlBHxyxe4JrSZxgekwb9AYR5xFxuVE9dTnDSo+gCaW/CQ=="
  }
}
```

Example Esempio: generare una firma per un file di dati

```
aws-cloudhsm > crypto sign ed25519 \
    --key-filter attr.label=ed25519-private \
    --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x00000000006c1507",
    "signature": "NjWz797ntYSLFwg7nKYYdn
+On3cCMj4zKz059wadVVlBHxyxe4JrSZxgekwb9AYR5xFxuVE9dTnDSo+gCaW/CQ=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare. I dati decodificati non devono superare i 16.000 byte.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare. Il contenuto del file non deve superare i 16.000 byte.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro data)

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di selezione di una chiave corrispondente.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta. Attributi chiave per la CLI di CloudHSM

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

Obbligatorio: no

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI
- Verifica una firma firmata con il meccanismo PureEd DSA nella CLI di CloudHSM
- Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM

Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM

> **Important**
>
> HashEdLe operazioni di firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.

Usa il crypto sign ed25519ph comando in CloudHSM CLI per generare una firma utilizzando una chiave privata Ed25519 e il meccanismo di firma DSA. HashEd Per ulteriori informazioni sul HashEd DSA, vedere NIST SP 186-5, Sezione 7.8.

Per utilizzare il crypto sign ed25519ph comando, è necessario innanzitutto disporre di una chiave privata Ed25519 nel cluster. AWS CloudHSM È possibile generare una chiave privata Ed25519 utilizzando il Genera una coppia di key pair EC asimmetrica con CloudHSM CLI comando con il `curve` parametro impostato su `ed25519` e l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate con sottocomandi. AWS CloudHSM La categoria di verifica crittografica in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.
- HashEdLe operazioni di firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.

Sintassi

```
aws-cloudhsm > help crypto sign ed25519ph
```

```
Sign with the Ed25519ph mechanism

Usage: crypto sign ed25519ph [OPTIONS] --key-filter [<KEY_FILTER>...] --data-
type <DATA_TYPE> --hash-function <HASH_FUNCTION> <--data-path <DATA_PATH>|--
data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --data-path <DATA_PATH>
          The path to the file containing the data to be signed
      --data <DATA>
          Base64 Encoded data to be signed
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
      --hash-function <HASH_FUNCTION>
          Hash function [possible values: sha512]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per generare una firma utilizzando il meccanismo di firma e crypto sign ed25519ph la funzione hash ED25519Ph. `sha512` Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per i dati codificati in base 64

```
aws-cloudhsm > crypto sign ed25519ph \
    --key-filter attr.label=ed25519-private \
    --data-type raw \
    --hash-function sha512 \
    --data YWJj
{
```

```
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000401cdf",
    "signature": "mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h2OUmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg=="
  }
}
```

Example Esempio: generare una firma per un file di dati

```
aws-cloudhsm > crypto sign ed25519ph \
    --key-filter attr.label=ed25519-private \
    --data-type raw \
    --hash-function sha512 \
    --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000401cdf",
    "signature": "mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h2OUmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro di dati)

<HASH_FUNCTION>

Specifica la funzione hash. ED25519ph supporta solo SHA512.

Valori validi:

- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di ATTR.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE per selezionare una chiave corrispondente.

Per un elenco Attributi chiave per la CLI di CloudHSM degli attributi chiave CLI di CloudHSM supportati, consulta.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

Obbligatorio: no

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Valori validi:

- raw
- digerire

Campo obbligatorio: sì

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Genera una firma con il ML-DSA meccanismo della CLI di CloudHSM

Usa il crypto sign ml-dsa comando nell'interfaccia a riga di comando di CloudHSM per generare una firma utilizzando una chiave ML-DSA privata e il ML-DSA meccanismo di firma.

Per utilizzare il crypto sign ml-dsa comando, devi prima avere una chiave ML-DSA privata nel tuo cluster. AWS CloudHSM È possibile generare una chiave ML-DSA privata utilizzando il Genera una coppia di ML-DSA chiavi asimmetrica con CloudHSM CLI comando con l'`sign`attributo impostato su`true`.

> **Note**
>
> A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

> **Note**
>
> È possibile verificare le firme utilizzando i sottocomandi. AWS CloudHSM La categoria di verifica crittografica in CloudHSM CLI

È possibile generare una firma utilizzando due tipi di dati: dati grezzi o dati mu esterni calcolati.

Passaggi per generare mu esterno

Per utilizzare il tipo di `external-mu` dati, devi prima generare un digest mu a 64 byte dal tuo messaggio utilizzando la chiave pubblica: ML-DSA

1. Genera il file PEM con chiave pubblica utilizzando l'interfaccia a riga di comando di CloudHSM:

   ```
   aws-cloudhsm > key generate-file --encoding pem --path /tmp/public_key.pem --filter
    attr.label="mldsa-public-key-example"
   ```

   Per ulteriori informazioni, consulta Esporta una chiave asimmetrica con CloudHSM CLI.
2. Costruisci il prefisso del messaggio:

   Il prefisso contiene un separatore di dominio, la lunghezza di qualsiasi contesto e il contesto. L'impostazione predefinita per il separatore di dominio e la lunghezza del contesto è zero.
3. Anteponi il prefisso del messaggio al messaggio.

4. Usa SHAKE256 per eseguire l'hashing della chiave pubblica e anteporla al risultato del passaggio 3.
5. Esegui l'hashing del risultato per produrre un mu esterno a 64 byte:

Esempio di OpenSSL

L'esempio seguente utilizza OpenSSL 3.5 per costruire il mu esterno:

```
{
    openssl asn1parse -inform PEM -in public_key.pem -strparse 17 -noout -out - 2>/dev/
null |
    openssl dgst -provider default -shake256 -xoflen 64 -binary;

    printf '\x00\x00';

    echo -n "your message"
} | openssl dgst -provider default -shake256 -xoflen 64 -binary | base64
```

Utilizzate il digest a 64 byte risultante con codifica in base64 come parametro con. `--data --data-type external-mu`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto sign ml-dsa
Signs data with a ML-DSA private key

Usage: crypto sign ml-dsa [OPTIONS] --key-filter <KEY_FILTER> --data-type <DATA_TYPE>
 --hash-function <HASH_FUNCTION>

Options:
```

```
      --key-filter <KEY_FILTER>
          The key reference or a list of key attributes to filter for a key to sign
 with

      --hash-function <HASH_FUNCTION>
          The hash function to use with ML-DSA. [possible values: shake256]

      --data <DATA>
          Data to be signed

      --data-path <DATA_PATH>
          Path to file containing data to be signed

      --data-type <DATA_TYPE>
          Specifies if the data is raw or has been processed using external mu.
 [possible values: raw, external-mu]

      --approval <APPROVAL>
          File path of signed quorum token file to approve operation. Necessary for
 keys with quorum enabled

  -h, --help
          Print help
```

Esempi

Esempio: firma i dati grezzi con una chiave privata ML-DSA

Usa questo comando per firmare i dati grezzi con una chiave ML-DSA privata.

```
aws-cloudhsm > crypto sign ml-dsa \
    --key-filter attr.label=ml-dsa-private-key-example \
    --data-type raw \
    --hash-function shake256 \
    --data YWJjMTIz
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000000f63",
    "signature": "+kO53l20YLjfUzDPbiJkFK/IWNUXobfqamJ7y2Tt7Nn3S6w3/h
+oPZm939ieNafVIwn1DsHyGm/
tLHFShZlFooi8alFyl76+VdXo5iJTmiBwYI5AK57ly8/2vj6zDRugnsLXbW40iMDlckWdZiwbYg/
yKfbHVFrYXywPkGbKk5d
+jZwvHlUAwnlc4MPAWKoauwkO86zlKjhuydihrMRBIPhUiXR21nkJ7dFMngzqHgBtwR08oPzMI5BLRpGL3euCFm816BWHzC
```

```
+Y5m0cPKbdjl/UTwFvh2UBytaqBu0wx6QcCsxsbZnBaEkspxYE62BuXZmM62B2VJsMG2j05MpaUULl6/
Yl2EeyqWEh3SqwAlN0QYRRqM4T3NkBwVfJ1JE5n4tgp91fL33iL4BoVeqStRJ0McA9+s
+EkYQ8UBM150CnHhqA47tMGmM0KfWPPx8H0xR7MqEaAAJup2D0SQfYfBbh4XKDVJpw1tpY/
hXlzEiWTTA6GVyYRVbsVGRDcxmyrzNf1DRSaHSgJksioSZ+xn0X8TKYnYp/
Fx80HnpO002RyQrzr+N8HvMemjVLhiY2eYYPQQsHSMZMZNbyG3SztiIYordUjc+6Ufwf
+fn9fbRzfBtkQJI7dJxgp518Z90FOzDigOq9p3S36W3K9RqUP9g3PeduucOm3YONdgl8TVpiuG0uC1JUUgUkh5yTNc49hPz
RKD3GVv6c2Fscj4KnL68j0N93aTCBpEWNv0hFWrIZJiRFinVx9hZPDeiFkomhMKR1+MjPeMY9hUE1lhOXUxbVSrtiMJBRjV
pi7pGKuNpm7VYbhRrbJ2hwPKiYgwkxhHf+SEoO5mkpzfQjADt5LOTUa+ZVlv5hWyp8v7YxDhV5Qv2jsII1h/
d1KX4A1BOrwS420/vT3haTKG1if9LphHziWdOFdRqJ40zt2MPaSZO9ARYN1Os1dqq
+Mf6dvzFHrcllcTrSlZvg2y31f0CP5IpjyV6mT3UjW0ccyIYTyaRTgcjJU7xj8qOHbEMbr8waGRSypNrBP4LJYCc6OJsljV
nGdGiPXCrQH6TH4ukV/dFaxueq9/WzFtHG6d6gQibytw1qrkYNSOv80cdyUNeZw6FJ5IlSvzM87N
+B18ghMo6J6yu/N6IJqNfyHtzRcgYtuo3O1bT5l+dvKoINw5ZXYW18ei/GXIVMn/z+TDZUl+TdUAUg6Q1C6zN
+TLX0Qdy+mirrjAvrrLT2aexOeVhqT8jgAcwQdZqnNg0nnp6IqNvepDtF1J5FnmRfkYCKxLXSXd/25q
+xeSMD3sVE9t8XwpjEyg9WGOLK7nBX8AgeeCFcYUyEIMq0sBZUNOeHWeY2R91Z6paj/W
+Is6FwHlP7dMe7zggiUxOk8oA8JWj9YlAI0KkijaBi047X78HJ/w5ywO5ybz663zBm7Cl7SOAARhGsPJwOSfbh/
U3Pq1ATNFv73O7dIoRY4E4czXIKHfKKkO1v4N5B6HneHlicCUAsz/
o78VuF36h5V29CsEL9qt0a01A1Q6jNigO4eTZ6BEqiRiM7LJoNjLrUn9ZK1czctr2hpjDXNi/
R+3zWwC1PqODNblqiQTV/2kxfjGIf+tIUsuPPPOzpRE3zgUrFcWBEpqBGIYMS0fBwN09Y
+RiE4jhtW+UM+rZB7lxSHUbAKme7Y5KRX9aIoRtmxJE5BnWnyeztZcQ02xBdTHSvXk
+fN2FkAU3rIPrsTaoTjmppmlwJoMaIO0DxSEsdGTpMgFwE
+B3Zc2StD/9zPCSHAC6lRHZL7BVqKOtm3Vc3CBzoH619HKu6jOteEDCRrNqfOzC8i7/
nv1GenadN6p7Ln+agE47/mOl2E/K254ULcl51eSbZpLpI+VDbN+yAFl40ErVxlWf
+YWg1RZziTnXCnHMWZOuJzqwsPuvaTCUTrh31MVSKvz0xGYZsepvsX8ho4xL003PaF9cOXnW9qgBryXb5JCQPbL8nFtxKcm
gPbDGOovkPwbAvRkYHjX6FbnysUqR7en0hAhiFsx0OZ90bRp1k5XYOW0YhC5degOoqpn9kTzbEnc58J9JnVq
+J5sHCaobp8yUXN6/cbM7l9+6G12oowTCbitkGOJ/
nZozeqnN8kdTw9HvVnOOMCD29ks38UEzFg1CaTgixLjjFMxgnN5twyK31HN0SCkcpmVkSnS/
oGZpMVqsfq5oIvN7+E3Ehi/KHHw52bhAgTiAq/ZTDpC8sVK0BwU2LgIXJUT/4dVTocOyRQb
+6fig4FaN4f9MGzB0cYiGKCACCZ+X7dtFgHzxqx9/
Qo4AQkmRODUiYs8STXKUF7igDBQsBEfPxsU6W24IgXrTz5JnIL9x/r2l2zz82ei2DWCxS4qlymz0Zs5H/
gjFsazYFbM8fYOEpCaB9vtuMSzVDILFFkSRVJrSAloG2vFu8fcjK5BA7a2h3uhBfaftmJFKiaEsCal
+mw0PzhDaMsuLgXXEm+RXuF1t8Bpiv+d47VjMV6sC6FaNi1gY+eH7/
oj1D3KC2xG2aSZjgdyhEm0uXLhCpRz1X7bpfELcxs1FqAGMLJFYGSOmUlrqrL4WROQT4FJ
+iZ0UFbqcujetJuWv9z+kBf9s/5r3ZoGd5VXNnNKQz7BV/
+wxjPk9Ui184ZHOOL2xM7BStdEDgTuMAbUsI5qDMCIE77g1JqZGWpeRuFf/
voJ2TNM7yUEikvf52UOihQbdiT6+feaK9urR4wcPDIwDB0/p8tObmVOCmKbhzVtXsnAjylpSk/
YHQ4fj6cxsMKSVz9IZihpga/DX12ki7Oacj/AJeGFIZYjBo/Kr2iEtoq5b0YQfC8Lp
+CItAelNzFzJ5IXbhmC8WtuhIk6ylQP5xyPofdlOJ1KBw46QckdxnMl3asO7rI4lHj2oXB4QiKVwgCbmvTw5U5O7NFB3rbp
UGTy7MHhf2hT9CbdJfxUUMXuv30nqc9xxkNxDozRw1KEAaG
+Ng/4p3WGmLRcSFj1JUHODkJyrubrD0djb5f4gNklKe4aNvsDF0tff4AUUQ0d/gYaLpdLr8/
sREhsfIzSgqKmyv9bi9QAAAAAAAAAAAAAAAAAAAAAAAAAAABIgLTs="
  }
}
```

Esempio: firma i dati mu esterni con una chiave ML-DSA privata

Usa questo comando per firmare dati mu esterni con una chiave ML-DSA privata.

```
aws-cloudhsm > crypto sign ml-dsa \
    --key-filter attr.label="MLDSA_private_key" \
    --data-type external-mu \
    --hash-function shake256 \
    --data DgFbkWgb9fbejQUSbJ09/
Nmp7phb714RTkWdNAwQqd9u7M7XCWbI8EoaIeGcZCUwIzaG6kDavZDYxCcaEicbTw==
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000000be2",
    "signature": "jxSKx9W/5Tu0lFySyrJPpNo4v/
tcNGfeC4Zt/0ubAAt7IQxH8MEjwcA7OO9AyaQntxDpzua8m+m5fEwiVrwmH
+zZhgw2EvXrFDU7zz6n6RRUI2RyXG6siistsBPwDZUU1G8oMp33k3E4LEL1al1AaYDncO9fQx1nS5wjg33OxzQbZi7GqRKB
+1MCPlSC3TshojRXlg2ELx32rRdzGhWAGZ/
A8qa9FIcS6LZlF35NZiJbK3W2mIFl8U5S2ZMXgIdfqm0wSNDibB4p23AR4SVz1Wghbu1To0pbeeaz74LxbZ3CIhpW
+UmtF8VZfGw/AbOQK2K1Z27VQuIc2Ru3L6PpSuDl4LM96ZdGPW6n/xjBMax83pAz9qwm
+cy/yjD28GCUKqwzNrjxB49ak6RGCYiUjCFG9UfEQ46xmJjOhvGaepf9XqjqmcO66rk
+mGThL96CGOpz1+QI91OQRSu3bDV9g2+Zl
+vEArvTUPLV5nbqZ598Uppswfg94n248S1P8M/5vkjiYBUD8asNcuQp2lQBXL4iDpuazewUyTLy2GADbwlfEYwxQ
+xELWxE4Gv8HMiQX2x97pC2BNjxlN1z4wMoPqsVlw
+G5xS2juTepUW3G9O0jgHvIVLuq2qC0XhSPRAvxWBE3iFVTDXwlpfaqjdRuhp5UwUfrL6m
+jKuLDRFxtzGFUmUfHtImSoZ0oNiQl+45Tlf+AZsYJno4chx5lQkVca7KI8jkvDLRqoxZFPb3T/
GlHoQ1R2y9EzB6LI0dCSAGWiU1kRDXcoXYls4Q6JfDZW6TRj0IAo0StVT7Au1S
+N+jPPKfpU7lJfROJglbt6gQM7rGy8naKsfEfBij+ufd89iKC+qwDP2WSx
+XsLMzfBBWHA8hJMGQHwEHeMvrjTJaSgKSP0M4pdcAV8g5A5TIaLgic4pszSVmc2PDmGw6ssAvS2M3SbnGCxUIcC7ziI74X
dy/uiZ8CirYhcvCH7jUsWR4J2JO+vAjnQJZZRXmUmmh3HHQoihV/P2kpmM8YxIVein1+
+gizJC9eijEuDh4rTr3SinO8vMoiclUG0+ah3yEKF6LjqtbWnNHY8Hnvm2w22yX1jjK4fsspsc9XbS3Og9un8qkaSpLphtz
f3CYiGLLFKupYjAKckU5J6t6LEdP8xoQF3alGVo/2lGQGIJrlg/qS2/7XpK5feeERtDxTCVZZhFj2w27OMsLTX/
Al0Koz8KCQFk5OWjQEfY3yicwP/+JB/ftQ
+5Uhp3CmOqSqNU7hRgL5e6QYi1E3D1KB8Je35Lz2PuzuClTttgWaExXe747fPvAbu3OJEjApdCYkaj0VokYhk2LH
+icpsCQfnMQjddei8WRyPMehp64PJRAXcCuX9qPiwfeCNxjplG+0PnY1SYKVRCQdbuuWORrAbLv8q7wSpLVB2A/
R0MISoZT6IGLiLxKbh7KZ7cSCV0rypD1+2Z7lYbvaUztP2CniJzqOuCbMDhlFzelbQaxY9UBlCiWjpxj7WShBnJuDD23Lhj
++W3mpaYBhUDMGMX0EWluSF7q2hAH8HcfMOUYDjaoV
+MoJupVb1yA6+NM5b6PmZR7RPiGPVBpoRR11GUAI9icFfYOr6gsc5yJZDsGtURz1PzpnSb21ZPlib9tHRW88kl
+2GijZes7DvyHhyavvfeO5M83uTCaYuohrVxKMJVewpW+O6nnPDxi
+dlWZ0NRQoU1CibRQiQi3ExDpmdQWMHHjkqVxxx6cnvn7wH60ztq5lsu3FoZ5gJSV8qabs3yjeh/
Tv5KlA7CaSVqcsYkqMEU2aT40Nysk2PO3v5b8ZmR9uU6YbQx7une+RZcmDhYXKtTG6NFaSNJjU3RwL805vaX/
hTVBc5o4jNK40JNIo1efrLzKp0V/cYk6/uBAoDAS4dbVhrf1GAyqlAbTc52dJJdJhzYaBJxddcdT
+CJAS8J1omVcPeMKn+Ny+Fy2sR53fHbP2E7OExbfYxToktPC4BRucIeFKWoYHYTSsYnennIbZBBfT4rPc7cA/
lWP94oxgjpCwbwQykLXeHh+MOgEDkk/Z+Gw/
```

```
xweTOWp6reiGUhERkXtniK2f3qjcRT03b9K6wNA7M08AKDSEtm3jT9KHhlJH9Tb1wOpPhUmj2MbIvbyG2yZknmiDIIwQmhM
PuG1yZMn9z+4tKFk6PYwbZON1GPPsVmhgKp+8AVIyLKVOoDNHX4io0u+Xk3+hAu5qU3kEJV29crhOCCJ8G2tBF
+y0Ho7S3ECAhGqFDxEKEQUFjXvU0gr8yGrqvgP1yC9ZxCu6TR5EESKtJc/
X2Acqyjg2Hx1IABe8ztmPT79FJyiAKDz5XsIfgBp0/
b1gBc3QV0VjYx4SvuYlVB0bFHqQmfs1bD2Ah9iCqPCe3qQli8q1T677shw07ijbyAYD6rfZqUDTwZr6EAQMC90v9sSaQY/5
FCquY2jZb5fTlDBk6TLYB8mMN2DJP+cDJ0FfhHOrt9/QIvrud2ZoldhAFPAFA841AYWb+BDVcZmLsIh
+RrFS0+vqyNyGN3TvOdxXxobFYuXA3QfcjcM6zMG+zcntK4b552QwKhVRzD9cvGB4AUM0ab0ivF
+8sfZVKtTYHepP/
FUiXk0QnMBpwydVrw0drz3dYRmvVHLSpJJemSVrT4hYmBrN0nMPWiiz9fANBHHC6F0TOwzeqmlrpXPXIiWOAp8ox/
fF9HcrULwkqJ5Bu6GVwXSCe19nFWVjt+8FPaa2B3mRaBgz71OLcpG4/LUD0rF0lUBvebl/
r8sEE5K2lhuak6Y/rg4/brTwVrFZdaYcmBUOSmboUpvox08HLY57XhJe8YoCQGQUezkL
+ZhXGMaYmphxmEp/4aWs6PvOfOWh849FCfQp9aEqfiX3d1rqNFK6NkLBSNEYf2D
+ikY/dLW98XUqDPHBipSlLY926ThO1Fc+BUFeeR9g6czyiB3K2Dcgi70cb61kAQ/
hHWNaaCdWfOdC7swjOlSlTMBo7CF6K
+FMA2uM9+8ZGOdlQOQUEgSutETmG9vNraGhC3qSZyhJBRtiDLSCxEQYu9BpgDDiAkK3ySpqu1urvD0uXs8fL5CiErOkBcZm
  }
}
```

Esempio: firma i dati da un percorso di file

Usa questo comando per firmare i dati di un file con una chiave ML-DSA privata.

```
aws-cloudhsm > crypto sign ml-dsa \
    --key-filter attr.label=ml-dsa-private-key-example \
    --data-type raw \
    --hash-function shake256 \
    --data-path /tmp/data.bin
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000000f63",
    "signature": "..."
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite dati)

<HASH_FUNCTION>

Specifica la funzione hash da utilizzare con. ML-DSA

Valori validi:

- shake256

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di ATTR.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE per selezionare una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un file di token quorum firmato per approvare un'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

Obbligatorio: no

<DATA_TYPE>

Specifica il tipo di dati forniti. Da utilizzare `raw` per dati senza hash; da utilizzare `external-mu` per dati mu esterni calcolati.

Valori validi:

- raw
- external-mu

Campo obbligatorio: sì

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- Verifica una firma firmata con il ML-DSA meccanismo nella CLI di CloudHSM
- Genera una coppia di ML-DSA chiavi asimmetrica con CloudHSM CLI

Genera una firma con il RSA-PKCS meccanismo della CLI di CloudHSM

Usa il crypto sign rsa-pkcs comando in CloudHSM CLI per generare una firma utilizzando una chiave privata RSA e il meccanismo di firma. RSA-PKCS

Per utilizzare il crypto sign rsa-pkcs comando, è necessario innanzitutto disporre di una chiave privata RSA nel cluster. AWS CloudHSM È possibile generare una chiave privata RSA utilizzando il Genera una coppia di key pair RSA asimmetrica con CloudhSM CLI comando con l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate AWS CloudHSM con La categoria di verifica crittografica in CloudHSM CLI sottocomandi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto sign rsa-pkcs
Sign with the RSA-PKCS mechanism

Usage: crypto sign rsa-pkcs --key-filter [<KEY_FILTER>>...] --hash-
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --hash-function <HASH_FUNCTION>
          [possible values: sha1, sha224, sha256, sha384, sha512]
      --data-path <DATA_PATH>
          The path to the file containing the data to be signed
      --data <DATA>
          Base64 Encoded data to be signed
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto sign rsa-pkcs generare una firma utilizzando il meccanismo di firma e la RSA-PKCS funzione SHA256 hash. Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per i dati codificati in base 64

```
aws-cloudhsm > crypto sign rsa-pkcs \
    --key-filter attr.label=rsa-private \
    --hash-function sha256 \
    --data YWJjMTIz
```

```
{
  "error_code": 0,
  "data": {
    "key-reference": "0x00000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}
```

Example Esempio: generare una firma per un file di dati

```
aws-cloudhsm > crypto sign rsa-pkcs \
    --key-filter attr.label=rsa-private \
    --hash-function sha256 \
    --data-path data.txt
{
  "error_code": 0,
  "data": {
    "key-reference": "0x00000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Per RSA-PKCS, i dati devono essere trasmessi nel formato codificato DER come specificato in RFC 8017, Sezione 9.2

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Genera una firma con il RSA-PKCS-PSS meccanismo della CLI di CloudHSM

Usa il crypto sign rsa-pkcs-pss comando in CloudHSM CLI per generare una firma utilizzando una chiave privata RSA e il meccanismo di firma. `RSA-PKCS-PSS`

Per utilizzare il crypto sign rsa-pkcs-pss comando, è necessario innanzitutto disporre di una chiave privata RSA nel cluster. AWS CloudHSM È possibile generare una chiave privata RSA utilizzando il Genera una coppia di key pair RSA asimmetrica con CloudhSM CLI comando con l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate AWS CloudHSM con La categoria di verifica crittografica in CloudHSM CLI sottocomandi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto sign rsa-pkcs-pss
Sign with the RSA-PKCS-PSS mechanism

Usage: crypto sign rsa-pkcs-pss [OPTIONS] --key-filter [<KEY_FILTER>...] --
hash-function <HASH_FUNCTION> --mgf <MGF> --salt-length <SALT_LENGTH> <--data-
path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>        Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]   Key reference (e.g. key-
reference=0xabc) or space separated list of key attributes in the form of
 attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key
      --hash-function <HASH_FUNCTION>  [possible values: sha1, sha224, sha256, sha384,
 sha512]
      --data-path <DATA_PATH>          The path to the file containing the data to be
 signed
      --data <DATA>                    Base64 Encoded data to be signed
      --mgf <MGF>                      The mask generation function [possible values:
 mgf1-sha1, mgf1-sha224, mgf1-sha256, mgf1-sha384, mgf1-sha512]
      --salt-length <SALT_LENGTH>      The salt length
      --approval <APPROVAL>            Filepath of signed quorum token file to approve
 operation
      --data-type <DATA_TYPE>          The type of data passed in, either raw or digest
 [possible values: raw, digest]
  -h, --help                           Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto sign rsa-pkcs-pss generare una firma utilizzando il meccanismo di firma e la `RSA-PKCS-PSS` funzione `SHA256` hash. Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per i dati codificati in base 64

```
aws-cloudhsm > crypto sign rsa-pkcs-pss \
    --key-filter attr.label=rsa-private \
    --hash-function sha256 \
    --data YWJjMTIz \
    --salt-length 10 \
```

```
    --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg=="
  }
}
```

Example Esempio: generare una firma per un file di dati

```
aws-cloudhsm > crypto sign rsa-pkcs-pss \
    --key-filter attr.label=rsa-private \
    --hash-function sha256 \
    --data-path data.txt \
    --salt-length 10 \
    --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "key-reference": "0x0000000000007008db",
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<MGF>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<SALT_LENGTH>

Specifica la lunghezza del sale.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di utilizzo delle chiavi della chiave privata è maggiore di 1.

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Argomenti correlati

- La categoria di verifica crittografica in CloudHSM CLI

La categoria di verifica crittografica in CloudHSM CLI

Nella CLI di CloudHSM, crypto verify è una categoria principale per un gruppo di comandi che, se combinata con la categoria principale, conferma se un file è stato firmato con una determinata chiave. crypto verifyha i seguenti sottocomandi:

- crypto verify ecdsa
- verifica crittografica ed25519
- verifica crittografica ed25519ph
- verifica crittografica ml-dsa
- verifica crittografica rsa-pkcs
- verifica crittografica rsa-pkcs-pss

Il crypto verify comando confronta un file firmato con un file sorgente e analizza se sono correlati crittograficamente in base a una determinata chiave pubblica e meccanismo di firma.

> **Note**
>
> È possibile accedere ai file con l'operazione. AWS CloudHSM La categoria delle firme crittografiche in CloudHSM CLI

Verifica una firma firmata con il meccanismo ECDSA nell'interfaccia CLI di CloudHSM

Usa il crypto verify ecdsa comando in CloudHSM CLI per completare le seguenti operazioni:

- Verifica che un file sia stato firmato nell'HSM con una determinata chiave pubblica.
- Verifica che la firma sia stata generata utilizzando il meccanismo di firma ECDSA.
- Confronta un file firmato con un file sorgente e determina se i due sono collegati crittograficamente in base a una determinata chiave pubblica ecdsa e a un determinato meccanismo di firma.
- La funzione di verifica ECDSA prevede la firma nel formato`r||s`, in cui i componenti r e s sono concatenati come dati binari grezzi.

Per utilizzare il crypto verify ecdsa comando, devi prima avere una chiave pubblica EC nel tuo cluster. AWS CloudHSM È possibile importare una chiave pubblica EC utilizzando il Importa una chiave in formato PEM con CloudHSM CLI comando con l'`verify`attributo impostato su`true`.

> **Note**
>
> È possibile generare una firma nella CLI di CloudHSM con sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto verify ecdsa
Verify with the ECDSA mechanism

Usage: crypto verify ecdsa --key-filter [<KEY_FILTER>...] --hash-
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>> <--signature-
path <SIGNATURE_PATH>|--signature <SIGNATURE>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --hash-function <HASH_FUNCTION>
          [possible values: sha1, sha224, sha256, sha384, sha512]
      --data-path <DATA_PATH>
          The path to the file containing the data to be verified
      --data <DATA>
          Base64 encoded data to be verified
      --signature-path <SIGNATURE_PATH>
```

```
          The path to where the signature is located
      --signature <SIGNATURE>
          Base64 encoded signature to be verified
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto verify ecdsa verificare una firma generata utilizzando il meccanismo di firma e la funzione hash ECDSA. SHA256 Questo comando utilizza una chiave pubblica nell'HSM.

Example Esempio: verifica una firma codificata Base64 con dati codificati Base64

```
aws-cloudhsm > crypto verify ecdsa \
    --hash-function sha256 \
    --key-filter attr.label=ec-public \
    --data YWJjMTIz \
    --signature 4zki+FzjhP7Z/KqoQvh4ueMAxQQVp7FQguZ2wOS3Q5bzk
+Hc5irV5iTkuxQbropPttVFZ8V6FgR2fz+sPegwCw==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica un file di firma con un file di dati

```
aws-cloudhsm > crypto verify ecdsa \
    --hash-function sha256 \
    --key-filter attr.label=ec-public \
    --data-path data.txt \
    --signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
```

```
}
```

Example Esempio: dimostrare una falsa relazione di firma

Questo comando verifica se i dati presenti in sono `/home/data` stati firmati da una chiave pubblica con l'etichetta `ecdsa-public` utilizzando il meccanismo di firma ECDSA per produrre la firma situata in. `/home/signature` Poiché gli argomenti forniti non costituiscono una vera relazione di firma, il comando restituisce un messaggio di errore.

```
aws-cloudhsm > crypto verify ecdsa \
    --hash-function sha256 \
    --key-filter attr.label=ec-public \
    --data aW52YWxpZA== \
    --signature +ogk7M7S3iTqFg3SndJfd91dZFr5Qo6YixJl8JwcvqqVgsVuO6o+VKvTRjz0/
V05kf3JJbBLr87Q+wLWcMAJfA==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<SIGNATURE>

Firma codificata in Base64.

Obbligatoria: Sì (a meno che non sia fornita tramite il percorso di firma)

<SIGNATURE_PATH>

Specifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Verifica una firma firmata con il meccanismo PureEd DSA nella CLI di CloudHSM

> **Important**
>
> PureEdLe operazioni di verifica della firma DSA sono supportate solo sulle istanze hsm2m.medium in modalità non FIPS.

Usa il crypto verify ed25519 comando in CloudHSM CLI per completare le seguenti operazioni:

- Verifica le firme dei dati o dei file utilizzando una determinata chiave pubblica Ed25519.
- Conferma che la firma è stata generata utilizzando il meccanismo di firma PureEd DSA (Ed25519).

> **Note**
>
> La dimensione massima dei messaggi per PureEd DSA è di 16.000 byte. Per messaggi di dimensioni maggiori, utilizzare. Verifica una firma firmata con il meccanismo HashEd DSA nella CLI di CloudhSM

Per utilizzare il crypto verify ed25519 comando, devi prima avere una chiave pubblica Ed25519 nel tuo AWS CloudHSM cluster. È possibile generare una coppia di chiavi Ed25519 utilizzando il Genera una coppia di key pair EC asimmetrica con CloudHSM CLI comando con il `curve` parametro impostato su `ed25519` e l'`verify`attributo impostato su`true`, oppure importare una chiave pubblica Ed25519 utilizzando il Importa una chiave in formato PEM con CloudHSM CLI comando con l'attributo impostato su. `verify` `true`

> **Note**
>
> È possibile generare una firma nella CLI di CloudHSM con sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.
- PureEdLe operazioni di verifica della firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.
- I dati del messaggio non devono superare i 16.000 byte.

Sintassi

```
aws-cloudhsm > help crypto verify ed25519
Verify with the Ed25519 mechanism

Usage: crypto verify ed25519 [OPTIONS] --key-filter [<KEY_FILTER>...] <--
data-path <DATA_PATH>|--data <DATA>> <--signature-path <SIGNATURE_PATH>|--
signature <SIGNATURE>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --data-path <DATA_PATH>
          The path to the file containing the data to be verified
      --data <DATA>
          Base64 encoded data to be verified
      --signature-path <SIGNATURE_PATH>
          The path to where the signature is located
      --signature <SIGNATURE>
          Base64 encoded signature to be verified
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto verify ed25519 verificare una firma generata utilizzando il meccanismo di firma PureEd DSA. Questo comando utilizza una chiave pubblica Ed25519 nell'HSM.

Example Esempio: verifica una firma codificata Base64 con dati codificati Base64

```
aws-cloudhsm > crypto verify ed25519 \
    --key-filter attr.label=ed25519-public \
    --data YWJj \
    --signature NjWz797ntYSLFwg7nKYYdn
+On3cCMj4zKz059wadVVlBHxyxe4JrSZxgekwb9AYR5xFxuVE9dTnDSo+gCaW/CQ==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica un file di firma con un file di dati

```
aws-cloudhsm > crypto verify ed25519 \
    --key-filter attr.label=ed25519-public \
    --data-path data.txt \
    --signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: dimostrare una relazione di falsa firma

Questo esempio mostra che la verifica ha esito negativo quando i dati, la firma o la chiave non corrispondono.

```
aws-cloudhsm > crypto verify ed25519 \
    --key-filter attr.label=ed25519-public \
    --data bWVzc2FnZQ== \
```

```
    --signature NjWz797ntYSLFwg7nKYYdn
+On3cCMj4zKz059wadVVlBHxyxe4JrSZxgekwb9AYR5xFxuVE9dTnDSo+gCaW/CQ==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da verificare. I dati decodificati non devono superare i 16.000 byte.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da verificare. Il contenuto del file non deve superare i 16.000 byte.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro data)

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di selezione di una chiave corrispondente.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta. Attributi chiave per la CLI di CloudHSM

Obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatoria: Sì (a meno che non sia fornita tramite il percorso di firma)

<SIGNATURE_PATH>

Specifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro di firma)

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI
- Genera una firma con il meccanismo PureEd DSA nella CLI di CloudHSM
- Verifica una firma firmata con il meccanismo HashEd DSA nella CLI di CloudhSM

Verifica una firma firmata con il meccanismo HashEd DSA nella CLI di CloudhSM

> **Important**
>
> HashEdLe operazioni di verifica della firma DSA sono supportate solo sulle istanze hsm2m.medium in modalità non FIPS.

Utilizzate il crypto verify ed25519ph comando nella CLI di CloudHSM per completare le seguenti operazioni:

- Verifica le firme di dati o file utilizzando una determinata chiave pubblica Ed25519.
- Conferma che la firma è stata generata utilizzando il meccanismo di firma HashEd DSA. Per ulteriori informazioni su HashEd DSA, vedere NIST SP 186-5, Sezione 7.8.

Per utilizzare il crypto verify ed25519ph comando, è necessario innanzitutto disporre di una chiave pubblica Ed25519 nel cluster. AWS CloudHSM È possibile generare una coppia di chiavi Ed25519 utilizzando il Genera una coppia di key pair EC asimmetrica con CloudHSM CLI comando con il `curve` parametro impostato su `ed25519` e l'`verify`attributo impostato su`true`, oppure importare una chiave pubblica Ed25519 utilizzando il Importa una chiave in formato PEM con CloudHSM CLI comando con l'`verify`attributo impostato su. `true`

> **Note**
>
> È possibile generare una firma nella CLI di CloudHSM con sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.
- HashEdLe operazioni di verifica della firma DSA sono supportate solo su istanze hsm2m.medium in modalità non FIPS.

Sintassi

```
aws-cloudhsm > help crypto verify ed25519ph
Verify with the Ed25519ph mechanism

Usage: crypto verify ed25519ph [OPTIONS] --key-filter [<KEY_FILTER>...] --data-
type <DATA_TYPE> --hash-function <HASH_FUNCTION> <--data-path <DATA_PATH>|--
data <DATA>> <--signature-path <SIGNATURE_PATH>|--signature <SIGNATURE>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --data-path <DATA_PATH>
          The path to the file containing the data to be verified
      --data <DATA>
          Base64 encoded data to be verified
      --signature-path <SIGNATURE_PATH>
          The path to where the signature is located
      --signature <SIGNATURE>
          Base64 encoded signature to be verified
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
```

```
      --hash-function <HASH_FUNCTION>
          Hash function [possible values: sha512]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come verificare una firma generata utilizzando crypto verify ed25519ph il meccanismo di firma e la funzione hash ED25519ph. `sha512` Questo comando utilizza una chiave pubblica Ed25519 nell'HSM.

Example Esempio: verifica una firma codificata Base64 con dati codificati Base64

```
aws-cloudhsm > crypto verify ed25519ph \
    --hash-function sha512 \
    --key-filter attr.label=ed25519-public \
    --data-type raw \
    --data YWJj \
    --signature mKcCIvC4Ehqp0w+BPWg/gJ5GK0acf/h2OUmbuU5trkEx
+FBCRjwqNVogA9BirfWqoQuMYeY2Biqq0RwqJgg0Bg==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica un file di firma con un file di dati

```
aws-cloudhsm > crypto verify ed25519ph \
    --hash-function sha512 \
    --key-filter attr.label=ed25519-public \
    --data-type raw \
    --data-path data.txt \
    --signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da verificare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Speciifica la posizione dei dati da verificare.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro data)

<HASH_FUNCTION>

Specifica la funzione hash. ED25519ph supporta solo. SHA512

Valori validi:

- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta. Attributi chiave per la CLI di CloudHSM

Obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatorio: Sì (a meno che non sia fornita tramite il percorso della firma)

<SIGNATURE_PATH>

Speciifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il parametro signature)

<DATA_TYPE>

Speciifica se il valore del parametro data deve essere sottoposto a hash come parte dell'algoritmo di verifica. Utilizzare `raw` per dati non sottoposti a hash; utilizzare `digest` per digest già sottoposti a hash.

Valori validi:

- raw
- digerire

Campo obbligatorio: sì

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI
- Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM

Verifica una firma firmata con il ML-DSA meccanismo nella CLI di CloudHSM

Usa il crypto verify ml-dsa comando nella CLI di CloudHSM per confermare l'integrità e l'autenticità dei dati dopo la firma. ML-DSA-based Con questo comando puoi:

- Conferma che una determinata chiave pubblica ha firmato un file nell'HSM.
- Verifica che il meccanismo di ML-DSA firma abbia generato la firma.
- Confronta un file firmato con un file sorgente per determinare se sono correlati crittograficamente in base a una determinata chiave ML-DSA pubblica.

Per utilizzare il crypto verify ml-dsa comando, è necessario innanzitutto disporre di una chiave ML-DSA pubblica nel AWS CloudHSM cluster con l'`verify`attributo impostato su. `true`

> **Note**
>
> A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

> **Note**
>
> È possibile generare una firma nella CLI di CloudHSM con sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

> **Note**
>
> Per utilizzare il tipo di `external-mu` dati, devi prima generare un mu digest a 64 byte. Per i passaggi, vedi Passaggi per generare mu esterno nella documentazione di sign ml-dsa.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto verify ml-dsa
Verifies a signature with a ML-DSA public key

Usage: crypto verify ml-dsa [OPTIONS] --key-filter <KEY_FILTER> --data-type <DATA_TYPE>
 --signature <SIGNATURE> --hash-function <HASH_FUNCTION>

Options:
      --cluster-id <CLUSTER_ID>
```

```
          Unique Id to choose which of the clusters in the config file to run the
operation against. If not provided, will fall back to the value provided when
interactive mode was started, or error

      --key-filter <KEY_FILTER>
          The key reference or a list of key attributes to filter for a key to verify
with

      --hash-function <HASH_FUNCTION>
          The hash function to use with ML-DSA. [possible values: shake256]

      --data <DATA>
          Data that was signed

      --data-path <DATA_PATH>
          Path to file containing data that was signed

      --data-type <DATA_TYPE>
          Specifies if the data is raw or processed using external mu [possible values:
raw, external-mu]

      --signature-path <SIGNATURE_PATH>
          The path to where the signature is located

      --signature <SIGNATURE>
          Base64 encoded signature to be verified

  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare per crypto verify ml-dsa verificare le firme generate dal meccanismo di firma. ML-DSA Questo comando utilizza una chiave pubblica nell'HSM.

Example Esempio: verifica una firma con dati non elaborati

Usa questo comando per verificare una firma per i dati grezzi con una chiave ML-DSA pubblica.

```
aws-cloudhsm > crypto verify ml-dsa \
    --key-filter attr.label=ml-dsa-public-key-example \
    --data-type raw \
    --hash-function shake256 \
    --data YWJjMTIz \
```

```
    --signature "+kO53l20YLjfUzDPbiJkFK/IWNUXobfqamJ7y2Tt7Nn3S6w3/h
+oPZm939ieNafVIwn1DsHyGm/
tLHFShZlFooi8alFyl76+VdXo5iJTmiBwYI5AK57ly8/2vj6zDRugnsLXbW40iMDlckWdZiwbYg/
yKfbHVFrYXywPkGbKk5d
+jZwvHlUAwnlc4MPAWKoauwkO86zlKjhuydihrMRBIPhUiXR21nkJ7dFMngzqHgBtwR08oPzMI5BLRpGL3euCFm816BWHzC
+Y5m0cPKbdjl/UTwFvh2UBytaqBu0wx6QcCsxsbZnBaEkspxYE62BuXZmM62B2VJsMG2j05MpaUULl6/
Yl2EeyqWEh3SqwAlN0QYRRqM4T3NkBwVfJ1JE5n4tgp91fL33iL4BoVeqStRJ0McA9+s
+EkYQ8UBM150CnHhqA47tMGmM0KfWPPx8H0xR7MqEaAAJup2D0SQfYfBbh4XKDVJpw1tpY/
hXlzEiWTTA6GVyYRVbsVGRDcxmyrzNf1DRSaHSgJksioSZ+xn0X8TKYnYp/
Fx80HnpO002RyQrzr+N8HvMemjVLhiY2eYYPQQsHSMZMZNbyG3SztiIYordUjc+6Ufwf
+fn9fbRzfBtkQJI7dJxgp518Z90FOzDigOq9p3S36W3K9RqUP9g3PeduucOm3YONdgl8TVpiuG0uC1JUUgUkh5yTNc49hPz
RKD3GVv6c2Fscj4KnL68j0N93aTCBpEWNv0hFWrIZJiRFinVx9hZPDeiFkomhMKR1+MjPeMY9hUE1lhOXUxbVSrtiMJBRjV
pi7pGKuNpm7VYbhRrbJ2hwPKiYgwkxhHf+SEoO5mkpzfQjADt5LOTUa+ZVlv5hWyp8v7YxDhV5Qv2jsII1h/
d1KX4A1BOrwS420/vT3haTKG1if9LphHziWdOFdRqJ40zt2MPaSZO9ARYN1Os1dqq
+Mf6dvzFHrcllcTrSlZvg2y31f0CP5IpjyV6mT3UjW0ccyIYTyaRTgcjJU7xj8qOHbEMbr8waGRSypNrBP4LJYCc6OJsljV
nGdGiPXCrQH6TH4ukV/dFaxueq9/WzFtHG6d6gQibytw1qrkYNSOv80cdyUNeZw6FJ5IlSvzM87N
+B18ghMo6J6yu/N6IJqNfyHtzRcgYtuo3O1bT5l+dvKoINw5ZXYW18ei/GXIVMn/z+TDZUl+TdUAUg6Q1C6zN
+TLX0Qdy+mirrjAvrrLT2aexOeVhqT8jgAcwQdZqnNg0nnp6IqNvepDtF1J5FnmRfkYCKxLXSXd/25q
+xeSMD3sVE9t8XwpjEyg9WGOLK7nBX8AgeeCFcYUyEIMq0sBZUNOeHWeY2R91Z6paj/W
+Is6FwHlP7dMe7zggiUxOk8oA8JWj9YlAI0KkijaBi047X78HJ/w5ywO5ybz663zBm7Cl7SOAARhGsPJwOSfbh/
U3Pq1ATNFv73O7dIoRY4E4czXIKHfKKkO1v4N5B6HneHlicCUAsz/
o78VuF36h5V29CsEL9qt0a01A1Q6jNigO4eTZ6BEqiRiM7LJoNjLrUn9ZK1czctr2hpjDXNi/
R+3zWwC1PqODNblqiQTV/2kxfjGIf+tIUsuPPPOzpRE3zgUrFcWBEpqBGIYMS0fBwN09Y
+RiE4jhtW+UM+rZB7lxSHUbAKme7Y5KRX9aIoRtmxJE5BnWnyeztZcQ02xBdTHSvXk
+fN2FkAU3rIPrsTaoTjmppmlwJoMaIO0DxSEsdGTpMgFwE
+B3Zc2StD/9zPCSHAC6lRHZL7BVqKOtm3Vc3CBzoH619HKu6jOteEDCRrNqfOzC8i7/
nv1GenadN6p7Ln+agE47/mOl2E/K254ULcl51eSbZpLpI+VDbN+yAFl40ErVxlWf
+YWg1RZziTnXCnHMWZOuJzqwsPuvaTCUTrh31MVSKvz0xGYZsepvsX8ho4xL003PaF9cOXnW9qgBryXb5JCQPbL8nFtxKcm
gPbDGOovkPwbAvRkYHjX6FbnysUqR7en0hAhiFsx0OZ90bRp1k5XYOW0YhC5degOoqpn9kTzbEnc58J9JnVq
+J5sHCaobp8yUXN6/cbM7l9+6G12oowTCbitkGOJ/
nZozeqnN8kdTw9HvVnOOMCD29ks38UEzFg1CaTgixLjjFMxgnN5twyK31HN0SCkcpmVkSnS/
oGZpMVqsfq5oIvN7+E3Ehi/KHHw52bhAgTiAq/ZTDpC8sVK0BwU2LgIXJUT/4dVTocOyRQb
+6fig4FaN4f9MGzB0cYiGKCACCZ+X7dtFgHzxqx9/
Qo4AQkmRODUiYs8STXKUF7igDBQsBEfPxsU6W24IgXrTz5JnIL9x/r2l2zz82ei2DWCxS4qlymz0Zs5H/
gjFsazYFbM8fYOEpCaB9vtuMSzVDILFFkSRVJrSAloG2vFu8fcjK5BA7a2h3uhBfaftmJFKiaEsCal
+mw0PzhDaMsuLgXXEm+RXuF1t8Bpiv+d47VjMV6sC6FaNi1gY+eH7/
oj1D3KC2xG2aSZjgdyhEm0uXLhCpRz1X7bpfELcxs1FqAGMLJFYGSOmUlrqrL4WROQT4FJ
+iZ0UFbqcujetJuWv9z+kBf9s/5r3ZoGd5VXNnNKQz7BV/
+wxjPk9Ui184ZHOOL2xM7BStdEDgTuMAbUsI5qDMCIE77g1JqZGWpeRuFf/
voJ2TNM7yUEikvf52UOihQbdiT6+feaK9urR4wcPDIwDB0/p8tObmVOCmKbhzVtXsnAjylpSk/
YHQ4fj6cxsMKSVz9IZihpga/DX12ki7Oacj/AJeGFIZYjBo/Kr2iEtoq5b0YQfC8Lp
+CItAelNzFzJ5IXbhmC8WtuhIk6ylQP5xyPofdlOJ1KBw46QckdxnMl3asO7rI4lHj2oXB4QiKVwgCbmvTw5U5O7NFB3rbp
UGTy7MHhf2hT9CbdJfxUUMXuv30nqc9xxkNxDozRw1KEAaG
```

```
+Ng/4p3WGmLRcSFj1JUHODkJyrubrD0djb5f4gNklKe4aNvsDF0tff4AUUQ0d/gYaLpdLr8/
sREhsfIzSgqKmyv9bi9QAAAAAAAAAAAAAAAAAAAAAAAAAAABIgLTs="
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica una firma con dati mu esterni

Usa questo comando per verificare una firma per i dati mu esterni con una chiave ML-DSA pubblica.

```
aws-cloudhsm > crypto verify ml-dsa \
    --key-filter attr.label=ml-dsa-public-key-example \
    --data-type external-mu \
    --hash-function shake256 \
    --data DgFbkWgb9fbejQUSbJ09/
Nmp7phb714RTkWdNAwQqd9u7M7XCWbI8EoaIeGcZCUwIzaG6kDavZDYxCcaEicbTw== \
    --signature "jxSKx9W/5Tu0lFySyrJPpNo4v/
tcNGfeC4Zt/0ubAAt7IQxH8MEjwcA7OO9AyaQntxDpzua8m+m5fEwiVrwmH
+zZhgw2EvXrFDU7zz6n6RRUI2RyXG6siistsBPwDZUU1G8oMp33k3E4LEL1al1AaYDncO9fQx1nS5wjg33OxzQbZi7GqRKB
+1MCPlSC3TshojRXlg2ELx32rRdzGhWAGZ/
A8qa9FIcS6LZlF35NZiJbK3W2mIFl8U5S2ZMXgIdfqm0wSNDibB4p23AR4SVz1Wghbu1To0pbeeaz74LxbZ3CIhpW
+UmtF8VZfGw/AbOQK2K1Z27VQuIc2Ru3L6PpSuDl4LM96ZdGPW6n/xjBMax83pAz9qwm
+cy/yjD28GCUKqwzNrjxB49ak6RGCYiUjCFG9UfEQ46xmJjOhvGaepf9XqjqmcO66rk
+mGThL96CGOpz1+QI91OQRSu3bDV9g2+Zl
+vEArvTUPLV5nbqZ598Uppswfg94n248S1P8M/5vkjiYBUD8asNcuQp2lQBXL4iDpuazewUyTLy2GADbwlfEYwxQ
+xELWxE4Gv8HMiQX2x97pC2BNjxlN1z4wMoPqsVlw
+G5xS2juTepUW3G9O0jgHvIVLuq2qC0XhSPRAvxWBE3iFVTDXwlpfaqjdRuhp5UwUfrL6m
+jKuLDRFxtzGFUmUfHtImSoZ0oNiQl+45Tlf+AZsYJno4chx5lQkVca7KI8jkvDLRqoxZFPb3T/
GlHoQ1R2y9EzB6LI0dCSAGWiU1kRDXcoXYls4Q6JfDZW6TRj0IAo0StVT7Au1S
+N+jPPKfpU7lJfROJglbt6gQM7rGy8naKsfEfBij+ufd89iKC+qwDP2WSx
+XsLMzfBBWHA8hJMGQHwEHeMvrjTJaSgKSP0M4pdcAV8g5A5TIaLgic4pszSVmc2PDmGw6ssAvS2M3SbnGCxUIcC7ziI74X
dy/uiZ8CirYhcvCH7jUsWR4J2JO+vAjnQJZZRXmUmmh3HHQoihV/P2kpmM8YxIVein1+
+gizJC9eijEuDh4rTr3SinO8vMoiclUG0+ah3yEKF6LjqtbWnNHY8Hnvm2w22yX1jjK4fsspsc9XbS3Og9un8qkaSpLphtz
f3CYiGLLFKupYjAKckU5J6t6LEdP8xoQF3alGVo/2lGQGIJrlg/qS2/7XpK5feeERtDxTCVZZhFj2w27OMsLTX/
Al0Koz8KCQFk5OWjQEfY3yicwP/+JB/ftQ
+5Uhp3CmOqSqNU7hRgL5e6QYi1E3D1KB8Je35Lz2PuzuClTttgWaExXe747fPvAbu3OJEjApdCYkaj0VokYhk2LH
+icpsCQfnMQjddei8WRyPMehp64PJRAXcCuX9qPiwfeCNxjplG+0PnY1SYKVRCQdbuuWORrAbLv8q7wSpLVB2A/
R0MISoZT6IGLiLxKbh7KZ7cSCV0rypD1+2Z7lYbvaUztP2CniJzqOuCbMDhlFzelbQaxY9UBlCiWjpxj7WShBnJuDD23Lhj
++W3mpaYBhUDMGMX0EWluSF7q2hAH8HcfMOUYDjaoV
+MoJupVb1yA6+NM5b6PmZR7RPiGPVBpoRR11GUAI9icFfYOr6gsc5yJZDsGtURz1PzpnSb21ZPlib9tHRW88kl
+2GijZes7DvyHhyavvfeO5M83uTCaYuohrVxKMJVewpW+O6nnPDxi
```

```
+dlWZ0NRQoU1CibRQiQi3ExDpmdQWMHHjkqVxxx6cnvn7wH60ztq5lsu3FoZ5gJSV8qabs3yjeh/
Tv5KlA7CaSVqcsYkqMEU2aT40Nysk2PO3v5b8ZmR9uU6YbQx7une+RZcmDhYXKtTG6NFaSNJjU3RwL805vaX/
hTVBc5o4jNK40JNIo1efrLzKp0V/cYk6/uBAoDAS4dbVhrf1GAyqlAbTc52dJJdJhzYaBJxddcdT
+CJAS8J1omVcPeMKn+Ny+Fy2sR53fHbP2E7OExbfYxToktPC4BRucIeFKWoYHYTSsYnennIbZBBfT4rPc7cA/
lWP94oxgjpCwbwQykLXeHh+MOgEDkk/Z+Gw/
xweTOWp6reiGUhERkXtniK2f3qjcRT03b9K6wNA7M08AKDSEtm3jT9KHhlJH9Tb1wOpPhUmj2MbIvbyG2yZknmiDIIwQmhM
PuG1yZMn9z+4tKFk6PYwbZON1GPPsVmhgKp+8AVIyLKVOoDNHX4io0u+Xk3+hAu5qU3kEJV29crhOCCJ8G2tBF
+y0Ho7S3ECAhGqFDxEKEQUFjXvU0gr8yGrqvgP1yC9ZxCu6TR5EESKtJc/
X2Acqyjg2Hx1IABe8ztmPT79FJyiAKDz5XsIfgBp0/
b1gBc3QV0VjYx4SvuYlVB0bFHqQmfs1bD2Ah9iCqPCe3qQli8q1T677shw07ijbyAYD6rfZqUDTwZr6EAQMC90v9sSaQY/5
FCquY2jZb5fTlDBk6TLYB8mMN2DJP+cDJ0FfhHOrt9/QIvrud2ZoldhAFPAFA841AYWb+BDVcZmLsIh
+RrFS0+vqyNyGN3TvOdxXxobFYuXA3QfcjcM6zMG+zcntK4b552QwKhVRzD9cvGB4AUM0ab0ivF
+8sfZVKtTYHepP/
FUiXk0QnMBpwydVrw0drz3dYRmvVHLSpJJemSVrT4hYmBrN0nMPWiiz9fANBHHC6F0TOwzeqmlrpXPXIiWOAp8ox/
fF9HcrULwkqJ5Bu6GVwXSCe19nFWVjt+8FPaa2B3mRaBgz71OLcpG4/LUD0rF0lUBvebl/
r8sEE5K2lhuak6Y/rg4/brTwVrFZdaYcmBUOSmboUpvox08HLY57XhJe8YoCQGQUezkL
+ZhXGMaYmphxmEp/4aWs6PvOfOWh849FCfQp9aEqfiX3d1rqNFK6NkLBSNEYf2D
+ikY/dLW98XUqDPHBipSlLY926ThO1Fc+BUFeeR9g6czyiB3K2Dcgi70cb61kAQ/
hHWNaaCdWfOdC7swjOlSlTMBo7CF6K
+FMA2uM9+8ZGOdlQOQUEgSutETmG9vNraGhC3qSZyhJBRtiDLSCxEQYu9BpgDDiAkK3ySpqu1urvD0uXs8fL5CiErOkBcZm
```

```
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica una firma dal percorso di un file

Usa questo comando per verificare una firma utilizzando i percorsi dei file per i dati e la firma.

```
aws-cloudhsm > crypto verify ml-dsa \
    --key-filter attr.label=ml-dsa-public-key-example \
    --data-path /tmp/data.bin \
    --data-type raw \
    --hash-function shake256 \
    --signature-path /tmp/signature.bin
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da verificare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da verificare.

Obbligatorio: Sì (a meno che non sia fornito tramite dati)

<HASH_FUNCTION>

Specifica la funzione hash da utilizzare con. ML-DSA

Valori validi:

- shake256

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<SIGNATURE>

Firma codificata in Base64.

Obbligatoria: Sì (a meno che non sia fornita tramite il percorso di firma)

<SIGNATURE_PATH>

Specifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite firma)

<DATA_TYPE>

Specifica il tipo di dati forniti. Da utilizzare `raw` per dati senza hash; da utilizzare `external-mu` per dati mu esterni calcolati.

Valori validi:

- raw
- external-mu

Campo obbligatorio: sì

Argomenti correlati

- La categoria di verifica crittografica in CloudHSM CLI
- Genera una firma con il ML-DSA meccanismo della CLI di CloudHSM
- Genera una coppia di ML-DSA chiavi asimmetrica con CloudHSM CLI

Verifica una firma firmata con il RSA-PKCS meccanismo nella CLI di CloudHSM

Usa il crypto verify rsa-pkcs comando nella CLI di CloudHSM per completare le seguenti operazioni:

- Verifica che un file sia stato firmato nell'HSM con una determinata chiave pubblica.
- Verifica che la firma sia stata generata utilizzando il meccanismo di `RSA-PKCS` firma.
- Confronta un file firmato con un file sorgente e determina se i due sono collegati crittograficamente in base a una determinata chiave pubblica RSA e a un determinato meccanismo di firma.

Per utilizzare il crypto verify rsa-pkcs comando, devi prima avere una chiave pubblica RSA nel tuo cluster. AWS CloudHSM

> **Note**
>
> È possibile generare una firma utilizzando l'interfaccia a riga di comando di CloudHSM con i sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto verify rsa-pkcs
Verify with the RSA-PKCS mechanism

Usage: crypto verify rsa-pkcs --key-filter [<KEY_FILTER>...] --hash-
function <HASH_FUNCTION> <--data-path <DATA_PATH>|--data <DATA>> <--signature-
path <SIGNATURE_PATH>|--signature <SIGNATURE>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --hash-function <HASH_FUNCTION>
          [possible values: sha1, sha224, sha256, sha384, sha512]
      --data-path <DATA_PATH>
          The path to the file containing the data to be verified
      --data <DATA>
          Base64 encoded data to be verified
      --signature-path <SIGNATURE_PATH>
```

```
          The path to where the signature is located
      --signature <SIGNATURE>
          Base64 encoded signature to be verified
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto verify rsa-pkcs verificare una firma generata utilizzando il meccanismo di firma e la funzione RSA-PKCS hash. SHA256 Questo comando utilizza una chiave pubblica nell'HSM.

Example Esempio: verifica una firma codificata Base64 con dati codificati Base64

```
aws-cloudhsm > crypto verify rsa-pkcs \
    --hash-function sha256 \
    --key-filter attr.label=rsa-public \
    --data YWJjMTIz \
    --signature XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica un file di firma con un file di dati

```
aws-cloudhsm > crypto verify rsa-pkcs \
    --hash-function sha256 \
    --key-filter attr.label=rsa-public \
    --data-path data.txt \
    --signature-path signature-file
{
  "error_code": 0,
```

```
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: dimostrare una falsa relazione di firma

Questo comando verifica se i dati non validi sono stati firmati da una chiave pubblica con l'etichetta `rsa-public` utilizzando il meccanismo di firma RSAPKCS per produrre la firma situata in. `/home/signature` Poiché gli argomenti forniti non costituiscono una vera relazione di firma, il comando restituisce un messaggio di errore.

```
aws-cloudhsm > crypto verify rsa-pkcs \
    --hash-function sha256 \
    --key-filter attr.label=rsa-public \
    --data aW52YWxpZA== \
    --signature XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<SIGNATURE>

Firma codificata in Base64.

Obbligatoria: Sì (a meno che non sia fornita tramite il percorso di firma)

<SIGNATURE_PATH>

Specifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Per RSA-PKCS, i dati devono essere trasmessi nel formato codificato DER come specificato in RFC 8017, Sezione 9.2

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Verifica una firma firmata con il RSA-PKCS-PSS meccanismo nella CLI di CloudHSM

Usa il crypto sign rsa-pkcs-pss comando in CloudHSM CLI per completare le seguenti operazioni.

- Verifica che un file sia stato firmato nell'HSM con una determinata chiave pubblica.
- Verifica che la firma sia stata generata utilizzando il meccanismo di RSA-PKCS-PSS firma.
- Confronta un file firmato con un file sorgente e determina se i due sono collegati crittograficamente in base a una determinata chiave pubblica RSA e a un determinato meccanismo di firma.

Per utilizzare il crypto verify rsa-pkcs-pss comando, devi prima avere una chiave pubblica RSA nel tuo cluster. AWS CloudHSM È possibile importare una chiave pubblica RSA utilizzando il Importa una chiave in formato PEM con CloudHSM CLI comando o il unwrap comando con l'`verify`attributo impostato su. `true`

> **Note**
>
> È possibile generare una firma utilizzando l'interfaccia a riga di comando di CloudHSM con i sottocomandi. La categoria delle firme crittografiche in CloudHSM CLI

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help crypto verify rsa-pkcs-pss
Verify with the RSA-PKCS-PSS mechanism

Usage: crypto verify rsa-pkcs-pss --key-filter [<KEY_FILTER>...] --hash-
function <HASH_FUNCTION> --mgf <MGF> --salt-length >SALT_LENGTH< <--data-
path <DATA_PATH>|--data <DATA> <--signature-path <SIGNATURE_PATH>|--
signature <SIGNATURE>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --key-filter [<KEY_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key
      --hash-function <HASH_FUNCTION>
          [possible values: sha1, sha224, sha256, sha384, sha512]
      --data-path <DATA_PATH>
          The path to the file containing the data to be verified
      --data <DATA>
          Base64 encoded data to be verified
      --signature-path <SIGNATURE_PATH>
          The path to where the signature is located
      --signature <SIGNATURE>
          Base64 encoded signature to be verified
      --data-type <DATA_TYPE>
          The type of data passed in, either raw or digest [possible values: raw,
 digest]
      --mgf <MGF>
          The mask generation function [possible values: mgf1-sha1, mgf1-sha224, mgf1-
sha256, mgf1-sha384, mgf1-sha512]
      --salt-length <SALT_LENGTH>
          The salt length
  -h, --help
```

```
          Print help
```

Esempio

Questi esempi mostrano come utilizzare per crypto verify rsa-pkcs-pss verificare una firma generata utilizzando il meccanismo di firma e la funzione RSA-PKCS-PSS hash. SHA256 Questo comando utilizza una chiave pubblica nell'HSM.

Example Esempio: verifica una firma codificata Base64 con dati codificati Base64

```
aws-cloudhsm > crypto verify rsa-pkcs-pss \
    --key-filter attr.label=rsa-public \
    --hash-function sha256 \
    --data YWJjMTIz \
    --salt-length 10 \
    --mgf mgf1-sha256 \
    --signature H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg==
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: verifica un file di firma con un file di dati

```
aws-cloudhsm > crypto verify rsa-pkcs-pss \
    --key-filter attr.label=rsa-public \
    --hash-function sha256 \
    --data-path data.txt \
    --salt-length 10 \
    --mgf mgf1-sha256 \
    --signature-path signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
```

```
}
```

Example Esempio: dimostrare una falsa relazione di firma

Questo comando verifica se i dati non validi sono stati firmati da una chiave pubblica con l'etichetta `rsa-public` utilizzando il meccanismo di firma RSAPKCSPSS per produrre la firma situata in. `/home/signature` Poiché gli argomenti forniti non costituiscono una vera relazione di firma, il comando restituisce un messaggio di errore.

```
aws-cloudhsm > crypto verify rsa-pkcs-pss \
    --key-filter attr.label=rsa-public \
    --hash-function sha256 \
    --data aW52YWxpZA== \
    --salt-length 10 \
    --mgf mgf1-sha256 \
    --signature H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg==
{
  "error_code": 1,
  "data": "Signature verification failed"
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specifica la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<KEY_FILTER>

Riferimento a una chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente.

Per un elenco degli attributi chiave CloudHSM CLI supportati, consulta Attributi chiave per CloudHSM CLI.

Obbligatorio: sì

<MFG>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256

- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatoria: Sì (a meno che non sia fornita tramite il percorso di firma)

<SIGNATURE_PATH>

Specifica la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

<DATA_TYPE>

Specifica se il valore del parametro data deve essere sottoposto a hashing come parte dell'algoritmo di firma. Da utilizzare `raw` per dati senza hash; da utilizzare `digest` per i riassunti, che sono già sottoposti a hash.

Valori validi:

- raw
- digerire

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

La categoria chiave dell'interfaccia CLI di CloudHSM

Nella CLI di CloudHSM, key è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per le chiavi. Attualmente, questa categoria comprende i seguenti comandi:

- Elimina
- Generazione-file

- Generazione chiavi-asimmetriche-coppia
  - Generazione chiavi-asimmetriche-coppia rsa
  - Generazione chiavi-asimmetriche-coppia ec
  - key generate-asimmetric-pair ml-dsa
- Generazione chiavi-simmetriche
  - Generazione chiavi-simmetriche aes
  - Generazione chiavi-simmetriche generiche-segrete
- importa pem
- elenco
- replicare
- Set-attributi
- Condivisione
- Annulla condivisione
- unwrap
- avvolgere

Eliminare una chiave con CloudHSM CLI

Utilizzate il key delete comando nella CLI di CloudHSM per eliminare una chiave da un cluster. AWS CloudHSM Puoi eliminare soltanto una chiave alla volta. L'eliminazione di una chiave di una coppia di chiavi non influisce sull'altra chiave della coppia.

Solo il CU che ha creato la chiave e di conseguenza la possiede può eliminarla. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni di crittografia, ma non possono eliminarla.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key delete
Delete a key in the HSM cluster

Usage: key delete [OPTIONS] --filter [<FILTER>...]

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --filter [<FILTER>...]     Key reference (e.g. key-reference=0xabc)
 or space separated list of key attributes in the form of
 attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key for deletion
  -h, --help                     Print help
```

Esempio

```
aws-cloudhsm > key delete --filter attr.label="ec-test-public-key"
{
  "error_code": 0,
  "data": {
    "message": "Key deleted successfully"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente da eliminare.

Per un elenco degli attributi delle chiavi supportati dalla CLI di CloudHSM, vedi Attributi chiave per la CLI di CloudHSM

Campo obbligatorio: sì

Argomenti correlati

- Elenca le chiavi per un utente con CLI CloudhSM
- Esporta una chiave asimmetrica con CloudHSM CLI
- Annullare la condivisione di una chiave utilizzando la CLI di CloudhSM
- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Esporta una chiave asimmetrica con CloudHSM CLI

Usa il key generate-file comando in CloudHSM CLI per esportare una chiave asimmetrica dal modulo di sicurezza hardware (HSM). Se la destinazione è una chiave privata, il riferimento alla chiave privata verrà esportato in un falso formato PEM. Se la destinazione è una chiave pubblica, i byte della chiave pubblica verranno esportati in formato PEM.

Il falso file PEM, che non contiene la chiave privata effettiva ma fa riferimento invece alla chiave privata nell'HSM, può essere utilizzato per stabilire l' SSL/TLS offload dal server Web a. AWS CloudHSM Per ulteriori informazioni, vedere offloading. SSL/TLS

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key generate-file
Generate a key file from a key in the HSM cluster. This command does not export any
 private key data from the HSM

Usage: key generate-file --encoding <ENCODING> --path <PATH> --filter [<FILTER>...]
```

```
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --encoding <ENCODING>
          Encoding format for the key file

          Possible values:
          - reference-pem:          PEM formatted key reference (supports private keys)
          - pem:                    PEM format (supports public keys)
          - cloudhsm-reference-pem: CloudHSM reference PEM format (supports private
 keys)

      --path <PATH>
          Filepath where the key file will be written

      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key for file generation

  -h, --help
          Print help (see a summary with '-h')
```

Esempi

Questi esempi mostrano come utilizzare key generate-file per generare un file chiave da una chiave nel AWS CloudHSM cluster.

Example Genera un file PEM di riferimento per una chiave privata RSA o EC

```
aws-cloudhsm > key generate-file --encoding reference-pem --path /tmp/ec-private-
key.pem --filter attr.label="ec-test-private-key"
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Example Genera un file PEM di riferimento CloudHSM per una chiave privata o Ed25519 ML-DSA

```
aws-cloudhsm > key generate-file --encoding cloudhsm-reference-pem --path /tmp/ed25519-private-key.pem --filter attr.label="ed25519-test-private-key"
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione di una chiave corrispondente da eliminare.

Per un elenco degli attributi chiave della CLI di CloudHSM supportati, vedi Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<ENCODING>

Specifica il formato di codifica per il file di chiavi

Obbligatorio: sì

<PATH>

Specifica il percorso del file in cui verrà scritto il file di chiavi

Obbligatorio: sì

Generazione di riferimenti chiave KSP (Windows)

> **Note**
>
> Questa funzionalità è disponibile solo nella versione SDK 5.16.0 e successive.

Prerequisiti

- È possibile generare riferimenti chiave KSP solo su piattaforme Windows.
- Devi accedere come utente crittografico (CU).

Posizione del file

Per impostazione predefinita, AWS CloudHSM archivia i file generati in: `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`

Per specificare una posizione diversa, utilizza il `--path` parametro.

Sintassi

```
aws-cloudhsm > help key generate-file --encoding ksp-key-reference
Generate a key file from a key in the HSM cluster. This command does not export any
 private key data from the HSM

Usage: key generate-file --encoding <ENCODING> --path <PATH> --filter [<FILTER>...]

Options:
      --encoding <ENCODING>
        Encoding format for the key file

        Possible values:
        - reference-pem:     PEM formatted key reference (supports private keys)
        - pem:               PEM format (supports public keys)
        - ksp-key-reference: KSP key reference format

      --cluster-id <CLUSTER_ID>
        Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided with multiple clusters configured, will error

      --path <PATH>
        Directory path where the key file will be written
```

```
      --filter [<FILTER>...]
        Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key for file generation

      --all
        Generate ksp key reference for all available key pairs in HSM

  -h, --help
        Print help (see a summary with '-h')
```

Esempio: generazione di un riferimento a una chiave KSP utilizzando un filtro di attributi di una chiave privata

L'esempio seguente genera un riferimento di chiave KSP per una chiave privata con un'etichetta specifica.

Example

```
aws-cloudhsm > key generate-file --encoding ksp-key-reference --path  --filter
 attr.label="ec-test-private-key"
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Esempio: generazione di riferimenti di chiave KSP per tutte le coppie di chiavi

L'esempio seguente genera riferimenti di chiave KSP per tutte le coppie di chiavi nel cluster.

Example

```
aws-cloudhsm > key generate-file --encoding ksp-key-reference --all
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM
- La categoria di generazione di coppie asimmetriche in CloudHSM CLI
- La categoria generate-symmetric nella CLI di CloudhSM

La categoria di generazione di coppie asimmetriche in CloudHSM CLI

Nella CLI di CloudHSM, key generate-asymmetric-pair è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando che genera coppie di chiavi asimmetriche. Attualmente, questa categoria comprende i seguenti comandi:

- Generazione chiavi-asimmetriche-coppia ec
- key generate-asimmetric-pair ml-dsa
- Generazione chiavi-asimmetriche-coppia rsa

Genera una coppia di key pair EC asimmetrica con CloudHSM CLI

Utilizza il key asymmetric-pair ec comando nella CLI di CloudHSM per generare una coppia di chiavi a curva ellittica (EC) asimmetrica nel cluster. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- CUsUtenti Crypto ()

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key generate-asymmetric-pair ec
Generate an Elliptic-Curve Cryptography (ECC) key pair

Usage: key generate-asymmetric-pair ec [OPTIONS] --public-label <PUBLIC_LABEL> --
private-label <PRIVATE_LABEL> --curve <CURVE>
```

```
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --public-label <PUBLIC_LABEL>
          Label for the public key
      --private-label <PRIVATE_LABEL>
          Label for the private key
      --session
          Creates a session key pair that exists only in the current session. The key
 cannot be recovered after the session ends
      --curve <CURVE>
          Elliptic curve used to generate the key pair [possible values: prime256v1,
 secp256r1, secp224r1, secp384r1, secp256k1, secp521r1, ed25519]
      --public-attributes [<PUBLIC_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated EC public key
 in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --private-attributes [<PRIVATE_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated EC private
 key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --share-crypto-users [<SHARE_CRYPTO_USERS>...]
          Space separated list of Crypto User usernames to share the EC private key
 with
      --manage-private-key-quorum-value <MANAGE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key management operations for the private key
      --use-private-key-quorum-value <USE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key usage operations for the private key
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il comando key generate-asymmetric-pair ec per creare una coppia di chiavi EC.

Example Esempio: Creare una coppia di chiavi EC

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve secp224r1 \
    --public-label ec-public-key-example \
    --private-label ec-private-key-example
{
```

```
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000012000b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 57,
```

```
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
        "curve": "secp224r1"
      }
    },
"private_key": {
      "key-reference": "0x000000000012000c",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-private-key-example",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
```

```
        "wrap-with-trusted": false,
        "key-length-bytes": 122,
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
        "curve": "secp224r1"
      }
    }
  }
}
```

Example Esempio: Creare una coppia di chiavi EC con attributi opzionali

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve secp224r1 \
    --public-label ec-public-key-example \
    --private-label ec-private-key-example \
    --public-attributes encrypt=true \
    --private-attributes decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "id": "",
        "check-value": "0xedef86",
        "class": "public-key",
```

```
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 57,
        "ec-point":
"0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
        "curve": "secp224r1"
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280c82",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-private-key-example",
        "id": "",
```

```
        "check-value": "0xedef86",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 122,
        "ec-point":
 "0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
        "curve": "secp224r1"
      }
    }
  }
}
```

Example Esempio: creazione di una coppia di chiavi EC con valori quorum

Quando si genera una chiave con controlli del quorum, la chiave deve essere associata a un numero minimo di utenti pari al valore del quorum chiave massimo. Gli utenti associati includono il proprietario della chiave e gli utenti Crypto con cui viene condivisa la chiave. Per determinare il numero minimo di utenti con cui condividere la chiave, ottieni il valore di quorum più alto tra il valore del quorum di utilizzo della chiave e il valore del quorum di gestione delle chiavi e sottrai 1 per tenere conto del proprietario della chiave, che per impostazione predefinita è associato alla chiave. Per condividere la chiave con più utenti, usa il comando. Condividi una chiave utilizzando la CLI di CloudHSM

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve secp224r1 \
    --public-label ec-public-key-example \
```

```
    --private-label ec-private-key-example \
    --public-attributes verify=true \
    --private-attributes sign=true
    --share-crypto-users cu2 cu3 cu4 \
    --manage-private-key-quorum-value 4 \
    --use-private-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-public-key-example",
        "id": "",
        "check-value": "0xedef86",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
```

```
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 57,
        "ec-point":
"0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
        "curve": "secp224r1"
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280c82",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu3",
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          },
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 4,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "ec-private-key-example",
        "id": "",
```

```
        "check-value": "0xedef86",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 122,
        "ec-point":
 "0x0487af31882189ec29eddf17a48e8b9cebb075b7b5afc5522fe9c83a029a450cc68592889a1ebf45f32240da514
        "curve": "secp224r1"
      }
    }
  }
}
```


Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<CURVE>

Specifica l'identificatore per la curva ellittica.

- prime256v1
- secp256r1
- secp224r1

- secp384r1
- secp256k1
- secp521r1
- ed25519 (supportato solo su istanze hsm2m.medium in modalità non FIPS)

Obbligatorio: sì

<PUBLIC_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi delle chiavi da impostare per la chiave pubblica EC generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `verify=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica. La dimensione massima consentita è di 127 caratteri per `label` Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<PRIVATE_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi delle chiavi da impostare per la chiave privata EC generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per impostazione predefinita, le chiavi generate sono chiavi persistenti (token). L'inserimento dell'argomento <SESSIONE> modifica la situazione, assicurando che una chiave generata con questo argomento sia una chiave di sessione (effimera).

Obbligatorio: no

<SHARE_CRYPTO_USERS>

Specifica un elenco separato da spazi di nomi utente Crypto User con cui condividere la chiave privata EC

Obbligatorio: no

<MANAGE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di gestione delle chiavi della chiave privata. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

<USE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di utilizzo delle chiavi private. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Genera una coppia di ML-DSA chiavi asimmetrica con CloudHSM CLI

Usa il key generate-asymmetric-pair ml-dsa comando in CloudHSM CLI per generare una coppia di chiavi asimmetrica nel tuo cluster. ML-DSA AWS CloudHSM ML-DSA è un algoritmo di firma digitale post-quantistica. Genera una coppia di ML-DSA chiavi quando devi firmare o verificare i dati utilizzando un algoritmo a resistenza quantistica.

> **Note**
>
> A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key generate-asymmetric-pair ml-dsa
Generate an ML-DSA key pair

Usage: key generate-asymmetric-pair ml-dsa [OPTIONS] --public-label <PUBLIC_LABEL> --
private-label <PRIVATE_LABEL> --mldsa-algorithm <MLDSA_ALGORITHM>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --public-label <PUBLIC_LABEL>
          Label for the public key
      --private-label <PRIVATE_LABEL>
          Label for the private key
      --session
```

```
          Creates a session key pair that exists only in the current session. The key
 cannot be recovered after the session ends
      --mldsa-algorithm <MLDSA_ALGORITHM>
          ML-DSA algorithm [possible values: ML-DSA-44, ML-DSA-65, ML-DSA-87]
      --public-attributes [<PUBLIC_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated ML-DSA public
 key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --private-attributes [<PRIVATE_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated ML-DSA
 private key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --share-crypto-users [<SHARE_CRYPTO_USERS>...]
          Space separated list of Crypto User usernames to share the ML-DSA key with
      --manage-private-key-quorum-value <MANAGE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key management operations for the private key
      --use-private-key-quorum-value <USE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key usage operations for the private key
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come usare per creare una coppia di chiavi. `key generate-asymmetric-pair ml-dsa` ML-DSA

Example Esempio: creare una coppia di ML-DSA-44 chiavi

```
aws-cloudhsm > key generate-asymmetric-pair ml-dsa \
    --mldsa-algorithm ML-DSA-44 \
    --public-label ml-dsa-public-key-example \
    --private-label ml-dsa-private-key-example
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000000240b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
```

```
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "ml-dsa-public-key-example",
        "id": "0x",
        "check-value": "0x988721",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "mldsa-algorithm": "ML-DSA-44",
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1312
      }
    },
    "private_key": {
      "key-reference": "0x0000000000000f63",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
```

```
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "ml-dsa-private-key-example",
        "id": "0x",
        "check-value": "0x988721",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "mldsa-algorithm": "ML-DSA-44",
        "sensitive": true,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 2560
      }
    }
  }
}
```

Example Esempio: creare una coppia di ML-DSA chiavi con attributi opzionali

```
aws-cloudhsm > key generate-asymmetric-pair ml-dsa \
    --mldsa-algorithm ML-DSA-65 \
    --public-label ml-dsa-public-key-example \
    --private-label ml-dsa-private-key-example \
```

```
    --public-attributes verify=true \
    --private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "ml-dsa-public-key-example",
        "mldsa-algorithm": "ML-DSA-65",
        "verify": true,
        "key-length-bytes": 1952
      }
    },
    "private_key": {
      "key-reference": "0x00000000002806ec",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
```

```
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "ml-dsa-private-key-example",
        "mldsa-algorithm": "ML-DSA-65",
        "sign": true,
        "key-length-bytes": 4032
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<MLDSA_ALGORITHM>

Specifica l' ML-DSA algoritmo.

- ML-DSA-44
- ML-DSA-65
- ML-DSA-87

Obbligatorio: sì

<PUBLIC_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi chiave da impostare per la chiave ML-DSA pubblica generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio,). `verify=true`

Per un elenco degli attributi di chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti ha un limite di 126 caratteri.

Obbligatorio: sì

<PRIVATE_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi chiave da impostare per la chiave ML-DSA privata generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio,). `sign=true`

Per un elenco degli attributi di chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti ha un limite di 126 caratteri.

Obbligatorio: sì

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Usa questo parametro quando hai bisogno di una chiave solo brevemente.

Per impostazione predefinita, l'HSM genera chiavi permanenti (token). Il trasferimento <SESSION>modifica questa situazione, assicurando che l'HSM generi invece una chiave di sessione (effimera).

Obbligatorio: no

<SHARE_CRYPTO_USERS>

Specifica un elenco separato da spazi di nomi utente Crypto con cui condividere la chiave privata. ML-DSA

Obbligatorio: no

<MANAGE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di gestione delle chiavi della chiave privata. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Sono inclusi gli utenti con cui la chiave è condivisa e il proprietario della chiave. Il valore massimo è 8.

Obbligatorio: no

<USE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di utilizzo della chiave privata. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Sono inclusi gli utenti con cui la chiave è condivisa e il proprietario della chiave. Il valore massimo è 8.

Obbligatorio: no

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Genera una firma con il ML-DSA meccanismo della CLI di CloudHSM
- Verifica una firma firmata con il ML-DSA meccanismo nella CLI di CloudHSM

Genera una coppia di key pair RSA asimmetrica con CloudhSM CLI

Usa il key generate-asymmetric-pair rsa comando nella CLI di CloudHSM per generare una coppia di key pair RSA asimmetrica nel tuo cluster. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- CUsUtenti Crypto ()

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key generate-asymmetric-pair rsa
Generate an RSA key pair

Usage: key generate-asymmetric-pair rsa [OPTIONS] --public-label <PUBLIC_LABEL>
 --private-label <PRIVATE_LABEL> --modulus-size-bits <MODULUS_SIZE_BITS> --public-
exponent <PUBLIC_EXPONENT>

Options:
```

```
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
operation against. If not provided, will fall back to the value provided when
interactive mode was started, or error
      --public-label <PUBLIC_LABEL>
          Label for the public key
      --private-label <PRIVATE_LABEL>
          Label for the private key
      --session
          Creates a session key pair that exists only in the current session. The key
cannot be recovered after the session ends
      --modulus-size-bits <MODULUS_SIZE_BITS>
          Modulus size in bits used to generate the RSA key pair
      --public-exponent <PUBLIC_EXPONENT>
          Public exponent used to generate the RSA key pair
      --public-attributes [<PUBLIC_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated RSA public
key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --private-attributes [<PRIVATE_KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated RSA private
key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --share-crypto-users [<SHARE_CRYPTO_USERS>...]
          Space separated list of Crypto User usernames to share the RSA key with
      --manage-private-key-quorum-value <MANAGE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key management operations for the private key
      --use-private-key-quorum-value <USE_PRIVATE_KEY_QUORUM_VALUE>
          The quorum value for key usage operations for the private key
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare `key generate-asymmetric-pair rsa` per creare una coppia di chiavi RSA.

Example Esempio: Creare una coppia di chiavi RSA

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example
{
```

```
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000160010",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
```

```
        "modulus":
 "0xdfca0669dc8288ed3bad99509bd21c7e6192661407021b3f4cdf4a593d939dd24f4d641af8e4e73b04c847731c6
e89a065e7d1a46ced96b46b909db2ab6be871ee700fd0a448b6e975bb64cae77c49008749212463e37a577baa57ce3e
bcebb7d20bd6df1948ae336ae23b52d73b7f3b6acc2543edb6358e08d326d280ce489571f4d34e316a2ea1904d513ca
        "modulus-size-bits": 2048
      }
    },
"private_key": {
      "key-reference": "0x0000000000160011",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": false,
        "trusted": false,
        "unwrap": false,
```

```
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
 "0xdfca0669dc8288ed3bad99509bd21c7e6192661407021b3f4cdf4a593d939dd24f4d641af8e4e73b04c847731c6
        "modulus-size-bits": 2048
      }
    }
  }
}
```

Example Esempio: Creare una coppia di chiavi RSA con attributi opzionali

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes encrypt=true \
--private-attributes decrypt=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
```

```
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa01
f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d510
133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e834697e
ac3160f0ca9725d38318b7",
        "modulus-size-bits": 2048
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280cc7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
```

```
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
        "modulus-size-bits": 2048
      }
    }
  }
}
```

Example Esempio: creare una coppia di key pair RSA con valori quorum

Quando si genera una chiave con controlli del quorum, la chiave deve essere associata a un numero minimo di utenti pari al valore del quorum della chiave massimo. Gli utenti associati includono il

proprietario della chiave e gli utenti Crypto con cui viene condivisa la chiave. Per determinare il numero minimo di utenti con cui condividere la chiave, ottieni il valore di quorum più alto tra il valore del quorum di utilizzo della chiave e il valore del quorum di gestione delle chiavi e sottrai 1 per tenere conto del proprietario della chiave, che per impostazione predefinita è associato alla chiave. Per condividere la chiave con più utenti, usa il comando. Condividi una chiave utilizzando la CLI di CloudHSM

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label rsa-public-key-example \
--private-label rsa-private-key-example \
--public-attributes verify=true \
--private-attributes sign=true
--share-crypto-users cu2 cu3 cu4 \
--manage-private-key-quorum-value 4 \
--use-private-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": false,
```

```
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa01
f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d510
133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e834697e
ac3160f0ca9725d38318b7",
        "modulus-size-bits": 2048
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280cc7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu3",
```

```
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          },
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 4,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1d4
        "modulus-size-bits": 2048
      }
    }
```

```
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<MODULUS_SIZE_BITS>

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Obbligatorio: sì

<PRIVATE_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi di chiavi da impostare per la chiave privata RSA generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`)

Per un elenco degli attributi di chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<PUBLIC_EXPONENT>

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Obbligatorio: sì

<PUBLIC_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi chiave da impostare per la chiave pubblica RSA generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `verify=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per impostazione predefinita, le chiavi generate sono chiavi persistenti (token). L'inserimento dell'argomento <SESSIONE> modifica la situazione, assicurando che una chiave generata con questo argomento sia una chiave di sessione (effimera).

Obbligatorio: no

<SHARE_CRYPTO_USERS>

Specifica un elenco separato da spazi di nomi utente Crypto User con cui condividere la chiave privata RSA

Obbligatorio: no

<MANAGE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di gestione delle chiavi della chiave privata. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

<USE_PRIVATE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di utilizzo delle chiavi private. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

La categoria generate-symmetric nella CLI di CloudhSM

Nella CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando che genera chiavi key generate-symmetric simmetriche. Attualmente, questa categoria comprende i seguenti comandi:

- Generazione chiavi-simmetriche aes
- Generazione chiavi-simmetriche generiche-segrete

Genera una chiave AES simmetrica con CloudhSM CLI

Usa il key generate-symmetric aes comando nella CLI di CloudHSM per generare una chiave AES simmetrica nel tuo cluster. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti CUs Crypto ()

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key generate-symmetric aes
```

```
Generate an AES key

Usage: key generate-symmetric aes [OPTIONS] --label <LABEL> --key-length-
bytes <KEY_LENGTH_BYTES>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --label <LABEL>
          Label for the key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --key-length-bytes <KEY_LENGTH_BYTES>
          Key length in bytes
      --attributes [<KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated AES key in
 the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --share-crypto-users [<SHARE_CRYPTO_USERS>...]
          Space separated list of Crypto User usernames to share the AES key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>
          The quorum value for key management operations
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE>
          The quorum value for key usage operations
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il comando key generate-symmetric aes per creare una chiave AES.

Example Esempio: creare una chiave AES

```
aws-cloudhsm > key generate-symmetric aes \
--label example-aes \
--key-length-bytes 24
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e06bf",
```

```
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-aes",
        "id": "",
        "check-value": "0x9b94bd",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 24
      }
    }
  }
}
```

Example Esempio: creare una chiave AES con attributi opzionali

```
aws-cloudhsm > key generate-symmetric aes \
--label example-aes \
--key-length-bytes 24 \
--attributes decrypt=true encrypt=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-aes",
        "id": "",
        "check-value": "0x9b94bd",
        "class": "secret-key",
        "encrypt": true,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
```

```
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 24
      }
    }
  }
}
```

Example Esempio: crea una chiave AES con valori di quorum

Quando si genera una chiave con controlli del quorum, la chiave deve essere associata a un numero minimo di utenti pari al valore del quorum chiave massimo. Gli utenti associati includono il proprietario della chiave e gli utenti Crypto con cui viene condivisa la chiave. Per determinare il numero minimo di utenti con cui condividere la chiave, ottieni il valore di quorum più alto tra il valore del quorum di utilizzo della chiave e il valore del quorum di gestione delle chiavi e sottrai 1 per tenere conto del proprietario della chiave, che per impostazione predefinita è associato alla chiave. Per condividere la chiave con più utenti, usa il comando. Condividi una chiave utilizzando la CLI di CloudHSM

```
aws-cloudhsm > key generate-symmetric aes \
--label example-aes \
--key-length-bytes 24 \
--attributes decrypt=true encrypt=true
--share-crypto-users cu2 cu3 cu4 \
--manage-key-quorum-value 4 \
--use-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e06bf",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
```

```
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu3",
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          },
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 4,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-aes",
        "id": "",
        "check-value": "0x9b94bd",
        "class": "secret-key",
        "encrypt": true,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 24
      }
```

```
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi delle chiavi da impostare per la chiave AES generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`).

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- 16, 24 e 32

Obbligatorio: sì

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave AES. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra

chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per impostazione predefinita, le chiavi generate sono chiavi persistenti (token). L'inserimento dell'argomento <SESSIONE> modifica la situazione, assicurando che una chiave generata con questo argomento sia una chiave di sessione (effimera).

Obbligatorio: no

<SHARE_CRYPTO_USERS>

Specifica un elenco separato da spazi di nomi utente Crypto User con cui condividere la chiave AES

Obbligatorio: no

<MANAGE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di gestione delle chiavi. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

<USE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di utilizzo delle chiavi. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Genera una chiave segreta generica simmetrica con la CLI di CloudhSM

Usa il key generate-symmetric generic-secret comando nella CLI di CloudHSM per generare una chiave segreta generica simmetrica nel tuo cluster. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- CUsUtenti Crypto ()

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > key help generate-symmetric generic-secret
Generate a generic secret key

Usage: key generate-symmetric generic-secret [OPTIONS] --label <LABEL> --key-length-
bytes <KEY_LENGTH_BYTES>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --label <LABEL>
          Label for the key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --key-length-bytes <KEY_LENGTH_BYTES>
          Key length in bytes
      --attributes [<KEY_ATTRIBUTES>...]
          Space separated list of key attributes to set for the generated generic
 secret key in the form of KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE
      --share-crypto-users [<SHARE_CRYPTO_USERS>...]
          Space separated list of Crypto User usernames to share the generic secret key
 with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE>
          The quorum value for key management operations
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE>
          The quorum value for key usage operations
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il comando key generate-symmetric generic-secret per creare una chiave generica segreta.

Example Esempio: creare una chiave generica segreta

```
aws-cloudhsm > key generate-symmetric generic-secret \
--label example-generic-secret \
--key-length-bytes 256
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
```

```
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 256
      }
    }
  }
}
```

Example Esempio: creare una chiave generica segreta con attributi opzionali

```
aws-cloudhsm > key generate-symmetric generic-secret \
--label example-generic-secret \
--key-length-bytes 256 \
--attributes encrypt=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
```

```
        "class": "secret-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 256
      }
    }
  }
}
```

Example Esempio: crea una chiave segreta generica con valori di quorum

Quando si genera una chiave con controlli quorum, la chiave deve essere associata a un numero minimo di utenti pari al valore del quorum della chiave massimo. Gli utenti associati includono il proprietario della chiave e gli utenti Crypto con cui viene condivisa la chiave. Per determinare il numero minimo di utenti con cui condividere la chiave, ottieni il valore di quorum più alto tra il valore del quorum di utilizzo della chiave e il valore del quorum di gestione delle chiavi e sottrai 1 per tenere conto del proprietario della chiave, che per impostazione predefinita è associato alla chiave. Per condividere la chiave con più utenti, usa il comando. Condividi una chiave utilizzando la CLI di CloudHSM

```
aws-cloudhsm > key generate-symmetric generic-secret \
--label example-generic-secret \
--key-length-bytes 256 \
--attributes encrypt=true
--share-crypto-users cu2 cu3 cu4 \
--manage-key-quorum-value 4 \
```

```
--use-key-quorum-value 2
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e08fd",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [
          {
            "username": "cu2",
            "key-coverage": "full"
          },
          {
            "username": "cu3",
            "key-coverage": "full"
          },
          {
            "username": "cu4",
            "key-coverage": "full"
          },
        ],
        "key-quorum-values": {
          "manage-key-quorum-value": 4,
          "use-key-quorum-value": 2
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
        "class": "secret-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
```

```
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 256
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi delle chiavi da impostare per la chiave AES generata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio `sign=true`).

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- Da 1 a 800

Obbligatorio: sì

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave generica segreta. La dimensione massima consentita `label` è di 127 caratteri per Client SDK 5.11 e versioni successive. Client SDK 5.10 e versioni precedenti hanno un limite di 126 caratteri.

Obbligatorio: sì

<SESSION>

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per impostazione predefinita, le chiavi generate sono chiavi persistenti (token). L'inserimento dell'argomento <SESSIONE> modifica la situazione, assicurando che una chiave generata con questo argomento sia una chiave di sessione (effimera).

Obbligatorio: no

<SHARE_CRYPTO_USERS>

Elenco separato da spazi di nomi utente Crypto User con cui condividere la chiave segreta generica

Obbligatorio: no

<MANAGE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di gestione delle chiavi. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

<USE_KEY_QUORUM_VALUE>

Il valore del quorum per le operazioni di utilizzo delle chiavi. Questo valore deve essere inferiore o uguale al numero di utenti a cui è associata la chiave. Ciò include gli utenti con cui viene condivisa la chiave e il proprietario della chiave. Valore massimo di 8.

Obbligatorio: no

Argomenti correlati

- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Importa una chiave in formato PEM con CloudHSM CLI

Utilizzate il key import pem comando in AWS CloudHSM per importare una chiave di formato PEM in un modulo di sicurezza hardware (HSM). È possibile utilizzarlo per importare le chiavi pubbliche generate al di fuori del HSM.

> **Note**
>
> ML-DSA le chiavi non possono essere importate utilizzando il comando. key import pem Per utilizzare ML-DSA le chiavi, generale direttamente sull'HSM utilizzando key generate-asimmetric-pair ml-dsa.

> **Note**
>
> Utilizzate il Esporta una chiave asimmetrica con CloudHSM CLI comando per creare un file PEM standard da una chiave pubblica o per creare un file PEM di riferimento da una chiave privata.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key import pem
Import key from a PEM file

Usage: key import pem [OPTIONS] --path <PATH> --label <LABEL> --key-type-
class <KEY_TYPE_CLASS>
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --path <PATH>
          Path where the key is located in PEM format
      --label <LABEL>
          Label for the imported key
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of the imported key [possible values: ec-public, rsa-
public]
      --attributes [<IMPORT_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the imported key
  -h, --help
          Print help
```

Esempi

Questo esempio mostra come utilizzare il key import pem comando per importare una chiave pubblica RSA da un file in formato PEM.

Example Esempio: importazione di una chiave pubblica RSA

```
aws-cloudhsm > key import pem --path /home/example --label example-imported-key --key-
type-class rsa-public
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001e08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
```

```
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
        "modulus":
"0x8e9c172c37aa22ed1ce25f7c3a7c936dadc532201400128b044ebb4b96#··3e4930ab910df5a2896eaeb8853cfe
        "modulus-size-bits": 2048
      }
    },
    "message": "Successfully imported key"
 }
```

```
}
```

Example Esempio: importazione di una chiave pubblica RSA con attributi opzionali

```
aws-cloudhsm > key import pem --path /home/example --label example-imported-key-with-
attributes --key-type-class rsa-public --attributes verify=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001e08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key-with-attributes",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
```

```
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
        "modulus":
 "0x8e9c172c37aa22ed1ce25f7c3a7c936dadc532201400128b044ebb4b96#··3e4930ab910df5a2896eaeb8853cfe
        "modulus-size-bits": 2048
      }
    },
    "message": "Successfully imported key"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PATH>

Specifica il percorso del file in cui si trova il file chiave.

Obbligatorio: sì

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave importata. La dimensione massima per l'`label` è 126 caratteri.

Obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe della chiave racchiusa.

Valori possibili:

- ec-public
- rsa-pubblico

Obbligatorio: sì

<IMPORT_KEY_ATTRIBUTES>

Specifica un elenco separato da spazi di attributi chiave da impostare per la chiave importata sotto forma di `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` (ad esempio,). `sign=true` Per un elenco degli attributi di chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: no

Argomenti correlati

- La categoria delle firme crittografiche in CloudHSM CLI
- La categoria di verifica crittografica in CloudHSM CLI

Elenca le chiavi per un utente con CLI CloudhSM

Utilizza il key list comando nella CLI di CloudHSM per trovare tutte le chiavi per l'utente corrente presente nel cluster. AWS CloudHSM L'output include le chiavi che l'utente possiede e condivide e tutte le chiavi pubbliche ne cluster del CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Amministratori () COs
- Utenti Crypto () CUs

Sintassi

```
aws-cloudhsm > help key list
List the keys the current user owns, shares, and all public keys in the HSM cluster

Usage: key list [OPTIONS]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
```

```
        Key reference (e.g. key-reference=0xabc) or space separated list of key
attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select
matching key(s) to list
     --max-items <MAX_ITEMS>
        The total number of items to return in the command's output. If the total
number of items available is more than the value specified, a next-token is provided
in the command's output. To resume pagination, provide the next-token value in the
starting-token argument of a subsequent command [default: 10]
     --starting-token <STARTING_TOKEN>
        A token to specify where to start paginating. This is the next-token from a
previously truncated response
 -v, --verbose
        If included, prints all attributes and key information for each matched key.
By default each matched key only displays its key-reference and label attribute. This
flag when used by Admins has no effect
 -h, --help
        Print help
```

Esempi

Nell'esempio seguente vengono illustrati tipi diversi di esecuzione del comando key list. Gli esempi seguenti mostrano gli output di un utente crittografico.

Example Esempio: Trova tutte le chiavi (impostazione predefinita)

Questo comando elenca le chiavi dell'utente registrato presente nel AWS CloudHSM cluster.

> **Note**
>
> Per impostazione predefinita, vengono visualizzate solo 10 chiavi dell'utente attualmente connesso e solo la `key-reference` e la `label` vengono visualizzate come output. Utilizza le opzioni di impaginazione appropriate per visualizzare più o meno chiavi come output.

```
aws-cloudhsm > key list
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000000003d5",
        "attributes": {
```

```
          "label": "test_label_1"
        }
      },
      {
        "key-reference": "0x0000000000000626",
        "attributes": {
          "label": "test_label_2"
        }
      },.
      ...8 keys later...
    ],
    "total_key_count": 56,
    "returned_key_count": 10,
    "next_token": "10"
  }
}
```

Example Esempio: Trova tutte le chiavi — verbose

L'output include le chiavi che l'utente possiede e condivide, oltre a tutte le chiavi pubbliche di. HSMs

> **Note**
>
> Nota: per impostazione predefinita, vengono visualizzate solo 10 chiavi dell'utente attualmente connesso. Utilizza le opzioni di impaginazione appropriate per visualizzare più o meno chiavi come output.

```
aws-cloudhsm > key list --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000012000c",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
```

```
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "ec",
          "label": "ec-test-private-key",
          "id": "",
          "check-value": "0x2a737d",
          "class": "private-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": false,
          "trusted": false,
          "unwrap": false,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 122,
          "ec-point":
"0x0442d53274a6c0ec1a23c165dcb9ccdd72c64e98ae1a9594bb5284e752c746280667e11f1e983493c1c605e0a80
          "curve": "secp224r1"
        }
      },
      {
        "key-reference": "0x000000000012000d",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
```

```
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "ec",
          "label": "ec-test-public-key",
          "id": "",
          "check-value": "0x2a737d",
          "class": "public-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": false,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": false,
          "sign": false,
          "trusted": false,
          "unwrap": false,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 57,
          "ec-point":
"0x0442d53274a6c0ec1a23c165dcb9ccdd72c64e98ae1a9594bb5284e752c746280667e11f1e983493c1c605e0a80
          "curve": "secp224r1"
        }
      }
    ],
      ...8 keys later...
    "total_key_count": 1580,
    "returned_key_count": 10
 }
```

```
}
```

Example Esempio: ritorno impaginato

L'esempio seguente mostra un sottoinsieme impaginato di chiavi che mostra solo due chiavi. L'esempio prevede quindi una chiamata successiva per visualizzare le due chiavi successive.

```
aws-cloudhsm > key list --verbose --max-items 2
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000000030",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "98a6688d1d964ed7b45b9cec5c4b1909",
          "id": "",
          "check-value": "0xb28a46",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
```

```
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": false,
          "verify": true,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      },
      {
        "key-reference": "0x0000000000000042",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "4ad6cdcbc02044e09fa954143efde233",
          "id": "",
          "check-value": "0xc98104",
          "class": "secret-key",
          "encrypt": true,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
```

```
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": true,
          "wrap": true,
          "wrap-with-trusted": false,
          "key-length-bytes": 16
        }
      }
    ],
    "total_key_count": 1580,
    "returned_key_count": 2,
    "next_token": "2"
  }
}
```

Per visualizzare le 2 chiavi successive, è possibile effettuare una chiamata successiva:

```
aws-cloudhsm > key list --verbose --max-items 2 --starting-token 2
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x0000000000000081",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
```

```
        "label": "6793b8439d044046982e5b895791e47f",
        "id": "",
        "check-value": "0x3f986f",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 32
      }
    },
    {
      "key-reference": "0x0000000000000089",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
      "key-quorum-values": {
        "manage-key-quorum-value": 0,
        "use-key-quorum-value": 0
      },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "56b30fa05c6741faab8f606d3b7fe105",
```

```
          "id": "",
          "check-value": "0xe9201a",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": false,
          "verify": true,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      }
    ],
    "total_key_count": 1580,
    "returned_key_count": 2,
    "next_token": "4"
  }
}
```

Per altri esempi che dimostrano come funziona il meccanismo di filtraggio chiave nella CLI di CloudHSM, vedi Filtrare le chiavi utilizzando la CLI di CloudHSM.

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire l'operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione delle chiavi corrispondenti da elencare.

Per un elenco degli attributi chiave della CLI di CloudHSM supportati, vedi Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<MAX_ITEMS>

Il numero totale di elementi da restituire nell'output del comando. Se il numero totale di elementi disponibili supera il valore specificato, viene fornito un token-successivo nell'output del comando. Per riprendere l'impaginazione, specifica il valore del token-successivo nell'argomento token-iniziale di un comando successivo.

Obbligatorio: no

<STARTING_TOKEN>

Token per specificare dove iniziare l'impaginazione. Si tratta del token-successivo da una risposta precedentemente troncata.

Obbligatorio: no

<VERBOSE>

Se incluso, stampa tutti gli attributi e le informazioni della chiave per ogni chiave abbinata. Per impostazione predefinita, ogni chiave abbinata mostra solo il riferimento alla chiave e l'etichetta dell'attributo. Questo flag, se usato dagli amministratori, non ha alcun effetto.

Obbligatorio: no

Argomenti correlati

- Eliminare una chiave con CloudHSM CLI
- Esporta una chiave asimmetrica con CloudHSM CLI
- Annullare la condivisione di una chiave utilizzando la CLI di CloudhSM
- Attributi chiave per la CLI di CloudHSM
- Filtrare le chiavi utilizzando la CLI di CloudHSM

Replica di una chiave con la CLI di CloudhSM

Utilizza il key replicate comando nella CLI di CloudHSM per replicare una chiave da AWS CloudHSM un cluster di origine a un cluster di destinazione. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Amministratori () COs
- Utenti Crypto () CUs

> **Note**
>
> Gli utenti Crypto devono possedere la chiave per utilizzare questo comando.

Requisiti

- I cluster di origine e di destinazione devono essere cloni. Ciò significa che uno è stato creato da un backup dell'altro o entrambi sono stati creati da un backup comune. Per ulteriori informazioni, consulta Creazione di cluster dai backup.
- Il proprietario della chiave deve esistere nel cluster di destinazione. Inoltre, se la chiave è condivisa con qualsiasi utente, tali utenti devono esistere anche nel cluster di destinazione.
- Per eseguire questo comando, devi accedere come utente crittografico o amministratore sia nel cluster di origine che in quello di destinazione.
  - In modalità a comando singolo, il comando utilizzerà le variabili ambientali CLOUDHSM_PIN e CLOUDHSM_ROLE per l'autenticazione nel cluster di origine. Per ulteriori informazioni, consulta Modalità di comando singolo. Per fornire le credenziali per il cluster di destinazione, è necessario impostare due variabili ambientali aggiuntive: DESTINATION_CLOUDHSM_PIN e DESTINATION_CLOUDHSM_ROLE:

    ```
    $ export DESTINATION_CLOUDHSM_ROLE=<role>
    ```

    ```
    $ export DESTINATION_CLOUDHSM_PIN=<username:password>
    ```

  - In modalità interattiva, gli utenti dovranno accedere in modo esplicito ai cluster di origine e di destinazione.

Sintassi

```
aws-cloudhsm > help key replicate
Replicate a key from a source to a destination cluster

Usage: key replicate --filter [<FILTER>...] --source-cluster-id <SOURCE_CLUSTER_ID> --
destination-cluster-id <DESTINATION_CLUSTER_ID>

Options:
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select
 matching key on the source cluster
      --source-cluster-id <SOURCE_CLUSTER_ID>
          Source cluster ID
      --destination-cluster-id <DESTINATION_CLUSTER_ID>
          Destination cluster ID
  -h, --help
          Print help
```

Esempi

Example Esempio: chiave di replica

Questo comando replica una chiave da un cluster di origine in un cluster di destinazione clonato. L'esempio seguente mostra l'output quando si accede come utente crittografico su entrambi i cluster.

```
crypto-user-1@cluster-1234abcdefg > key replicate \
      --filter attr.label=example-key \
      --source-cluster-id cluster-1234abcdefg \
      --destination-cluster-id cluster-2345bcdefgh
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x0000000000300006",
      "key-info": {
        "key-owners": [
          {
            "username": "crypto-user-1",
            "key-coverage": "full"
          }
        ],
```

```
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-key",
        "id": "0x",
        "check-value": "0x5e118e",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": true,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    },
    "message": "Successfully replicated key"
  }
}
```

Arguments (Argomenti)

<FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente nel cluster di origine.

Per un elenco degli attributi chiave della CLI di CloudHSM supportati, vedi Attributi chiave per la CLI di CloudHSM

Obbligatorio: sì

<SOURCE_CLUSTER_ID>

L'ID del cluster di origine.

Obbligatorio: sì

<DESTINATION_CLUSTER_ID>

L'ID del cluster di destinazione.

Campo obbligatorio: sì

Argomenti correlati

- Connessione a più cluster con la CLI CloudhSM

Imposta gli attributi delle chiavi con CloudhSM CLI

Usa il key set-attribute comando nella CLI di CloudHSM per impostare gli attributi delle chiavi nel tuo cluster. AWS CloudHSM Solo il CU che ha creato la chiave e di conseguenza la possiede può modificare gli attributi della chiave.

Per un elenco degli attributi delle chiavi che possono essere utilizzati nella CLI di CloudHSM, vedi Attributi chiave per la CLI di CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Gli utenti Crypto (CUs) possono eseguire questo comando.

- Gli admin possono impostare l'attributo affidabile.

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU. Per impostare l'attributo affidabile, è necessario aver effettuato l'accesso come utente admin.

Sintassi

```
aws-cloudhsm > help key set-attribute
Set an attribute for a key in the HSM cluster

Usage: cloudhsm-cli key set-attribute [OPTIONS] --filter [<FILTER>...] --
name <KEY_ATTRIBUTE> --value <KEY_ATTRIBUTE_VALUE>

Options:
      --cluster-id <CLUSTER_ID>         Unique Id to choose which of the clusters in
 the config file to run the operation against. If not provided, will fall back to the
 value provided when interactive mode was started, or error
      --filter [<FILTER>...]            Key reference (e.g. key-
reference=0xabc) or space separated list of key attributes in the form of
 attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a matching key to modify
      --name <KEY_ATTRIBUTE>            Name of attribute to be set
      --value <KEY_ATTRIBUTE_VALUE>...  Attribute value to be set
      --approval <APPROVAL>            Filepath of signed quorum token file to approve
 operation
  -h, --help                            Print help
```

Esempio: impostazione di un attributo della chiave

L'esempio seguente mostra come utilizzare il comando key set-attribute per impostare l'etichetta.

Example

1. Utilizza la chiave con l'etichetta `my_key`, come mostrato di seguito:

```
aws-cloudhsm > key set-attribute --filter attr.label=my_key --name encrypt --value
 false
{
  "error_code": 0,
  "data": {
    "message": "Attribute set successfully"
```

```
  }
}
```

2. Utilizza il comando key list per confermare che l'attributo `encrypt` è cambiato:

```
aws-cloudhsm > key list --filter attr.label=my_key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000006400ec",
        "key-info": {
          "key-owners": [
            {
              "username": "bob",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "my_key",
          "id": "",
          "check-value": "0x6bd9f7",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": true,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
```

```
          "sign": true,
          "trusted": true,
          "unwrap": true,
          "verify": true,
          "wrap": true,
          "wrap-with-trusted": false,
          "key-length-bytes": 32
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<KEY_ATTRIBUTE>

Specifica il nome dell'attributo della chiave.

Obbligatorio: sì

<FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente da eliminare.

Per un elenco degli attributi chiave della CLI di CloudHSM supportati, vedi Attributi chiave per la CLI di CloudHSM

Obbligatorio: no

<KEY_ATTRIBUTE_VALUE>

Specifica il valore dell'attributo della chiave.

Obbligatorio: sì

<KEY_REFERENCE>

Una rappresentazione esadecimale o decimale della chiave. (ad esempio un handle della chiave).

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave è maggiore di 1.

Argomenti correlati

- Filtrare le chiavi utilizzando la CLI di CloudHSM
- Attributi chiave per la CLI di CloudHSM

Condividi una chiave utilizzando la CLI di CloudHSM

Usa il key share comando nella CLI di CloudHSM per condividere una chiave CUs con altri membri del tuo cluster. AWS CloudHSM

Solo il CU che ha creato la chiave e di conseguenza la possiede può condividere la chiave. Gli utenti con cui viene condivisa una chiave possono utilizzare la chiave nelle operazioni crittografiche, ma non possono eliminare, esportare, condividere, annullare la condivisione, derivare o avvolgere la chiave. Inoltre, questi utenti non possono modificare gli attributi della chiave.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti crittografici () CUs

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key share
Share a key in the HSM cluster with another user

Usage: key share --filter [<FILTER>...] --username <USERNAME> --role <ROLE>
```

```
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key for sharing

      --username <USERNAME>
          A username with which the key will be shared

      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
          - admin:       An Admin has the ability to manage user accounts

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation

  -h, --help
          Print help (see a summary with '-h')
```

Esempio: condividere una chiave con un altro CU

L'esempio seguente mostra come utilizzare il comando key share per condividere una chiave con il CU `alice`.

Example

1. Esegui il comando key share per condividere la chiave con `alice`.

```
aws-cloudhsm > key share --filter attr.label="rsa_key_to_share" attr.class=private-
key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key shared successfully"
```

```
  }
}
```

2. Esegui il comando key list.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-
key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
```

```
            },
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
"0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
```

```
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

3. Nell'elenco precedente, la verifica `alice` è nell'elenco di `shared-users`

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente da eliminare.

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: sì

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il trattino basso (_). In questo comando il nome utente non è sensibile alle maiuscole e minuscole, il nome utente viene sempre visualizzato in minuscolo.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. Per ottenere il ruolo dell'utente, utilizzare il comando user list. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per CloudHSM CLI.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave è maggiore di 1.

Argomenti correlati

- Filtrare le chiavi utilizzando la CLI di CloudHSM
- Attributi chiave per la CLI di CloudHSM

Annullare la condivisione di una chiave utilizzando la CLI di CloudhSM

Utilizza il key unshare comando nella CLI di CloudHSM per annullare la condivisione di una chiave con altri membri del cluster. CUs AWS CloudHSM

Solo il CU che ha creato la chiave e di conseguenza la possiede può annullare la condivisione della chiave. Gli utenti con cui è condivisa la chiave possono utilizzarla in operazioni di crittografia, ma non possono esportarla, eliminarla, condividerla o annullarne la condivisione. Inoltre, questi utenti non possono modificare gli attributi della chiave.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unshare
Unshare a key in the HSM cluster with another user

Usage: key unshare --filter [<FILTER>...] --username <USERNAME> --role <ROLE>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 matching key for unsharing
```

```
    --username <USERNAME>
        A username with which the key will be unshared

    --role <ROLE>
        Role the user has in the cluster

        Possible values:
        - crypto-user: A CryptoUser has the ability to manage and use keys
        - admin:       An Admin has the ability to manage user accounts

    --approval <APPROVAL>
        Filepath of signed quorum token file to approve operation

-h, --help
        Print help (see a summary with '-h')
```

Esempio: annullare la condivisione di una chiave con un altro CU

L'esempio seguente mostra come utilizzare il comando key unshare per annullare la condivisione di una chiave con il CU `alice`.

Example

1. Esegui il comando key list e filtra in base alla specifica chiave di cui desideri annullare la condivisione con `alice`.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-
key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
```

```
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
            },
            {
              "username": "alice",
              "key-coverage": "full"
            }
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
```

```
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

2. Conferma che `alice` è presente nell'output `shared-users` ed esegui il seguente comando key unshare per annullare la condivisione della chiave con `alice`.

```
aws-cloudhsm > key unshare --filter attr.label="rsa_key_to_share"
 attr.class=private-key --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "message": "Key unshared successfully"
  }
}
```

3. Esegui nuovamente il comando `key list` per confermare che condivisione della chiave con `alice` è stata annullata.

```
aws-cloudhsm > key list --filter attr.label="rsa_key_to_share" attr.class=private-
key --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x00000000001c0686",
        "key-info": {
          "key-owners": [
            {
              "username": "cu3",
              "key-coverage": "full"
            }
          ],
          "shared-users": [
            {
              "username": "cu2",
              "key-coverage": "full"
            },
            {
              "username": "cu1",
              "key-coverage": "full"
            },
            {
              "username": "cu4",
              "key-coverage": "full"
            },
            {
              "username": "cu5",
              "key-coverage": "full"
            },
            {
              "username": "cu6",
              "key-coverage": "full"
            },
            {
              "username": "cu7",
              "key-coverage": "full"
            },
          ],
          "key-quorum-values": {
            "manage-key-quorum-value": 0,
```

```
            "use-key-quorum-value": 0
          },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "rsa",
          "label": "rsa_key_to_share",
          "id": "",
          "check-value": "0xae8ff0",
          "class": "private-key",
          "encrypt": false,
          "decrypt": true,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": true,
          "trusted": false,
          "unwrap": true,
          "verify": false,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 1219,
          "public-exponent": "0x010001",
          "modulus":
 "0xa8855cba933cec0c21a4df0450ec31675c024f3e65b2b215a53d2bda6dcd191f75729150b59b4d86df58254
          "modulus-size-bits": 2048
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave corrispondente da eliminare.

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiave per la CLI di CloudHSM.

Obbligatorio: sì

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il trattino basso (_). In questo comando il nome utente non è sensibile alle maiuscole e minuscole, il nome utente viene sempre visualizzato in minuscolo.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. Per ottenere il ruolo dell'utente, utilizzare il comando user list. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per CloudHSM CLI.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave è maggiore di 1.

Argomenti correlati

- Filtrare le chiavi utilizzando la CLI di CloudHSM
- Attributi chiave per la CLI di CloudHSM

Il comando key unwrap nella CLI di CloudHSM

Il comando key unwrap principale in CloudHSM CLI importa una chiave privata simmetrica o asimmetrica crittografata (racchiusa) da un file e nell'HSM. Questo comando è progettato per importare le chiavi crittografate incluse nel Il comando key wrap nella CLI di CloudHSM comando, ma può essere utilizzato anche per estrarre le chiavi che sono state racchiuse con altri strumenti. Tuttavia, in tali situazioni, ti consigliamo di utilizzare le librerie software PKCS # 11 o JCE per annullare il wrapping della chiave.

> **Note**
>
> ML-DSA le chiavi non possono essere decompresse. Per utilizzare ML-DSA le chiavi, generale direttamente sull'HSM utilizzando key generate-asimmetric-pair ml-dsa.

- aes-gcm
- aes-no-pad
- aes-pkcs5-pad
- aes-zero-pad
- cloudhsm-aes-gcm
- rsa-aes
- rsa-oaep
- rsa-pkcs

Estrarre una chiave con AES-GCM utilizzando la CLI di CloudhSM

Utilizzate il key unwrap aes-gcm comando nella CLI di CloudHSM per estrarre una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `AES-GCM`

Le chiavi non impacchettate possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap aes-gcm comando, è necessario disporre della chiave di wrapping AES nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap aes-gcm
Usage: key unwrap aes-gcm [OPTIONS] --filter [<FILTER>...] --tag-length-
bits <TAG_LENGTH_BITS> --key-type-class <KEY_TYPE_CLASS> --label <LABEL> --iv <IV> <--
data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --aad <AAD>
          Aes GCM Additional Authenticated Data (AAD) value, in hex
      --tag-length-bits <TAG_LENGTH_BITS>
```

```
          Aes GCM tag length in bits
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
 generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --iv <IV>
          Initial value used to wrap the key, in hex
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap aes-gcm comando utilizzando una chiave AES con il valore dell'`unwrap`attributo impostato su`true`.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap aes-gcm --key-type-class aes --label aes-unwrapped
 --filter attr.label=aes-example --tag-length-bits 64  --aad 0x10 --iv
 0xf90613bb8e337ec0339aad21 --data xvslgrtg8kHzrvekny97tLSIeokpPwV8
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e4",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
```

```
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap aes-gcm --key-type-class aes --label aes-unwrapped
 --filter attr.label=aes-example --tag-length-bits 64  --aad 0x10 --iv
 0xf90613bb8e337ec0339aad21 --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e4",
      "key-info": {
```

```
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<AAD>

Come valore AAD (Additional Authenticated Data) di GCM, in esadecimale.

Obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag Aes GCM in bit.

Obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3ec-private`,`generic-secret`]. `rsa-private`

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<IV>

Valore iniziale usato per racchiudere la chiave, in esadecimale.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave AES-NO-PAD utilizzando la CLI di CloudhSM

Utilizzate il key unwrap aes-no-pad comando nella CLI di CloudHSM per estrarre una chiave di payload AWS CloudHSM nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `AES-NO-PAD`

Le chiavi non incluse possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap aes-no-pad comando, è necessario disporre della chiave di wrapping AES nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap aes-no-pad
Usage: key unwrap aes-no-pad [OPTIONS] --filter [<FILTER>...] --key-type-
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
```

```
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
 -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap aes-no-pad comando utilizzando una chiave AES con il valore dell'`unwrap`attributo impostato su`true`.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap aes-no-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data eXK3PMAOnKM9y3YX6brbhtMoC060EOH9
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ec",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
```

```
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap aes-no-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ec",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3` `ec-private`generic-secret,`rsa-private`].

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave con AES- PKCS5 -PAD utilizzando la CLI CloudhSM

Utilizzate il key unwrap aes-pkcs5-pad comando nella CLI di CloudHSM per decomprimere una chiave di payload utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `AES-PKCS5-PAD`

Le chiavi non impacchettate possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap aes-pkcs5-pad comando, è necessario disporre della chiave di wrapping AES nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap aes-pkcs5-pad
Usage: key unwrap aes-pkcs5-pad [OPTIONS] --filter [<FILTER>...] --key-type-
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap aes-pkcs5-pad comando utilizzando una chiave AES con il valore dell'`unwrap`attributo impostato su`true`.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap aes-pkcs5-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data MbuYNresfOKyGNnxKWen88nSfX+uUE/0qmGofSisicY=
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
```

```
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap aes-pkcs5-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
```

```
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3` `ec-private`generic-secret,`rsa-private`].

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Recuperare una chiave AES-ZERO-PAD utilizzando CloudHSM CLI

Usa il key unwrap aes-zero-pad comando in CloudHSM CLI per estrarre una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. AWS CloudHSM `AES-ZERO-PAD`

Le chiavi non impacchettate possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap aes-zero-pad comando, è necessario disporre della chiave di wrapping AES nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap aes-zero-pad
Usage: key unwrap aes-zero-pad [OPTIONS] --filter [<FILTER>...] --key-type-
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
```

```
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap aes-zero-pad comando utilizzando una chiave AES con il valore dell'`unwrap`attributo `true` impostato su.

Example Esempio: Estrarre una chiave di payload dai dati della chiave racchiusa codificati in Base64

```
aws-cloudhsm > key unwrap aes-zero-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data L1wVlL/YeBNVAw6Mpk3owFJZXBzDLONt
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
```

```
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: Estrarre una chiave di payload fornita tramite un percorso di dati

```
aws-cloudhsm > key unwrap aes-zero-pad --key-type-class aes --label aes-unwrapped --
filter attr.label=aes-example --data-path payload-key.pem
{
  "error_code": 0,
```

```
  "data": {
    "key": {
      "key-reference": "0x00000000001c08e7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
```

```
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento chiave (ad esempio`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave con cui aprire la cartella.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati in Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo e classe della chiave racchiusa [valori possibili:`aes`,`des3`,`ec-private`,`generic-secret`,`rsa-private`].

Obbligatorio: sì

<LABEL>

Etichetta per la chiave non imballata.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore quorum del servizio di gestione delle chiavi della chiave unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave CLOUDHSM-AES-GCM utilizzando la CLI di CloudhSM

Utilizzate il key unwrap cloudhsm-aes-gcm comando nella CLI di CloudHSM per estrarre una chiave di payload AWS CloudHSM nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `CLOUDHSM-AES-GCM`

Le chiavi non impacchettate possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap cloudhsm-aes-gcm comando, è necessario disporre della chiave di wrapping AES nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap cloudhsm-aes-gcm
Usage: key unwrap cloudhsm-aes-gcm [OPTIONS] --filter [<FILTER>...] --tag-length-
bits <TAG_LENGTH_BITS> --key-type-class <KEY_TYPE_CLASS> --label <LABEL> <--data-
path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --aad <AAD>
          Aes GCM Additional Authenticated Data (AAD) value, in hex
      --tag-length-bits <TAG_LENGTH_BITS>
          Aes GCM tag length in bits
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
 generic-secret, rsa-private]
```

```
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap cloudhsm-aes-gcm comando utilizzando una chiave AES con il valore dell'`unwrap`attributo impostato su`true`.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap cloudhsm-aes-gcm --key-type-class aes --label aes-
unwrapped --filter attr.label=aes-example --tag-length-bits 64  --aad 0x10 --data
 6Rn8nkjEriDYlnP3P8nPkYQ8hplOEJ899zsrF+aTB0i/fIlZ
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001408e8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
```

```
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap cloudhsm-aes-gcm --key-type-class aes --label aes-unwrapped
 --filter attr.label=aes-example --tag-length-bits 64  --aad 0x10 --data-path payload-
key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001408e8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
```

```
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<AAD>

Come valore AAD (Additional Authenticated Data) di GCM, in esadecimale.

Obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag Aes GCM in bit.

Obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3ec-private`,`generic-secret`]. `rsa-private`

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave con RSA-AES utilizzando la CLI di CloudhSM

Utilizzate il key unwrap rsa-aes comando nella CLI di CloudHSM per decomprimere una chiave di payload utilizzando una chiave privata RSA e il meccanismo di unwrapping. `RSA-AES`

Le chiavi aperte possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare ilkey unwrap rsa-aes, è necessario disporre della chiave privata RSA della chiave di wrapping pubblica RSA nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

> **Note**
>
> Questo comando è disponibile solo con CloudHSM CLI 5.11+.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti CUs Crypto ()

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap rsa-aes
Usage: key unwrap rsa-aes [OPTIONS] --filter [<FILTER>...] --hash-
function <HASH_FUNCTION> --mgf <MGF> --key-type-class <KEY_TYPE_CLASS> --label <LABEL>
 <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --hash-function <HASH_FUNCTION>
          Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
      --mgf <MGF>
```

```
          Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
mgf1-sha256, mgf1-sha384, mgf1-sha512]
     --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
generic-secret, rsa-private]
     --label <LABEL>
          Label for the unwrapped key
     --session
          Creates a session key that exists only in the current session. The key cannot
be recovered after the session ends
     --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
 -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare il key unwrap rsa-aes comando utilizzando la chiave privata RSA con il valore dell'`unwrap`attributo `true` impostato su.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap rsa-aes --key-type-class aes --label aes-unwrapped
 --filter attr.label=rsa-private-key-example --hash-function sha256 --
mgf mgf1-sha256 --data HrSE1DEyLjIeyGdPa9R+ebiqB5TIJGyamPker31ZebPwRA
+NcerbAJO8DJ1lXPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4DOmU4meSofqREYvtb3EoIKwjy
+RL5WGXKe4nAboAkC5GO7veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VRl6ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
```

```
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap rsa-aes --key-type-class aes --label aes-unwrapped --filter
 attr.label=rsa-private-key-example --hash-function sha256 --mgf mgf1-sha256 --data-
path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
```

```
      "key-reference": "0x00000000001808e2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
```

```
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3` `ec-private`generic-secret,`rsa-private`].

Obbligatorio: sì

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<MGF>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave con RSA-OAEP utilizzando la CLI di CloudhSM

Utilizzate il key unwrap rsa-oaep comando nella CLI di CloudHSM per decomprimere una chiave di payload utilizzando la chiave privata RSA e il meccanismo di unwrapping. `RSA-OAEP`

Le chiavi aperte possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il key unwrap rsa-oaep comando, è necessario disporre della chiave privata RSA della chiave di wrapping pubblica RSA nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti crittografici () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap rsa-oaep
Usage: key unwrap rsa-oaep [OPTIONS] --filter [<FILTER>...] --hash-
function <HASH_FUNCTION> --mgf <MGF> --key-type-class <KEY_TYPE_CLASS> --label <LABEL>
 <--data-path <DATA_PATH>|--data <DATA>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --hash-function <HASH_FUNCTION>
          Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
      --mgf <MGF>
          Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
 mgf1-sha256, mgf1-sha384, mgf1-sha512]
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
 generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
```

```
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap rsa-oaep comando utilizzando la chiave privata RSA con il valore dell'`unwrap`attributo `true` impostato su.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap rsa-oaep --key-type-class aes --label aes-unwrapped --filter
 attr.label=rsa-private-example-key --hash-function sha256 --mgf mgf1-sha256 --data
 OjJe4msobPLz9TuSAdULEu17T5rMDWtSlLyBSkLbaZnYzzpdrhsbGLbwZJCtB/jGkDNdB4qyTAOQwEpggGf6v
+Yx6JcesNeKKNU8XZal/YBoHC8noTGUSDI2qr+u2tDc84NPv6d+F2KOONXsSxMhmxzzNG/
gzTVIJhOuy/B1yHjGP4mOXoDZf5+7f5M1CjxBmz4Vva/wrWHGCSG0yOaWblEvOiHAIt3UBdyKmU+/
My4xjfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCuSZ9yWThDT9as6i7e3htnyDhIhGWaoK8JU855cN/
YNKAUqkNpC4FPL3iw==
{
  "data": {
    "key": {
      "key-reference": "0x00000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
```

```
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap rsa-oaep --key-type-class aes --label aes-unwrapped --filter
 attr.label=rsa-private-example-key --hash-function sha256 --mgf mgf1-sha256 --data-
path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
```

```
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3` `ec-private`generic-secret,`rsa-private`].

Obbligatorio: sì

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Obbligatorio: sì

<MGF>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi:

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM

- Il comando key unwrap nella CLI di CloudHSM

Estrarre una chiave con RSA-PKCS utilizzando la CLI di CloudhSM

Utilizzate il key unwrap rsa-pkcs comando nella CLI di CloudHSM per decomprimere una chiave di payload utilizzando la chiave privata RSA e il meccanismo di unwrapping. `RSA-PKCS`

Le chiavi decomposte possono essere utilizzate nello stesso modo delle chiavi generate da. AWS CloudHSM Per indicare che non sono state generate localmente, il loro `local` attributo è impostato su. `false`

Per utilizzare il unwrap rsa-pkcs comando da tastiera, è necessario disporre della chiave privata RSA della chiave di wrapping pubblica RSA nel AWS CloudHSM cluster e il relativo `unwrap` attributo deve essere impostato su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key unwrap rsa-pkcs
Usage: key unwrap rsa-pkcs [OPTIONS] --filter [<FILTER>...] --key-type-
class <KEY_TYPE_CLASS> --label <LABEL> <--data-path <DATA_PATH>|--data <DATA>>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --filter [<FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a key
 to unwrap with
      --data-path <DATA_PATH>
```

```
          Path to the binary file containing the wrapped key data
      --data <DATA>
          Base64 encoded wrapped key data
      --attributes [<UNWRAPPED_KEY_ATTRIBUTES>...]
          Space separated list of key attributes in the form of
 KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE for the unwrapped key
      --share-crypto-users [<SHARE_CRYPTO_USERS;...]
          Space separated list of Crypto User usernames to share the unwrapped key with
      --manage-key-quorum-value <MANAGE_KEY_QUORUM_VALUE;
          The quorum value for key management operations for the unwrapped key
      --use-key-quorum-value <USE_KEY_QUORUM_VALUE;
          The quorum value for key usage operations for the unwrapped key
      --key-type-class <KEY_TYPE_CLASS>
          Key type and class of wrapped key [possible values: aes, des3, ec-private,
 generic-secret, rsa-private]
      --label <LABEL>
          Label for the unwrapped key
      --session
          Creates a session key that exists only in the current session. The key cannot
 be recovered after the session ends
      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap rsa-oaep comando utilizzando una chiave AES con il valore dell'`unwrap`attributo impostato su`true`.

Example Esempio: scartare una chiave di payload dai dati chiave avvolti codificati in Base64

```
aws-cloudhsm > key unwrap rsa-pkcs --key-type-class aes --label
 aes-unwrapped --filter attr.label=rsa-private-key-example --data
 am0Nc7+YE8FWs+5HvU7sIBcXVb24QA0l65nbNAD+1bK+e18BpSfnaI3P+r8Dp+pLu1ofoUy/
vtzRjZoCiDofcz4EqCFnGl4GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYRO1lGlfq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcpOOobJlITCoXhtMRepSdO4OggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg==
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ef",
```

```
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Example Esempio: scartare una chiave di payload fornita tramite un percorso dati

```
aws-cloudhsm > key unwrap rsa-pkcs --key-type-class aes --label aes-unwrapped --filter
 attr.label=rsa-private-key-example --data-path payload-key.pem
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000001c08ef",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "aes",
        "label": "aes-unwrapped",
        "id": "0x",
        "check-value": "0x8d9099",
        "class": "secret-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": false,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
```

```
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 16
      }
    }
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di selezione di una chiave da utilizzare.

Obbligatorio: sì

<DATA_PATH>

Percorso del file binario contenente i dati chiave racchiusi.

Obbligatorio: Sì (a meno che non sia fornito tramite dati codificati Base64)

<DATA>

Dati chiave avvolti codificati in Base64.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<ATTRIBUTES>

Elenco separato da spazi degli attributi chiave sotto forma `KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di chiave racchiusa.

Obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,, `des3` `ec-private`generic-secret,`rsa-private`].

Obbligatorio: sì

<LABEL>

Etichetta per la chiave aperta.

Obbligatorio: sì

<SESSION>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Obbligatorio: no

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di unwrapping è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Il comando key wrap nella CLI di CloudHSM

Il key wrap comando nella CLI di CloudHSM esporta una copia crittografata di una chiave privata simmetrica o asimmetrica dal modulo di sicurezza hardware (HSM) in un file. Quando si eseguekey wrap, si specificano due cose: la chiave da esportare e il file di output. La chiave da esportare è una chiave sull'HSM che crittograferà (avvolgerà) la chiave da esportare.

Il key wrap comando non rimuove la chiave dall'HSM né impedisce di utilizzarla nelle operazioni crittografiche. È possibile esportare la stessa chiave più volte. Per reimportare la chiave crittografata nell'HSM, utilizzare. Il comando key unwrap nella CLI di CloudHSM Solo il proprietario di una chiave,

ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti con cui la chiave è condivisa possono utilizzarla solo per operazioni crittografiche.

Il key wrap comando è composto dai seguenti sottocomandi:

- aes-gcm
- aes-no-pad
- aes-pkcs5-pad
- aes-zero-pad
- cloudhsm-aes-gcm
- rsa-aes
- rsa-oaep
- rsa-pkcs

Inserisci una chiave con AES-GCM utilizzando la CLI di CloudhSM

Utilizzate il key wrap aes-gcm comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave AES sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `AES-GCM` L'attributo della chiave di payload deve essere impostato su. `extractable true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap aes-gcm comando, è necessario innanzitutto disporre di una chiave AES nel AWS CloudHSM cluster. È possibile generare una chiave AES per il wrapping con il Genera una chiave AES simmetrica con CloudhSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap aes-gcm
Usage: key wrap aes-gcm [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-
filter [<WRAPPING_FILTER>...] --tag-length-bits <TAG_LENGTH_BITS>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
      --aad <AAD>
          Aes GCM Additional Authenticated Data (AAD) value, in hex
      --tag-length-bits <TAG_LENGTH_BITS>
          Aes GCM tag length in bits
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap aes-gcm comando utilizzando una chiave AES.

Example

```
aws-cloudhsm > key wrap aes-gcm --payload-filter attr.label=payload-key --wrapping-
filter attr.label=aes-example --tag-length-bits 64  --aad 0x10
{
  "error_code": 0,
  "data": {
```

```
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "iv": "0xf90613bb8e337ec0339aad21",
    "wrapped_key_data": "xvslgrtg8kHzrvekny97tLSIeokpPwV8"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<AAD>

Valore AES GCM Additional Authenticated Data (AAD), in esadecimale.

Obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag AES GCM in bit.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Speciifica il percorso del file di un token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Avvolgi una chiave AES-NO-PAD utilizzando la CLI di CloudhSM

Utilizzate il key wrap aes-no-pad comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave AES sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `AES-NO-PAD` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap aes-no-pad comando, è necessario innanzitutto disporre di una chiave AES nel AWS CloudHSM cluster. È possibile generare una chiave AES per il wrapping utilizzando il Genera una chiave AES simmetrica con CloudhSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap aes-no-pad
Usage: key wrap aes-no-pad [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-
filter [<WRAPPING_FILTER>...]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap aes-no-pad comando utilizzando una chiave AES con il valore dell'`wrap`attributo impostato su`true`.

Example

```
aws-cloudhsm > key wrap aes-no-pad --payload-filter attr.label=payload-key --wrapping-
filter attr.label=aes-example
{
```

```
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "eXK3PMAOnKM9y3YX6brbhtMoC060EOH9"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Specificate il percorso del file di un file token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Avvolgi una chiave con AES- PKCS5 -PAD utilizzando la CLI CloudhSM

Utilizzate il key wrap aes-pkcs5-pad comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave AES sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `AES-PKCS5-PAD` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap aes-pkcs5-pad comando, è necessario innanzitutto disporre di una chiave AES nel AWS CloudHSM cluster. È possibile generare una chiave AES per il wrapping utilizzando il Genera una chiave AES simmetrica con CloudhSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap aes-pkcs5-pad
```

```
Usage: key wrap aes-pkcs5-pad [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --
wrapping-filter [<WRAPPING_FILTER>...]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap aes-pkcs5-pad comando utilizzando una chiave AES con il valore dell'`wrap`attributo impostato su`true`.

Example

```
aws-cloudhsm > key wrap aes-pkcs5-pad --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "MbuYNresfOKyGNnxKWen88nSfX+uUE/0qmGofSisicY="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Specificate il percorso del file di un file token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM

- Il comando key unwrap nella CLI di CloudHSM

Avvolgi una chiave AES-ZERO-PAD utilizzando la CLI di CloudhSM

Utilizzate il key wrap aes-zero-pad comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave AES sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `AES-ZERO-PAD` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap aes-zero-pad comando, è necessario innanzitutto disporre di una chiave AES nel AWS CloudHSM cluster. È possibile generare una chiave AES per il wrapping utilizzando il Genera una chiave AES simmetrica con CloudhSM CLI comando con l'`wrap`attributo `true` impostato su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap aes-zero-pad
Usage: key wrap aes-zero-pad [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --
wrapping-filter [<WRAPPING_FILTER>...]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
```

```
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap aes-zero-pad comando utilizzando una chiave AES con il valore dell'`wrap`attributo impostato su`true`.

Example

```
aws-cloudhsm > key wrap aes-zero-pad --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "L1wVlL/YeBNVAw6Mpk3owFJZXBzDLONt"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Specificate il percorso del file di un file token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Avvolgi una chiave CLOUDHSM-AES-GCM utilizzando la CLI di CloudhSM

Utilizzate il key wrap cloudhsm-aes-gcm comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave AES sul modulo di sicurezza hardware (HSM) e

sul meccanismo di wrapping. `CLOUDHSM-AES-GCM` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap cloudhsm-aes-gcm comando, è necessario innanzitutto disporre di una chiave AES nel AWS CloudHSM cluster. È possibile generare una chiave AES per il wrapping con il Genera una chiave AES simmetrica con CloudhSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap cloudhsm-aes-gcm
Usage: key wrap cloudhsm-aes-gcm [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --
wrapping-filter [<WRAPPING_FILTER>...] --tag-length-bits <TAG_LENGTH_BITS>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
```

```
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
      --aad <AAD>
          Aes GCM Additional Authenticated Data (AAD) value, in hex
      --tag-length-bits <TAG_LENGTH_BITS>
          Aes GCM tag length in bits
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap cloudhsm-aes-gcm comando utilizzando una chiave AES.

Example

```
aws-cloudhsm > key wrap cloudhsm-aes-gcm --payload-filter attr.label=payload-key --
wrapping-filter attr.label=aes-example --tag-length-bits 64 --aad 0x10
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000001c08ea",
    "wrapped_key_data": "6Rn8nkjEriDYlnP3P8nPkYQ8hplOEJ899zsrF+aTB0i/fIlZ"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<AAD>

Valore AES GCM Additional Authenticated Data (AAD), in esadecimale.

Obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag AES GCM in bit.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Speciifica il percorso del file di un token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Inserisci una chiave con RSA-AES utilizzando la CLI di CloudhSM

Utilizzate il key wrap rsa-aes comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping RSA-AES. L'attributo della chiave payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap rsa-aes comando, è necessario innanzitutto disporre di una chiave RSA nel cluster AWS CloudHSM . È possibile generare una coppia di chiavi RSA utilizzando il La categoria di generazione di coppie asimmetriche in CloudHSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap rsa-aes
Usage: key wrap rsa-aes [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-
filter [<WRAPPING_FILTER>...] --hash-function <HASH_FUNCTION> --mgf <MGF>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
```

```
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
      --hash-function <HASH_FUNCTION>
          Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
      --mgf <MGF>
          Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
 mgf1-sha256, mgf1-sha384, mgf1-sha512]
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap rsa-ae comando utilizzando una chiave pubblica RSA con il valore dell'`wrap`attributo `true` impostato su.

Example

```
aws-cloudhsm > key wrap rsa-aes --payload-filter attr.label=payload-key --wrapping-
filter attr.label=rsa-public-key-example --hash-function sha256 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "payload-key-reference": "0x00000000001c08f1",
    "wrapping-key-reference": "0x00000000007008da",
    "wrapped-key-data": "HrSE1DEyLjIeyGdPa9R+ebiqB5TIJGyamPker31ZebPwRA
+NcerbAJO8DJ1lXPygZcI21vIFSZJuWMEiWpe1R9D/5WSYgxLVKex30xCFqebtEzxbKuv4DOmU4meSofqREYvtb3EoIKwjy
+RL5WGXKe4nAboAkC5GO7veI5yHL1SaKlssSJtTL/CFpbSLsAFuYbv/NUCWwMY5mwyVTCSlw+HlgKK
+5TH1MzBaSi8fpfyepLT8sHy2Q/VRl6ifb49p6m0KQFbRVvz/OWUd6l4d97BdgtaEz6ueg=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<MGF>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<WRAPPING_APPROVALR>

Speciifica il percorso del file di un token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Inserisci una chiave con RSA-OAEP utilizzando la CLI di CloudhSM

Utilizzate il key wrap rsa-oaep comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `RSA-OAEP` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap rsa-oaep comando, è necessario innanzitutto disporre di una chiave RSA nel cluster AWS CloudHSM . È possibile generare una coppia di chiavi RSA utilizzando il La categoria di generazione di coppie asimmetriche in CloudHSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap rsa-oaep
Usage: key wrap rsa-oaep [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-
filter [<WRAPPING_FILTER>...] --hash-function <HASH_FUNCTION> --mgf <MGF>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
      --hash-function <HASH_FUNCTION>
          Hash algorithm [possible values: sha1, sha224, sha256, sha384, sha512]
      --mgf <MGF>
          Mask Generation Function algorithm [possible values: mgf1-sha1, mgf1-sha224,
mgf1-sha256, mgf1-sha384, mgf1-sha512]
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap rsa-oaep comando utilizzando una chiave pubblica RSA con il valore dell'`wrap`attributo `true` impostato su.

Example

```
aws-cloudhsm > key wrap rsa-oaep --payload-filter attr.label=payload-key --wrapping-
filter attr.label=rsa-public-key-example --hash-function sha256 --mgf mgf1-sha256
{
  "error_code": 0,
  "data": {
    "payload-key-reference": "0x00000000001c08f1",
    "wrapping-key-reference": "0x00000000007008da",
    "wrapped-key-data": "OjJe4msobPLz9TuSAdULEu17T5rMDWtSlLyBSkLbaZnYzzpdrhsbGLbwZJCtB/
jGkDNdB4qyTAOQwEpggGf6v+Yx6JcesNeKKNU8XZal/YBoHC8noTGUSDI2qr+u2tDc84NPv6d
+F2KOONXsSxMhmxzzNG/gzTVIJhOuy/B1yHjGP4mOXoDZf5+7f5M1CjxBmz4Vva/
wrWHGCSG0yOaWblEvOiHAIt3UBdyKmU+/
My4xjfJv7WGGu3DFUUIZ06TihRtKQhUYU1M9u6NPf9riJJfHsk6QCuSZ9yWThDT9as6i7e3htnyDhIhGWaoK8JU855cN/
YNKAUqkNpC4FPL3iw=="
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.`KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE di una chiave di avvolgimento.

Obbligatorio: sì

<MGF>

Specifica la funzione di generazione della maschera.

> **Note**
>
> La funzione hash della funzione di generazione della maschera deve corrispondere alla funzione hash del meccanismo di firma.

Valori validi

- mgf1-sha1
- mgf1-sha224
- mgf1-sha256
- mgf1-sha384
- mgf1-sha512

Obbligatorio: sì

<WRAPPING_APPROVALR>

Speciifica il percorso del file di un token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Inserisci una chiave con RSA-PKCS utilizzando la CLI di CloudhSM

Utilizzate il key wrap rsa-pkcs comando nella CLI di CloudHSM per eseguire il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sul modulo di sicurezza hardware (HSM) e sul meccanismo di wrapping. `RSA-PKCS` L'attributo della chiave di payload deve essere impostato su. `extractable` `true`

Solo il proprietario di una chiave, ovvero l'utente crittografico (CU) che ha creato la chiave, può impacchettare la chiave. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni crittografiche.

Per utilizzare il key wrap rsa-pkcs comando, è necessario innanzitutto disporre di una chiave RSA nel cluster AWS CloudHSM . È possibile generare una coppia di chiavi RSA utilizzando il La categoria di generazione di coppie asimmetriche in CloudHSM CLI comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti Crypto () CUs

Requisiti

- Per eseguire questo comando, è necessario aver effettuato l'accesso come CU.

Sintassi

```
aws-cloudhsm > help key wrap rsa-pkcs
Usage: key wrap rsa-pkcs [OPTIONS] --payload-filter [<PAYLOAD_FILTER>...] --wrapping-
filter [<WRAPPING_FILTER>...]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
      --payload-filter [<PAYLOAD_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 payload key
```

```
      --wrapping-filter [<WRAPPING_FILTER>...]
          Key reference (e.g. key-reference=0xabc) or space separated list of key
 attributes in the form of attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE to select a
 wrapping key
      --path <PATH>
          Path to the binary file where the wrapped key data will be saved
      --wrapping-approval <WRAPPING_APPROVALR>
          File path of signed quorum token file to approve operation for wrapping key
      --payload-approval <PAYLOAD_APPROVALR>
          File path of signed quorum token file to approve operation for payload key
  -h, --help
          Print help
```

Esempio

Questo esempio mostra come utilizzare il key wrap rsa-pkcs comando utilizzando una chiave pubblica RSA.

Example

```
aws-cloudhsm > key wrap rsa-pkcs --payload-filter attr.label=payload-key --wrapping-
filter attr.label=rsa-public-key-example
{
  "error_code": 0,
  "data": {
    "payload_key_reference": "0x00000000001c08f1",
    "wrapping_key_reference": "0x00000000007008da",
    "wrapped_key_data": "am0Nc7+YE8FWs+5HvU7sIBcXVb24QA0l65nbNAD+1bK+e18BpSfnaI3P+r8Dp
+pLu1ofoUy/
vtzRjZoCiDofcz4EqCFnGl4GdcJ1/3W/5WRvMatCa2d7cx02swaeZcjKsermPXYRO1lGlfq6NskwMeeTkV8R7Rx9artFrs1
c3XdFJ2+0Bo94c6og/
yfPcpOOobJlITCoXhtMRepSdO4OggYq/6nUDuHCtJ86pPGnNahyr7+sAaSI3a5ECQLUjwaIARUCyoRh7EFK3qPXcg=="
  }
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PAYLOAD_FILTER>

Riferimento alla chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi degli attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di payload.

Obbligatorio: sì

<PATH>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Obbligatorio: no

<WRAPPING_FILTER>

Riferimento chiave (ad esempio,`key-reference=0xabc`) o elenco separato da spazi di attributi chiave sotto forma di selezione `attr.KEY_ATTRIBUTE_NAME=KEY_ATTRIBUTE_VALUE` di una chiave di avvolgimento.

Obbligatorio: sì

<WRAPPING_APPROVALR>

Specificate il percorso del file di un file token quorum firmato per approvare l'operazione di wrapping key. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di wrapping è maggiore di 1.

<PAYLOAD_APPROVALR>

Speciifica il percorso del file di un file di token quorum firmato per approvare l'operazione per la chiave di payload. Richiesto solo se il valore del quorum del servizio di gestione delle chiavi della chiave di payload è maggiore di 1.

Argomenti correlati

- Il comando key wrap nella CLI di CloudHSM
- Il comando key unwrap nella CLI di CloudHSM

Accedi a un HSM utilizzando CloudHSM CLI

È possibile utilizzare il login comando nella CLI di CloudHSM per accedere e disconnettersi da ogni sicurezza hardware (HSM) in un cluster. AWS CloudHSM Questo comando ha il seguente sottocomando:

- mfa-token-sign

> **Note**
>
> Se superi cinque tentativi di accesso errati, il tuo account viene bloccato. Per sbloccare l'account, un admin deve reimpostare la password utilizzando il comando user change-password nella cloudhsm_cli.

Per risolvere i problemi di accesso e disconnessione

Se disponi di più HSM nel cluster, potresti avere consentiti più tentativi di accesso errati prima che l'account venga bloccato. Questo perché il client CloudHSM bilancia il carico tra diversi tipi. HSMs Pertanto, il tentativo di accesso potrebbe non iniziare sullo stesso HSM ogni volta. Se stai testando questa funzionalità, ti consigliamo di farlo su un cluster con un solo HSM attivo.

Se il cluster è stato creato prima di febbraio 2018, l'account viene bloccato dopo 20 tentativi di accesso errati.

Tipo di utente

Gli utenti seguenti possono eseguire questi comandi.

- Admin non attivato
- Admin
- Crypto user (CU)

Sintassi

```
aws-cloudhsm > help login
Login to your cluster
```

```
USAGE:
    cloudhsm-cli login [OPTIONS] --username <USERNAME> --role <ROLE> [COMMAND]

Commands:
  mfa-token-sign  Login with token-sign mfa
  help            Print this message or the help of the given subcommand(s)

OPTIONS:
        --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

        --username <USERNAME>
            Username to access the Cluster

        --role <ROLE>
            Role the user has in the Cluster

           Possible values:
           - crypto-user: A CryptoUser has the ability to manage and use keys
           - admin:       An Admin has the ability to manage user accounts

        --password <PASSWORD>
           Optional: Plaintext user's password. If you do not include this argument you
 will be prompted for it

  -h, --help
          Print help (see a summary with '-h')
```

Esempio

Example

Questo comando consente di accedere a tutti gli HSMs utenti di un cluster con le credenziali di un utente amministratore denominato. `admin1`

```
aws-cloudhsm > login --username admin1 --role admin
Enter password:
{
  "error_code": 0,
  "data": {
```

```
    "username": "admin1",
    "role": "admin"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire l'operazione.

Obbligatorio: se sono stati configurati più cluster.

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il trattino basso (_). In questo comando il nome utente non è sensibile alle maiuscole e minuscole, il nome utente viene sempre visualizzato in minuscolo.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. I valori validi sono admin, crypto-user.

Per ottenere il ruolo dell'utente, usa il user list comando. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

Obbligatorio: sì

<PASSWORD>

Speciifica la password dell'utente che accede a. HSMs

Obbligatorio: i clienti devono fornire la password tramite l'argomento della `--password` riga di comando o ometterla per richiederla in modo interattivo.

> **Note**
>
> L'uso dell'`--password`argomento sulla riga di comando può esporre le credenziali nella cronologia della shell. Valuta la possibilità di richiedere la password in modo interattivo o di cancellare la cronologia della shell dopo l'uso.

Argomenti correlati

- Guida introduttiva alla CLI di CloudHSM
- Attivazione del cluster

Accedi con MFA a un HSM utilizzando CloudHSM CLI

Utilizza il login mfa-token-sign comando nella AWS CloudHSM CLI di CloudHSM per accedere a un modulo di sicurezza hardware (HSM) utilizzando l'autenticazione a più fattori (MFA). Per utilizzare questo comando, devi prima configurare MFA per la CLI del CloudHSM.

Tipo di utente

Gli utenti seguenti possono eseguire questi comandi.

- Admin
- Crypto user (CU)

Sintassi

```
aws-cloudhsm > help login mfa-token-sign
Login with token-sign mfa

USAGE:
    login --username <username> --role <role> mfa-token-sign --token <token>

OPTIONS:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --token <TOKEN>            Filepath where the unsigned token file will be written
  -h, --help                     Print help
```

Esempio

Example

```
aws-cloudhsm > login --username test_user --role admin mfa-token-sign --token /home/
valid.token
Enter password:
```

```
Enter signed token file path (press enter if same as the unsigned token file):
{
  "error_code": 0,
  "data": {
    "username": "test_user",
    "role": "admin"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<TOKEN>

Percorso del file in cui verrà scritto il file token non firmato.

Campo obbligatorio: sì

Argomenti correlati

- Guida introduttiva alla CLI di CloudHSM
- Attivazione del cluster
- Utilizzare la CLI di CloudHSM per gestire l'MFA

Esci da un HSM utilizzando CloudHSM CLI

Usa il logout comando nella CLI di CloudHSM per disconnetterti da ogni modulo di sicurezza hardware (HSM) in un cluster. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Crypto user (CU)

Sintassi

```
aws-cloudhsm > help logout
Logout of your cluster

USAGE:
    logout

OPTIONS:
        --cluster-id <CLUSTER_ID> Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
    -h, --help                    Print help information
    -V, --version                 Print version information
```

Esempio

Example

Questo comando ti disconnette da tutti gli utenti di un cluster. HSMs

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

Argomenti correlati

- Guida introduttiva alla CLI di CloudHSM
- Attivazione del cluster

La categoria di utenti nella CLI di CloudHSM

Nella user CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per gli utenti. Attualmente, la categoria utente è composta dai seguenti comandi:

- user change-mfa
- user change-password

- user create
- user delete
- user list
- replica utente

La categoria user change-mfa nella CLI di CloudhSM

Nella user change-mfa CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per modificare l'autenticazione a più fattori (MFA) per gli utenti.

Attualmente, questa categoria è composta dal seguente sottocomando:

- token-sign

Modificare la configurazione MFA di un utente con CloudhSM CLI

Utilizza il user change-mfa token-sign comando nella CLI di CloudHSM per aggiornare la configurazione dell'autenticazione a più fattori (MFA) di un account utente. Qualsiasi account utente può eseguire questo comando. Gli account con il ruolo di admin possono eseguire questo comando per altri utenti.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Crypto user

Sintassi

Attualmente, è disponibile una sola strategia a più fattori per gli utenti: Token Sign.

```
aws-cloudhsm > help user change-mfa
Change a user's Mfa Strategy

Usage:
    user change-mfa <COMMAND>
```

```
Commands:
  token-sign  Register or Deregister a public key using token-sign mfa strategy
  help        Print this message or the help of the given subcommand(s)
```

La strategia Token Firma richiede un file Token su cui scrivere token non firmati.

```
aws-cloudhsm > help user change-mfa token-sign
Register or Deregister a public key using token-sign mfa strategy

Usage: user change-mfa token-sign [OPTIONS] --username <USERNAME> --role <ROLE> <--
token <TOKEN>|--deregister>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --username <USERNAME>
          Username of the user that will be modified

      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
          - admin:       An Admin has the ability to manage user accounts

      --change-password <CHANGE_PASSWORD>
          Optional: Plaintext user's password. If you do not include this argument you
 will be prompted for it

      --token <TOKEN>
          Filepath where the unsigned token file will be written. Required for enabling
 MFA for a user

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation

      --deregister
          Deregister the MFA public key, if present
```

```
      --change-quorum
          Change the Quorum public key along with the MFA key

  -h, --help
          Print help (see a summary with '-h')
```

Esempio

Questo comando scriverà un token non firmato per HSM nel cluster nel file specificato da `token`. Quando ti viene richiesto, firma i token presenti nel file.

Example: Scrive un token non firmato per HSM nel tuo cluster

```
aws-cloudhsm > user change-mfa token-sign --username cu1 --change-password password --
role crypto-user --token /path/myfile
Enter signed token file path (press enter if same as the unsigned token file):
Enter public key PEM file path:/path/mypemfile
{
  "error_code": 0,
  "data": {
    "username": "test_user",
    "role": "admin"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<ROLE>

Specifica il ruolo assegnato all'account utente. Questo parametro è obbligatorio. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

Valori validi

- Admin: gli admin possono gestire gli utenti, ma non possono gestire le chiavi.
- Crypto user: gli utenti di crittografia possono creare e gestire le chiavi e utilizzarle nelle operazioni di crittografia.

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è un trattino basso (_).

Non puoi modificare il nome di un utente dopo che è stato creato. Nei comandi della CLI di CloudHSM, il ruolo e la password sono sensibili alle maiuscole e alle minuscole, il nome utente no.

Campo obbligatorio: sì

<CHANGE_PASSWORD>

Specifica in testo semplice la nuova password dell'utente la cui MFA viene registrata/annullata.

Obbligatorio: sì

<TOKEN>

Percorso del file in cui verrà scritto il file token non firmato.

Campo obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

<DEREGISTER>

Annulla la registrazione della chiave pubblica MFA, se presente.

<CHANGE-QUORUM>

Modifica la chiave pubblica del quorum insieme alla chiave MFA.

Argomenti correlati

- Capire la 2FA per gli utenti dell'HSM

Modifica della password di un utente con CloudhSM CLI

Utilizza il user change-password comando nella CLI di CloudHSM per modificare la password di un utente esistente nel cluster. AWS CloudHSM Per abilitare l'MFA per un utente, utilizzare il comando `user change-mfa`.

Qualsiasi utente può modificare la propria password. Inoltre, gli utenti con il ruolo di admin possono modificare la password di un altro utente nel cluster. Non è necessario immettere la password attuale per effettuare la modifica.

> **Note**
>
> Tuttavia, non potrai modificare la password di un utente che è attualmente connesso al cluster.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Crypto user (CU)

Sintassi

> **Note**
>
> Per abilitare l'autenticazione a più fattori (MFA) per un utente, usa user change-mfa il comando.

```
aws-cloudhsm > help user change-password
Change a user's password

    Usage:
        cloudhsm-cli user change-password [OPTIONS] --username <USERNAME> --role <ROLE>
 [--password <PASSWORD>]

    Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --username <USERNAME>
          Username of the user that will be modified
```

```
      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
          - admin:       An Admin has the ability to manage user accounts

      --password <PASSWORD>
          Optional: Plaintext user's password. If you do not include this argument you
 will be prompted for it

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation

      --deregister-mfa <DEREGISTER-MFA>
          Deregister the user's mfa public key, if present

      --deregister-quorum <DEREGISTER-QUORUM>
          Deregister the user's quorum public key, if present
 -h, --help
          Print help (see a summary with '-h')
```

Esempio

I seguenti esempi mostrano come utilizzare user change-password per reimpostare la password per l'utente corrente o qualsiasi altro utente nel cluster.

Example: modifica la tua password

Qualsiasi utente del cluster può utilizzare il comando user change-password per modificare la propria password.

Il seguente output indica che Bob è attualmente connesso come crypto user (CU).

```
aws-cloudhsm > user change-password --username bob --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "bob",
    "role": "crypto-user"
```

```
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

<DEREGISTER-MFA>

Annulla la registrazione della chiave pubblica MFA, se presente.

<DEREGISTER-QUORUM>

Annulla la registrazione della chiave pubblica del quorum, se presente.

<PASSWORD>

Specifica la nuova password dell'utente in testo semplice. I seguenti caratteri non sono consentiti ':'

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato all'account utente. Questo parametro è obbligatorio. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

Valori validi

- Admin: gli admin possono gestire gli utenti, ma non possono gestire le chiavi.
- Crypto user: gli utenti di crittografia possono creare e gestire le chiavi e utilizzarle nelle operazioni di crittografia.

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è un trattino basso (_).

Non puoi modificare il nome di un utente dopo che è stato creato. Nei comandi della CLI di CloudHSM, il ruolo e la password sono sensibili alle maiuscole e alle minuscole, il nome utente no.

Campo obbligatorio: sì

Argomenti correlati

- user list
- user create
- user delete

La categoria del quorum di modifica degli utenti nella CLI di CloudhSM

Nella user change-quorum CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per la modifica del quorum per gli utenti.

user change-quorum viene utilizzato per registrare l'autenticazione del quorum degli utenti utilizzando una specifica strategia di quorum. A partire da SDK 5.8.0, è disponibile una sola strategia di quorum per gli utenti, come illustrato di seguito.

Attualmente, questa categoria è composta dalla seguente categoria e sottocomando:

- Token-firma
  - Registra

La categoria token-sign per la modifica del quorum degli utenti nella CLI di CloudHSM

Nella user change-quorum token-sign CLI di CloudHSM, è presente una categoria principale per i comandi che, se combinati con questa categoria principale, creano un comando specifico per le operazioni quorum con firma token.

Attualmente, questa categoria comprende i seguenti comandi:

- Registra

Registra la strategia del quorum di firma dei token di un utente utilizzando la CLI di CloudHSM

Utilizza il comando user change-quorum token-sign register nella CLI di CloudHSM per registrare la strategia del quorum token-firma per un utente admin.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Sintassi

```
aws-cloudhsm > help user change-quorum token-sign register
Register a user for quorum authentication with a public key

Usage: user change-quorum token-sign register --public-key <PUBLIC_KEY> --signed-
token <SIGNED_TOKEN>

Options:
      --cluster-id <CLUSTER_ID>      Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
      --public-key <PUBLIC_KEY>      Filepath to public key PEM file
      --signed-token <SIGNED_TOKEN>  Filepath with token signed by user private key
  -h, --help Print help (see a summary with '-h')
```

Esempio

Example

Per eseguire questo comando devi accedere come l' utente per il quale desideri register quorum token-sign.

```
aws-cloudhsm > login --username admin1 --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
```

```
}
```

Il comando user change-quorum token-sign register registrerà la tua chiave pubblica con l'HSM. Di conseguenza, ti qualificherà come approvatore per le operazioni richieste dal quorum che richiedono che un utente ottenga le firme del quorum per raggiungere la soglia del quorum necessaria.

```
aws-cloudhsm > user change-quorum token-sign register \
    --public-key /home/mypemfile \
    --signed-token /home/mysignedtoken
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

Ora puoi eseguire il comando user list e confermare che il quorum del token-firma è stato registrato per questo utente.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin1",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [
          {
            "strategy": "token-sign",
            "status": "enabled"
```

```
          }
        ],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<PUBLIC-KEY>

Percorso del file PEM della chiave pubblica.

Campo obbligatorio: sì

<SIGNED-TOKEN>

Percorso del file della chiave privata dell'utente con token firmato.

Campo obbligatorio: sì

Argomenti correlati

- Utilizzo della CLI di CloudHSM per la gestione dell'autenticazione del quorum
- Utilizzo dell'autenticazione del quorum per admin: prima configurazione
- Modifica del valore minimo del quorum per gli amministratori
- Nomi e tipi di servizi che supportano l'autenticazione del quorum

Crea un AWS CloudHSM utente con CloudHSM CLI

Il user create comando nella CLI di CloudHSM crea un utente nel cluster. AWS CloudHSM Solo gli account utente con ruolo di admin possono eseguire questo comando.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Admin

Requisiti

Per eseguire questo comando, è necessario aver effettuato l'accesso come admin

Sintassi

```
aws-cloudhsm > help user create
Create a new user

Usage: cloudhsm-cli user create [OPTIONS] --username <USERNAME> --role <ROLE> [--
password <PASSWORD>]

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --username <USERNAME>
          Username to access the HSM cluster

      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
          - admin:       An Admin has the ability to manage user accounts

      --password <PASSWORD>
          Optional: Plaintext user's password. If you do not include this argument you
 will be prompted for it

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation

  -h, --help
          Print help (see a summary with '-h')
```

Esempio

Questi esempi mostrano come utilizzare per user create creare nuovi utenti nel tuo. HSMs

Example: creare un crypto user

Questo esempio crea un account nel AWS CloudHSM cluster con il ruolo di utente crittografico.

```
aws-cloudhsm > user create --username alice --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "alice",
    "role": "crypto-user"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il trattino basso (_). In questo comando il nome utente non è sensibile alle maiuscole e minuscole, il nome utente viene sempre visualizzato in minuscolo.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. I valori validi sono admin, crypto-user.

Per ottenere il ruolo dell'utente, usa il comando user list. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

<PASSWORD>

Speciifica la password dell'utente che accede a. HSMs I seguenti caratteri non sono consentiti ':'

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

Argomenti correlati

- user list
- user delete
- user change-password

Eliminare un AWS CloudHSM utente con CloudHSM CLI

Il user delete comando nella CLI di CloudHSM elimina un utente dal cluster. AWS CloudHSM Solo gli account utente con ruolo di admin possono eseguire questo comando. Non è possibile eliminare un utente che è attualmente connesso a un HSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Admin

Requisiti

- Non è possibile eliminare gli account utente che possiedono chiavi.
- Il tuo account utente deve avere il ruolo di admin per eseguire questo comando.

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
aws-cloudhsm > help user delete
Delete a user

Usage: user delete [OPTIONS] --username <USERNAME> --role <ROLE>
```

```
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --username <USERNAME>
          Username to access the HSM cluster

      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
          - admin:       An Admin has the ability to manage user accounts

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
```

Esempio

```
aws-cloudhsm > user delete --username alice --role crypto-user
{
  "error_code": 0,
  "data": {
    "username": "alice",
    "role": "crypto-user"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<USERNAME>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il trattino basso (_). In questo comando il nome utente non è sensibile alle maiuscole e minuscole, il nome utente viene sempre visualizzato in minuscolo.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. I valori validi sono admin, crypto-user.

Per ottenere il ruolo dell'utente, usa il comando user list. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

Obbligatorio: sì

<APPROVAL>

Specifica il percorso del file di un token firmato del quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

Campo obbligatorio: sì

Argomenti correlati

- user list
- user create
- user change-password

Elenca tutti gli AWS CloudHSM utenti con CLI CloudHSM

Il user list comando nella CLI di CloudHSM elenca gli account utente presenti nel cluster. AWS CloudHSM Non è necessario che tu abbia eseguito l'accesso alla CLI di CloudHSM per eseguire questo comando.

> **Note**
>
> Se aggiungi o elimini HSMs, aggiorna i file di configurazione utilizzati dal AWS CloudHSM client e dagli strumenti della riga di comando. In caso contrario, le modifiche apportate potrebbero non essere valide per tutti HSMs gli utenti del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
aws-cloudhsm > help user list
List the users in your cluster

USAGE:
    user list

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Questo comando elenca gli utenti presenti nel cluster CloudHSM.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      },
      {
        "username": "test_user",
        "role": "admin",
        "locked": "false",
        "mfa": [
          {
            "strategy": "token-sign",
            "status": "enabled"
          }
        ],
```

```
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

L'output include i seguenti attributi degli utenti:

- Nome utente: Visualizza il nome intuitivo definito dall'utente. Il nome utente viene sempre visualizzato in lettere minuscole.
- Ruolo: stabilisce quali operazioni può eseguire l'utente sull'HSM.
- Bloccato: indica se questo account utente è stato bloccato.
- MFA: indica i meccanismi di autenticazione a più fattori supportati per questo account utente.
- Copertura del cluster: Indica la disponibilità a livello di cluster di questo account utente.

Argomenti correlati

- listUsers in key_mgmt_util
- user create
- user delete
- user change-password

Replica un utente con CloudHSM CLI

Utilizza il user replicate comando nella CLI di CloudHSM per replicare un utente da AWS CloudHSM un cluster di origine a un cluster di destinazione. AWS CloudHSM

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Amministratori () COs

Requisiti

- I cluster di origine e di destinazione devono essere cloni. Ciò significa che uno è stato creato da un backup dell'altro o entrambi sono stati creati da un backup comune. Per ulteriori informazioni, consulta Creazione di cluster dai backup.
- Per eseguire questo comando, è necessario accedere come amministratore sia nel cluster di origine che in quello di destinazione.
  - In modalità a comando singolo, il comando utilizzerà le variabili ambientali CLOUDHSM_PIN e CLOUDHSM_ROLE per l'autenticazione nel cluster di origine. Per ulteriori informazioni, consulta Modalità di comando singolo. Per fornire le credenziali per il cluster di destinazione, è necessario impostare due variabili ambientali aggiuntive: DESTINATION_CLOUDHSM_PIN e DESTINATION_CLOUDHSM_ROLE:

    ```
    $ export DESTINATION_CLOUDHSM_ROLE=<role>
    ```

    ```
    $ export DESTINATION_CLOUDHSM_PIN=<username:password>
    ```

  - In modalità interattiva, gli utenti dovranno accedere in modo esplicito ai cluster di origine e di destinazione.

Sintassi

```
aws-cloudhsm > help user replicate
Replicate a user from a source to a destination cluster

Usage: user replicate --username <USERNAME> --role <ROLE> --source-cluster-
id <SOURCE_CLUSTER_ID> --destination-cluster-id <DESTINATION_CLUSTER_ID>

Options:
      --username <USERNAME>
          Username of the user to replicate

      --role <ROLE>
          Role the user has in the cluster

          Possible values:
          - crypto-user: A CryptoUser has the ability to manage and use keys
```

```
          - admin:       An Admin has the ability to manage user accounts

      --source-cluster-id <SOURCE_CLUSTER_ID>
          Source cluster ID

      --destination-cluster-id <DESTINATION_CLUSTER_ID>
          Destination cluster ID

  -h, --help
          Print help (see a summary with '-h')
```

Esempi

Example Esempio: utente replicato

Questo comando replica un utente da un cluster di origine a un cluster di destinazione clonato. L'esempio seguente mostra l'output quando si accede come amministratore su entrambi i cluster.

```
admin-user@cluster-1234abcdefg > user replicate \
      --username example-admin \
      --role admin \
      --source-cluster-id cluster-1234abcdefg \
      --destination-cluster-id cluster-2345bcdefgh
{
  "error_code": 0,
  "data": {
    "user": {
      "username": "example-admin",
      "role": "admin",
      "locked": "false",
      "mfa": [],
      "quorum": [],
      "cluster-coverage": "full"
    },
    "message": "Successfully replicated user"
  }
}
```

Arguments (Argomenti)

<USERNAME>

Speciifica il nome utente dell'utente da replicare nel cluster di origine.

Obbligatorio: sì

<ROLE>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. I valori validi sono admin, crypto-user.

Per ottenere il ruolo dell'utente, usa il comando user list. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Capire gli utenti dell'HSM.

Obbligatorio: sì

<SOURCE_CLUSTER_ID>

L'ID del cluster di origine.

Obbligatorio: sì

<DESTINATION_CLUSTER_ID>

L'ID del cluster di destinazione.

Campo obbligatorio: sì

Argomenti correlati

- Connessione a più cluster con la CLI CloudhSM

La categoria quorum nella CLI di CloudhSM

Nella quorum CLI di CloudHSM, è una categoria principale per un gruppo di comandi che, se quorum combinato con, crea un comando specifico per l'autenticazione quorum, o operazioni M of N.. Attualmente, questa categoria è costituita dalla sottocategoria token-sign che comprende i propri comandi. Per informazioni dettagliate, clicca sul seguente link.

- token-sign

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admin come la creazione e l'eliminazione di utenti, la modifica delle password degli utenti, l'impostazione dei valori del quorum e la disattivazione delle funzionalità quorum e MFA.

Crypto User Services: l'autenticazione Quorum viene utilizzata per i servizi privilegiati degli utenti crittografici associati a una chiave specifica, come la firma con una chiave, una chiave, una chiave e

l'impostazione dell'attributo di sharing/unsharing una chiave. wrapping/unwrapping Il valore quorum di una chiave associata viene configurato quando la chiave viene generata, importata o aperta. Il valore del quorum deve essere uguale o inferiore al numero di utenti a cui è associata la chiave, che include gli utenti con cui la chiave è condivisa e il proprietario della chiave.

Ogni tipo di servizio è ulteriormente suddiviso in un nome di servizio qualificante, che contiene un set specifico di operazioni di servizio supportate dal quorum che possono essere eseguite.

Nome del servizio	Tipo di servizio	Operazioni di servizio
Utente	Admin	• user create • user delete • user change-password • user change-mfa
quorum	Admin	• segno del token del quorum set-quorum-value
gruppo [1]	Admin	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gestione delle chiavi	Utente Crypto	• portachiavi • scartare le chiavi • Condivisione chiave • Annulla condivisione chiave • key set-attribute
utilizzo delle chiavi	Utente Crypto	• segno chiave

[1] Il servizio cluster è disponibile esclusivamente su hsm2m.medium

Argomenti correlati

- Imposta l'autenticazione quorum per AWS CloudHSM amministratori che utilizzano CloudhSM CLI

- Gestisci l'autenticazione del quorum (controllo degli accessi M of N) utilizzando la CLI CloudhSM

La categoria quorum token-sign nella CLI di CloudhSM

Nella quorum token-sign CLI di CloudHSM, è una categoria per un gruppo di comandi che, se quorum token-sign combinati con, creano un comando specifico per l'autenticazione quorum, o operazioni M of N..

Attualmente, questa categoria comprende i seguenti comandi:

- Elimina
- Generazione
- elenco
- list-quorum-values
- set-quorum-value

Eliminare i token del quorum utilizzando la CLI di CloudhSM

Utilizza il comando quorum token-sign delete nella CLI di CloudHSM per eliminare uno o più token per un servizio autorizzato dal quorum.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Sintassi

```
aws-cloudhsm > help quorum token-sign delete
Delete one or more Quorum Tokens

Usage: quorum token-sign delete --scope <SCOPE>

Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error
```

```
    --scope <SCOPE>
        Scope of which token(s) will be deleted

        Possible values:
        - user: Deletes all token(s) of currently logged in user
        - all:  Deletes all token(s) on the HSM
-h, --help
        Print help (see a summary with '-h')
```

Esempio

L'esempio seguente mostra come utilizzare il comando quorum token-sign delete nella CLI di CloudHSM per eliminare uno o più token per un servizio autorizzato dal quorum.

Example: Eliminare uno o più token per un servizio autorizzato dal quorum

```
aws-cloudhsm > quorum token-sign delete --scope all
{
  "error_code": 0,
  "data": "Deletion of quorum token(s) successful"
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<SCOPE>

L'ambito in cui i token verranno eliminati nel AWS CloudHSM cluster.

Valori validi

- Utente: utilizzato per eliminare solo i token di proprietà dell'utente connesso.
- Tutti: utilizzato per eliminare tutti i token nel AWS CloudHSM cluster.

Argomenti correlati

- user list

- user create
- user delete

Genera un token di quorum utilizzando la CLI di CloudhSM

Utilizza il comando quorum token-sign generate nella CLI di CloudHSM per generare un token per un servizio autorizzato dal quorum.

Esiste un limite all'ottenimento di un token attivo per utente per servizio su un cluster HSM per i servizi utente e il quorum. Questo limite non si applica ai token relativi ai servizi chiave.

> **Note**
>
> Solo gli amministratori e gli utenti Crypto possono generare token di servizio specifici. Per ulteriori informazioni sui tipi e sui nomi dei servizi, consulta Nomi e tipi di servizi che supportano l'autenticazione quorum

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admin come la creazione e l'eliminazione di utenti, la modifica delle password degli utenti, l'impostazione dei valori del quorum e la disattivazione delle funzionalità quorum e MFA.

Crypto User Services: l'autenticazione quorum viene utilizzata per i servizi privilegiati degli utenti crittografici associati a una chiave specifica, come la firma con una chiave, sharing/unsharing una chiave e l'impostazione dell'attributo di wrapping/unwrapping una chiave. Il valore quorum di una chiave associata viene configurato quando la chiave viene generata, importata o aperta. Il valore del quorum deve essere uguale o inferiore al numero di utenti a cui è associata la chiave, che include gli utenti con cui la chiave è condivisa e il proprietario della chiave.

Ogni tipo di servizio è ulteriormente suddiviso in un nome di servizio qualificante, che contiene un set specifico di operazioni di servizio supportate dal quorum che possono essere eseguite.

Nome del servizio	Tipo di servizio	Operazioni di servizio
Utente	Admin	• user create • user delete • user change-password

Nome del servizio	Tipo di servizio	Operazioni di servizio
		• user change-mfa
quorum	Admin	• segno del token del quorum set-quorum-value
gruppo [1]	Admin	• cluster mtls register-trust-anchor • cluster mtls deregister-trust-anchor • cluster mtls set-enforcement
gestione delle chiavi	Utente Crypto	• involucro per chiavi • scartare le chiavi • Condivisione chiave • Annulla condivisione chiave • key set-attribute
utilizzo delle chiavi	Utente Crypto	• segno chiave

[1] Il servizio cluster è disponibile esclusivamente su hsm2m.medium

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Crypto user (CU)

Sintassi

```
aws-cloudhsm > help quorum token-sign generate
Generate a token

Usage: quorum token-sign generate --service <SERVICE> --token <TOKEN>

Options:
```

```
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
operation against. If not provided, will fall back to the value provided when
interactive mode was started, or error

      --service <SERVICE>
          Service the token will be used for

          Possible values:
          - user:
            User management service is used for executing quorum authenticated user
management operations
          - quorum:
            Quorum management service is used for setting quorum values for any quorum
service
          - cluster:
            Cluster management service is used for executing quorum for cluster
wide configuration managements like mtls enforcement, mtls registration and mtls
deregistration
          - registration:
            Registration service is used for registering a public key for quorum
authentication
          - key-usage:
            Key usage service is used for executing quorum authenticated key usage
operations
          - key-management:
            Key management service is used for executing quorum authenticated key
management operations

      --token <TOKEN>
          Filepath where the unsigned token file will be written
  -h, --help                     Print help
```

Esempio

Questo comando scriverà un token non firmato per HSM nel cluster nel file specificato da `token`.

Example: Scrive un token non firmato per HSM nel tuo cluster

```
aws-cloudhsm > quorum token-sign generate --service user --token /home/tfile
{
  "error_code": 0,
  "data": {
```

```
    "filepath": "/home/tfile"
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<SERVICE>

Specifica il servizio autorizzato dal quorum per cui generare un token. Questo parametro è obbligatorio.

Valori validi

- Utente: il servizio di gestione degli utenti utilizzato per eseguire operazioni di gestione degli utenti autorizzati dal quorum.
- Quorum: il servizio di gestione del quorum utilizzato per impostare i valori autorizzati del quorum per qualsiasi servizio autorizzato del quorum.
- cluster: il servizio di gestione del cluster utilizzato per l'esecuzione del quorum per la gestione della configurazione a livello di cluster come mtls enforcement, mtls registration e mtls deregistration.
- Registra: genera un token non firmato da utilizzare per la registrazione di una chiave pubblica per l'autorizzazione del quorum.
- key-usage: genera un token non firmato che viene utilizzato per eseguire operazioni di utilizzo delle chiavi autorizzate dal quorum.
- gestione delle chiavi: genera un token non firmato che viene utilizzato per eseguire operazioni di gestione delle chiavi autorizzate dal quorum.

Obbligatorio: sì

<TOKEN>

Percorso del file in cui verrà scritto il file token non firmato.

Campo obbligatorio: sì

Argomenti correlati

- Nomi e tipi di servizi che supportano l'autenticazione del quorum

Elenca i token del quorum utilizzando la CLI CloudhSM

Usa il quorum token-sign list comando nella CLI di CloudHSM per elencare tutti i token quorum con firma a token presenti nel cluster. AWS CloudHSM Ciò include i token generati da altri utenti. Un token è associato a un utente, quindi, sebbene tu possa vedere i token di altri utenti, potrai utilizzare solo i token associati all'utente attualmente connesso.

Per ulteriori informazioni sui tipi e sui nomi dei servizi, vedi Nomi e tipi di servizio che supportano l'autenticazione del quorum. Per ulteriori informazioni sul contenuto visualizzato dai token elencati, vedere rispettivamente the section called "Gestione e utilizzo delle chiavi con quorum (M of N)" i token associati a `key-management` e `key-usage` i servizi e vedere the section called "Gestione degli utenti con quorum (M of N)" i token associati `user` a o al servizio. `quorum` `cluster`

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Crypto user (CU)

Sintassi

```
aws-cloudhsm > help quorum token-sign list
List the token-sign tokens in your cluster

Usage: quorum token-sign list

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Questo comando elencherà tutti i token token-sign presenti nel cluster. AWS CloudHSM Ciò include i token generati da altri utenti. Un token è associato a un utente, quindi, sebbene tu possa vedere i token di altri utenti, potrai utilizzare solo i token associati all'utente attualmente connesso.

Example

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
  "data": {
    "tokens": [
      {
        "username": "admin",
        "service": "quorum",
        "approvals-required": 2,
        "number-of-approvals": 0,
        "token-timeout-seconds": 397,
        "cluster-coverage": "full"
      },
      {
        "username": "admin",
        "service": "user",
        "approvals-required": 2,
        "number-of-approvals": 0,
        "token-timeout-seconds": 588,
        "cluster-coverage": "full"
      },
      {
        "username": "crypto_user1",
        "service": "key-management",
        "key-reference": "0x00000000002c33f7",
        "minimum-token-count": 1
      },
      {
        "username": "crypto_user1",
        "service": "key-usage",
        "key-reference": "0x00000000002c33f7",
        "minimum-token-count": 1
      }
    ]
  }
```

```
}
```

Argomenti correlati

- quorum token-sign generate

Mostra i valori del quorum utilizzando la CLI CloudhSM

Utilizza il quorum token-sign list-quorum-values comando nella CLI di CloudHSM per elencare i valori del quorum impostati nel cluster. AWS CloudHSM

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
aws-cloudhsm > help quorum token-sign list-quorum-values
List current quorum values

Usage: quorum token-sign list-quorum-values

Options:
      --cluster-id <CLUSTER_ID>  Unique Id to choose which of the clusters in the
 config file to run the operation against. If not provided, will fall back to the value
 provided when interactive mode was started, or error
  -h, --help                     Print help
```

Esempio

Questo comando elenca i valori di quorum impostati nel cluster per ogni servizio. AWS CloudHSM

Example

hsm1.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
```

```
  "data": {
    "user": 1,
    "quorum": 1
  }
}
```

hsm2m.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 1,
    "cluster": 1
  }
}
```

Argomenti correlati

- Nomi e tipi di servizi che supportano l'autenticazione del quorum
- Configurazione MTL (consigliata)

Aggiornare un valore di quorum utilizzando la CLI di CloudhSM

Utilizza il comando quorum token-sign set-quorum-value nella CLI di CloudHSM per impostare un nuovo valore di quorum per un servizio autorizzato.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Sintassi

```
aws-cloudhsm > help quorum token-sign set-quorum-value
Set a quorum value

Usage: quorum token-sign set-quorum-value [OPTIONS] --service <SERVICE> --value <VALUE>
```

```
Options:
      --cluster-id <CLUSTER_ID>
          Unique Id to choose which of the clusters in the config file to run the
 operation against. If not provided, will fall back to the value provided when
 interactive mode was started, or error

      --service <SERVICE>
          Service the token will be used for

          Possible values:
          - user:
            User management service is used for executing quorum authenticated user
 management operations
          - quorum:
            Quorum management service is used for setting quorum values for any quorum
 service
          - cluster:
            Cluster management service is used for executing quorum for cluster
 wide configuration managements like mtls enforcement, mtls registration and mtls
 deregistration

      --value <VALUE>
          Value to set for service

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation

  -h, --help
          Print help (see a summary with '-h')
```

Esempio

Example

Nell'esempio seguente, questo comando scrive un token non firmato per l'HSM nel cluster nel file specificato dal token. Quando ti viene richiesto, firma i token presenti nel file.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2
{
  "error_code": 0,
  "data": "Set Quorum Value successful"
}
```

Puoi quindi eseguire il comando list-quorum-values per confermare che il valore del quorum per il servizio di gestione del quorum è stato impostato:

hsm1.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 2
  }
}
```

hsm2m.medium:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 2,
    "cluster": 1
  }
}
```

Arguments (Argomenti)

<CLUSTER_ID>

L'ID del cluster su cui eseguire questa operazione.

Obbligatorio: se sono stati configurati più cluster.

<APPROVAL>

Il percorso del file token firmato da approvare sull'HSM.

<SERVICE>

Specifica il servizio autorizzato dal quorum per cui generare un token. Questo parametro è obbligatorio. Per ulteriori informazioni sui tipi e sui nomi dei servizi, vedi Nomi e tipi di servizio che supportano l'autenticazione del quorum.

Valori validi

- Utente: Il servizio di gestione degli utenti. Servizio utilizzato per eseguire operazioni di gestione degli utenti autorizzate dal quorum.
- Quorum: il servizio di gestione del quorum. Servizio utilizzato per impostare i valori autorizzati del quorum per qualsiasi servizio autorizzato dal quorum.
- cluster: il servizio di gestione del cluster utilizzato per l'esecuzione del quorum per la gestione della configurazione a livello di cluster come mtls enforcement, mtls registration e mtls deregistration.
- Registra: genera un token non firmato da utilizzare per registrare una chiave pubblica per l'autorizzazione del quorum.

Campo obbligatorio: sì

<VALUE>

Specifica il valore del quorum da impostare. Il valore del quorum massimo è otto (8).

Campo obbligatorio: sì

Argomenti correlati

- firma del token del quorum list-quorum-values
- Nomi e tipi di servizi che supportano l'autenticazione del quorum
- Configurazione MTL (consigliata)

AWS CloudHSM Utilità di gestione (CMU)

Lo strumento da riga di comando cloudhsm_mgmt_util aiuta i responsabili delle criptovalute a gestire gli utenti nei moduli di sicurezza hardware () nei cluster. HSMs AWS CloudHSM La AWS CloudHSM Management Utility (CMU) include strumenti che creano, eliminano ed elencano gli utenti e modificano le password degli utenti.

La CMU e la Key Management Utility (KMU) fanno parte della suite Client SDK 3. Client SDK 3 e i relativi strumenti da riga di comando (Key Management Utility e CloudHSM Management Utility) sono disponibili solo nel tipo HSM hsm1.medium.

cloudhsm_mgmt_util include anche comandi che consentono agli utenti crypto () di condividere chiavi e ottenere e impostare gli attributi chiave. CUs Questi comandi completano i comandi di gestione delle chiavi nello strumento di gestione delle chiavi primarie, key_mgmt_util.

Per una guida rapida, vedi Guida introduttiva a AWS CloudHSM Management Utility (CMU). Per informazioni dettagliate sui comandi cloudhsm_mgmt_util ed esempi di utilizzo dei comandi, vedi Riferimento per i comandi AWS CloudHSM dell'utilità di gestione.

Argomenti

- Piattaforme supportate per AWS CloudHSM Management Utility
- Guida introduttiva a AWS CloudHSM Management Utility (CMU)
- Installa e configura il AWS CloudHSM client per CMU (Linux)
- Installare e configurare il AWS CloudHSM client per CMU (Windows)
- Riferimento per i comandi AWS CloudHSM dell'utilità di gestione

Piattaforme supportate per AWS CloudHSM Management Utility

Questo argomento descrive le piattaforme Linux e Windows supportate dalla AWS CloudHSM Management Utility (CMU).

Supporto di Linux

- Amazon Linux
- Amazon Linux 2
- CentOS 6.10+
- CentOS 7.3+
- CentOS 8
- Red Hat Enterprise Linux (RHEL) 6.10+
- Red Hat Enterprise Linux (RHEL) 7.9+
- Red Hat Enterprise Linux (RHEL) 8
- Ubuntu 16.04 LTS
- Ubuntu 18.04 LTS

Supporto Windows

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Guida introduttiva a AWS CloudHSM Management Utility (CMU)

AWS CloudHSM Management Utility (CMU) consente di gestire gli utenti dei moduli di sicurezza hardware (HSM). Utilizza questo argomento per iniziare con le attività di base di gestione degli utenti HSM, come la creazione di utenti, l'elenco degli utenti e la connessione della CMU al cluster.

1. Per utilizzare CMU, è necessario innanzitutto utilizzare lo strumento di configurazione per aggiornare la configurazione CMU locale con il `--cmu` parametro e un indirizzo IP di uno dei componenti del HSMs cluster. Esegui questa operazione ogni volta che utilizzi CMU per assicurarti di gestire gli utenti HSM su tutti i moduli HSM del cluster.

   Linux

   ```
   $ sudo /opt/cloudhsm/bin/configure --cmu <IP address>
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" --cmu <IP address>
   ```

2. Immetti il seguente comando per avviare la CLI in modalità interattiva.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\cloudhsm_mgmt_util.exe" C:
   \ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_util.cfg
   ```

L'output dovrebbe essere simile al seguente, a seconda di quanti ne HSMs avete.

```
Connecting to the server(s), it may take time
depending on the server(s) load, please wait...

Connecting to server '10.0.2.9': hostname '10.0.2.9', port 2225...
Connected to server '10.0.2.9': hostname '10.0.2.9', port 2225.

Connecting to server '10.0.3.11': hostname '10.0.3.11', port 2225...
Connected to server '10.0.3.11': hostname '10.0.3.11', port 2225.

Connecting to server '10.0.1.12': hostname '10.0.1.12', port 2225...
Connected to server '10.0.1.12': hostname '10.0.1.12', port 2225.
```

Il prompt cambia in `aws-cloudhsm>` quando cloudhsm_mgmt_util è in esecuzione.

3. Utilizza il comando loginHSM per connetterti al cluster. Qualsiasi utente di qualsiasi tipo può utilizzare il comando per accedere al cluster.

   Il comando dell'esempio seguente accede a admin, che è il crypto officer (CO) predefinito. Imposta la password di questo utente una volta attivato il cluster. Puoi usare il parametro `-hpswd` per nascondere la tua password.

   ```
   aws-cloudhsm>loginHSM CO admin -hpswd
   ```

   Il sistema ti invita a inserire la tua password. Si immette la password, il sistema la nasconde e l'output mostra che il comando è stato eseguito con successo e che l'utente si è connesso a tutti gli elementi HSMs del cluster.

   ```
   Enter password:


   loginHSM success on server 0(10.0.2.9)
   loginHSM success on server 1(10.0.3.11)
   loginHSM success on server 2(10.0.1.12)
   ```

4. Usa listUsers per elencare tutti gli utenti del cluster.

   ```
   aws-cloudhsm>listUsers
   ```

CMU elenca tutti gli utenti del cluster.

```
Users on server 0(10.0.2.9):
Number of users found:2

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
Users on server 1(10.0.3.11):
Number of users found:2

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
Users on server 2(10.0.1.12):
Number of users found:2

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
```

5. Usa createUser per creare un utente CU denominato **`example_user`** con una password di **`password1`**.

   Utilizza gli utenti CU nelle tue applicazioni per eseguire operazioni crittografiche e di gestione delle chiavi. Puoi creare utenti CU perché nel passaggio 3 è stato effettuato l'accesso come utente CO. Solo gli utenti CO possono eseguire attività di gestione degli utenti con CMU, come la creazione e l'eliminazione di utenti e la modifica delle password di altri utenti.

   ```
   aws-cloudhsm>createUser CU example_user password1
   ```

CMU richiede informazioni sull'operazione di creazione utente.

```
**************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
******************************************************************

Do you want to continue(y/n)?
```

6. Per creare l'utente CU **`example_user`**, digita **`y`**.
7. Utilizza listUsers per elencare tutti gli utenti del cluster.

```
aws-cloudhsm>listUsers
```

CMU elenca tutti gli utenti del cluster, incluso il nuovo utente CU appena creato.

```
Users on server 0(10.0.2.9):
Number of users found:3

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
         3              CU              example_user                            NO
               0               NO
Users on server 1(10.0.3.11):
Number of users found:3

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                                   NO
               0               NO
         2              AU              app_user                                NO
               0               NO
```

```
         3              CU              example_user                    NO
               0              NO
Users on server 2(10.0.1.12):
Number of users found:3

    User Id             User Type       User Name
 MofnPubKey    LoginFailureCnt         2FA
         1              CO              admin                           NO
               0              NO
         2              AU              app_user                        NO
               0              NO
         3              CU              example_user                    NO
               0              NO
```

8. Usa il logoutHSM comando per disconnetterti da. HSMs

```
aws-cloudhsm>logoutHSM


logoutHSM success on server 0(10.0.2.9)
logoutHSM success on server 1(10.0.3.11)
logoutHSM success on server 2(10.0.1.12)
```

9. Chiudi il comando quit per arrestare cloudhsm_mgmt_util.

```
aws-cloudhsm>quit


disconnecting from servers, please wait...
```

Installa e configura il AWS CloudHSM client per CMU (Linux)

Per interagire con il modulo di sicurezza hardware (HSM) del AWS CloudHSM cluster utilizzando cloudhsm_mgmt_util (CMU), è necessario il software client per Linux. AWS CloudHSM È necessario installarlo sull'istanza client Linux Amazon EC2 creata in precedenza. Puoi installare un client anche se utilizzi Windows. Per ulteriori informazioni, vedi Installare e configurare il AWS CloudHSM client per CMU (Windows).

Processi

- Passaggio 1. Installa il AWS CloudHSM client e gli strumenti da riga di comando

- Passaggio 2. Modifica la configurazione del client

Passaggio 1. Installa il AWS CloudHSM client e gli strumenti da riga di comando

Connect all'istanza client ed esegui i seguenti comandi per scaricare e installare il AWS CloudHSM client e gli strumenti da riga di comando.

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb
```

Passaggio 2. Modifica la configurazione del client

Prima di poter utilizzare il AWS CloudHSM client per connettersi al cluster, è necessario modificare la configurazione del client.

Per modificare la configurazione del client

1. Se installi Client SDK 3 su cloudhsm_mgmt_util, completa i seguenti passaggi per assicurarti che tutti i nodi del cluster siano sincronizzati.

   a. Esegui configure -a ***<IP of one of the HSMs>***.

   b. Riavvia il servizio del client.

   c. Esegui configure -m.

2. Copia il certificato di emissione, quello utilizzato per firmare il certificato del cluster, nel seguente percorso sull'istanza del client: `/opt/cloudhsm/etc/customerCA.crt`. Per copiare il certificato in questa posizione, sono necessarie le autorizzazioni dell'utente root dell'istanza sull'istanza del client.

3. Utilizzate il seguente comando configure per aggiornare i file di configurazione per il AWS CloudHSM client e gli strumenti della riga di comando, specificando l'indirizzo IP dell'HSM nel cluster. Per ottenere l'indirizzo IP dell'HSM, visualizza il cluster nella AWS CloudHSM console o esegui il comando. describe-clusters AWS CLI Nell'output del comando, l'indirizzo IP del modulo HSM è il valore del campo `EniIp`. Se disponi di più di un HSM, scegli l'indirizzo IP per ognuno di HSMs essi, indipendentemente da quale.

   ```
   sudo /opt/cloudhsm/bin/configure -a <IP address>

   Updating server config in /opt/cloudhsm/etc/cloudhsm_client.cfg
   Updating server config in /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

4. Passa a Attiva il cluster in AWS CloudHSM.

Installare e configurare il AWS CloudHSM client per CMU (Windows)

Per utilizzare un modulo di sicurezza hardware (HSM) nel AWS CloudHSM cluster su Windows utilizzando cloudhsm_mgmt_util (CMU), è necessario il software client per Windows. AWS CloudHSM È consigliabile installarlo nell'istanza di Windows Server creata in precedenza.

> **Note**
>
> - Se aggiorni il client, i file di configurazione esistenti di installazioni precedenti non verranno sovrascritti.
> - Il programma di installazione AWS CloudHSM del client per Windows registra automaticamente l'API di crittografia: Next Generation (CNG) e Key Storage Provider (KSP). Per disinstallare il client, esegui di nuovo il programma di installazione e segui le istruzioni per la disinstallazione.
> - Se usi Linux, puoi installare il client Linux. Per ulteriori informazioni, consulta Installa e configura il AWS CloudHSM client per CMU (Linux).

Per installare (o aggiornare) le versioni più recenti del client Windows e degli strumenti a riga di comando

1. Connettersi all'istanza di Windows Server.
2. Scarica il programma di installazione.msi. AWSCloudHSMClient-latest
3. Se installi Client SDK 3 su cloudhsm_mgmt_util, completa i seguenti passaggi per assicurarti che tutti i nodi del cluster siano sincronizzati.
   a. Esegui configure.exe -a ***<IP of one of the HSMs>***.
   b. Riavvia il servizio del client.
   c. Esegui configure.exe -m.
4. Vai al percorso di download ed esegui il programma di installazione (AWSCloudHSMClient-latest.msi) con privilegi amministrativi.
5. Segui le istruzioni del programma di installazione, quindi scegli Chiudi al termine dell'installazione.
6. Copia il certificato di emissione autofirmato, quello utilizzato per firmare il certificato del cluster, nella cartella `C:\ProgramData\Amazon\CloudHSM`.
7. Esegui questo comando per aggiornare i file di configurazione. Assicurati di arrestare e avviare il client durante la riconfigurazione se decidi di aggiornarla:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

8. Passa a Attiva il cluster in AWS CloudHSM.

Riferimento per i comandi AWS CloudHSM dell'utilità di gestione

Lo strumento da riga di comando AWS CloudHSM cloudhsm_mgmt_util aiuta i responsabili delle criptovalute a gestire gli utenti nei moduli di sicurezza hardware () del cluster. HSMs AWS CloudHSM Include anche comandi che consentono agli utenti crittografici (CUs) di condividere le chiavi e ottenere e impostare gli attributi chiave. Questi comandi completano i comandi di gestione delle chiavi primarie nello strumento a riga di comando key_mgmt_util.

Per una guida rapida, vedi Guida introduttiva a AWS CloudHSM Management Utility (CMU).

Prima di eseguire qualsiasi comando cloudhsm_mgmt_util devi avviare key_mgmt_util e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Per elencare tutti i comandi cloudhsm_mgmt_util, esegui il comando seguente:

```
aws-cloudhsm> help
```

Per ottenere la sintassi di un comando cloudhsm_mgmt_util, esegui il comando seguente:

```
aws-cloudhsm> help <command-name>
```

> **Note**
>
> Utilizza la sintassi illustrata nella documentazione. Sebbene il software integrato di aiuto può offrire opzioni aggiuntive, queste non devono essere considerate come supportate e non devono essere utilizzate nel codice di produzione.

Per eseguire un comando, immetti il nome del comando o una parte del nome sufficiente a distinguerlo dai nomi degli altri comandi cloudhsm_mgmt_util.

Ad esempio, per ottenere un elenco di utenti su HSMs, inserisci listUsers olistU.

```
aws-cloudhsm> listUsers
```

Per terminare la sessione di cloudhsm_mgmt_util, esegui il comando seguente:

```
aws-cloudhsm> quit
```

Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

I seguenti argomenti descrivono i comandi in cloudhsm_mgmt_util.

> **Note**
>
> Alcuni comandi in key_mgmt_util e cloudhsm_mgmt_mgmt_util hanno lo stesso nome. Tuttavia, i comandi hanno in genere una sintassi diversa, un output diverso e funzionalità leggermente diverse.

Comando	Description	Tipo di utente
changePswd	Modifica le password degli utenti su. HSMs Qualsiasi utente può modificare la propria password. COs può cambiare la password di chiunque.	CO
createUser	Crea utenti di tutti i tipi su HSMs.	CO
deleteUser	Elimina utenti di tutti i tipi da. HSMs	CO
findAllKeys	Ottiene le chiavi che un utente possiede o condivide . Ottiene inoltre un hash della proprietà delle chiavi e dei dati di condivisione per tutte le chiavi su ciascun modulo HSM.	CO, AU

Comando	Description	Tipo di utente
OttieniAttributo	Ottiene un valore di attributo per una AWS CloudHSM chiave e lo scrive in un file o in uno stdout (output standard).	CU
getHSMInfo	Ottiene informazioni sull'hard ware su cui è in esecuzione un modulo HSM.	Tutti. L'accesso non è necessario.
getKeyInfo	Ottiene i proprietari, gli utenti condivisi e lo stato di autentica zione del quorum di una chiave.	Tutti. L'accesso non è necessario.
Info	Ottiene informazioni su un modulo HSM, inclusi indirizzo IP, nome host, porta e utente corrente.	Tutti. L'accesso non è necessario.
ElencaUtenti	Ottiene gli utenti in ciascuno di essi HSMs, il tipo e l'ID utente e altri attributi.	Tutti. L'accesso non è necessario.
loginHSM e logoutHSM	Permette di eseguire l'accesso e la disconnessione da un modulo HSM.	Tutti.
Esci	Esci da cloudhsm_mgmt_util.	Tutti. L'accesso non è necessario.
server	Permette di entrare e uscire dalla modalità server in un modulo HSM.	Tutti.
registerQuorumPubChiave	Associa un utente HSM a una coppia di chiavi RSA-2048 asimmetriche.	CO

Comando	Description	Tipo di utente
setAttribute	Modifica i valori degli attributi di etichette, crittografa, decodifica, wrap e relativo annullamento di una chiave esistente.	CU
shareKey	Condivide una chiave esistente con altri utenti.	CU
syncKey	Sincronizza una chiave tra cluster AWS CloudHSM clonati.	CU, CO
syncUser	Sincronizza un utente tra cluster clonati. AWS CloudHSM	CO

Modificare la password di un utente utilizzando CMU

Utilizzate il changePswd comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per modificare la password di un utente esistente sui moduli di sicurezza hardware (HSM) nel cluster. AWS CloudHSM

Qualsiasi utente può modificare la propria password. Inoltre, i responsabili di Crypto (COs and PCOs) possono modificare la password di un altro CO o utente crittografico (CU). Non è necessario immettere la password attuale per effettuare la modifica.

> **Note**
>
> Non è possibile modificare la password di un utente che è attualmente connesso al AWS CloudHSM client o a key_mgmt_util.

Per risolvere i problemi relativi a changePswd

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungete o eliminate, aggiornate i file di configurazione per HSMs CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Crypto officer (CO)
- Crypto user (CU)

Sintassi

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per abilitare l'autenticazione a due fattori (2FA) per un utente CO, utilizza il parametro `-2fa` e includi un percorso di file. Per ulteriori informazioni, vedi the section called "Arguments (Argomenti)".

```
changePswd <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

Esempi

Gli esempi seguenti mostrano come changePassword reimpostare la password per l'utente corrente o per qualsiasi altro utente del tuo HSMs.

Example: modifica la tua password

Qualsiasi utente su HSMs può utilizzare changePswd per modificare la propria password. Prima di modificare la password, utilizza info per ottenere informazioni su ciascuno degli elementi del HSMs cluster, inclusi il nome utente e il tipo di utente dell'utente che ha effettuato l'accesso.

Il seguente output indica che Bob è attualmente connesso come crypto user (CU).

```
        aws-cloudhsm> info server 0

Id      Name                    Hostname         Port   State           Partition
  LoginState
0       10.1.9.193              10.1.9.193      2225    Connected       hsm-jqici4covtv
  Logged in as 'bob(CU)'

aws-cloudhsm> info server 1

Id      Name                    Hostname         Port   State           Partition
  LoginState
1       10.1.10.7               10.1.10.7       2225    Connected       hsm-ogi3sywxbqx
  Logged in as 'bob(CU)'
```

Per modificare la password, Bob esegue il comando changePswd seguito da tipo di utente, nome utente e una nuova password.

```
aws-cloudhsm> changePswd CU bob newPassword

************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?y
Changing password for bob(CU) on 2 nodes
```

Example: modifica la password di un altro utente

È necessario essere un CO o PCO per modificare la password di un altro CO o CU su. HSMs Prima di modificare la password di un altro utente, utilizzare il comando info per confermare che il tuo utente sia di tipo CO o PCO.

Il seguente output conferma che Alice, un utente CO, ha attualmente eseguito l'accesso.

```
aws-cloudhsm>info server 0
```

```
Id      Name             Hostname         Port   State           Partition
 LoginState
0       10.1.9.193       10.1.9.193        2225   Connected      hsm-jqici4covtv
 Logged in as 'alice(CO)'


aws-cloudhsm>info server 1

Id      Name             Hostname         Port   State           Partition
 LoginState
0       10.1.10.7        10.1.10.7        2225   Connected       hsm-ogi3sywxbqx
 Logged in as 'alice(CO)'
```

Alice desidera reimpostare la password di un altro utente, John. Prima di modificare la password, utilizza il comando listUsers per verificare il tipo di utente di John.

Il seguente output elenca John come utente CO.

```
aws-cloudhsm> listUsers
Users on server 0(10.1.9.193):
Number of users found:5

    User Id             User Type       User Name            MofnPubKey
 LoginFailureCnt         2FA
         1              PCO             admin                    YES              0
             NO
         2              AU              jane                      NO              0
             NO
         3              CU              bob                       NO              0
             NO
         4              CU              alice                     NO              0
             NO
         5              CO              john                      NO              0
             NO
Users on server 1(10.1.10.7):
Number of users found:5

    User Id             User Type       User Name            MofnPubKey
 LoginFailureCnt         2FA
         1              PCO             admin                    YES              0
             NO
```

```
        2               AU              jane                    NO              0
            NO
        3               CU              bob                     NO              0
            NO
        4               CO              alice                   NO              0
            NO
        5               CO              john                    NO              0
            NO
```

Per modificare la password, Alice esegue changePswd seguito dal tipo di utente di John, dal nome utente e da una nuova password.

```
aws-cloudhsm>changePswd CO john newPassword

**************************CAUTION*********************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?y
Changing password for john(CO) on 2 nodes
```

Arguments (Argomenti)

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per abilitare la 2FA per un utente CO, utilizzate il parametro `-2fa` e includete un percorso del file. Per ulteriori informazioni sull'utilizzo di 2FA, vedi Gestisci l'autenticazione 2FA degli utenti

```
changePswd <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

<tipo-utente>

Specifica il tipo dell'utente di cui stai modificando la password. Non potrai utilizzare changePswd per modificare il tipo di utente.

I valori validi sono `CO`, `CU`, `PCO` e `PRECO`.

Per ottenere il tipo di utente, utilizza ElencaUtenti. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per Management Utility AWS CloudHSM.

Obbligatorio: sì

<nome-utente>

Specifica il nome intuitivo dell'utente. Questo parametro non fa distinzione tra maiuscole e minuscole. Non potrai utilizzare changePswd per modificare il nome utente.

Obbligatorio: sì

<password | -hpswd >

Specifica la nuova password dell'utente. Inserisci una stringa di lunghezza compresa tra 7 e 32 caratteri. Questo valore prevede la distinzione tra lettere maiuscole e minuscole. La password viene visualizzata in testo normale quando la digiti. Per nascondere la password, utilizza il parametro `-hpswd` al posto della password e segui le istruzioni.

Obbligatorio: sì

[-2 anni </ >path/to/authdata]

Specifica l'attivazione della 2FA per questo utente CO. Per ottenere i dati necessari per configurare la 2FA, includi un percorso verso una posizione nel file system con un nome di file dopo il parametro `-2fa`. Per ulteriori informazioni sull'utilizzo di 2FA, vedi Gestisci l'autenticazione 2FA degli utenti.

Obbligatorio: no

Argomenti correlati

- Info
- Elenco Utenti
- createUser
- deleteUser

Creare un AWS CloudHSM utente con CMU

Utilizzate il createUser comando in cloudhsm_mgmt_util (CMU) per creare un utente sui moduli di sicurezza hardware (HSM) del cluster. AWS CloudHSM Solo gli ufficiali addetti alle criptovalute (and) possono eseguire questo comando. COs PRECOs Quando il comando ha esito positivo, crea l'utente HSMs in tutto il cluster.

Per risolvere i problemi relativi a createUser

Se la configurazione HSM non è accurata, è possibile che l'utente non venga creato su tutti. HSMs Per aggiungere l'utente a qualsiasi utente HSMs in cui è mancante, utilizzare il comando SyncUser o CreateUser solo HSMs sugli utenti mancanti. Per evitare errori di configurazione, esegui lo strumento di configurazione con l'opzione `-m`.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungete o eliminate HSMs, aggiornate i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO, PRECO)

Sintassi

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per creare un utente CO con autenticazione a due fattori (2FA), utilizza il parametro `-2fa` e includi un percorso del file. Per ulteriori informazioni, vedi the section called "Arguments (Argomenti)".

```
createUser <user-type> <user-name> <password> |-hpswd> [-2fa </path/to/authdata>]
```

Esempi

Questi esempi mostrano come utilizzare createUser per creare nuovi utenti nel tuo HSMs.

Example: creare un crypto officer

Questo esempio crea un ufficiale di crittografia (CO) HSMs in un cluster. Il primo comando utilizza loginHSM per accedere al modulo HSM come crypto officer.

```
aws-cloudhsm> loginHSM CO admin 735782961
```

```
loginHSM success on server 0(10.0.0.1)
loginHSM success on server 1(10.0.0.2)
loginHSM success on server 1(10.0.0.3)
```

Il secondo comando utilizza createUser per creare `alice`, un nuovo crypto officer sul modulo HSM.

Il messaggio di avvertenza spiega che il comando crea utenti su tutto il cluster. HSMs Tuttavia, se il comando fallisce su uno di essi HSMs, l'utente non esisterà su di essi HSMs. Per continuare, digita `y`.

L'output mostra che il nuovo utente è stato creato su tutti e tre i HSMs componenti del cluster.

```
aws-cloudhsm> createUser CO alice 391019314

************************CAUTION*********************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
****************************************************************

Do you want to continue(y/n)?Invalid option, please type 'y' or 'n'

Do you want to continue(y/n)?y
Creating User alice(CO) on 3 nodes
```

Al termine del comando, `alice` dispone delle stesse autorizzazioni sull'HSM dell'utente `admin` CO, inclusa la modifica della password di qualsiasi utente su. HSMs

Il comando finale utilizza il comando ListUsers per verificare che `alice` esista su tutti e tre i componenti del HSMs cluster. L'output mostra anche che ad `alice` è assegnato un ID utente `3`.. L'ID utente viene utilizzato per identificarsi `alice` in altri comandi, ad esempio. findAllKeys

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:3

    User Id             User Type       User Name                      MofnPubKey
  LoginFailureCnt         2FA
        1               PRECO           admin                              YES
          0               NO
```

```
         2              AU              app_user                                 NO
           0               NO
         3              CO              alice                                    NO
           0               NO
Users on server 1(10.0.0.2):
Number of users found:3

    User Id             User Type       User Name                           MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                                    YES
           0               NO
         2              AU              app_user                                 NO
           0               NO
         3              CO              alice                                    NO
           0               NO

Users on server 1(10.0.0.3):
Number of users found:3

    User Id             User Type       User Name                           MofnPubKey
  LoginFailureCnt         2FA
         1              PRECO           admin                                    YES
           0               NO
         2              AU              app_user                                 NO
           0               NO
         3              CO              alice                                    NO
           0               NO
```

Example: creare un crypto user

In questo esempio viene creato un crypto user (CU), bob, sul modulo HSM. I crypto user possono creare e gestire le chiavi, ma non possono gestire gli utenti.

Dopo aver digitato y per rispondere al messaggio di avvertenza, l'output mostra che bob è stato creato su tutti e tre HSMs i messaggi del cluster. Il nuovo CU può accedere al modulo HSM per creare e gestire le chiavi.

Il comando ha utilizzato un valore di password `defaultPassword`. In seguito, bob o qualsiasi CO possono utilizzare il comando changePswd per modificare la password.

```
aws-cloudhsm> createUser CU bob defaultPassword

************************CAUTION*********************************
```

```
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?Invalid option, please type 'y' or 'n'

Do you want to continue(y/n)?y
Creating User bob(CU) on 3 nodes
```

Arguments (Argomenti)

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per creare un utente CO con 2FA abilitato, utilizza il parametro `-2fa` e includi un percorso del file. Per ulteriori informazioni sulla 2FA, vedi Gestisci l'autenticazione 2FA degli utenti.

```
createUser <user-type> <user-name> <password> |-hpswd> [-2fa </path/to/authdata>]
```

<tipo-utente>

Specifica il tipo di utente. Questo parametro è obbligatorio.

Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per Management Utility AWS CloudHSM.

Valori validi:

- CO: i crypto officer possono gestire gli utenti ma non le chiavi.
- CU: i crypto user possono creare e gestire le chiavi e utilizzarle nelle operazioni di crittografia.

Il tipo PRECO viene convertito in CO quando assegni una password durante l'attivazione del modulo HSM.

Obbligatorio: sì

<nome-utente>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è un trattino basso (_).

Non puoi modificare il nome di un utente dopo che è stato creato. Nei comandi cloudhsm_mgmt_util, il tipo di utente e la password sono sensibili alle maiuscole e alle minuscole, il nome utente no.

Obbligatorio: sì

<password | -hpswd >

Specifica una password per l'utente. Inserisci una stringa di lunghezza compresa tra 7 e 32 caratteri. Questo valore prevede la distinzione tra lettere maiuscole e minuscole. La password viene visualizzata in testo normale quando la digiti. Per nascondere la password, utilizza il parametro `-hpswd` al posto della password e segui le istruzioni.

Per modificare una password utente, utilizza changePswd. Qualsiasi utente HSM può modificare la propria password, ma gli utenti CO possono modificare la password di qualsiasi utente (di qualsiasi tipo) su. HSMs

Obbligatorio: sì

[-2fa </ >path/to/authdata]

Specifica la creazione di un utente CO con 2FA abilitata. Per ottenere i dati necessari per configurare l'autenticazione 2FA, includi un percorso verso una posizione nel file system con un nome di file dopo il parametro `-2fa`. Per informazioni sull'impostazione e il funzionamento della 2FA, vedi Gestisci l'autenticazione 2FA degli utenti.

Obbligatorio: no

Argomenti correlati

- ElencaUtenti
- deleteUser
- syncUser
- changePswd

Eliminare un AWS CloudHSM utente utilizzando CMU

Utilizzate il deleteUser comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per eliminare un utente dai moduli di sicurezza hardware (HSM) del cluster. AWS CloudHSM Solo i crypto officer (CO)

possono eseguire questo comando. Non è possibile eliminare un utente che è attualmente connesso a un HSM. Per ulteriori informazioni sull'eliminazione degli utenti, vedi Come eliminare gli utenti HSM.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- CO

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
deleteUser <user-type> <user-name>
```

Esempio

Questo esempio elimina un ufficiale crittografico (CO) dall' HSMs interno di un cluster. Il primo comando utilizza ListUsers per elencare tutti gli utenti su. HSMs

L'output mostra che l'utente3,`alice`, è un CO su. HSMs

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:3

    User Id             User Type       User Name                           MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                                    YES
           0               NO
         2              AU              app_user                                  NO
           0               NO
         3              CO              alice                                     NO
           0               NO
Users on server 1(10.0.0.2):
Number of users found:3
```

```
    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                               YES
           0               NO
         2              AU              app_user                             NO
           0               NO
         3              CO              alice                                NO
           0               NO

Users on server 1(10.0.0.3):
Number of users found:3

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                               YES
           0               NO
         2              AU              app_user                             NO
           0               NO
         3              CO              alice                                NO
           0               NO
```

Il secondo comando utilizza il deleteUser comando per eliminare `alice` da HSMs.

L'output mostra che il comando ha avuto esito positivo su tutti e tre i HSMs componenti del cluster.

```
aws-cloudhsm> deleteUser CO alice
Deleting user alice(CO) on 3 nodes
deleteUser success on server 0(10.0.0.1)
deleteUser success on server 0(10.0.0.2)
deleteUser success on server 0(10.0.0.3)
```

Il comando finale utilizza il listUsers comando per verificare che `alice` venga eliminato da tutti e tre i componenti HSMs del cluster.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:2

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                               YES
           0               NO
```

```
         2              AU              app_user                                 NO
           0               NO
Users on server 1(10.0.0.2):
Number of users found:2

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                                YES
           0               NO
         2              AU              app_user                                 NO
           0               NO
Users on server 1(10.0.0.3):
Number of users found:2

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                                YES
           0               NO
         2              AU              app_user                                 NO
           0               NO
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
deleteUser <user-type> <user-name>
```

<tipo-utente>

Specifica il tipo di utente. Questo parametro è obbligatorio.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

I valori validi sono CO, CU.

Per ottenere il tipo di utente, utilizzare ElencaUtenti. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per Management Utility AWS CloudHSM.

Obbligatorio: sì

<nome-utente>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è un trattino basso (_).

Non puoi modificare il nome di un utente dopo che è stato creato. Nei comandi cloudhsm_mgmt_util, il tipo di utente e la password sono sensibili alle maiuscole e alle minuscole, il nome utente no.

Campo obbligatorio: sì

Argomenti correlati

- ElencaUtenti
- createUser
- syncUser
- changePswd

Elenca le chiavi possedute da un utente AWS CloudHSM crittografico utilizzando CMU

Usa il findAllKeys comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per ottenere le chiavi che un utente crittografico (CU) specificato possiede o condivide. AWS CloudHSM Il comando restituisce anche un hash dei dati utente su ciascuno di. HSMs È possibile utilizzare l'hash per determinare a colpo d'occhio se gli utenti, la proprietà delle chiavi e i dati di condivisione delle chiavi sono gli stessi su tutti i HSMs componenti del cluster. Nell'output, le chiavi di proprietà dell'utente vengono annotate da `(o)` e le chiavi condivise vengono annotate da `(s)`.

findAllKeysrestituisce le chiavi pubbliche solo quando la CU specificata possiede la chiave, anche se tutti gli CUs elementi dell'HSM possono utilizzare qualsiasi chiave pubblica. Questo comportamento è diverso rispetto a findKey in key_mgmt_util, che restituisce chiavi pubbliche per tutti gli utenti CU.

Solo i responsabili delle criptovalute (COs and PCOs) e gli utenti dell'appliance (AUs) possono eseguire questo comando. Gli utenti Crypto (CUs) possono eseguire i seguenti comandi:

- listUsers per trovare tutti gli utenti
- findKey in key_mgmt_util per trovare le chiavi che è possibile utilizzare

- getKeyInfoin key_mgmt_util per trovare il proprietario e gli utenti condivisi di una particolare chiave che possiedono o condividono

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini HSMs, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Crypto officer (CO, PCO)
- Utenti dell'appliance (AU)

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
findAllKeys <user id> <key hash (0/1)> [<output file>]
```

Esempi

Questi esempi mostrano come trovare tutte le chiavi per un utente e ottenere un hash di informazioni chiave sull'utente su ciascuna di esse. `findAllKeys` HSMs

Example: trovare le chiavi per un CU

Questo esempio utilizza findAllKeys per trovare le chiavi nella cartella HSMs che l'utente 4 possiede e condivide. Il comando utilizza il valore `0` per il secondo argomento per eliminare il valore hash. Poiché viene omesso il nome file opzionale, il comando scrive in stdout (output standard).

L'output indica che l'utente 4 può utilizzare 6 chiavi: 8, 9, 17, 262162, 19 e 31. L'output utilizza un `(s)` per indicare le chiavi che sono esplicitamente condivise dall'utente. Le chiavi che l'utente possiede sono indicate da un `(o)` e includono chiavi simmetriche e private che l'utente non condivide e chiavi pubbliche disponibili per tutti i crypto user.

```
aws-cloudhsm> findAllKeys 4 0
Keys on server 0(10.0.0.1):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 0(10.0.0.1)

Keys on server 1(10.0.0.2):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 1(10.0.0.2)

Keys on server 1(10.0.0.3):
Number of keys found 6
number of keys matched from start index 0::6
8(s),9(s),17,262162(s),19(o),31(o)
findAllKeys success on server 1(10.0.0.3)
```

Example: Verifica della sincronizzazione dei dati utente

Questo esempio serve findAllKeys a verificare che tutte le informazioni del HSMs cluster contengano gli stessi utenti, la proprietà delle chiavi e gli stessi valori di condivisione delle chiavi. Per eseguire questa operazione, ottiene un hash dei dati dell'utente delle chiavi su ciascun HSM e confronta i valori hash.

Per ottenere l'hash delle chiavi, il comando utilizza il valore 1 nel secondo argomento. Il nome del file opzionale viene omesso, pertanto il comando scrive l'hash della chiave in stdout.

L'esempio specifica l'utente6, ma il valore hash sarà lo stesso per tutti gli utenti che possiedono o condividono una delle chiavi del. HSMs Se l'utente specificato non possiede o condivide alcuna chiave, come accade per un CO, il comando non restituisce un valore hash.

L'output mostra che l'hash della chiave è identico a entrambi quelli del cluster HSMs . Se uno degli HSM avesse utenti diversi, proprietari delle chiavi diversi o utenti condivisi diversi, i valori hash delle chiavi non sarebbero uguali.

```
aws-cloudhsm> findAllKeys 6 1
Keys on server 0(10.0.0.1):
Number of keys found 3
number of keys matched from start index 0::3
```

```
8(s),9(s),11,17(s)
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 0(10.0.0.1)
Keys on server 1(10.0.0.2):
Number of keys found 3
number of keys matched from start index 0::3
8(s),9(s),11(o),17(s)
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 1(10.0.0.2)
```

Questo comando dimostra che il valore hash rappresenta i dati utente per tutte le chiavi nell'HSM. Il comando utilizza findAllKeys per l'utente 3. A differenza dell'utente 6, che possiede o condivide solo 3 chiavi, l'utente 3 possiede o condivide 17 chiavi, ma il valore hash delle chiavi è lo stesso.

```
aws-cloudhsm> findAllKeys 3 1
Keys on server 0(10.0.0.1):
Number of keys found 17
number of keys matched from start index 0::17
6(o),7(o),8(s),11(o),12(o),14(o),262159(o),262160(o),17(s),262162(s),19(s),20(o),21(o),262177(o
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 0(10.0.0.1)
Keys on server 1(10.0.0.2):
Number of keys found 17
number of keys matched from start index 0::17
6(o),7(o),8(s),11(o),12(o),14(o),262159(o),262160(o),17(s),262162(s),19(s),20(o),21(o),262177(o
Key Hash:
55655676c95547fd4e82189a072ee1100eccfca6f10509077a0d6936a976bd49

findAllKeys success on server 1(10.0.0.2)
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
findAllKeys <user id> <key hash (0/1)> [<output file>]
```

<id utente>

Ottiene tutte le chiavi che l'utente specificato possiede o condivide. Immettere l'ID utente di un utente su. HSMs Per trovare l'utente IDs di tutti gli utenti, usa ListUsers.

Tutti gli utenti IDs sono validi, ma `findAllKeys` restituiscono le chiavi solo per gli utenti crittografici (CUs).

Obbligatorio: sì

<hash chiave>

Include (`1`) o esclude (`0`) un hash della proprietà dell'utente e dei dati di condivisione per tutte le chiavi in ciascun modulo HSM.

Quando l'argomento `user id` rappresenta un utente che possiede o condivide le chiavi, l'hash delle chiavi è popolato. Il valore hash delle chiavi è identico per tutti gli utenti che possiedono o condividono chiavi sull'HSM, anche se possiedono e condividono chiavi diverse. Tuttavia, quando `user id` rappresenta un utente che non possiede o condivide alcuna chiave, ad esempio un CO, il valore hash non è popolato.

Obbligatorio: sì

<file output>

Scrive l'output nel file specificato.

Obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- changePswd
- deleteUser
- ElencaUtenti
- syncUser
- findKey in key_mgmt_util
- getKeyInfoin key_mgmt_util

Ottieni un valore di attributo AWS CloudHSM chiave utilizzando CMU

Utilizzate il getAttribute comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per ottenere un valore di attributo per una chiave da tutti i moduli di sicurezza hardware (HSM) del AWS CloudHSM cluster e lo scrive su stdout (output standard) o su un file. CUsSolo gli utenti crypto () possono eseguire questo comando.

Gli attributi chiave sono le proprietà di una chiave. Includono caratteristiche, quali tipo di chiave, classe, etichetta e ID e valori che rappresentano le azioni che è possibile eseguire sulla chiave, ad esempio crittografare, decodificare, wrap, firmare e verificare.

Puoi utilizzare il comando getAttribute solo sulle chiavi di tua proprietà e su quelle condivise con te. È possibile eseguire questo comando o il comando getAttribute in key_mgmt_util che scrive uno o tutti i valori degli attributi di una chiave su un file.

Per ottenere un elenco di attributi e delle costanti che li rappresentano, utilizza il comando listAttributes. Per modificare i valori degli attributi delle chiavi esistenti, utilizza setAttribute in key_mgmt_util e setAttribute in cloudhsm_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi. AWS CloudHSM riferimento agli attributi chiave per KMU

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungete o eliminate HSMs, aggiornate i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Crypto user (CU)

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
getAttribute <key handle> <attribute id> [<filename>]
```

Esempio

Questo esempio ottiene il valore dell'attributo extractable per una chiave in. HSMs È possibile utilizzare un comando come questo per determinare se è possibile esportare una chiave da. HSMs

Il primo comando utilizza listAttributes per trovare la costante che rappresenta l'attributo estraibile. L'output indica che la costante per `OBJ_ATTR_EXTRACTABLE` è `354`. È anche possibile trovare queste informazioni con le descrizioni degli attributi e i relativi valori in AWS CloudHSM riferimento agli attributi chiave per KMU.

```
aws-cloudhsm> listAttributes

Following are the possible attribute values for getAttribute:

      OBJ_ATTR_CLASS                = 0
      OBJ_ATTR_TOKEN                = 1
      OBJ_ATTR_PRIVATE              = 2
      OBJ_ATTR_LABEL                = 3
      OBJ_ATTR_TRUSTED              = 134
      OBJ_ATTR_KEY_TYPE             = 256
      OBJ_ATTR_ID                   = 258
      OBJ_ATTR_SENSITIVE            = 259
      OBJ_ATTR_ENCRYPT              = 260
      OBJ_ATTR_DECRYPT              = 261
      OBJ_ATTR_WRAP                 = 262
      OBJ_ATTR_UNWRAP               = 263
      OBJ_ATTR_SIGN                 = 264
      OBJ_ATTR_VERIFY               = 266
      OBJ_ATTR_DERIVE               = 268
      OBJ_ATTR_LOCAL                = 355
      OBJ_ATTR_MODULUS              = 288
      OBJ_ATTR_MODULUS_BITS         = 289
      OBJ_ATTR_PUBLIC_EXPONENT      = 290
      OBJ_ATTR_VALUE_LEN            = 353
      OBJ_ATTR_EXTRACTABLE          = 354
      OBJ_ATTR_NEVER_EXTRACTABLE    = 356
      OBJ_ATTR_ALWAYS_SENSITIVE     = 357
      OBJ_ATTR_DESTROYABLE          = 370
      OBJ_ATTR_KCV                  = 371
      OBJ_ATTR_WRAP_WITH_TRUSTED    = 528
      OBJ_ATTR_WRAP_TEMPLATE        = 1073742353
      OBJ_ATTR_UNWRAP_TEMPLATE      = 1073742354
      OBJ_ATTR_ALL                  = 512
```

Il secondo comando utilizza getAttribute per ottenere il valore dell'attributo extractable per la chiave con maniglia di chiave 262170 in. HSMs Per specificare l'attributo estraibile, il comando utilizza 354, la costante che rappresenta l'attributo. Poiché il comando non specifica un nome file, getAttribute scrive l'output in stdout.

L'output indica che il valore dell'attributo estraibile è 1 in tutti i moduli HSM. Tale valore indica che il proprietario della chiave può esportarlo. Quando il valore è 0 (0x0), non può essere esportato da. HSMs È possibile impostare il valore dell'attributo estraibile al momento della creazione di una chiave, ma non è possibile modificarlo.

```
aws-cloudhsm> getAttribute 262170 354

Attribute Value on server 0(10.0.1.10):
OBJ_ATTR_EXTRACTABLE
0x00000001

Attribute Value on server 1(10.0.1.12):
OBJ_ATTR_EXTRACTABLE
0x00000001

Attribute Value on server 2(10.0.1.7):
OBJ_ATTR_EXTRACTABLE
0x00000001
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
getAttribute <key handle> <attribute id> [<filename>]
```

<handle-chiave>

Specifica l'handle della chiave di destinazione. È possibile specificare una sola chiave in ogni comando. Per individuare l'handle di una chiave, utilizza findKey in key_mgmt_util.

È necessario essere in possesso della chiave specificata altrimenti è necessario richiederne la condivisione. Per trovare gli utenti di una chiave, usa getKeyInfokey_mgmt_util.

Obbligatorio: sì

<id attributo>

Identifica l'attributo. Inserisci una costante che rappresenti un attributo oppure immetti `512` per tutti gli attributi. Ad esempio, per ottenere il tipo di chiave, immetti `256`, che è la costante per l'attributo `OBJ_ATTR_KEY_TYPE`.

Per elencare gli attributi e le relative costanti, utilizza listAttributes. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Obbligatorio: sì

<nome file>

Scrive l'output nel file specificato. È necessario immettere un percorso di file.

Se il file specificato esiste, getAttribute lo sovrascrive senza preavviso.

Obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- getAttribute in key_mgmt_util
- listAttributes
- setAttribute in cloudhsm_mgmt_util
- setAttribute in key_mgmt_util
- Riferimento per l'attributo della chiave

Ottieni informazioni sull'hardware per ogni HSM in un AWS CloudHSM cluster con CMU

Usa il getHSMInfo comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per ottenere informazioni sull'hardware su cui viene eseguito ogni modulo di sicurezza hardware (HSM), inclusi il modello, il numero di serie, lo stato FIPS, la memoria, la temperatura e i numeri di versione dell'hardware e del firmware. Le informazioni includono anche l'ID del server utilizzato da cloudhsm_mgmt_util per fare riferimento all'HSM.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini, aggiorna i file di configurazione per CMU. HSMs In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

Questo comando non ha parametri.

```
getHSMInfo
```

Esempio

Questo esempio utilizza getHSMInfo per ottenere informazioni sulla HSMs presenza nel cluster.

```
aws-cloudhsm> getHSMInfo
Getting HSM Info on 3 nodes
                *** Server 0 HSM Info ***

        Label                :cavium
        Model                :NITROX-III CNN35XX-NFBE

        Serial Number        :3.0A0101-ICM000001
        HSM Flags            :0
        FIPS state           :2 [FIPS mode with single factor authentication]

        Manufacturer ID      :
        Device ID            :10
        Class Code           :100000
        System vendor ID     :177D
        SubSystem ID         :10
```

```
        TotalPublicMemory    :560596
        FreePublicMemory     :294568
        TotalPrivateMemory   :0
        FreePrivateMemory    :0

        Hardware Major       :3
        Hardware Minor       :0

        Firmware Major       :2
        Firmware Minor       :03

        Temperature          :56 C

        Build Number         :13

        Firmware ID          :xxxxxxxxxxxxxx

...
```

Argomenti correlati

- Info

Ottieni informazioni AWS CloudHSM utente su una chiave utilizzando CMU

Usa il getKeyInfo comando contenuto in AWS CloudHSM key_mgmt_util (KMU) per restituire l'utente IDs del modulo di sicurezza hardware (HSM) degli utenti che possono usare la chiave, inclusi il proprietario e gli utenti crittografici (CU) con cui la chiave è condivisa. Quando su una chiave è abilitata la funzionalità di autenticazione del quorum, getKeyInfo restituisce anche il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave. Puoi eseguire il comando getKeyInfo solo sulle chiavi di tua proprietà e su quelle condivise con te.

Quando esegui il comando getKeyInfo su chiavi pubbliche, getKeyInfo restituisce solo il proprietario della chiave, anche se tutti gli utenti HSM possono utilizzare la chiave pubblica. Per trovare l'utente HSM IDs degli utenti nel tuo HSMs, usa ListUsers. Per trovare le chiavi di un utente specifico, utilizza findKey key_mgmt_util. Gli ufficiali crittografici possono utilizzarlo findAllKeysin cloudhsm_mgmt_util.

Le chiavi che hai creato sono di tua proprietà. Puoi condividere una chiave con altri utenti nel momento in cui la crei. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini HSMs, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Sintassi

```
getKeyInfo -k <key-handle> [<output file>]
```

Esempi

Questi esempi mostrano come utilizzare getKeyInfo per ottenere informazioni sugli utenti di una chiave.

Example: ottenere gli utenti di una chiave asimmetrica

Questo comando ottiene gli utenti che possono utilizzare la chiave AES (asimmetrica) con l'handle di chiave 262162. L'output mostra che l'utente 3 possiede la chiave e l'ha condivisa con gli utenti 4 e 6.

Solo gli utenti 3, 4 e 6 possono eseguire getKeyInfo sulla chiave 262162.

```
aws-cloudhsm>getKeyInfo 262162
Key Info on server 0(10.0.0.1):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 2 user(s):

                4
                6
```

```
Key Info on server 1(10.0.0.2):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 2 user(s):

                4
                6
```

Example: ottieni gli utenti di una coppia di chiavi simmetriche

Questi comandi utilizzano getKeyInfo per ottenere gli utenti che possono utilizzare le chiavi in una coppia di chiavi ECC (simmetriche). La chiave pubblica ha l'handle 262179. La chiave privata presenta l'handle 262177.

Quando esegui getKeyInfo sulla chiave privata (262177), restituisce il proprietario della chiave (3) e gli utenti crittografici (CUs) 4, con cui la chiave è condivisa.

```
aws-cloudhsm>getKeyInfo -k 262177
Key Info on server 0(10.0.0.1):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 1 user(s):

                4
Key Info on server 1(10.0.0.2):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 1 user(s):

                4
```

Quando esegui getKeyInfo sulla chiave pubblica (262179), il comando restituisce solo il proprietario della chiave, l'utente 3.

```
aws-cloudhsm>getKeyInfo -k 262179
Key Info on server 0(10.0.3.10):

        Token/Flash Key,

        Owned by user 3

Key Info on server 1(10.0.3.6):

        Token/Flash Key,

        Owned by user 3
```

Per verificare se l'utente 4 può utilizzare la chiave pubblica (e tutte le chiavi pubbliche sul modulo HSM), utilizza il parametro `-u` di findKey in key_mgmt_util.

L'output indica che nella coppia di chiavi l'utente 4 può utilizzare sia la chiave pubblica (262179) sia quella privata (262177). L'utente 4 può inoltre utilizzare tutte le altre chiavi pubbliche e qualsiasi chiave privata che abbia creato o che sia stata condivisa con lui.

```
Command:  findKey -u 4

Total number of keys present 8

 number of keys matched from start index 0::7
11, 12, 262159, 262161, 262162, 19, 20, 21, 262177, 262179

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: ottenere il valore di autenticazione del quorum (m_value) per una chiave

Questo esempio illustra come ottenere il valore `m_value` per una chiave. L'm_value è il numero di utenti del quorum che deve approvare qualsiasi operazione di crittografia che utilizza la chiave e le operazioni di condivisione e annullamento della condivisione della chiave.

Quando l'autenticazione del quorum è abilitata su una chiave, il quorum degli utenti deve approvare tutte le operazioni di crittografia che utilizzano la chiave. Per abilitare l'autenticazione del quorum e impostarne le dimensioni, utilizza il parametro `-m_value` durante la creazione della chiave.

Questo comando consente genSymKeydi creare una chiave AES a 256 bit condivisa con l'utente 4. Il comando utilizza il parametro `m_value` per abilitare l'autenticazione del quorum e impostarne le dimensioni a due utenti. Il numero di utenti deve essere sufficiente per fornire le approvazioni necessarie.

L'output indica che il comando ha creato la chiave 10.

```
Command:  genSymKey -t 31 -s 32 -l aes256m2 -u 4 -m_value 2

       Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

       Symmetric Key Created.  Key Handle: 10

       Cluster Error Status
       Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
       Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Questo comando utilizza getKeyInfo in cloudhsm_mgmt_util per ottenere informazioni sugli utenti della chiave 10. L'output indica che la chiave è di proprietà dell'utente 3 ed è condivisa con l'utente 4. Inoltre, mostra che un quorum di due utenti deve approvare ogni operazione di crittografia che utilizza la chiave.

```
aws-cloudhsm>getKeyInfo 10

Key Info on server 0(10.0.0.1):

       Token/Flash Key,

       Owned by user 3

       also, shared to following 1 user(s):

                4
        2 Users need to approve to use/manage this key
Key Info on server 1(10.0.0.2):

       Token/Flash Key,

       Owned by user 3

       also, shared to following 1 user(s):
```

```
                4
        2 Users need to approve to use/manage this key
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
getKeyInfo -k <key-handle> <output file>
```

<handle-chiave>

Specifica l'handle di una chiave nel modulo HSM. Specifica l'handle di una chiave di cui sei proprietario o che è condivisa con te. Questo parametro è obbligatorio.

Obbligatorio: sì

<file output>

Scrive l'output nel file specificato, invece che in stdout. Se il file esiste, il comando lo sovrascrive senza preavviso.

Obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- getKeyInfoin key_mgmt_util
- findKey in key_mgmt_util
- findAllKeysin cloudhsm_mgmt_util
- ElencaUtenti
- shareKey

Ottieni informazioni per ogni HSM in un AWS CloudHSM cluster utilizzando CMU

Usa il info comando in AWS CloudHSM cloudhsm_mgmt_util (CMU) per ottenere informazioni su ciascuno dei moduli di sicurezza hardware (HSM) del AWS CloudHSM cluster, inclusi il nome host,

la porta, l'indirizzo IP e il nome e il tipo di utente che ha effettuato l'accesso a cloudhsm_mgmt_util sull'HSM.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se HSMs aggiungi o elimini, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
info server <server ID>
```

Esempio

Questo esempio utilizza info per ottenere informazioni su un HSM nel cluster. Il comando utilizza 0 per fare riferimento al primo HSM del cluster. L'output mostra l'indirizzo IP, la porta, nonché il tipo e il nome dell'utente corrente.

```
aws-cloudhsm> info server 0
Id      Name                    Hostname         Port   State           Partition
         LoginState
0       10.0.0.1                10.0.0.1        2225    Connected       hsm-udw0tkfg1ab
         Logged in as 'testuser(CU)'
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
info server <server ID>
```

<id server>

Specifica l'ID server dell'HSM. HSMs Vengono assegnati numeri ordinali che rappresentano l'ordine in cui vengono aggiunti al cluster, a partire da 0. Per trovare l'ID del server di un HSM, usa get. HSMInfo

Campo obbligatorio: sì

Argomenti correlati

- getHSMInfo
- loginHSM e logoutHSM

Elenca gli attributi di una AWS CloudHSM chiave usando CMU

Usa il listAttributes comando nella AWS CloudHSM CMU cloudhsm_mgmt_util per elencare gli attributi di una chiave e le costanti che li rappresentano. AWS CloudHSM Puoi utilizzare queste costanti per identificare gli attributi nei comandi getAttribute e setAttribute.

Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
listAttributes [-h]
```

Esempio

Questo comando elenca gli attributi della chiave che puoi ottenere e modificare in key_mgmt_util e le costanti che li rappresentano. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU. Per rappresentare tutti gli attributi utilizza 512.

```
Command: listAttributes

    Description
===========
The following are all of the possible attribute values for getAttribute.

      OBJ_ATTR_CLASS                 = 0
      OBJ_ATTR_TOKEN                 = 1
      OBJ_ATTR_PRIVATE               = 2
      OBJ_ATTR_LABEL                 = 3
      OBJ_ATTR_TRUSTED               = 134
      OBJ_ATTR_KEY_TYPE              = 256
      OBJ_ATTR_ID                    = 258
      OBJ_ATTR_SENSITIVE             = 259
      OBJ_ATTR_ENCRYPT               = 260
      OBJ_ATTR_DECRYPT               = 261
      OBJ_ATTR_WRAP                  = 262
      OBJ_ATTR_UNWRAP                = 263
      OBJ_ATTR_SIGN                  = 264
      OBJ_ATTR_VERIFY                = 266
      OBJ_ATTR_DERIVE                = 268
      OBJ_ATTR_LOCAL                 = 355
      OBJ_ATTR_MODULUS               = 288
      OBJ_ATTR_MODULUS_BITS          = 289
      OBJ_ATTR_PUBLIC_EXPONENT       = 290
      OBJ_ATTR_VALUE_LEN             = 353
      OBJ_ATTR_EXTRACTABLE           = 354
      OBJ_ATTR_NEVER_EXTRACTABLE     = 356
      OBJ_ATTR_ALWAYS_SENSITIVE      = 357
      OBJ_ATTR_DESTROYABLE           = 370
      OBJ_ATTR_KCV                    = 371
      OBJ_ATTR_WRAP_WITH_TRUSTED      = 528
      OBJ_ATTR_WRAP_TEMPLATE          = 1073742353
      OBJ_ATTR_UNWRAP_TEMPLATE       = 1073742354
      OBJ_ATTR_ALL                   = 512
```

Parameters

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

Argomenti correlati

- OttieniAttributo
- setAttribute
- Riferimento per l'attributo della chiave

Elenca tutti AWS CloudHSM gli utenti che utilizzano CMU

Utilizzate il listUsers comando in AWS CloudHSM cloudhsm_mgmt_util per inserire gli utenti in ciascuno dei moduli di sicurezza hardware (HSM), insieme al tipo di utente e ad altri attributi. Questo comando può essere eseguito da tutti i tipi di utenti. Non è necessario che tu abbia eseguito l'accesso a loudhsm_mgmt_util per eseguire questo comando.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini, aggiorna i file di configurazione per CMU. HSMs In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

Questo comando non ha parametri.

```
listUsers
```

Esempio

Questo comando elenca gli utenti di ogni elemento HSMs del cluster e ne visualizza gli attributi. È possibile utilizzare l'attributo `User ID` per identificare gli utenti in altri comandi, ad esempio deleteUser, changePswd e findAllKeys.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
Number of users found:6

    User Id             User Type       User Name            MofnPubKey
 LoginFailureCnt         2FA
         1              PCO             admin                   YES              0
             NO
         2              AU              app_user                 NO              0
             NO
         3              CU              crypto_user1             NO              0
             NO
         4              CU              crypto_user2             NO              0
             NO
         5              CO              officer1                YES              0
             NO
         6              CO              officer2                 NO              0
             NO
Users on server 1(10.0.0.2):
Number of users found:5

    User Id             User Type       User Name            MofnPubKey
 LoginFailureCnt         2FA
         1              PCO             admin                   YES              0
             NO
         2              AU              app_user                 NO              0
             NO
         3              CU              crypto_user1             NO              0
             NO
         4              CU              crypto_user2             NO              0
             NO
         5              CO              officer1                YES              0
             NO
```

L'output include i seguenti attributi degli utenti:

- ID utente: identifica l'utente nei comandi key_mgmt_util e cloudhsm_mgmt_util.
- Tipo utente: stabilisce quali operazioni può eseguire l'utente sull'HSM.
- Nome utente: visualizza il nome intuitivo definito dall'utente.
- MofnPubKey: indica se l'utente ha registrato una coppia di chiavi per la firma dei token di autenticazione del quorum.

- LoginFailureCnt: indica il numero di volte in cui l'utente ha effettuato l'accesso senza successo.
- 2FA: indica che l'utente ha abilitato l'autenticazione a più fattori.

Argomenti correlati

- listUsers in key_mgmt_util
- createUser
- deleteUser
- changePswd

Accedere e disconnettersi da un HSM utilizzando AWS CloudHSM Management Utility

Usa i logoutHSM comandi loginHSM and in AWS CloudHSM cloudhsm_mgmt_util per accedere e disconnetterti da ogni HSM in un cluster. Qualsiasi utente di qualsiasi tipo può utilizzare questi comandi.

> **Note**
>
> Se superi cinque tentativi di accesso errati, il tuo account viene bloccato. Per sbloccare l'account, un responsabile della crittografia (CO) deve reimpostare la password utilizzando il comando changePswd in cloudhsm_mgmt_util.

Per risolvere i problemi relativi a LoginHSM e LogoutHSM

Prima di eseguire questi comandi cloudhsm_mgmt_util, devi avviare cloudhsm_mgmt_util.

Se aggiungi o elimini HSMs, aggiorna i file di configurazione utilizzati dal client e dagli strumenti a riga di comando. AWS CloudHSM In caso contrario, le modifiche apportate potrebbero non essere valide per tutti HSMs gli utenti del cluster.

Se disponi di più HSM nel cluster, potresti avere consentiti più tentativi di accesso errati prima che l'account venga bloccato. Questo perché il client CloudHSM bilancia il carico tra diversi. HSMs Pertanto, il tentativo di accesso potrebbe non iniziare sullo stesso HSM ogni volta. Se stai testando questa funzionalità, ti consigliamo di farlo su un cluster con un solo HSM attivo.

Se il cluster è stato creato prima di febbraio 2018, l'account viene bloccato dopo 20 tentativi di accesso errati.

Tipo di utente

Gli utenti seguenti possono eseguire questi comandi.

- Funzionario pre-Crypto (PRECO)
- Crypto officer (CO)
- Crypto user (CU)

Sintassi

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per accedere con l'autenticazione a due fattori (2FA), utilizza il parametro `-2fa` e includi un percorso del file. Per ulteriori informazioni, vedi the section called "Arguments (Argomenti)".

```
loginHSM <user-type> <user-name> <password> |-hpswd> [-2fa </path/to/authdata>]
```

```
logoutHSM
```

Esempi

Questi esempi mostrano come utilizzare loginHSM e come logoutHSM accedere e disconnettersi da tutti HSMs in un cluster.

Example: Accedere HSMs a un cluster

Questo comando consente di accedere HSMs a tutti gli utenti di un cluster con le credenziali di un utente CO denominato `admin` e una password di. `co12345` L'output mostra che il comando è stato eseguito correttamente e che ci si è connessi a HSMs (che, in questo caso, sono `server 0` `eserver 1`).

```
aws-cloudhsm>loginHSM CO admin co12345

loginHSM success on server 0(10.0.2.9)
loginHSM success on server 1(10.0.3.11)
```

Example: accedi con una password nascosta

Questo comando è lo stesso dell'esempio precedente, tranne che questa volta si specifica che il sistema deve nascondere la password.

```
aws-cloudhsm>loginHSM CO admin -hpswd
```

Il sistema ti invita a inserire la tua password. Si immette la password, il sistema la nasconde e l'output mostra che il comando ha avuto successo e che ci si è connessi al HSMs.

```
Enter password:

loginHSM success on server 0(10.0.2.9)
loginHSM success on server 1(10.0.3.11)

aws-cloudhsm>
```

Example: Disconnettiti da un modulo HSM

Questo comando consente di disconnettersi dai file HSMs a cui si è attualmente connessi (che, in questo caso, sono `server 0` e). `server 1` L'output mostra che il comando è stato eseguito correttamente e che l'utente si è disconnesso da. HSMs

```
aws-cloudhsm>logoutHSM

logoutHSM success on server 0(10.0.2.9)
logoutHSM success on server 1(10.0.3.11)
```

Arguments (Argomenti)

Immetti gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizza il parametro `-hpswd` per mascherare la password. Per accedere con l'autenticazione a due fattori (2FA), utilizza il parametro `-2fa` e includi un percorso del file. Per ulteriori informazioni sull'utilizzo della 2FA, vedi Gestisci l'autenticazione 2FA degli utenti

```
loginHSM <user-type> <user-name> <password> |-hpswd> [-2fa </path/to/authdata>]
```

<tipo utente>

Speciifica il tipo di utente che accede a. HSMs Per ulteriori informazioni, vedi Tipo di utente sopra.

Obbligatorio: sì

<nome utente>

Speciifica il nome utente dell'utente che accede a. HSMs

Obbligatorio: sì

<password | -hpswd >

Specifica la password dell'utente che accede a. HSMs Per nascondere la password, utilizza il parametro `-hpswd` al posto della password e segui le istruzioni.

Obbligatorio: sì

[-2fa </ >] path/to/authdata

Specifica che il sistema deve utilizzare un secondo fattore per autenticare questo utente CO abilitato alla 2FA. Per ottenere i dati necessari per l'accesso con 2FA, includi un percorso verso una posizione nel file system con un nome di file dopo il parametro `-2fa`. Per ulteriori informazioni sull'utilizzo della 2FA, vedi Gestisci l'autenticazione 2FA degli utenti.

Obbligatorio: no

Argomenti correlati

- Nozioni di base su cloudhsm_mgmt_util
- Attivazione del cluster

Associa AWS CloudHSM gli utenti alle chiavi utilizzando CMU

Usa il registerQuorumPubKey comando in AWS CloudHSM cloudhsm_mgmt_util per associare gli utenti del modulo di sicurezza hardware (HSM) a coppie di chiavi RSA-2048 asimmetriche. Una volta associati gli utenti HSM alle chiavi, tali utenti possono utilizzare la chiave privata per approvare le richieste del quorum e il cluster può utilizzare la chiave pubblica registrata per verificare che la firma provenga dall'utente. Per ulteriori informazioni sull'autenticazione del quorum, vedi Gestire l'Autenticazione del Quorum (Controllo Acessi M of N).

> **Tip**
>
> Nella AWS CloudHSM documentazione, l'autenticazione quorum viene talvolta definita M of N (MoFN), il che significa un minimo di M approvatori su un numero totale di N approvatori.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO)

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
registerQuorumPubKey <user-type> <user-name> <registration-token> <signed-registration-token> <public-key>
```

Esempi

Questo esempio mostra come usare registerQuorumPubKey per registrare i crypto officer (CO) come approvatori delle richieste di autenticazione del quorum. Per eseguire questo comando, è necessario disporre di una coppia di chiavi RSA-2048 asimmetriche, un token firmato e un token non firmato. Per ulteriori informazioni sui requisiti, vedi the section called "Arguments (Argomenti)".

Example: Registra un utente HSM per l'autenticazione del quorum

Questo esempio registra un CO denominato `quorum_officer` come approvatore per l'autenticazione del quorum.

```
aws-cloudhsm> registerQuorumPubKey CO <quorum_officer> </path/to/quorum_officer.token> </path/to/quorum_officer.token.sig> </path/to/quorum_officer.pub>

*************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
***************************************************************

Do you want to continue(y/n)?y
registerQuorumPubKey success on server 0(10.0.0.1)
```

Il comando finale utilizza il comando ListUsers per verificare che `quorum_officer` sia registrato come utente MoFN.

```
aws-cloudhsm> listUsers
Users on server 0(10.0.0.1):
```

```
Number of users found:3

    User Id             User Type       User Name                       MofnPubKey
  LoginFailureCnt         2FA
         1              PCO             admin                             NO
           0              NO
         2              AU              app_user                          NO
           0              NO
         3              CO              quorum_officer                   YES
           0              NO
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
registerQuorumPubKey <user-type> <user-name> <registration-token> <signed-registration-token> <public-key>
```

<tipo-utente>

Specifica il tipo di utente. Questo parametro è obbligatorio.

Per informazioni dettagliate sui tipi di utente su un HSM, vedi Tipi di utente HSM per Management Utility AWS CloudHSM.

Valori validi:

- CO: i crypto officer possono gestire gli utenti ma non le chiavi.

Obbligatorio: sì

<nome-utente>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è un trattino basso (_).

Non puoi modificare il nome di un utente dopo che è stato creato. Nei comandi cloudhsm_mgmt_util, il tipo di utente e la password sono sensibili alle maiuscole e alle minuscole, il nome utente no.

Obbligatorio: sì

<registrazione-token>

Specifica il percorso di un file che contiene un token di registrazione non firmato. Può contenere qualsiasi dato casuale della dimensione massima del file di 245 byte. Per ulteriori informazioni sulla creazione di un token di registrazione non firmato, vedi Creazione e firma di un token di registrazione.

Obbligatorio: sì

<signed-registration-token>

Speciifica il percorso di un file che contiene l'hash firmato dal meccanismo SHA256 _PKCS del token di registrazione. Per ulteriori informazioni, vedi Creazione e firma di un token di registrazione.

Obbligatorio: sì

<chiavi-pubbliche>

Specifica il percorso di un file che contiene la chiave pubblica di una coppia di chiavi RSA-2048 asimmetriche. Utilizza la chiave privata per firmare il token di registrazione. Per ulteriori informazioni, vedi Creazione di una coppia di chiavi RSA.

Obbligatorio: sì

> **Note**
>
> Il cluster utilizza la stessa chiave per l'autenticazione del quorum e per l'autenticazione a due fattori (2FA). Ciò significa che non è possibile ruotare una chiave quorum per un utente che ha abilitato la 2FA usando registerQuorumPubKey. Per ruotare la chiave, è necessario utilizzare changePswd. Per ulteriori informazioni sull'utilizzo dell'autenticazione del quorum e dell'autenticazione a due fattori, vedi Autenticazione del quorum e 2FA.

Argomenti correlati

- Creazione di una coppia di chiavi RSA
- Crea e firma un token di registrazione
- Registrazione della chiave pubblica con HSM
- Gestisci Autenticazione del Quorum (controllo accessi M of N)
- Autenticazione quorum e 2FA

- ElencaUtenti

Interagisci con un HSM in un AWS CloudHSM cluster utilizzando CMU

Utilizzate il server comando in AWS CloudHSM cloudhsm_mgmt_util per accedere alla modalità server e interagire direttamente con una particolare istanza del modulo di sicurezza hardware (HSM).

Normalmente, quando si esegue un comando in cloudhsm_mgmt_util, il comando ha effetto su tutto il cluster designato (modalità globale). HSMs In alcuni casi, tuttavia, potrebbe essere necessario eseguire i comandi in un singolo modulo HSM. Ad esempio, nel caso in cui la sincronizzazione automatica non riesca, potresti aver bisogno di sincronizzare le chiavi e gli utenti in un modulo HSM, per mantenere la coerenza nel cluster.

Dopo la corretta inizializzazione, il prompt dei comandi `aws-cloudhsm >` viene sostituito dal prompt dei comandi `server >`.

Per uscire dalla modalità server, utilizza il comando `exit`. Una volta completata l'uscita, verrà di nuovo visualizzato il prompt dei comandi cloudhsm_mgmt_util.

Prima di eseguire qualsiasi comando key_mgmt_util, devi avviare key_mgmt_util.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti.

Prerequisiti

Per passare alla modalità server, è necessario conoscere il numero di server del modulo HSM di destinazione. I numeri di server sono elencati nell'output di traccia generato da cloudhsm_mgmt_util all'avvio. I numeri dei server vengono assegnati nello stesso ordine in cui appaiono nel file di configurazione. HSMs Per questo esempio, supponiamo che `server 0` sia il server corrispondente al modulo HSM desiderato.

Sintassi

Per avviare la modalità server:

```
server <server-number>
```

Per uscire dalla modalità server:

```
server> exit
```

Esempio

Questo comando permette di passare alla modalità server in un modulo HSM con numero di server `0`.

```
aws-cloudhsm> server 0

Server is in 'E2' mode...
```

Per uscire dalla modalità server, utilizza il comando exit.

```
server0> exit
```

Arguments (Argomenti)

```
server <server-number>
```

<numero-server>

Specifica il numero di server del modulo HSM di destinazione.

Obbligatorio: sì

Il comando `exit` non ha argomenti.

Argomenti correlati

- syncKey
- createUser
- deleteUser

Imposta gli attributi delle AWS CloudHSM chiavi usando CMU

Usa il setAttribute comando in AWS CloudHSM cloudhsm_mgmt_util per modificare il valore degli attributi label, encrypt, decrypt, wrap e unwrap di una chiave in. HSMs Puoi utilizzare il comando

setAttribute anche in key_mgmt_util per convertire una chiave di sessione in una chiave persistente. Puoi modificare solo gli attributi di chiavi di tua proprietà.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini, aggiorna i file di configurazione per CMU. HSMs In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Crypto user (CU)

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
setAttribute <key handle> <attribute id>
```

Esempio

Questo esempio illustra come disattivare la funzionalità di decodifica di una chiave simmetrica. Puoi utilizzare un comando come questo per configurare una chiave di wrapping in grado di eseguire e annullare il wrapping di altre chiavi ma non di crittografare o decodificare dati.

Per prima cosa, è necessario creare la chiave di wrapping. Questo comando utilizza genSymKeykey_mgmt_util per generare una chiave simmetrica AES a 256 bit. L'output mostra che la nuova chiave presenta l'handle 14.

```
$  genSymKey -t 31 -s 32 -l aes256

Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 14
```

```
        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Poi, è necessario confermare il valore corrente dell'attributo di decodifica. Per ottenere l'ID dell'attributo di decodifica, utilizza listAttributes. L'output indica che la costante che rappresenta l'attributo OBJ_ATTR_DECRYPT è 261. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

```
aws-cloudhsm> listAttributes

Following are the possible attribute values for getAttribute:

      OBJ_ATTR_CLASS                = 0
      OBJ_ATTR_TOKEN                = 1
      OBJ_ATTR_PRIVATE              = 2
      OBJ_ATTR_LABEL                = 3
      OBJ_ATTR_TRUSTED              = 134
      OBJ_ATTR_KEY_TYPE             = 256
      OBJ_ATTR_ID                   = 258
      OBJ_ATTR_SENSITIVE            = 259
      OBJ_ATTR_ENCRYPT              = 260
      OBJ_ATTR_DECRYPT              = 261
      OBJ_ATTR_WRAP                 = 262
      OBJ_ATTR_UNWRAP               = 263
      OBJ_ATTR_SIGN                 = 264
      OBJ_ATTR_VERIFY               = 266
      OBJ_ATTR_DERIVE               = 268
      OBJ_ATTR_LOCAL                = 355
      OBJ_ATTR_MODULUS              = 288
      OBJ_ATTR_MODULUS_BITS         = 289
      OBJ_ATTR_PUBLIC_EXPONENT      = 290
      OBJ_ATTR_VALUE_LEN            = 353
      OBJ_ATTR_EXTRACTABLE          = 354
      OBJ_ATTR_NEVER_EXTRACTABLE    = 356
      OBJ_ATTR_ALWAYS_SENSITIVE     = 357
      OBJ_ATTR_DESTROYABLE          = 370
      OBJ_ATTR_KCV                  = 371
      OBJ_ATTR_WRAP_WITH_TRUSTED    = 528
      OBJ_ATTR_WRAP_TEMPLATE        = 1073742353
      OBJ_ATTR_UNWRAP_TEMPLATE      = 1073742354
      OBJ_ATTR_ALL                  = 512
```

Per ottenere il valore corrente dell'attributo di decodifica per la chiave 14, il comando successivo utilizza getAttribute in cloudhsm_mgmt_util.

L'output mostra che il valore dell'attributo decrypt è true (1) su entrambi i componenti del cluster. HSMs

```
aws-cloudhsm> getAttribute 14 261

Attribute Value on server 0(10.0.0.1):
OBJ_ATTR_DECRYPT
0x00000001

Attribute Value on server 1(10.0.0.2):
OBJ_ATTR_DECRYPT
0x00000001
```

Questo comando utilizza setAttribute per modificare il valore dell'attributo di decodifica (attributo 261) della chiave 14 in `0`. In questo modo viene disabilitata la funzionalità di decodifica sulla chiave.

L'output mostra che il comando ha avuto successo su entrambi i componenti del cluster HSMs .

```
aws-cloudhsm> setAttribute 14 261 0
************************CAUTION*********************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
************************************************************

Do you want to continue(y/n)? y
setAttribute success on server 0(10.0.0.1)
setAttribute success on server 1(10.0.0.2)
```

Il comando finale ripete il comando getAttribute. E, come in precedenza, ottiene l'attributo di decodifica (attributo 261) della chiave 14.

Questa volta, l'output mostra che il valore dell'attributo decrypt è falso (0) su entrambi i componenti del cluster HSMs .

```
aws-cloudhsm > getAttribute 14 261
Attribute Value on server 0(10.0.3.6):
```

```
OBJ_ATTR_DECRYPT
0x00000000

Attribute Value on server 1(10.0.1.7):
OBJ_ATTR_DECRYPT
0x00000000
```

Arguments (Argomenti)

```
setAttribute <key handle> <attribute idb
```

<handle-chiave>

Specifica l'handle della chiave posseduta. È possibile specificare una sola chiave in ogni comando. Per individuare l'handle di una chiave, utilizza findKey in key_mgmt_util. Per trovare gli utenti di una chiave, usa. getKeyInfo

Obbligatorio: sì

<attributo id>

Specifica la costante che rappresenta l'attributo da modificare. È possibile specificare un solo attributo in ogni comando. Per ottenere gli attributi e i valori interi, utilizza listAttributes. Per informazioni sull'interpretazione degli attributi chiave, vedi. AWS CloudHSM riferimento agli attributi chiave per KMU

Valori validi:

- 3 – OBJ_ATTR_LABEL.
- 134 – OBJ_ATTR_TRUSTED.
- 260 – OBJ_ATTR_ENCRYPT.
- 261 – OBJ_ATTR_DECRYPT.
- 262 – OBJ_ATTR_WRAP.
- 263 – OBJ_ATTR_UNWRAP.
- 264 – OBJ_ATTR_SIGN.
- 266 – OBJ_ATTR_VERIFY.
- 268 – OBJ_ATTR_DERIVE.
- 370 – OBJ_ATTR_DESTROYABLE.

- 528 – `OBJ_ATTR_WRAP_WITH_TRUSTED`.
- 1073742353 – `OBJ_ATTR_WRAP_TEMPLATE`.
- 1073742354 – `OBJ_ATTR_UNWRAP_TEMPLATE`.

Campo obbligatorio: sì

Argomenti correlati

- setAttribute in key_mgmt_util
- getAttribute
- listAttributes
- Riferimento per l'attributo della chiave

Uscire dalla CMU

Usa il quit comando in cloudhsm_mgmt_util per AWS CloudHSM uscire da cloudhsm_mgmt_util. Qualsiasi tipo di utente può utilizzare questo comando.

Prima di eseguire qualsiasi comando key_mgmt_util, devi avviare key_mgmt_util.

Tipo di utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti. Non è necessario che tu abbia eseguito l'accesso per eseguire questo comando.

Sintassi

```
quit
```

Esempio

Questo comando ti fa uscire da cloudhsm_mgmt_util. Una volta completata l'operazione, viene visualizzata di nuovo la riga di comando standard. Questo comando non ha parametri di output.

```
aws-cloudhsm> quit
```

```
disconnecting from servers, please wait...
```

Argomenti correlati

- Guida introduttiva a cloudhsm_mgmt_util

Condivisione AWS CloudHSM delle chiavi tramite CMU

Usa il shareKey comando in AWS CloudHSM cloudhsm_mgmt_util per condividere e annullare la condivisione delle chiavi che possiedi con altri utenti crittografici. Soltanto il proprietario della chiave può condividere e annullare la condivisione di una chiave. Puoi inoltre condividere una chiave quando viene creata.

Gli utenti che condividono la chiave possono utilizzarla in operazioni di crittografia, ma non possono eliminarla, esportarla, condividerla, annullarne la condivisione o modificarne gli attributi. Quando l'autenticazione del quorum è abilitata su una chiave, il quorum deve approvare tutte le operazioni che condividono o annullano la condivisione della chiave.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini, aggiorna i file di configurazione per CMU. HSMs In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto user (CU)

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

Tipo di utente: crypto user (CU)

```
shareKey <key handle> <user id> <(share/unshare key?) 1/0>
```

Esempio

I seguenti esempi mostrano come utilizzare shareKey per condividere e annullare la condivisione delle chiavi che possiedi insieme ad altri crypto user.

Example: condividi una chiave

Questo esempio utilizza shareKey la condivisione di una chiave privata ECC di proprietà dell'utente corrente con un altro utente crittografico su HSMs. Le chiavi pubbliche sono disponibili per tutti gli utenti dell'HSM, perciò non è possibile condividerle o annullarne la condivisione.

Il primo comando utilizza getKeyInfoper ottenere le informazioni utente per la chiave262177, una chiave privata ECC su HSMs.

L'output indica che la chiave 262177 è di proprietà dell'utente 3, ma non è condivisa.

```
aws-cloudhsm>getKeyInfo 262177

Key Info on server 0(10.0.3.10):

        Token/Flash Key,

        Owned by user 3

Key Info on server 1(10.0.3.6):

        Token/Flash Key,

        Owned by user 3
```

Questo comando utilizza shareKey per condividere la chiave 262177 con l'utente4, un altro utente crittografico su HSMs. L'argomento finale utilizza un valore di 1 per indicare un'operazione di condivisione.

L'output mostra che l'operazione è riuscita su entrambi i componenti HSMs del cluster.

```
aws-cloudhsm>shareKey 262177 4 1
************************CAUTION*********************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
```

```
*****************************************************************

Do you want to continue(y/n)?y
shareKey success on server 0(10.0.3.10)
shareKey success on server 1(10.0.3.6)
```

Per verificare l'esito positivo dell'operazione, l'esempio ripete il primo comando getKeyInfo.

L'output indica che la chiave 262177 è ora condivisa con l'utente 4.

```
aws-cloudhsm>getKeyInfo 262177

Key Info on server 0(10.0.3.10):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 1 user(s):

                4
Key Info on server 1(10.0.3.6):

        Token/Flash Key,

        Owned by user 3

        also, shared to following 1 user(s):

                4
```

Example: annulla la condivisione di una chiave

Questo esempio annulla la condivisione di una chiave simmetrica, ovvero elimina un crypto user dall'elenco di utenti con cui è condivisa la chiave.

Questo comando utilizza shareKey per rimuovere l'utente 4 dall'elenco di utenti con cui è condivisa la chiave 6. L'argomento finale utilizza un valore di 0 per indicare un'operazione di annullamento della condivisione.

L'output mostra che il comando è riuscito su entrambi. HSMs Di conseguenza, l'utente 4 non può più utilizzare la chiave 6 nelle operazioni di crittografia.

```
aws-cloudhsm>shareKey 6 4 0
************************CAUTION*******************************
This is a CRITICAL operation, should be done on all nodes in the
cluster. AWS does NOT synchronize these changes automatically with the
nodes on which this operation is not executed or failed, please
ensure this operation is executed on all nodes in the cluster.
**************************************************************

Do you want to continue(y/n)?y
shareKey success on server 0(10.0.3.10)
shareKey success on server 1(10.0.3.6)
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
shareKey <key handle> <user id> <(share/unshare key?) 1/0>
```

<handle-chiave>

Specifica l'handle della chiave posseduta. È possibile specificare una sola chiave in ogni comando. Per individuare l'handle di una chiave, utilizza findKey in key_mgmt_util. Per verificare di possedere una chiave, usa getKeyInfo.

Obbligatorio: sì

<id utente>

Specifica l'ID del crypto user (CU) con cui si condivide o si annulla la condivisione della chiave. Per trovare l'ID di un utente, utilizza ElencaUtenti.

Obbligatorio: sì

<condividi 1 o annulla condivisione 0>

Per condividere la chiave con l'utente specificato, digita 1. Per annullare la condivisione della chiave, ovvero per rimuovere l'utente specificato dall'elenco di utenti con cui è condivisa chiave, digita `0`.

Campo obbligatorio: sì

Argomenti correlati

- getKeyInfo

Sincronizza le chiavi nel AWS CloudHSM cluster utilizzando CMU

Usa il syncKey comando in AWS CloudHSM cloudhsm_mgmt_util per sincronizzare manualmente le chiavi tra istanze HSM all'interno di un cluster o tra cluster clonati. In generale, non è necessario utilizzare questo comando, in quanto le istanze dell'HSM in un cluster sincronizzano le chiavi automaticamente. Tuttavia, la sincronizzazione di chiavi tra cluster clonati deve essere eseguita manualmente. I cluster clonati vengono generalmente creati in regioni diverse per semplificare i processi di scalabilità globale e disaster recovery. AWS

Non è possibile utilizzare syncKey per sincronizzare le chiavi tra i cluster arbitrari: uno dei cluster deve essere stato creato da un backup dell'altro cluster. Inoltre, entrambi i cluster devono avere credenziali per CO e CU coerenti affinché l'operazione abbia successo. Per ulteriori informazioni, vedi Utenti HSM.

Per utilizzarlosyncKey, è necessario innanzitutto creare un file di AWS CloudHSM configurazione che specifichi un HSM dal cluster di origine e uno dal cluster di destinazione. Questo permetterà a cloudhsm_mgmt_util di connettersi a entrambe le istanze dell'HSM. Usa questo file di configurazione per avviare cloudhsm_mgmt_util. Quindi effettua l'accesso con le credenziali di un CO o un CU che possiede le chiavi che si desidera sincronizzare.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO)
- Crypto user (CU)

> **Note**
>
> COs può utilizzarlo syncKey su qualsiasi chiave, mentre CUs può utilizzare questo comando solo su chiavi di sua proprietà. Per ulteriori informazioni, consulta the section called "Tipi di utente".

Prerequisiti

Prima di iniziare, è necessario conoscere il `key handle` della chiave nel modulo HSM di origine affinché sia sincronizzato con il modulo HSM di destinazione. Per individuare `key handle`, utilizza il comando listUsers , per elencare tutti gli identificatori per gli utenti designati. Quindi, usa il findAllKeyscomando per trovare tutte le chiavi che appartengono a un particolare utente.

È inoltre necessario conoscere le `server IDs` assegnazioni all'origine e alla destinazione HSMs, che vengono mostrate nell'output di traccia restituito da cloudhsm_mgmt_util al momento dell'iniziazione. Questi vengono assegnati nello stesso ordine in cui appaiono nel file di configurazione. HSMs

Segui le istruzioni in Utilizzo di CMU tra cluster clonati e inizializza cloudhsm_mgmt_util con il nuovo file di configurazione. Quindi, passa alla modalità server nel modulo HSM di origine, eseguendo il comando server.

Sintassi

> **Note**
>
> Per eseguire syncKey, passa alla modalità server nel modulo HSM che contiene la chiave da sincronizzare.

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

Tipo di utente: crypto user (CU)

```
syncKey <key handle> <destination hsm>
```

Esempio

Esegui il comando server per accedere al modulo HSM di origine e passare alla modalità server. Per questo esempio, supponiamo che `server 0` sia l'HSM di origine.

```
aws-cloudhsm> server 0
```

Esegui il comando syncKey. In questo esempio, supponiamo che la chiave `261251` sia sincronizzata su `server 1`.

```
aws-cloudhsm> syncKey 261251 1
syncKey success
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
syncKey <key handle> <destination hsm>
```

<handle chiave>

Specifica l'handle della chiave da sincronizzare. È possibile specificare una sola chiave in ogni comando. Per ottenere l'handle di una chiave, usalo findAllKeysmentre sei connesso a un server HSM.

Obbligatorio: sì

<hsm di destinazione>

Specifica il numero del server sul quale si sta sincronizzando una chiave.

Campo obbligatorio: sì

Argomenti correlati

- ElencaUtenti
- findAllKeys
- descrivi i cluster in AWS CLI
- server

Sincronizza gli utenti nel AWS CloudHSM cluster utilizzando CMU

Usa il syncUser comando in AWS CloudHSM cloudhsm_mgmt_util per sincronizzare manualmente gli utenti crittografici () o gli ufficiali di crittografia () tra istanze HSM all'interno di un cluster o tra cluster clonati. CUs COs AWS CloudHSM non sincronizza automaticamente gli utenti. In genere, gli utenti vengono gestiti in modalità globale in modo che tutti i HSMs componenti di un cluster vengano aggiornati insieme. Potrebbe essere necessario utilizzare syncUser se un HSM viene

accidentalmente desincronizzato (per esempio, a causa di modifiche della password) o se si desidera ruotare le credenziali utente tra i cluster clonati. I cluster clonati vengono generalmente creati in diverse AWS regioni per semplificare la scalabilità globale e i processi di disaster recovery.

Prima di eseguire qualsiasi comando CMU è necessario avviare CMU e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account utente autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini HSMs, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere efficaci per tutti i membri HSMs del cluster.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO)

Prerequisiti

Prima di iniziare, è necessario conoscere il `user ID` dell'utente nel modulo HSM di origine affinché sia sincronizzato con il modulo HSM di destinazione. Per trovare il`user ID`, usa il comando ListUsers per elencare tutti gli utenti di un HSMs cluster.

È inoltre necessario conoscere i valori `server ID` assegnati all'origine e alla destinazione HSMs, che vengono visualizzati nell'output di traccia restituito da cloudhsm_mgmt_util al momento dell'avvio. Questi vengono assegnati nello stesso ordine in cui appaiono nel file di configurazione. HSMs

Se stai eseguendo la sincronizzazione HSMs tra cluster clonati, segui le istruzioni in Uso della CMU tra cluster clonati e inizializza cloudhsm_mgmt_util con il nuovo file di configurazione.

Quando tutto è pronto per l'esecuzione di syncUser, passa alla modalità server nell'HSM sorgente emettendo il comando server.

Sintassi

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
syncUser <user ID> <server ID>
```

Esempio

Esegui il comando server per accedere al modulo HSM di origine e passare alla modalità server. Per questo esempio, supponiamo che `server 0` sia l'HSM di origine.

```
aws-cloudhsm> server 0
```

Esegui il comando syncUser. Per questo esempio, supponiamo che l'utente 6 sia l'utente da sincronizzare e `server 1` l'HSM di destinazione.

```
server 0> syncUser 6 1
ExtractMaskedObject: 0x0 !
InsertMaskedObject: 0x0 !
syncUser success
```

Arguments (Argomenti)

Poiché questi comandi non dispongono di parametri denominati, è necessario immettere gli argomenti nell'ordine specificato nei diagrammi sintattici.

```
syncUser <user ID> <server ID>
```

<ID utente>

Specifica l'ID dell'utente da sincronizzare. È possibile specificare un solo utente in ogni comando. Per ottenere l'ID di un utente, utilizzare ElencaUtenti.

Obbligatorio: sì

<ID server>

Specifica il numero del server dell'HSM sul quale si sta sincronizzando un utente.

Campo obbligatorio: sì

Argomenti correlati

- ElencaUtenti
- descrivi i cluster in AWS CLI
- server

AWS CloudHSM Utilità di gestione delle chiavi (KMU)

L'utilità di gestione delle chiavi (KMU) è uno strumento a riga di comando AWS CloudHSM che aiuta gli utenti crittografici (CU) a gestire le chiavi sui moduli di sicurezza hardware (HSM). Include vari comandi per creare, eliminare, importare ed esportare le chiavi, ottenere e impostare attributi, trovare chiavi ed eseguire operazioni di crittografia.

KMU e CMU fanno parte della suite Client SDK 3.

Per una guida rapida, vedi Guida introduttiva a AWS CloudHSM key_mgmt_util. Per informazioni dettagliate sui comandi, vedi Riferimento per i comandi della AWS CloudHSM Key Management Utility. Per informazioni sull'interpretazione degli attributi chiave, vedi. AWS CloudHSM riferimento agli attributi chiave per KMU

Per utilizzare key_mgmt_util con Linux, effettua la connessione all'istanza del client e vedi Installa e configura il AWS CloudHSM client per KMU (Linux). Se utilizzi Windows, vedi Installa e configura il AWS CloudHSM client per KMU (Windows).

Argomenti

- Guida introduttiva a AWS CloudHSM key_mgmt_util
- Installa e configura il AWS CloudHSM client per KMU (Linux)
- Installa e configura il AWS CloudHSM client per KMU (Windows)
- Riferimento per i comandi della AWS CloudHSM Key Management Utility

Guida introduttiva a AWS CloudHSM key_mgmt_util

AWS CloudHSM include due strumenti da riga di comando con il software client.AWS CloudHSM Lo strumento cloudhsm_mgmt_util include comandi per gestire gli utenti HSM. Lo strumento key_mgmt_util include comandi per gestire le chiavi. Per iniziare a utilizzare lo strumento a riga di comando key_mgmt_util, vedi i seguenti argomenti.

Argomenti

- Configura AWS CloudHSM key_mgmt_util
- Accedi a un cluster usando KMU HSMs AWS CloudHSM
- Esci da un AWS CloudHSM cluster utilizzando KMU HSMs
- AWS CloudHSM Arresta key_mgmt_util

In caso di messaggio di errore o risultato imprevisto per un comando, vedi gli argomenti Risoluzione dei problemi AWS CloudHSM per ricevere assistenza. Per dettagli sui comandi key_mgmt_util, vedi Riferimento per i comandi della AWS CloudHSM Key Management Utility.

Configura AWS CloudHSM key_mgmt_util

Completa la seguente configurazione prima di utilizzare key_mgmt_util (KMU). AWS CloudHSM

Argomenti

- Passaggio 1. AWS CloudHSM Avvia il client
- Passaggio 2. Avvio di key_mgmt_util

Passaggio 1. AWS CloudHSM Avvia il client

Prima di utilizzare key_mgmt_util, è necessario avviare il client. AWS CloudHSM Il client è un demone che stabilisce end-to-end una comunicazione crittografata con i membri del cluster. HSMs Lo strumento key_mgmt_util utilizza la connessione client per comunicare con i membri del cluster. HSMs Senza ciò, key_mgmt_util non funziona.

Per AWS CloudHSM avviare il client

Usa il seguente comando per avviare il AWS CloudHSM client.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:
\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Passaggio 2. Avvio di key_mgmt_util

Dopo aver avviato il AWS CloudHSM client, utilizzate il seguente comando per avviare key_mgmt_util.

Amazon Linux

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Amazon Linux 2

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

CentOS 7

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

CentOS 8

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

RHEL 7

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

RHEL 8

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Ubuntu 16.04 LTS

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Ubuntu 18.04 LTS

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\key_mgmt_util.exe"
```

Il prompt cambia in `Command:` quando key_mgmt_util è in esecuzione.

Se il comando ha esito negativo, ad esempio se restituisce un messaggio `Daemon socket connection error`, prova ad aggiornare il file di configurazione.

Accedi a un cluster usando KMU HSMs AWS CloudHSM

Utilizzate il loginHSM comando in key_mgmt_util (KMU) per accedere ai moduli di sicurezza hardware (HSM) in un cluster. AWS CloudHSM Il comando seguente effettua l'accesso come crypto user (CU) nominato `example_user`. L'output indica un accesso riuscito per tutti e tre i membri del cluster. HSMs

```
Command:  loginHSM -u CU -s example_user -p <PASSWORD>
Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Di seguito è mostrata la sintassi del comando loginHSM.

```
Command:  loginHSM -u <USER TYPE> -s <USERNAME> -p <PASSWORD>
```

Esci da un AWS CloudHSM cluster utilizzando KMU HSMs

Utilizzate il logoutHSM comando in key_mgmt_util (KMU) per disconnettervi dai moduli di sicurezza hardware (HSM) in un cluster. AWS CloudHSM

```
Command:  logoutHSM
Cfm3LogoutHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

AWS CloudHSM Arresta key_mgmt_util

Usa il comando per fermare key_mgmt_util. exit AWS CloudHSM

```
Command:  exit
```

Installa e configura il AWS CloudHSM client per KMU (Linux)

Per interagire con il modulo di sicurezza hardware (HSM) del AWS CloudHSM cluster utilizzando key_mgmt_util (KMU), è necessario il software client per Linux. AWS CloudHSM Ti consigliamo di installarlo nell'istanza del client EC2 Linux creata precedentemente. Puoi installare un client anche se utilizzi Windows. Per ulteriori informazioni, vedi Installa e configura il AWS CloudHSM client per KMU (Windows).

Processi

- Passaggio 1. AWS CloudHSM Installa il client e gli strumenti da riga di comando
- Passaggio 2. Modifica la configurazione del client

Passaggio 1. AWS CloudHSM Installa il client e gli strumenti da riga di comando

Connect all'istanza client ed esegui i seguenti comandi per scaricare e installare il AWS CloudHSM client e gli strumenti da riga di comando.

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

```
sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

```
sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb
```

Passaggio 2. Modifica la configurazione del client

Prima di poter utilizzare il AWS CloudHSM client per connettersi al cluster, è necessario modificare la configurazione del client.

Per modificare la configurazione del client

1. Copia il certificato di emissione, quello utilizzato per firmare il certificato del cluster, nel seguente percorso sull'istanza del client: `/opt/cloudhsm/etc/customerCA.crt`. Per copiare il certificato in questa posizione, sono necessarie le autorizzazioni dell'utente root dell'istanza sull'istanza del client.
2. Utilizzate il seguente comando configure per aggiornare i file di configurazione per il AWS CloudHSM client e gli strumenti della riga di comando, specificando l'indirizzo IP dell'HSM nel cluster. Per ottenere l'indirizzo IP dell'HSM, visualizza il cluster nella AWS CloudHSM console o esegui il comando. describe-clusters AWS CLI Nell'output del comando, l'indirizzo IP del modulo HSM è il valore del campo `EniIp`. Se disponi di più di un HSM, scegli l'indirizzo IP per ognuno di HSMs essi, indipendentemente da quale.

   ```
   sudo /opt/cloudhsm/bin/configure -a <IP address>

   Updating server config in /opt/cloudhsm/etc/cloudhsm_client.cfg
   Updating server config in /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

3. Passa a Attiva il cluster in AWS CloudHSM.

Installa e configura il AWS CloudHSM client per KMU (Windows)

Per utilizzare un modulo di sicurezza hardware (HSM) nel AWS CloudHSM cluster su Windows utilizzando key_mgmt_util (KMU), è necessario il software client per Windows. AWS CloudHSM È consigliabile installarlo nell'istanza di Windows Server creata in precedenza.

Per installare (o aggiornare) le versioni più recenti del client Windows e degli strumenti a riga di comando

1. Connettersi all'istanza di Windows Server.
2. Scaricate la versione più recente (.msi) dalla pagina dei download. AWSCloudHSMClient-latest

3. Vai al percorso di download ed esegui il programma di installazione (AWSCloudHSMClient-latest.msi) con privilegi amministrativi.
4. Segui le istruzioni del programma di installazione, quindi scegli Chiudi al termine dell'installazione.
5. Copia il certificato di emissione autofirmato, quello utilizzato per firmare il certificato del cluster, nella cartella `C:\ProgramData\Amazon\CloudHSM`.
6. Esegui questo comando per aggiornare i file di configurazione. Assicurati di arrestare e avviare il client durante la riconfigurazione se decidi di aggiornarla:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM IP address>
```

7. Vai a Attiva il cluster in AWS CloudHSM.

Note:

- Se aggiorni il client, i file di configurazione esistenti di installazioni precedenti non verranno sovrascritti.
- Il programma di installazione del AWS CloudHSM client per Windows registra automaticamente l'API di crittografia: Next Generation (CNG) e Key Storage Provider (KSP). Per disinstallare il client, esegui di nuovo il programma di installazione e segui le istruzioni per la disinstallazione.
- Se usi Linux, puoi installare il client Linux. Per ulteriori informazioni, vedi Installa e configura il AWS CloudHSM client per KMU (Linux).

Riferimento per i comandi della AWS CloudHSM Key Management Utility

Lo strumento da riga di comando key_mgmt_util consente di gestire le chiavi nei moduli di sicurezza hardware (HSM) del AWS CloudHSM cluster, incluse la creazione, l'eliminazione e la ricerca delle chiavi e dei relativi attributi. Include vari comandi, ognuno dei quali è descritto in dettaglio in questo argomento.

Per una guida rapida, vedi Guida introduttiva a AWS CloudHSM key_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi. AWS CloudHSM riferimento agli attributi chiave per KMU Per informazioni sullo strumento a riga di comando cloudhsm_mgmt_util, che include i comandi di gestione dell'HSM e degli utenti del cluster, vedi AWS CloudHSM Utilità di gestione (CMU).

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Per elencare tutti i comandi key_mgmt_util, digita:

```
Command: help
```

Per ottenere assistenza su un determinato comando key_mgmt_util, digita:

```
Command: <command-name> -h
```

Per terminare la sessione key_mgmt_util, digita:

```
Command: exit
```

I seguenti argomenti descrivono i comandi in key_mgmt_util.

> **Note**
>
> Alcuni comandi in key_mgmt_util e cloudhsm_mgmt_util hanno lo stesso nome. Tuttavia, i comandi hanno in genere una sintassi diversa, un output diverso e funzionalità leggermente diverse.

Comando	Description
aesWrapUnwrap	Consente di crittografare e decodificare i contenuti di una chiave in un file.
EliminaChiave	Elimina una chiave da. HSMs
StringaErrore2	Consente di recuperare l'errore che corrisponde a un codice di errore esadecimale.
Esci	Esce da key_mgmt_util.
exportPrivateKey	Consente di esportare su un file su disco una copia di una chiave privata.
exportPubKey	Consente di esportare da un HSM su un file una copia di una chiave pubblica.

Comando	Description
exSymKey	Esporta una copia in testo semplice di una chiave simmetrica da a un file. HSMs
extractMaskedObject	Estrae una chiave da un HSM come un file oggetto nascosto.
TrovaChiave	Consente di cercare le chiavi in base al valore dell'attributo della chiave.
findSingleKey	Verifica che esista una chiave su tutti i componenti del cluster. HSMs
Gen DSAKey Pair	Genera una coppia di key pair DSA (Digital Signing Algorithm) nel tuo HSMs.
coppia di geni ECCKey	Genera una coppia di chiavi Elliptic Curve Cryptography (ECC) nel tuo. HSMs
coppia di geni RSAKey	Genera una coppia di chiavi asimmetriche RSA nel tuo. HSMs
genSymKey	Genera una chiave simmetrica nel tuo HSMs
OttieniAttributo	Consente di recuperare i valori degli attributi di una chiave di AWS CloudHSM e li scrive in un file.
getCaviumPrivChiave	Crea una versione falsa in formato PEM di una chiave privata e la esporta in un file.
OttieniCertificato	Consente di recuperare certificati di partizioni HSM e li salva in un file.
getKeyInfo	Ottiene l'utente HSM IDs degli utenti che possono utilizzare la chiave. Se la chiave è controllata dal quorum, consente di recuperare il numero di utenti del quorum.

Comando	Description
help	Visualizza informazioni di aiuto sui comandi disponibili in key_mgmt_util.
importPrivateKey	Importa una chiave privata in un HSM.
importPubKey	Importa una chiave pubblica in un HSM.
imSymKey	Consente di importare nell'HSM una copia in testo normale di una chiave simmetrica da un file.
insertMaskedObject	Inserisce un oggetto nascosto da un file su disco in un HSM contenuto nel cluster correlato al cluster di origine dell'oggetto. I cluster correlati sono qualsiasi cluster generati da un backup del cluster di origine.
???	Determina se un determinato file contiene o meno una chiave privata reale o una chiave PEM di esempio.
listAttributes	Elenca gli attributi di una AWS CloudHSM chiave e le costanti che li rappresentano.
ElencaUtenti	Ottiene gli utenti inclusi in HSMs, il tipo e l'ID utente e altri attributi.
loginHSM e logoutHSM	Accedere e disconnettersi da un cluster. HSMs
ImpostaAttributo	Consente di convertire una chiave di sessione in una chiave persistente.
Firma	Generare una firma per un file utilizzando una chiave privata scelta.
unWrapKey	Importa una chiave incapsulata (crittografata) da un file in. HSMs

Comando	Description
Verifica	Verifica se una determinata chiave è stata utilizzata per firmare un determinato file.
wrapChiave	Consente di esportare una copia crittografata di una chiave dai moduli HSM a un file.

Crittografa e decrittografa un AWS CloudHSM file usando KMU

Utilizzate il aesWrapUnwrap comando in AWS CloudHSM key_mgmt_util per crittografare o decrittografare il contenuto di un file su disco. Questo comando è concepito per eseguire e annullare il wrapping delle chiavi di crittografia, ma è possibile utilizzarlo su qualsiasi file che contenga meno di 4 KB (4096 byte) di dati.

aesWrapUnwrapaputilizza AES Key Wrap con imbottitura PKCS #5. Utilizza una chiave AES sull'HSM come chiave di wrapping o di annullamento del wrapping. Quindi scrive il risultato su un altro file su disco.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
aesWrapUnwrap -h

aesWrapUnwrap -m <wrap-unwrap mode>
              -f <file-to-wrap-unwrap>
              -w <wrapping-key-handle>
              [-i <wrapping-IV>]
              [-out <output-file>]
```

Esempi

Questi esempi mostrano come utilizzare aesWrapUnwrap per crittografare e decodificare una chiave di crittografia in un file.

Example: eseguire il wrapping di una chiave di crittografia

Questo comando utilizza aesWrapUnwrap per eseguire il wrapping di una chiave Triple DES simmetrica che è stata esportata dall'HSM in testo normale nel file `3DES.key`. È possibile utilizzare un comando simile per eseguire il wrapping di qualsiasi chiave salvata in un file.

Il comando utilizza il parametro `-m` con il valore `1` per indicare la modalità di wrapping. Utilizza il parametro `-w` per specificare la chiave AES nell'HSM (handle della chiave 6) come chiave di wrapping. Scrive la risultante chiave con wrapping nel file `3DES.key.wrapped`.

L'output indica che il comando ha avuto successo e che l'operazione ha utilizzato il valore IV predefinito, che è quello preferito.

```
 Command:  aesWrapUnwrap -f 3DES.key -w 6 -m 1 -out 3DES.key.wrapped

        Warning: IV (-i) is missing.
                 0xA6A6A6A6A6A6A6A6 is considered as default IV
result data:
49 49 E2 D0 11 C1 97 22
17 43 BD E3 4E F4 12 75
8D C1 34 CF 26 10 3A 8D
6D 0A 7B D5 D3 E8 4D C2
79 09 08 61 94 68 51 B7

result written to file 3DES.key.wrapped

        Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

Example: annullare il wrapping di una chiave di crittografia

Questo esempio mostra come utilizzare aesWrapUnwrap per annullare il wrapping (decodificare) di una chiave con wrapping (crittografata) in un file. È possibile eseguire un'operazione come questa prima di importare una chiave sull'HSM. Ad esempio, se si tenta di utilizzare il imSymKeycomando per importare una chiave crittografata, viene restituito un errore perché la chiave crittografata non ha il formato richiesto per una chiave di testo semplice di quel tipo.

Il comando annulla il wrapping della chiave nel file `3DES.key.wrapped` e scrive il testo normare sul file `3DES.key.unwrapped`. Il comando utilizza il parametro `-m` con il valore `0` per indicare la modalità di annullamento del wrapping. Utilizza il parametro `-w` per specificare la chiave AES nell'HSM (handle della chiave 6) come chiave di wrapping. Scrive la risultante chiave con wrapping nel file `3DES.key.unwrapped`.

```
 Command:  aesWrapUnwrap -m 0 -f 3DES.key.wrapped -w 6 -out 3DES.key.unwrapped

        Warning: IV (-i) is missing.
                 0xA6A6A6A6A6A6A6A6 is considered as default IV
result data:
14 90 D7 AD D6 E4 F5 FA
A1 95 6F 24 89 79 F3 EE
37 21 E6 54 1F 3B 8D 62

result written to file 3DES.key.unwrapped

        Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-m

Specifica la modalità. Per eseguire il wrapping (crittografare) il contenuto del file, digita 1; per annullare il wrapping (decodificare) il contenuto del file, digita 0.

Obbligatorio: sì

-f

Specifica il file su cui eseguire il wrapping. Inserisci un file che contiene meno di 4 KB (4096 byte) di dati. Questa operazione è stata progettata per eseguire e annullare il wrapping delle chiavi di crittografia.

Obbligatorio: sì

-w

Specifica la chiave di wrapping. Immetti l'handle di una chiave AES sull'HSM. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Per creare una chiave di wrapping, usa genSymKeyper generare una chiave AES (tipo 31).

Obbligatorio: sì

-i

Specifica un valore iniziale (IV) alternativo per l'algoritmo. Utilizza il valore predefinito a meno che non si abbia una condizione speciale che richiede un'alternativa.

Default: `0xA6A6A6A6A6A6A6A6`. Il valore predefinito è stabilito nella specifica dell'algoritmo Wrapping Chiave AES.

Obbligatorio: no

-output

Specifica un nome alternativo per il file di output che contiene la chiave con o senza wrapping. Il valore predefinito è `wrapped_key` (per le operazioni di wrapping) e `unwrapped_key` (per le operazioni di annullamento del wrapping) nella directory locale.

Se il file esiste, il comando aesWrapUnwrap lo sovrascrive senza preavviso. Se il comando ha esito negativo, aesWrapUnwrap crea un file di output senza contenuto.

Impostazione predefinita: per il wrapping: `wrapped_key`; per l'annullamento del wrapping: `unwrapped_key`.

Obbligatorio: no

Argomenti correlati

- exSymKey
- imSymKey
- unWrapKey
- wrapKey

Eliminare una AWS CloudHSM chiave usando KMU

Utilizzate il deleteKey comando contenuto in AWS CloudHSM key_mgmt_util per eliminare una chiave dal modulo di sicurezza hardware (HSM) in un cluster. AWS CloudHSM Puoi eliminare soltanto una chiave alla volta. L'eliminazione di una chiave di una coppia di chiavi non influisce sull'altra chiave della coppia.

Soltanto il proprietario della chiave può eliminarla. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni di crittografia, ma non eliminarla.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
deleteKey -h

deleteKey -k
```

Esempi

Questi esempi mostrano come utilizzare per eliminare le chiavi dal tuo. deleteKey HSMs

Example: eliminazione di una chiave

Questo comando elimina la chiave con l'handle 6. Se il comando ha esito positivo, deleteKey restituisce messaggi di operazione riuscita da ciascun HSM del cluster.

```
Command: deleteKey -k 6

        Cfm3DeleteKey returned: 0x00 : HSM Return: SUCCESS

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: eliminazione di una chiave (errore)

Se il comando non riesce perché nessuna chiave dispone dell'handle specificato, deleteKey restituisce un messaggio di errore di handle in oggetto non valido.

```
Command: deleteKey -k 252126

        Cfm3FindKey returned: 0xa8 : HSM Error: Invalid object handle is passed to this
 operation

        Cluster Error Status
        Node id 1 and err state 0x000000a8 : HSM Error: Invalid object handle is passed
 to this operation
        Node id 2 and err state 0x000000a8 : HSM Error: Invalid object handle is passed
 to this operation
```

Se il comando non riesce perché l'utente corrente non è il proprietario della chiave, il comando restituisce un errore di accesso negato.

```
Command:  deleteKey -k 262152

Cfm3DeleteKey returned: 0xc6 : HSM Error: Key Access is denied.
```

Parameters

-h

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

-k

Specifica l'handle della chiave da eliminare. Per cercare gli handle di chiave nell'HSM, utilizza findKey.

Campo obbligatorio: sì

Argomenti correlati

- findKey

Descrivi un AWS CloudHSM errore utilizzando KMU

Utilizzate il comando Error2String helper in key_mgmt_util per restituire l'errore che corrisponde a un codice di errore esadecimale AWS CloudHSM key_mgmt_util. Puoi usare questo comando per la risoluzione dei problemi di comandi e script.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
Error2String -h

Error2String -r <response-code>
```

Esempi

Questi esempi mostrano come utilizzare Error2String per ottenere la stringa di errore per un codice di errore key_mgmt_util.

Example: ottenere una descrizione dell'errore

Questo comando ottiene la descrizione dell'errore per il codice di errore `0xdb`. La descrizione spiega che un tentativo di accedere a key_mgmt_util non è riuscito perché il tipo di utente è errato. Solo gli utenti crittografici (CU) possono accedere a key_mgmt_util.

```
        Command:  Error2String -r 0xdb

        Error Code db maps to HSM Error: Invalid User Type.
```

Example: trovare il codice di errore

Questo esempio illustra dove trovare il codice di errore in un errore key_mgmt_util. Il codice di errore, `0xc6`, viene visualizzato dopo la stringa: `Cfm3<command-name> returned: .`

In questo esempio, getKeyInfoindica che l'utente corrente (utente 4) può utilizzare la chiave nelle operazioni crittografiche. Tuttavia, quando l'utente cerca di utilizzare deleteKey per eliminare la chiave, il comando restituisce il codice di errore `0xc6`.

```
        Command:  deleteKey -k 262162

        Cfm3DeleteKey returned: <0xc6> : HSM Error: Key Access is denied

        Cluster Error Status

        Command:  getKeyInfo -k 262162

        Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

       Owned by user 3

       also, shared to following 1 user(s):

                4
```

Se l'errore `0xc6` ti viene notificato, puoi utilizzare un comando Error2String come questo per individuare l'errore. In questo caso, il comando `deleteKey` ha avuto esito negativo causando un errore di accesso negato in quanto la chiave è condivisa con l'utente corrente, ma è di proprietà di un altro utente. Solo i proprietari delle chiavi dispongono dell'autorizzazione per eliminare una chiave.

```
        Command:  Error2String -r 0xa8

        Error Code c6 maps to HSM Error: Key Access is denied
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-r

Specifica un codice di errore esadecimale. L'indicatore esadecimale `0x` è obbligatorio.

Obbligatorio: sì

Uscire dalla AWS CloudHSM KMU

Utilizzate il exit comando in AWS CloudHSM key_mgmt_util per uscire da key_mgmt_util. Una volta completata la disconnessione, verrà di nuovo visualizzata la riga di comando standard.

Prima di eseguire qualsiasi comando key_mgmt_util, devi avviare key_mgmt_util.

Sintassi

```
exit
```

Parameters

Questo comando non ha parametri.

Argomenti correlati

- Avvio di key_mgmt_util

Esportazione di una AWS CloudHSM chiave privata tramite KMU

Utilizzate il exportPrivateKey comando in AWS CloudHSM key_mgmt_util per esportare una chiave privata asimmetrica da un modulo di sicurezza hardware (HSM) a un file. L'HSM non consente l'esportazione diretta di chiavi in chiaro. Il comando esegue il wrapping della chiave privata utilizzando una chiave di wrapping AES specificata dall'utente, decodifica i byte soggetti a wrapping e copia la chiave privata in chiaro in un file.

Tuttavia, il comando exportPrivateKey non rimuove la chiave da HSM, non ne modifica gli attributi della chiave oppure impedisce di utilizzare la chiave in altre operazioni di crittografia. È possibile esportare la stessa chiave più volte.

È possibile esportare solo le chiavi private che hanno il valore dell'attributo `OBJ_ATTR_EXTRACTABLE` impostato su 1. È necessario specificare una chiave di wrapping AES con un valore di attributo 1 `OBJ_ATTR_WRAP` e `OBJ_ATTR_DECRYPT`. Per trovare gli attributi della chiave, utilizza il comando getAttribute.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
exportPrivateKey -h

exportPrivateKey -k <private-key-handle>
                 -w <wrapping-key-handle>
                 -out <key-file>
                 [-m <wrapping-mechanism>]
                 [-wk <wrapping-key-file>]
```

Esempi

Questo esempio illustra come utilizzare exportPrivateKey per esportare una chiave privata da un HSM.

Example: Esporta una chiave privata

Questo comando esporta una chiave privata con handle 15 utilizzando una chiave di wrapping con handle 16 per un file PEM chiamato `exportKey.pem`. Se il comando ha esito positivo, exportPrivateKey restituisce un messaggio di operazione riuscita.

```
Command: exportPrivateKey -k 15 -w 16 -out exportKey.pem

Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS

PEM formatted private key is written to exportKey.pem
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-k`

Specifica l'handle della chiave privata da esportare.

Obbligatorio: sì

`-w`

Specifica l'handle di una chiave di wrapping. Questo parametro è obbligatorio. Per trovare le handle della chiave, utilizza il comando findKey.

Per determinare se una chiave può essere utilizzata come chiave di wrapping, utilizzare getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_WRAP` (262). Per creare una chiave di wrapping, utilizza genSymKey per creare una chiave AES (tipo 31).

Se si utilizza il parametro `-wk` per specificare una chiave di annullamento del wrapping esterna, la chiave di wrapping `-w` viene utilizzata per eseguire il wrapping della chiave durante l'esportazione, ma non per annullarlo.

Obbligatorio: sì

`-out`

Consente di specificare il nome del file in cui verrà scritta la chiave privata esportata.

Obbligatorio: sì

`-m`

Specifica il meccanismo di wrapping della chiave privata in fase di esportazione. L'unico valore valido è `4`, che rappresenta il `NIST_AES_WRAP mechanism.`

Default: 4 (`NIST_AES_WRAP`)

Obbligatorio: no

`-wk`

Specifica la chiave da utilizzare per eseguire l'annullamento del wrapping della chiave esportata. Inserire il percorso e il nome di un file che contiene una chiave AES non crittografata.

Quando includi questo parametro. exportPrivateKey utilizza la chiave nel file `-w` per eseguire il wrapping della chiave esportata e utilizza la chiave specificata dal parametro `-wk` per annullarlo.

Impostazione predefinita: Utilizza il codice di wrapping specificato nel parametro `-w` per eseguire il wrapping e per annullarlo.

Obbligatorio: no

Argomenti correlati

- importPrivateKey
- wrapKey
- unWrapKey
- genSymKey

Esporta una AWS CloudHSM chiave pubblica usando KMU

Utilizzate il exportPubKey comando in AWS CloudHSM key_mgmt_util per esportare una chiave pubblica in un HSM in un file. È possibile utilizzarlo per esportare le chiavi pubbliche che generi su un HSM. È inoltre possibile utilizzare questo comando per esportare le chiavi pubbliche importate in un HSM, ad esempio quelle importate tramite il comando importPubKey.

L'operazione exportPubKey copia il materiale della chiave su un file specificato. Tuttavia, non rimuove la chiave dall'HSM, non ne modifica gli attributi e neppure impedisce di utilizzare la chiave in altre operazioni di crittografia. È possibile esportare la stessa chiave più volte.

È possibile esportare solo le chiavi pubbliche che hanno il valore `OBJ_ATTR_EXTRACTABLE` pari a `1`. Per trovare gli attributi della chiave, utilizza il comando getAttribute.

Prima di eseguire un comando `key_mgmt_util`, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
exportPubKey -h

exportPubKey -k <public-key-handle>
             -out <key-file>
```

Esempi

Questo esempio illustra come utilizzare exportPubKey per esportare una chiave pubblica da un HSM.

Example: Esporta una chiave pubblica

Questo comando esporta una chiave pubblica con handle `10` su un file denominato `public.pem`. Se il comando ha esito positivo, exportPubKey restituisce un messaggio di operazione riuscita.

```
Command: exportPubKey -k 10 -out public.pem

PEM formatted public key is written to public.pem

Cfm3ExportPubKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-k`

Specifica l'handle della chiave pubblica da esportare.

Obbligatorio: sì

`-out`

Specifica il nome del file in cui verrà scritta la chiave pubblica esportata.

Campo obbligatorio: sì

Argomenti correlati

- importPubKey
- Genera Chiavi

Esporta una copia in testo semplice di una AWS CloudHSM chiave usando KMU

Utilizzate il exSymKey comando dello strumento AWS CloudHSM key_mgmt_util per esportare una copia in testo semplice di una chiave simmetrica dal modulo di sicurezza hardware (HSM) e salvarla in un file su disco. Per esportare una copia crittografata (su cui è stato eseguito il wrapping) di una chiave, usa wrapKey. Per importare una chiave in testo semplice, come quelle che esporta, usa. `exSymKey` imSymKey

Durante il processo di esportazione, il comando exSymKey utilizza una chiave AES specificata (la chiave di wrapping) per effettuare il wrapping (crittografia) e quindi annullare il wrapping (decodifica) della chiave da esportare. Tuttavia, il risultato dell'operazione di esportazione è una chiave di testo non crittografato (su cui è stato annullato il wrapping) su disco.

Soltanto il proprietario della chiave, ovvero l'utente CU che ha creato la chiave, è in grado di esportarla. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni di crittografia, ma non possono esportarla.

L'operazione exSymKey copia il materiale della chiave su un file specificato, ma non rimuove la chiave dall'HSM, non ne modifica gli attributi, né impedisce l'utilizzo della chiave nelle operazioni di crittografia. È possibile esportare la stessa chiave più volte.

exSymKey esporta solo le chiavi simmetriche. Per esportare le chiavi pubbliche, utilizzare exportPubKey. Per esportare le chiavi private, utilizzare exportPrivateKey.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
exSymKey -h

exSymKey -k <key-to-export>
         -w <wrapping-key>
         -out <key-file>
         [-m 4]
         [-wk <unwrapping-key-file> ]
```

Esempi

Questi esempi mostrano come utilizzare per exSymKey esportare le chiavi simmetriche che possiedi dal tuo. HSMs

Example: esportazione di una chiave simmetrica 3DES

Questo comando esporta una chiave simmetrica Triple DES (3DES) (handle chiave 7). Utilizza una chiave AES esistente (handle chiave 6) sull'HSM come chiave di wrapping. Quindi scrive il testo non crittografato della chiave 3DES sul file `3DES.key`.

L'output indica che la chiave 7 (la chiave 3DES) è stata sottoposta a wrapping e all'annullamento del wrapping e che è stata scritta sul file `3DES.key`.

> **Warning**
>
> Anche se l'output dice che una "Chiave simmetrica wrapped" è stata scritta sul file di output, il file di output contiene una chiave di testo non crittografata (unwrapped).

```
        Command: exSymKey -k 7 -w 6 -out 3DES.key

      Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

       Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS


Wrapped Symmetric Key written to file "3DES.key"
```

Example: esportazione con una chiave di wrapping solo per la sessione

Questo esempio illustra come utilizzare una chiave che esiste solo nella sessione come chiave di wrapping. Poiché sulla chiave da esportare è stato eseguito il wrapping che è poi stato immediatamente annullato ed è stata distribuita come testo non crittografato, non è necessario conservare la chiave di wrapping.

Questa serie di comandi esporta dall'HSM una chiave AES con handle di chiave 8. Utilizza una chiave di sessione AES creata specificatamente per questo scopo.

Il primo comando consente di genSymKeycreare una chiave AES a 256 bit. Utilizza il parametro `-sess` per creare una chiave che esiste solo nella sessione corrente.

L'output indica che l'HSM crea la chiave 262168.

```
        Command:  genSymKey -t 31 -s 32 -l AES-wrapping-key -sess

        Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 262168

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Quindi, l'esempio verifica che la chiave 8, la chiave da esportare, sia una chiave simmetrica estraibile. Inoltre verifica che la chiave di wrapping, la chiave 262168, sia una chiave AES che esiste solo nella sessione. È possibile utilizzare il comando findKey, ma questo esempio esporta gli attributi di entrambe le chiavi su file e utilizza `grep` per trovare i valori di attributo rilevanti nel file.

Questi comandi utilizzano `getAttribute` con un valore `-a` di `512` (tutti) per ottenere tutti gli attributi per le chiavi 8 e 262168. Per ulteriori informazioni sugli attributi delle chiavi, vedi the section called "Riferimento agli attributi chiave".

```
getAttribute -o 8 -a 512 -out attributes/attr_8
getAttribute -o 262168 -a 512 -out attributes/attr_262168
```

Questi comandi utilizzano `grep` per verificare gli attributi della chiave da esportare (chiave 8) e la chiave di wrapping valida solo per la sessione (chiave 262168).

```
    // Verify that the key to be exported is a symmetric key.
```

```
    $  grep -A 1 "OBJ_ATTR_CLASS" attributes/attr_8
    OBJ_ATTR_CLASS
    0x04

    // Verify that the key to be exported is extractable.
    $  grep -A 1 "OBJ_ATTR_KEY_TYPE" attributes/attr_8
    OBJ_ATTR_EXTRACTABLE
    0x00000001

    // Verify that the wrapping key is an AES key
    $  grep -A 1 "OBJ_ATTR_KEY_TYPE" attributes/attr_262168
    OBJ_ATTR_KEY_TYPE
    0x1f

    // Verify that the wrapping key is a session key
    $  grep -A 1 "OBJ_ATTR_TOKEN" attributes/attr_262168
    OBJ_ATTR_TOKEN
    0x00

    // Verify that the wrapping key can be used for wrapping
     $  grep -A 1 "OBJ_ATTR_WRAP" attributes/attr_262168
    OBJ_ATTR_WRAP
    0x00000001
```

Infine, utilizziamo un comando exSymKey per esportare la chiave 8 utilizzando la chiave di sessione (chiave 262168) come chiave di wrapping.

Quando la sessione scade, la chiave 262168 non è più disponibile.

```
        Command:  exSymKey -k 8 -w 262168 -out aes256_H8.key

        Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS


Wrapped Symmetric Key written to file "aes256_H8.key"
```

Example: utilizzo di una chiave di wrapping esterna

Questo esempio illustra come utilizzare una chiave di wrapping esterna per esportare una chiave dall'HSM.

Quando si esegue l'esportazione di una chiave dall'HSM, è necessario specificare una chiave AES nell'HSM che funga da chiave di wrapping. Per impostazione predefinita, la chiave di wrapping viene utilizzata per eseguire e annullare il wrapping della chiave da esportare. Tuttavia, è possibile utilizzare il parametro `-wk` per ordinare a exSymKey di utilizzare una chiave esterna in un file su disco per annullare il wrapping. Quando si esegue questa operazione, la chiave specificata dal parametro `-w` effettua il wrapping della chiave di destinazione e la chiave nel file specificata dal parametro `-wk` annulla il wrapping della chiave.

Poiché la chiave di wrapping deve essere una chiave AES, ovvero una chiave simmetrica, la chiave di wrapping nell'HSM e la chiave di unwrapping su disco devono avere lo stesso materiale chiave. Per eseguire questa operazione, è necessario importare la chiave di wrapping sull'HSM o esportare la chiave di wrapping dall'HSM prima dell'operazione di esportazione.

Questo esempio crea una chiave al di fuori dell'HSM e la importa nell'HSM. Utilizza la copia interna della chiave per effettuare il wrapping di una chiave simmetrica esportata e la copia della chiave nel file per annullare il wrapping.

Il primo comando utilizza OpenSSL per generare una chiave AES a 256 bit. Memorizza la chiave sul file `aes256-forImport.key`. Il comando OpenSSL non restituisce alcun output, ma è possibile utilizzare diversi comandi per confermare che l'operazione sia avvenuta con successo. Questo esempio utilizza lo strumento wc (conteggio delle parole), che conferma che il file contiene 32 byte di dati.

```
$  openssl rand -out keys/aes256-forImport.key 32

$ wc keys/aes256-forImport.key
 0  2 32 keys/aes256-forImport.key
```

Questo comando utilizza il comando imSymKey per importare la chiave AES dal file `aes256-forImport.key` all'HSM. Quando il comando viene completato, la chiave esiste nell'HSM con handle `262167` e nel file `aes256-forImport.key`.

```
Command:  imSymKey -f keys/aes256-forImport.key -t 31 -l aes256-imported -w 6

        Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
        Symmetric Key Unwrapped.  Key Handle: 262167

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Questo comando utilizza la chiave in un'operazione di esportazione. Il comando utilizza exSymKey per esportare la chiave 21, una chiave AES a 192 bit. Per effettuare il wrapping della chiave, utilizza la chiave 262167, che è la copia importata nell'HSM. Per annullare il wrapping della chiave, utilizza lo stesso materiale chiave nel file `aes256-forImport.key`. Quando il comando viene completato, la chiave 21 viene esportata sul file `aes192_h21.key`.

```
        Command:  exSymKey -k 21 -w 262167 -out aes192_H21.key -wk aes256-forImport.key

        Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

Wrapped Symmetric Key written to file "aes192_H21.key"
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-k

Specifica l'handle della chiave da esportare. Questo parametro è obbligatorio. Specifica l'handle della chiave simmetrica posseduta. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Per verificare che una chiave possa essere esportata, utilizza il comando getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_EXTRACTABLE`, che è rappresentato dalla costante `354`. Inoltre, puoi esportare solo le chiavi di tua proprietà. Per trovare il proprietario di una chiave, usa il getKeyInfocomando.

Obbligatorio: sì

-w

Specifica l'handle di una chiave di wrapping. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Una chiave di wrapping è una chiave nell'HSM che viene utilizzata per crittografare (eseguire il wrapping) e quindi decodificare (annullare il wrapping) della chiave da esportare. Solo le chiavi AES possono essere utilizzate come chiavi di wrapping.

Puoi usare qualsiasi chiave AES (di qualsiasi dimensione) come chiave di wrapping. Poiché la chiave di wrapping effettua e quindi annulla immediatamente il wrapping della chiave di destinazione, puoi utilizzare una chiave AES valida solo per la sessione come chiave di wrapping. Per determinare se una chiave può essere usata come chiave di wrapping, utilizza getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_WRAP`, che è rappresentato dalla costante `262`. Per creare una chiave di wrapping, utilizza genSymKey per creare una chiave AES (tipo 31).

Se utilizzi il parametro `-wk` per specificare una chiave di unwrapping esterna, la chiave di wrapping `-w` viene utilizzata per eseguire il wrapping della chiave durante l'esportazione, ma non per annullarlo.

> **Note**
>
> La chiave 4 rappresenta una chiave interna non supportata. Ti consigliamo di utilizzare una chiave AES che crei e gestisci come chiave di wrapping.

Obbligatorio: sì

-output

Specifica il percorso e il nome del file di output. Quando il comando viene completato, questo file contiene la chiave esportata in testo non crittografato. Se il file già esiste, il comando lo sovrascrive senza preavviso.

Obbligatorio: sì

-m

Specifica il meccanismo di wrapping. L'unico valore valido è `4`, che rappresenta il meccanismo `NIST_AES_WRAP`.

Obbligatorio: no

Impostazione predefinita: 4

-wk

Utilizza la chiave AES nel file specificato per annullare il wrapping della chiave esportata. Inserire il percorso e il nome di un file che contiene una chiave AES non crittografata.

Quando includi questo parametro. `exSymKey` utilizza la chiave nell'HSM specificata dal parametro `-w` per eseguire il wrapping della chiave esportata e utilizza la chiave nel file `-wk` per annullarne il wrapping. I valori di parametro `-w` e `-wk` devono determinare la stessa chiave non crittografata.

Obbligatorio: no

Impostazione predefinita: utilizzo della chiave di wrapping sull'HSM per annullare il wrapping.

Argomenti correlati

- genSymKey
- imSymKey
- wrapKey

Estrai una AWS CloudHSM chiave usando KMU

Utilizzate il extractMaskedObject comando in AWS CloudHSM key_mgmt_util per estrarre una chiave da un modulo di sicurezza hardware (HSM) e salvarla in un file come oggetto mascherato. Gli oggetti nascosti sono oggetti clonati che possono essere utilizzati solo dopo averli inseriti nuovamente nel cluster originale utilizzando il comando insertMaskedObject. È possibile inserire solo un oggetto mascherato nello stesso cluster da cui è stato generato, o un clone dello stesso cluster. Questo include qualsiasi versione clonata del cluster generata dalla copia di un backup tra le regioni e utilizzando tale backup per creare un nuovo cluster.

Gli oggetti mascherati sono un modo efficiente per scaricare e sincronizzare le chiavi, incluse cui le chiavi nonestraibili (ovvero chiavi che hanno un valore `OBJ_ATTR_EXTRACTABLE` di `0`). In questo modo, le chiavi possono essere sincronizzate in modo sicuro tra cluster correlati in diverse regioni senza la necessità di aggiornare il file di configurazione. AWS CloudHSM

> **Important**
>
> Dopo l'inserimento, gli oggetti mascherati vengono decodificati e viene loro affidato un handle diverso dall'handle della chiave originale. Un oggetto mascherato include tutti i metadati

associati alla chiave originale, tra cui attributi, proprietà e informazioni di condivisione, nonché le impostazioni del quorum. Se è necessario sincronizzare le chiavi tra i cluster in un'applicazione, utilizzare invece syncKey in the cloudhsm_mgmt_util.

Prima di eseguire qualsiasi comando è necessario avviare key_mgmt_util e accedere al modulo HSM. Il comando extractMaskedObject può essere utilizzato dal CU che possiede la chiave o qualsiasi CO.

Sintassi

```
extractMaskedObject -h

extractMaskedObject -o <object-handle>
                    -out <object-file>
```

Esempi

Questo esempio illustra come utilizzare extractMaskedObject per estrarre una chiave da un HSM come oggetto mascherato.

Example: Estrazione di un oggetto mascherato

Questo comando consente di estrarre un oggetto mascherato da un HSM di una chiave con handle `524295` e salvarlo come un file chiamato `maskedObj`. Se il comando ha esito positivo, extractMaskedObject restituisce un messaggio di operazione riuscita.

```
Command: extractMaskedObject -o 524295 -out maskedObj

Object was masked and written to file "maskedObj"

        Cfm3ExtractMaskedObject returned: 0x00 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-o`

Specifica l'handle della chiave da estrarre come oggetto mascherato.

Obbligatorio: sì

`-out`

Consente di specificare il nome del file in cui l'oggetto mascherato verrà salvato.

Campo obbligatorio: sì

Argomenti correlati

- insertMaskedObject
- syncKey
- Copiare un backup tra regioni
- Creazione di un AWS CloudHSM cluster da un backup precedente

Cerca AWS CloudHSM le chiavi per attributi usando KMU

Utilizzate il findKey comando in AWS CloudHSM key_mgmt_util per cercare le chiavi in base ai valori degli attributi chiave. Quando un chiave soddisfa tutti i criteri impostati, findKey restituisce l'handle della chiave. Senza parametri, findKey restituisce gli handle della chiave di tutte le chiavi utilizzabili nell'HSM. Per trovare i valori degli attributi di una determinata chiave, utilizzare getAttribute.

Come tutti i comandi key_mgmt_util, findKey è specifico per l'utente. Restituisce solo le chiavi che l'utente corrente può utilizzare nelle operazioni di crittografia. Ciò include le chiavi che l'utente corrente possiede e le chiavi che sono state condivise con l'utente corrente.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
findKey -h

findKey [-c <key class>]
        [-t <key type>]
        [-l <key label>]
        [-id <key ID>]
```

```
        [-sess (0 | 1)]
        [-u <user-ids>]
        [-m <modulus>]
        [-kcv <key_check_value>]
```

Esempi

Questi esempi mostrano come utilizzare per trovare e identificare le chiavi findKey nel tuo. HSMs

Example: trovare tutte le chiavi

Questo comando trova tutte le chiavi per l'utente corrente nell'HSM. L'output include le chiavi che l'utente possiede e condivide e tutte le chiavi pubbliche in HSMs.

Per ottenere gli attributi di una chiave con un determinato handle della chiave, utilizzare getAttribute. Per determinare se l'utente corrente possiede o condivide una particolare chiave, usa getKeyInfoo findAllKeysin cloudhsm_mgmt_util.

```
Command: findKey

Total number of keys present 13

 number of keys matched from start index 0::12
6, 7, 524296, 9, 262154, 262155, 262156, 262157, 262158, 262159, 262160, 262161, 262162

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: trovare le chiavi per tipo, utente e sessione

Questo comando trova le chiavi AES persistenti che l'utente corrente e l'utente 3 possono utilizzare (l'utente 3 potrebbe essere in grado di utilizzare altre chiavi che l'utente corrente non può visualizzare).

```
Command: findKey -t 31 -sess 0 -u 3
```

Example: trovare chiavi per classe ed etichetta

Questo comando trova tutte le chiavi pubbliche per l'utente corrente con l'etichetta `2018-sept`.

```
Command: findKey -c 2 -l 2018-sept
```

Example: trovare le chiavi RSA per modulo

Questo comando trova le chiavi RSA (tipo 0) per l'utente corrente, create utilizzando il modulo nel file `m4.txt`.

```
Command: findKey -t 0 -m m4.txt
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-t

Trova le chiavi del tipo specificato. Inserire la costante che rappresenta la classe della chiave. Ad esempio, per trovare le chiavi 3DES, digitare `-t 21`.

Valori validi:

- 0: RSA
- 1: DSA
- 3: EC
- 16: GENERIC_SECRET
- 18: RC4
- 21: Triple DES (3DES)
- 31: AES

Obbligatorio: no

-c

Trova le chiavi nella classe specificata. Inserire la costante che rappresenta la classe della chiave. Ad esempio, per trovare le chiavi pubbliche, digitare `-c 2`.

Valori validi per ogni tipo di chiave:

- 2: pubblica. Questa classe contiene le chiavi pubbliche delle coppie di chiavi pubbliche-private.
- 3: privata. Questa classe contiene le chiavi private delle coppie di chiavi pubbliche-private.
- 4: segreta. Questa classe contiene tutte le chiavi simmetriche.

Obbligatorio: no

-l

Trova le chiavi con l'etichetta specificata. Digitare l'etichetta esatta. Non è possibile utilizzare caratteri jolly o espressioni regolari nel valore `--l`.

Obbligatorio: no

-id

Trova la chiavi con l'ID specificato. Digitare la stringa ID esatta. Non è possibile utilizzare caratteri jolly o espressioni regolari nel valore `-id`.

Obbligatorio: no

-sessione

Trova le chiavi per stato della sessione. Per trovare le chiavi che sono valide solo nella sessione corrente, digitare `1`. Per trovare le chiavi persistenti, digitare `0`.

Obbligatorio: no

-u

Trova le chiavi che gli utenti specificati e l'utente corrente condividono. Digitare un elenco separato da virgole di utenti HSM IDs, ad esempio o. `-u 3 -u 4,7` Per trovare il numero IDs di utenti su un HSM, usa ListUsers.

Quando si specifica un ID utente, findKey restituisce le chiavi per quell'utente. Quando si specificano più utenti IDs, findKey restituisce le chiavi che tutti gli utenti specificati possono utilizzare.

Poiché findKey restituisce solo le chiavi che l'utente corrente può utilizzare, i risultati `-u` sono sempre identici alle chiavi dell'utente corrente o un sottoinsieme di queste. Per ottenere tutte le chiavi possedute o condivise con qualsiasi utente, i funzionari di crittografia (COs) possono utilizzare findAllKeysin cloudhsm_mgmt_util.

Obbligatorio: no

-m

Trova le chiavi create utilizzando il modulo RSA nel file specificato. Digitare il percorso del file che archivia il modulo.

-m specifica il file binario contenente il modulo RSA da associare (opzionale).

Obbligatorio: no

-kcv

Trova le chiavi con il valore di controllo della chiave specificato.

Il valore di controllo chiave (KCV) è un hash o checksum a 3 byte di una chiave che viene generato quando l'HSM importa o genera una chiave. Puoi anche calcolare un KCV al di fuori dell'HSM, ad esempio dopo aver esportato una chiave. È quindi possibile confrontare i valori KCV per confermare l'identità e l'integrità della chiave. Per ottenere il KCV di una chiave, usa getAttribute.

AWS CloudHSM utilizza il seguente metodo standard per generare un valore di controllo della chiave:

- Chiavi simmetriche: primi 3 byte del risultato della crittografia a blocchi zero con la chiave.
- Coppie di chiavi asimmetriche: primi 3 byte dell'hash SHA-1 della chiave pubblica.
- Chiavi HMAC: KCV per le chiavi HMAC attualmente non supportato.

Obbligatorio: no

Output

L'output findKey elenca il numero totale di chiavi corrispondenti e i relativi handle della chiave.

```
        Command:  findKey
Total number of keys present 10

 number of keys matched from start index 0::9
6, 7, 8, 9, 10, 11, 262156, 262157, 262158, 262159

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Argomenti correlati

- findSingleKey
- getKeyInfo
- OttieniAttributo
- findAllKeysin cloudhsm_mgmt_util
- Riferimento per l'attributo della chiave

Verifica una AWS CloudHSM chiave usando KMU

Utilizzate il findSingleKey comando nello strumento AWS CloudHSM key_mgmt_util per verificare che esista una chiave su tutti i moduli di sicurezza hardware (HSM) del cluster. AWS CloudHSM

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
findSingleKey -h

findSingleKey -k <key-handle>
```

Esempio

Example

Questo comando verifica che la chiave 252136 esista su tutti e tre i componenti del cluster. HSMs

```
Command: findSingleKey -k 252136
Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS

        Cluster Error Status
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-k

Specifica l'handle di una chiave nel modulo HSM. Questo parametro è obbligatorio.

Per trovare gli handle della chiave, utilizza il comando findKey.

Campo obbligatorio: sì

Argomenti correlati

- findKey
- getKeyInfo
- getAttribute

Genera una coppia di key pair AWS CloudHSM DSA usando KMU

Utilizzate il genDSAKeyPair comando nello strumento AWS CloudHSM key_mgmt_util per generare una coppia di chiavi DSA (Digital Signing Algorithm) nei moduli di sicurezza hardware (HSM). Devi specificare la lunghezza del modulo; il comando genera il valore del modulo. Puoi anche assegnare un ID, condividere la chiave con altri utenti HSM, creare chiavi non estraibili e chiavi che scadono al termine della sessione. Quando il comando viene eseguito correttamente, restituisce l'handle della chiave che l'HSM assegna alle chiavi pubbliche e private. Puoi utilizzare gli handle per identificare le chiavi per altri comandi.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

> **Tip**
>
> Per trovare gli attributi di una chiave che hai creato, ad esempio tipo, lunghezza, etichetta e ID, usa getAttribute. Per trovare le chiavi per un determinato utente, usa. getKeyInfo Per trovare le chiavi in base ai valori degli attributi, usa findKey.

Sintassi

```
genDSAKeyPair -h
```

```
genDSAKeyPair -m <modulus length>
              -l <label>
              [-id <key ID>]
              [-min_srv <minimum number of servers>]
              [-m_value <0..8>]
              [-nex]
              [-sess]
              [-timeout <number of seconds> ]
              [-u <user-ids>]
              [-attest]
```

Esempi

Questi esempi mostrano come utilizzare genDSAKeyPair per creare una coppia di chiavi DSA.

Example: creazione di una coppia di chiavi DSA

Questo comando crea una coppia di chiavi DSA con un'etichetta DSA. L'output indica che l'handle della chiave pubblica è 19 e l'handle della chiave privata è 21.

```
Command: genDSAKeyPair -m 2048 -l DSA

        Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 19    private key handle: 21

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: creazione di una coppia di chiavi DSA solo per la sessione

Questo comando crea una coppia di chiavi DSA valida solo nella sessione corrente. Il comando assegna un ID univoco di DSA_temp_pair oltre all'etichetta richiesta (non univoca). È possibile creare una coppia di chiavi come questa per firmare e verificare un token solo per la sessione. L'output indica che l'handle della chiave pubblica è 12 e l'handle della chiave privata è 14.

```
Command: genDSAKeyPair -m 2048 -l DSA-temp -id DSA_temp_pair -sess

        Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 12    private key handle: 14
```

```
        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Per confermare che la coppia di chiavi esiste solo nella sessione, utilizza il parametro `-sess` di findKey con un valore di 1 (vero).

```
  Command: findKey -sess 1

  Total number of keys present 2

 number of keys matched from start index 0::1
12, 14

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: creazione di una coppia di chiavi DSA non estraibili e condivise

Questo comando crea una coppia di chiavi DSA. La chiave privata è condivisa con altri tre utenti e non può essere esportata dall'HSM. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente e possono sempre essere estratte.

```
        Command:  genDSAKeyPair -m 2048 -l DSA -id DSA_shared_pair -nex -u 3,5,6

        Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 11    private key handle: 19

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: creazione di una coppia di chiavi controllate dal quorum

Questo comando crea una coppia di chiavi DSA con l'etichetta `DSA-mV2`. Il comando utilizza il parametro `-u` per condividere la chiave privata con gli utenti 4 e 6. Utilizzare il parametro `-m_value` per richiedere un quorum di almeno due approvazioni per le operazioni di crittografia che utilizzano la chiave privata: Viene inoltre utilizzato il parametro `-attest` per verificare l'integrità del firmware in cui la coppia di chiavi è generata.

L'output indica che il comando genera una chiave pubblica con handle 12 e una chiave privata con handle 17 e che il controllo di attestazione sul firmware del cluster ha avuto esito positivo.

```
Command:  genDSAKeyPair -m 2048 -l DSA-mV2 -m_value 2 -u 4,6 -attest

Cfm3GenerateKeyPair: returned: 0x00 : HSM Return: SUCCESS

Cfm3GenerateKeyPair:    public key handle: 12    private key handle: 17

Attestation Check : [PASS]

Cluster Error Status
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Questo comando utilizza getKeyInfola chiave privata (key handle17). L'output conferma che la chiave è di proprietà dell'utente corrente (utente 3) e che è condivisa con gli utenti 4 e 6 (e non con altri). L'output mostra anche che l'autenticazione del quorum è abilitata e la dimensione del quorum è due.

```
Command:  getKeyInfo -k 17

Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

Owned by user 3

also, shared to following 2 user(s):

         4
         6
 2 Users need to approve to use/manage this key
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-m

Specifica la lunghezza del modulo in bit. L'unico valore valido è `2048`.

Obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la coppia di chiavi. Digita una stringa. La stessa etichetta si applica a entrambe le chiavi della coppia. La dimensione massima per `label` è di 127 caratteri.

Puoi usare qualsiasi frase che ti aiuti a identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Obbligatorio: sì

-id

Specifica un identificatore definito dall'utente per la coppia di chiavi. Digita una stringa univoca nel cluster. L'impostazione predefinita è una stringa vuota. L'ID specificato si applica a entrambe le chiavi della coppia.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del `-timeout` parametro. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave privata della coppia. Digita un valore da `0` a `8`.

Questo parametro stabilisce un requisito di autenticazione del quorum per la chiave privata. Il valore predefinito `0` disabilita la funzionalità di autenticazione del quorum per la chiave. Quando l'autenticazione del quorum è abilitata, il numero specificato di utenti deve firmare un token per approvare le operazioni crittografiche che utilizzano la chiave privata e le operazioni che condividono o annullano la condivisione della chiave privata.

Per trovare il codice `m_value` di una chiave, usa getKeyInfo.

Questo parametro è valido soltanto quando il parametro `-u` nel comando condivide la coppia di chiavi con un numero sufficiente di utenti per soddisfare il requisito `m_value`.

Impostazione predefinita: 0

Obbligatorio: no

-nex

Rende la chiave privata non estraibile. La chiave privata generata non può essere esportata dall'HSM. Le chiavi pubbliche sono sempre estraibili.

Impostazione predefinita: sia la chiave pubblica che quella privata nella coppia di chiavi sono estraibili.

Obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende che una chiave venga sincronizzata con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CUs) il permesso di utilizzare la chiave privata nelle operazioni crittografiche. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digitate un elenco separato da virgole di utenti HSM, ad esempio -. IDs `u 5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'utente HSM IDs di CUs sull'HSM, usa ListUsers. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave privata.

Obbligatorio: no

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

Argomenti correlati

- RSAKeygen Pair
- genSymKey
- ECCKeycoppia di generi

Genera una coppia di chiavi AWS CloudHSM ECC usando KMU

Utilizzate il `genECCKeyPair` comando nello strumento AWS CloudHSM key_mgmt_util per generare una coppia di chiavi Elliptic Curve Cryptography (ECC) nei vostri moduli di sicurezza hardware (HSM). Quando si esegue il comando `genECCKeyPair`, è necessario specificare l'identificatore della curva ellittica e un'etichetta per la coppia di chiavi. Inoltre, è possibile condividere la chiave privata con altri utenti CU, creare chiavi non estraibili, chiavi controllate da quorum e chiavi che scadono al termine della sessione. Quando il comando viene completato con successo, restituisce gli handle che l'HSM assegna alle chiavi ECC pubbliche e private. Puoi utilizzare gli handle per identificare le chiavi per altri comandi.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

> **Tip**
>
> Per trovare gli attributi di una chiave che hai creato, ad esempio tipo, lunghezza, etichetta e ID, usa getAttribute. Per trovare le chiavi per un determinato utente, usa. getKeyInfo Per trovare le chiavi in base ai valori degli attributi, usa findKey.

Sintassi

```
genECCKeyPair -h

genECCKeyPair -i <EC curve id>
              -l <label>
              [-id <key ID>]
              [-min_srv <minimum number of servers>]
              [-m_value <0..8>]
              [-nex]
              [-sess]
              [-timeout <number of seconds> ]
              [-u <user-ids>]
              [-attest]
```

Esempi

Gli esempi seguenti mostrano come utilizzare genECCKeyPair per creare coppie di chiavi ECC in HSMs.

Example: creare ed esaminare una coppia di chiavi ECC

Questo comando usa una curva ellittica NID_secp384r1 e un'etichetta `ecc14` per creare una coppia di chiavi ECC. L'output indica che l'handle della chiave privata è 262177 e l'handle della chiave pubblica è 262179. L'etichetta si applica a entrambe le chiavi, sia pubblica che privata.

```
Command: genECCKeyPair -i 14 -l ecc14

        Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 262179    private key handle: 262177

        Cluster Error Status
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Dopo aver generato la chiave, è possibile esaminarne gli attributi. Utilizza getAttribute per scrivere tutti gli attributi (rappresentati dalla costante 512) della nuova chiave privata ECC sul file `attr_262177`.

```
Command: getAttribute -o 262177 -a 512 -out attr_262177
got all attributes of size 529 attr cnt 19
Attributes dumped into attr_262177

        Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

Quindi utilizza il comando `cat` per visualizzare il contenuto del file degli attributi `attr_262177`. L'output indica che la chiave è una chiave privata basata su una curva ellittica che può essere utilizzata per firmare, ma non per la crittografia, la decodifica, il wrapping, l'annullamento del wrapping o la verifica. La chiave è persistente ed esportabile.

```
$  cat attr_262177

OBJ_ATTR_CLASS
0x03
OBJ_ATTR_KEY_TYPE
0x03
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
```

```
0x01
OBJ_ATTR_ENCRYPT
0x00
OBJ_ATTR_DECRYPT
0x00
OBJ_ATTR_WRAP
0x00
OBJ_ATTR_UNWRAP
0x00
OBJ_ATTR_SIGN
0x01
OBJ_ATTR_VERIFY
0x00
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x01
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
ecc2
OBJ_ATTR_ID

OBJ_ATTR_VALUE_LEN
0x0000008a
OBJ_ATTR_KCV
0xbbb32a
OBJ_ATTR_MODULUS
044a0f9d01d10f7437d9fa20995f0cc742552e5ba16d3d7e9a65a33e20ad3e569e68eb62477a9960a87911e6121d112
OBJ_ATTR_MODULUS_BITS
0x0000019f
```

Example Utilizzo di una curva EEC non valida

Questo comando tenta di creare una coppia di chiavi ECC utilizzando una curva NID_X9_62_prime192v1. Poiché questa curva ellittica non è valida per la modalità FIPS HSMs, il comando ha esito negativo. Il messaggio segnala che un server del cluster non è disponibile, ma in genere ciò non indica un problema relativo al cluster. HSMs

```
Command:  genECCKeyPair -i 1 -l ecc1

        Cfm3GenerateKeyPair returned: 0xb3 : HSM Error: This operation violates the
 current configured/FIPS policies
```

```
        Cluster Error Status
        Node id 0 and err state 0x30000085 : HSM CLUSTER ERROR: Server in cluster is
 unavailable
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-i

Specifica l'identificatore per la curva ellittica. Inserisci un identificatore.

Valori validi:

- 2: NID_X9_62_prime256v1
- 14: NID_secp384r1
- 16: NID_secp256k1

Obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la coppia di chiavi. Digita una stringa. La stessa etichetta si applica a entrambe le chiavi della coppia. La dimensione massima per `label` è di 127 caratteri.

Puoi usare qualsiasi frase che ti aiuti a identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Obbligatorio: sì

-id

Specifica un identificatore definito dall'utente per la coppia di chiavi. Digita una stringa univoca nel cluster. L'impostazione predefinita è una stringa vuota. L'ID specificato si applica a entrambe le chiavi della coppia.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del `-timeout` parametro. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave privata della coppia. Digita un valore da `0` a `8`.

Questo parametro stabilisce un requisito di autenticazione del quorum per la chiave privata. Il valore predefinito `0` disabilita la funzionalità di autenticazione del quorum per la chiave. Quando l'autenticazione del quorum è abilitata, il numero specificato di utenti deve firmare un token per approvare le operazioni crittografiche che utilizzano la chiave privata e le operazioni che condividono o annullano la condivisione della chiave privata.

Per trovare il codice `m_value` di una chiave, usa getKeyInfo.

Questo parametro è valido soltanto quando il parametro `-u` nel comando condivide la coppia di chiavi con un numero sufficiente di utenti per soddisfare il requisito `m_value`.

Impostazione predefinita: 0

Obbligatorio: no

-nex

Rende la chiave privata non estraibile. La chiave privata generata non può essere esportata dall'HSM. Le chiavi pubbliche sono sempre estraibili.

Impostazione predefinita: sia la chiave pubblica che quella privata nella coppia di chiavi sono estraibili.

Obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende che una chiave venga sincronizzata con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CUs) il permesso di utilizzare la chiave privata nelle operazioni crittografiche. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digitate un elenco separato da virgole di utenti HSM, ad esempio -. IDs u `5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'utente HSM IDs di CUs sull'HSM, usa ListUsers. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave privata.

Obbligatorio: no

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

Argomenti correlati

- genSymKey
- RSAKeygen Pair
- DSAKeycoppia di generi

Genera una coppia di key pair AWS CloudHSM RSA usando KMU

Utilizzate il genRSAKeyPair comando nello strumento AWS CloudHSM key_mgmt_util per generare una coppia di chiavi asimmetrica RSA. Occorre specificare il tipo di chiave, la lunghezza del modulo e un esponente pubblico. Il comando genera un modulo della lunghezza specificata e crea la coppia di chiavi. È possibile assegnare un ID, condividere la chiave con altri utenti HSM, creare chiavi non estraibili e chiavi che scadono al termine della sessione. Quando il comando viene completato con successo, restituisce un handle che l'HSM assegna alla chiave. È possibile utilizzare l'handle per identificare la chiave per altri comandi.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

> **Tip**
>
> Per trovare gli attributi di una chiave che hai creato, ad esempio tipo, lunghezza, etichetta e ID, usa getAttribute. Per trovare le chiavi per un determinato utente, usa. getKeyInfo Per trovare le chiavi in base ai valori degli attributi, usa findKey.

Sintassi

```
genRSAKeyPair -h
```

```
genRSAKeyPair -m <modulus length>
              -e <public exponent>
              -l <label>
              [-id <key ID>]
              [-min_srv <minimum number of servers>]
              [-m_value <0..8>]
              [-nex]
              [-sess]
              [-timeout <number of seconds> ]
              [-u <user-ids>]
              [-attest]
```

Esempi

Questi esempi mostrano come utilizzare per genRSAKeyPair creare coppie di chiavi asimmetriche in. HSMs

Example: crea ed esamina una coppia di chiavi RSA

Questo comando crea una coppia di chiavi RSA con un modulo a 2048 bit e un esponente di 65537. L'output indica che l'handle della chiave pubblica è `2100177` e l'handle della chiave privata è `2100426`.

```
Command: genRSAKeyPair -m 2048 -e 65537 -l rsa_test

Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 2100177    private key handle:
 2100426

        Cluster Status:
        Node id 0 status: 0x00000000 : HSM Return: SUCCESS
        Node id 1 status: 0x00000000 : HSM Return: SUCCESS
```

Il comando successivo utilizza getAttribute per ottenere gli attributi della chiave pubblica appena creata. Scrive l'output nel file `attr_2100177`. È seguito da un comando cat che ottiene il contenuto del file degli attributi. Per informazioni sull'interpretazione degli attributi delle chiavi, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

I risultanti valori esadecimali confermano che è una chiave pubblica (`OBJ_ATTR_CLASS 0x02`) con un tipo di RSA (`OBJ_ATTR_KEY_TYPE 0x00`). È possibile utilizzare questa chiave pubblica per la

crittografia (OBJ_ATTR_ENCRYPT 0x01), ma non per la decodifica (OBJ_ATTR_DECRYPT 0x00). I risultati includono anche la lunghezza della chiave (512, 0x200), il modulo, la lunghezza del modulo (2048, 0x800) e l'esponente pubblico (65537, 0x10001).

```
Command:  getAttribute -o 2100177 -a 512 -out attr_2100177

Attribute size: 801, count: 26
Written to: attr_2100177 file

        Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS

$  cat attr_2100177
OBJ_ATTR_CLASS
0x02
OBJ_ATTR_KEY_TYPE
0x00
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x00
OBJ_ATTR_WRAP
0x01
OBJ_ATTR_UNWRAP
0x00
OBJ_ATTR_SIGN
0x00
OBJ_ATTR_VERIFY
0x01
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x00
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
rsa_test
OBJ_ATTR_ID

OBJ_ATTR_VALUE_LEN
```

```
0x00000200
OBJ_ATTR_KCV
0xc51c18
OBJ_ATTR_MODULUS
0xbb9301cc362c1d9724eb93da8adab0364296bde7124a241087d9436b9be57e4f7780040df03c2c
1c0fe6e3b61aa83c205280119452868f66541bbbffacbbe787b8284fc81deaeef2b8ec0ba25a077d
6983c77a1de7b17cbe8e15b203868704c6452c2810344a7f2736012424cf0703cf15a37183a1d2d0
97240829f8f90b063dd3a41171402b162578d581980976653935431da0c1260bfe756d85dca63857
d9f27a541676cb9c7def0ef6a2a89c9b9304bcac16fdf8183c0a555421f9ad5dfeb534cf26b65873
970cdf1a07484f1c128b53e10209cc6f7ac308669112968c81a5de408e7f644fe58b1a9ae1286fec
b3e4203294a96fae06f8f0db7982cb5d7f
OBJ_ATTR_MODULUS_BITS
0x00000800
OBJ_ATTR_PUBLIC_EXPONENT
0x010001
OBJ_ATTR_TRUSTED
0x00
OBJ_ATTR_WRAP_WITH_TRUSTED
0x00
OBJ_ATTR_DESTROYABLE
0x01
OBJ_ATTR_DERIVE
0x00
OBJ_ATTR_ALWAYS_SENSITIVE
0x00
OBJ_ATTR_NEVER_EXTRACTABLE
0x00
```

Example: genera una coppia di chiavi RSA condivise

Questo comando genera una coppia di chiavi RSA e condivide la chiave privata con l'utente 4, un altro CU sull'HSM. Il comando utilizza il parametro `m_value` per richiedere almeno due approvazioni prima che la chiave privata nella coppia possa essere utilizzata in un'operazione di crittografia. Quando si utilizza il parametro `m_value`, è necessario utilizzare anche `-u` nel comando e `m_value` non può superare il numero totale di utenti (numero di valori in `-u` + proprietario).

```
Command:  genRSAKeyPair -m 2048 -e 65537 -l rsa_mofn -id rsa_mv2 -u 4 -m_value 2

       Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

       Cfm3GenerateKeyPair:    public key handle: 27    private key handle: 28

       Cluster Error Status
```

```
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-m

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Obbligatorio: sì

-e

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la coppia di chiavi. Digita una stringa. La stessa etichetta si applica a entrambe le chiavi della coppia. La dimensione massima per `label` è di 127 caratteri.

Puoi usare qualsiasi frase che ti aiuti a identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Obbligatorio: sì

-id

Specifica un identificatore definito dall'utente per la coppia di chiavi. Digita una stringa univoca nel cluster. L'impostazione predefinita è una stringa vuota. L'ID specificato si applica a entrambe le chiavi della coppia.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del parametro. `-timeout` Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave privata della coppia. Digita un valore da `0` a `8`.

Questo parametro stabilisce un requisito di autenticazione del quorum per la chiave privata. Il valore predefinito `0` disabilita la funzionalità di autenticazione del quorum per la chiave. Quando l'autenticazione del quorum è abilitata, il numero specificato di utenti deve firmare un token per approvare le operazioni crittografiche che utilizzano la chiave privata e le operazioni che condividono o annullano la condivisione della chiave privata.

Per trovare il codice `m_value` di una chiave, usa getKeyInfo.

Questo parametro è valido soltanto quando il parametro `-u` nel comando condivide la coppia di chiavi con un numero sufficiente di utenti per soddisfare il requisito `m_value`.

Impostazione predefinita: 0

Obbligatorio: no

-nex

Rende la chiave privata non estraibile. La chiave privata generata non può essere esportata dall'HSM. Le chiavi pubbliche sono sempre estraibili.

Impostazione predefinita: sia la chiave pubblica che quella privata nella coppia di chiavi sono estraibili.

Obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende che una chiave venga sincronizzata con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CUs) il permesso di utilizzare la chiave privata nelle operazioni crittografiche. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digitate un elenco separato da virgole di utenti HSM, ad esempio -. IDs u `5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'utente HSM IDs di CUs sull'HSM, usa ListUsers. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave privata.

Obbligatorio: no

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

Argomenti correlati

- genSymKey
- DSAKeygen Pair
- ECCKeycoppia di generi

Genera una chiave AWS CloudHSM simmetrica usando KMU

Utilizzate il genSymKey comando nello strumento AWS CloudHSM key_mgmt_util per generare una chiave simmetrica nei vostri moduli di sicurezza hardware (HSM). È possibile specificare il tipo e le dimensioni della chiave, assegnare un ID e un'etichetta e condividere la chiave con altri utenti HSM. È anche possibile creare chiavi non estraibili e chiavi che scadono al termine della sessione. Quando il comando viene completato con successo, restituisce un handle che l'HSM assegna alla chiave. È possibile utilizzare l'handle per identificare la chiave per altri comandi.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
genSymKey -h

genSymKey -t <key-type>
          -s <key-size>
          -l <label>
          [-id <key-ID>]
          [-min_srv <minimum-number-of-servers>]
```

```
        [-m_value <0..8>]
        [-nex]
        [-sess]
        [-timeout <number-of-seconds> ]
        [-u <user-ids>]
        [-attest]
```

Esempi

Questi esempi mostrano come utilizzare per creare chiavi simmetriche nel tuo. genSymKey HSMs

> **Tip**
>
> Per utilizzare le chiavi create con questi esempi per le operazioni HMAC, è necessario impostare `OBJ_ATTR_SIGN` e `OBJ_ATTR_VERIFY` a `TRUE` dopo aver generato la chiave. Per impostare questi valori, utilizza setAttribute in CloudHSM Management Utility (CMU). Per ulteriori informazioni, vedi setAttribute.

Example Generazione di una chiave AES

Questo comando crea una chiave AES a 256 bit con un'etichetta `aes256`. L'output indica che l'handle della nuova chiave è 6.

```
Command: genSymKey -t 31 -s 32 -l aes256

        Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 6

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Creazione di una chiave di sessione

Questo comando crea una chiave AES non estraibile a 192 bit valida solo per la durata della sessione corrente. È possibile creare una chiave come questa per eseguire il wrapping (e subito dopo annullare il wrapping) di una chiave in fase di esportazione.

```
Command: genSymKey -t 31 -s 24 -l tmpAES -id wrap01 -nex -sess
```

Example: Risultato rapido

Questo comando crea una chiave generica a 512 byte con un'etichetta di `IT_test_key`. Il comando non attende che la chiave venga sincronizzata con tutte HSMs le chiavi del cluster. Restituisce invece un risultato non appena la chiave viene creata in qualsiasi HSM (`-min_srv 1`) o in 1 secondo (`-timeout 1`), qualunque sia la soluzione più rapida. Se la chiave non è sincronizzata al numero minimo specificato HSMs prima della scadenza del timeout, non viene generata. È possibile utilizzare un comando come questo in uno script che crea numerose chiavi, come il loop `for` nell'esempio seguente.

```
Command: genSymKey -t 16 -s 512 -l IT_test_key -min_srv 1 -timeout 1

$  for i in {1..30};
     do /opt/cloudhsm/bin/key_mgmt_util singlecmd loginHSM -u CU -s example_user -p
 example_pwd genSymKey -l aes -t 31 -s 32 -min_srv 1 -timeout 1;
 done;
```

Example: Creazione di una chiave generica con autorizzazione del quorum

Questo comando crea una chiave segreta generica a 2048 bit con l'etichetta `generic-mV2`.
Il comando utilizza il parametro `-u` per condividere la chiave con un altro utente di crittografia, l'utente 6. Usa il parametro `-m_value` per richiedere un quorum di almeno due approvazioni per le operazioni di crittografia che utilizzano la chiave. Il comando utilizza inoltre il parametro `-attest` per verificare l'integrità del firmware in cui la chiave viene generata.

L'output indica che il comando ha generato una chiave con handle 9 e che il controllo di attestazione sul firmware del cluster ha avuto esito positivo.

```
                Command:  genSymKey -t 16 -s 2048 -l generic-mV2 -m_value 2 -u 6 -
attest

        Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 9

        Attestation Check : [PASS]

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: Creazione e analisi di una chiave

Questo comando crea una chiave Triple DES con un'etichetta `3DES_shared` e un ID `IT-02`. La chiave può essere utilizzata dall'utente corrente e dagli utenti 4 e 5. Il comando ha esito negativo se l'ID non è univoco nel cluster o se l'utente corrente è l'utente 4 o 5.

L'output indica che la nuova chiave ha un handle 7.

```
Command: genSymKey -t 21 -s 24 -l 3DES_shared -id IT-02 -u 4,5

       Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 7

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Per verificare che la nuova chiave 3DES sia di proprietà dell'utente corrente e sia condivisa con gli utenti 4 e 5, utilizza getKeyInfo. Il comando usa l'handle assegnato alla nuova chiave (`Key Handle: 7`).

L'output conferma che la chiave è di proprietà dell'utente 3 ed è condivisa con gli utenti 4 e 5.

```
Command:  getKeyInfo -k 7

        Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

        Owned by user 3

        also, shared to following 2 user(s):

                4, 5
```

Per verificare le altre proprietà della chiave, utilizza getAttribute. Il primo comando usa `getAttribute` per ottenere tutti gli attributi (`-a 512`) dell'handle di chiave 7 (`-o 7`) e li scrive nel file `attr_7`. Il secondo comando usa `cat` per ottenere il contenuto del file `attr_7`.

Questo comando conferma che la chiave 7 è una chiave simmetrica (`OBJ_ATTR_CLASS 0x04`) 3DES (`OBJ_ATTR_KEY_TYPE 0x15`) a 192 bit (`OBJ_ATTR_VALUE_LEN 0x00000018` o 24 byte)

con un'etichetta `3DES_shared` (`OBJ_ATTR_LABEL 3DES_shared`) e un ID `IT_02` (`OBJ_ATTR_ID IT-02`). La chiave è persistente (`OBJ_ATTR_TOKEN 0x01`) ed estraibile (`OBJ_ATTR_EXTRACTABLE 0x01`) e può essere utilizzata per la crittografia, la decodifica e il wrapping.

> **Tip**
>
> Per trovare gli attributi di una chiave che hai creato, ad esempio tipo, lunghezza, etichetta e ID, usa getAttribute. Per trovare le chiavi per un determinato utente, usa. getKeyInfo Per trovare le chiavi in base ai valori degli attributi, usa findKey.

Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

```
Command:  getAttribute -o 7 -a 512 -out attr_7

got all attributes of size 444 attr cnt 17
Attributes dumped into attr_7 file

        Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS


$  cat attr_7

OBJ_ATTR_CLASS
0x04
OBJ_ATTR_KEY_TYPE
0x15
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x01
OBJ_ATTR_WRAP
0x00
OBJ_ATTR_UNWRAP
0x00
OBJ_ATTR_SIGN
0x00
```

```
OBJ_ATTR_VERIFY
0x00
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x01
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
3DES_shared
OBJ_ATTR_ID
IT-02
OBJ_ATTR_VALUE_LEN
0x00000018
OBJ_ATTR_KCV
0x59a46e
```

> **Tip**
>
> Per utilizzare le chiavi create con questi esempi per le operazioni HMAC, è necessario impostare `OBJ_ATTR_SIGN` e `OBJ_ATTR_VERIFY` a `TRUE` dopo aver generato la chiave. Per impostare questi valori, usa setAttribute in CMU. Per ulteriori informazioni, vedi setAttribute.

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-t

Specifica il tipo di chiave simmetrica. Inserisci la costante che rappresenta il tipo di chiave. Ad esempio, per creare una chiave AES, digita `-t 31`.

Valori validi:

- 16: GENERIC_SECRET. Una chiave segreta generica è una matrice di byte non conforme a un determinato standard, come i requisiti per una chiave AES.
- 18: RC4. RC4 le chiavi non sono valide in modalità FIPS HSMs

- 21: Triple DES (3DES). In conformità alle linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.
- 31: AES

Obbligatorio: sì

-s

Specifica le dimensioni della chiave in byte. Ad esempio, per creare una chiave a 192 bit, digita 24.

Valori validi per ogni tipo di chiave:

- AES: 16 (128 bit), 24 (192 bit), 32 (256 bit)
- 3DES: 24 (192 bit)
- Segreta generica: <3584 (28672 bit)

Obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la chiave. Digita una stringa.

Puoi usare qualsiasi frase che ti aiuti a identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Obbligatorio: sì

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

-id

Specifica un identificatore definito dall'utente per la chiave. Digita una stringa univoca nel cluster. L'impostazione predefinita è una stringa vuota.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del parametro. `-timeout` Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave importata. Digitare un valore da `0` a `8`.

Questo parametro stabilisce un requisito di autenticazione del quorum per la chiave. Il valore predefinito, `0`, disabilita la funzionalità di autenticazione del quorum per la chiave. Quando su una chiave è abilitata la funzionalità di autenticazione del quorum, restituisce anche il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave.

Per trovare il codice `m_value` di una chiave, usa getKeyInfo.

Questo parametro è valido soltanto quando il parametro `-u` nel comando condivide la chiave con un numero sufficiente di utenti per soddisfare il requisito `m_value`.

Impostazione Predefinita: 0

Obbligatorio: no

-nex

Rende la chiave non estraibile. La chiave generata non può essere esportata dall'HSM.

Impostazione predefinita: la chiave è estraibile.

Obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende che una chiave venga sincronizzata con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-u

Condivide la chiave con gli utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CUs) il permesso di utilizzare questa chiave nelle operazioni crittografiche.

Digitate un elenco separato da virgole di utenti HSM, ad esempio -. IDs u `5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'utente HSM IDs di CUs sull'HSM, usa ListUsers. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave.

Obbligatorio: no

Argomenti correlati

- exSymKey
- RSAKeygen Pair
- DSAKeycoppia di generi
- ECCKeycoppia di generi
- setAttribute

Ottieni un attributo AWS CloudHSM chiave usando KMU

Utilizzate il getAttribute comando in AWS CloudHSM key_mgmt_util per scrivere uno o tutti i valori degli attributi per una chiave di un file. AWS CloudHSM Se l'attributo specificato non esiste per il tipo di chiave, ad esempio il modulo di una chiave AES, getAttribute restituisce un errore.

Gli attributi chiave sono le proprietà di una chiave. Includono caratteristiche quali tipo di chiave, classe, etichetta e ID, nonché i valori che rappresentano le azioni eseguibili con la chiave, ad esempio crittografia, decodifica, wrapping, firma e verifica.

Puoi utilizzare il comando getAttribute solo sulle chiavi di tua proprietà e su quelle condivise con te. Puoi eseguire questo comando o il comando getAttribute in cloudhsm_mgmt_util, che ottiene un valore di attributo di una chiave da tutti i HSMs componenti di un cluster e lo scrive su stdout o su un file.

Per ottenere un elenco di attributi e delle costanti che li rappresentano, utilizza il comando listAttributes. Per modificare i valori degli attributi delle chiavi esistenti, utilizza setAttribute in key_mgmt_util e setAttribute in cloudhsm_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
getAttribute -h

getAttribute -o <key handle>
```

```
            -a <attribute constant>
            -out <file>
```

Esempi

Questi esempi mostrano come utilizzare per ottenere gli attributi delle chiavi nel tuo. getAttribute HSMs

Example: ottenere il tipo di chiave

Questo esempio mostra come ottenere il tipo di chiave, ad esempio per una chiave AES, 3DES o generica, ma anche una coppia di chiavi RSA o basata su curva ellittica.

Il primo comando esegue listAttributes, che ottiene gli attributi di una chiave e le costanti che li rappresentano. L'output indica che la costante per il tipo di chiave è 256. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

```
Command: listAttributes

Description
===========
The following are all of the possible attribute values for getAttributes.

      OBJ_ATTR_CLASS                = 0
      OBJ_ATTR_TOKEN                = 1
      OBJ_ATTR_PRIVATE              = 2
      OBJ_ATTR_LABEL                = 3
      OBJ_ATTR_KEY_TYPE             = 256
      OBJ_ATTR_ID                   = 258
      OBJ_ATTR_SENSITIVE            = 259
      OBJ_ATTR_ENCRYPT              = 260
      OBJ_ATTR_DECRYPT              = 261
      OBJ_ATTR_WRAP                 = 262
      OBJ_ATTR_UNWRAP               = 263
      OBJ_ATTR_SIGN                 = 264
      OBJ_ATTR_VERIFY               = 266
      OBJ_ATTR_LOCAL                = 355
      OBJ_ATTR_MODULUS              = 288
      OBJ_ATTR_MODULUS_BITS         = 289
      OBJ_ATTR_PUBLIC_EXPONENT      = 290
      OBJ_ATTR_VALUE_LEN            = 353
      OBJ_ATTR_EXTRACTABLE          = 354
```

```
      OBJ_ATTR_KCV                    = 371
```

Il secondo comando esegue getAttribute. Richiede il tipo di chiave (attributo 256) per l'handle di chiave 524296 e lo scrive nel file `attribute.txt`.

```
Command: getAttribute -o 524296 -a 256 -out attribute.txt
Attributes dumped into attribute.txt file
```

L'ultimo comando ottiene i contenuti del file della chiave. L'output indica che il tipo di chiave è `0x15` o 21, che è una chiave Triple DES (3DES). Per le definizioni dei valori della classe e del tipo, vedere il Riferimento agli attributi delle chiavi.

```
$  cat attribute.txt
OBJ_ATTR_KEY_TYPE
0x00000015
```

Example: ottieni tutti gli attributi di una chiave

Questo comando ottiene tutti gli attributi della chiave con handle 6 e li scrive nel file `attr_6`. Utilizza il valore di attributo 512, che rappresenta tutti gli attributi.

```
Command: getAttribute -o 6 -a 512 -out attr_6

got all attributes of size 444 attr cnt 17
Attributes dumped into attribute.txt file

        Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS>
```

Questo comando mostra i contenuti di un file degli attributi di esempio con i valori di tutti gli attributi. Tra i valori, indica che la chiave è di tipo AES a 256 bit con ID `test_01` ed etichetta `aes256`. La chiave è estraibile e persistente, ovvero non è una chiave solo di sessione. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

```
$  cat attribute.txt

OBJ_ATTR_CLASS
0x04
OBJ_ATTR_KEY_TYPE
```

```
0x15
OBJ_ATTR_TOKEN
0x01
OBJ_ATTR_PRIVATE
0x01
OBJ_ATTR_ENCRYPT
0x01
OBJ_ATTR_DECRYPT
0x01
OBJ_ATTR_WRAP
0x01
OBJ_ATTR_UNWRAP
0x01
OBJ_ATTR_SIGN
0x00
OBJ_ATTR_VERIFY
0x00
OBJ_ATTR_LOCAL
0x01
OBJ_ATTR_SENSITIVE
0x01
OBJ_ATTR_EXTRACTABLE
0x01
OBJ_ATTR_LABEL
aes256
OBJ_ATTR_ID
test_01
OBJ_ATTR_VALUE_LEN
0x00000020
OBJ_ATTR_KCV
0x1a4b31
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-o

Specifica l'handle della chiave di destinazione. È possibile specificare una sola chiave in ogni comando. Per individuare l'handle di una chiave, utilizza findKey.

La chiave specificata deve inoltre essere in tuo possesso o condivisa con te. Per trovare gli utenti di una chiave, usa getKeyInfo.

Obbligatorio: sì

-a

Identifica l'attributo. Inserisci una costante che rappresenti un attributo oppure immetti `512` per tutti gli attributi. Ad esempio, per ottenere il tipo di chiave, digita `256`, che è la costante per l'attributo `OBJ_ATTR_KEY_TYPE`.

Per elencare gli attributi e le relative costanti, utilizza listAttributes. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Obbligatorio: sì

-output

Scrive l'output nel file specificato. Immetti un percorso file. L'output non può essere scritto su `stdout`.

Se il file specificato esiste, getAttribute lo sovrascrive senza preavviso.

Campo obbligatorio: sì

Argomenti correlati

- getAttribute su cloudhsm_mgmt_util
- listAttributes
- setAttribute
- findKey
- Riferimento per l'attributo della chiave

Esporta una AWS CloudHSM chiave in un formato PEM falso usando KMU

Utilizzate il comando getCaviumPrivKey in AWS CloudHSM key_mgmt_util per esportare una chiave privata da un modulo di sicurezza hardware (HSM) in formato PEM falso. Il file PEM falso, che non contiene l'effettivo materiale della chiave privata ma fa invece riferimento alla chiave privata nell'HSM,

può quindi essere utilizzato per stabilire l'offload dal server Web a. SSL/TLS AWS CloudHSMPer ulteriori informazioni, consulta SSL/TLS Offload su Linux con Tomcat o SSL/TLS Offload su Linux con NGINX o Apache.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
getCaviumPrivKey -h

getCaviumPrivKey -k <private-key-handle>
                 -out <fake-PEM-file>
```

Esempi

Questo esempio illustra come utilizzare getCaviumPrivKey per esportare una chiave privata in un formato PEM falso.

Example: Esportare un file PEM falso

Questo comando crea ed esporta una versione PEM falsa di una chiave privata con handle 15 e la salva su un file chiamato `cavKey.pem`. Se il comando ha esito positivo, exportPrivateKey restituisce un messaggio di operazione riuscita.

```
Command: getCaviumPrivKey -k 15 -out cavKey.pem

Private Key Handle is written to cavKey.pem in fake PEM format

        getCaviumPrivKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

-k

Specifica l'handle della chiave privata da esportare in formato PEM falso.

Obbligatorio: sì

-out

Consente di specificare il nome del file in cui la chiave PEM falsa verrà scritta.

Campo obbligatorio: sì

Argomenti correlati

- importPrivateKey
- SSL/TLS Offload su Linux utilizzando Tomcat
- SSL/TLS Offload su Linux utilizzando NGINX o Apache

Ottieni certificati di partizione HSM usando KMU AWS CloudHSM

Usa il getCert comando contenuto in AWS CloudHSM key_mgmt_util per recuperare i certificati di partizione di un modulo di sicurezza hardware (HSM) e salvarli in un file. Quando esegui il comando, specifichi il tipo di certificato da recuperare. A tale scopo, utilizza uno dei corrispondenti numeri interi come descritto nella sezione Parametri che segue. Per ulteriori informazioni sul ruolo di ognuno di questi certificati, vedi la pagina relativa alla Verifica dell'identità del modulo HSM.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
getCert -h

getCert -f <file-name>
        -t <certificate-type>
```

Esempio

Questo esempio illustra come utilizzare getCert per recuperare un certificato root cliente del cluster e salvarlo come file.

Example: recuperare un certificato root cliente

Questo comando esporta un certificato root cliente (rappresentato da un numero intero 4) e lo salva in un file chiamato `userRoot.crt`. Se il comando ha esito positivo, getCert restituisce un messaggio di operazione riuscita.

```
Command: getCert -f userRoot.crt -s 4

Cfm3GetCert() returned 0 :HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-f`

Consente di specificare il nome del file in cui il certificato recuperato verrà salvato.

Obbligatorio: sì

-s

Valore intero che specifica il tipo di certificato da recuperare. Di seguito sono elencati i valori interi e i tipi di certificato corrispondenti:

- 1 – Certificato root del produttore
- 2 – Certificato hardware del produttore
- 4 – Certificato root del cliente
- 8 – Certificato del cluster (firmato dal certificato root del cliente)
- 16 – Certificato del cluster (concatenato al certificato root del cliente)

Campo obbligatorio: sì

Argomenti correlati

- Verifica dell'identità HSM

Ottieni gli utenti di una AWS CloudHSM chiave usando KMU

Utilizzate il getKeyInfo comando contenuto in AWS CloudHSM key_mgmt_util per restituire l'utente del modulo di sicurezza hardware (HSM) IDs degli utenti che possono utilizzare la chiave, inclusi il proprietario e gli utenti crittografici (CU) con cui la chiave è condivisa. Quando su una chiave è abilitata la funzionalità di autenticazione del quorum, getKeyInfo restituisce anche il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave. Puoi eseguire il comando getKeyInfo solo sulle chiavi di tua proprietà e su quelle condivise con te.

Quando esegui il comando getKeyInfo su chiavi pubbliche, getKeyInfo restituisce solo il proprietario della chiave, anche se tutti gli utenti HSM possono utilizzare la chiave pubblica. Per trovare l'utente HSM IDs degli utenti nel tuo HSMs, usa ListUsers. Per trovare le chiavi di un utente specifico, utilizza Trova chiave `-u`.

Le chiavi che hai creato sono di tua proprietà. Puoi condividere una chiave con altri utenti nel momento in cui la crei. Quindi, per condividere o interrompere la condivisione di una chiave esistente, utilizza shareKey in cloudhsm_mgmt_util.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
getKeyInfo -h

getKeyInfo -k <key-handle>
```

Esempi

Questi esempi mostrano come utilizzare getKeyInfo per ottenere informazioni sugli utenti di una chiave.

Example: ottenere gli utenti di una chiave simmetrica

Questo comando consente di ottenere gli utenti che possono utilizzare la chiave AES (simmetrica) con l'handle di chiave 9. L'output indica che l'utente 3 possiede la chiave e l'ha condivisa con l'utente 4.

```
Command:  getKeyInfo -k 9

       Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS
```

```
       Owned by user 3

       also, shared to following 1 user(s):

                4
```

Example: ottenere gli utenti di una coppia di chiavi asimmetriche

Questi comandi utilizzano getKeyInfo per ottenere gli utenti che possono utilizzare le chiavi in una coppia di chiavi RSA (asimmetriche). La chiave pubblica ha l'handle 21. La chiave privata presenta l'handle 20.

Quando esegui getKeyInfo sulla chiave privata (20), restituisce il proprietario della chiave (3) e gli utenti crittografici (CUs) 4 e 5, con cui la chiave è condivisa.

```
Command:  getKeyInfo -k 20

       Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

       Owned by user 3

       also, shared to following 2 user(s):

                4
                5
```

Quando esegui getKeyInfo sulla chiave pubblica (21), il comando restituisce solo il proprietario della chiave, l'utente (3).

```
Command:  getKeyInfo -k 21

       Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

       Owned by user 3
```

Per verificare se l'utente 4 può utilizzare la chiave pubblica (e tutte le chiavi pubbliche sul modulo HSM), utilizza il parametro -u di findKey.

L'output indica che nella coppia di chiavi l'utente 4 può utilizzare sia la chiave pubblica (21) sia quella privata (20). L'utente 4 può inoltre utilizzare tutte le altre chiavi pubbliche e qualsiasi chiave privata che abbia creato o che sia stata condivisa con lui.

```
Command:  findKey -u 4
Total number of keys present 8

 number of keys matched from start index 0::7
11, 12, 262159, 262161, 262162, 19, 20, 21

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Example: ottenere il valore di autenticazione del quorum (m_value) per una chiave

Questo esempio mostra come ottenere il valore `m_value` per una chiave, che corrisponde al numero di utenti nel quorum che deve approvare tutte le operazioni di crittografia che utilizzano la chiave.

Quando l'autenticazione del quorum è abilitata su una chiave, un quorum di utenti deve approvare tutte le operazioni di crittografia che utilizzano la chiave. Per abilitare l'autenticazione del quorum e impostarne le dimensioni, utilizza il parametro `-m_value` durante la creazione della chiave.

Questo comando utilizza gen RSAKey Pair per creare una coppia di chiavi RSA condivisa con l'utente 4. Si serve del parametro `m_value` per abilitare l'autenticazione del quorum sulla chiave privata nella coppia e per impostare il quorum a due utenti. Il numero di utenti deve essere sufficiente per fornire le approvazioni necessarie.

L'output mostra che il comando ha creato la chiave pubblica 27 e la chiave privata 28.

```
Command:  genRSAKeyPair -m 2048 -e 195193 -l rsa_mofn -id rsa_mv2 -u 4 -m_value 2

        Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

        Cfm3GenerateKeyPair:    public key handle: 27    private key handle: 28

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Questo comando utilizza getKeyInfo per ottenere informazioni sugli utenti della chiave privata. L'output indica che la chiave è di proprietà dell'utente 3 ed è condivisa con l'utente 4. Inoltre, mostra che un quorum di due utenti deve approvare ogni operazione di crittografia che utilizza la chiave.

```
Command:  getKeyInfo -k 28

        Cfm3GetKey returned: 0x00 : HSM Return: SUCCESS

        Owned by user 3

        also, shared to following 1 user(s):

                4
         2 Users need to approve to use/manage this key
```

Parameters

-h

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

-k

Specifica l'handle di una chiave nel modulo HSM. Specifica l'handle di una chiave di cui sei proprietario o che è condivisa con te. Questo parametro è obbligatorio.

Per trovare gli handle della chiave, utilizza il comando findKey.

Campo obbligatorio: sì

Argomenti correlati

- getKeyInfoin cloudhsm_mgmt_util
- ElencaUtenti
- findKey
- findAllKeysin cloudhsm_mgmt_util

Visualizza le informazioni di aiuto per AWS CloudHSM KMU

Utilizzate il comando in key_mgmt_util per visualizzare informazioni su tutti i help comandi key_mgmt_util disponibili. AWS CloudHSM

Prima di eseguire il comando help, devi avviare key_mgmt_util.

Sintassi

```
help
```

Esempio

Questo esempio mostra l'output del comando `help`.

Example

```
Command:  help

Help Commands Available:

Syntax: <command> -h


   Command               Description
   =======               ===========

   exit                   Exits this application
   help                   Displays this information

        Configuration and Admin Commands
   getHSMInfo             Gets the HSM Information
   getPartitionInfo       Gets the Partition Information
   listUsers              Lists all users of a partition
   loginStatus            Gets the Login Information
   loginHSM               Login to the HSM
   logoutHSM              Logout from the HSM

        M of N commands
   getToken               Initiate an MxN service and get Token
   delToken               delete Token(s)
   approveToken           Approves an MxN service
   listTokens             List all Tokens in the current partition

        Key Generation Commands

        Asymmetric Keys:
   genRSAKeyPair          Generates an RSA Key Pair
   genDSAKeyPair          Generates a DSA Key Pair
   genECCKeyPair          Generates an ECC Key Pair
```

```
     Symmetric Keys:
genPBEKey              Generates a PBE DES3 key
genSymKey              Generates a Symmetric keys

     Key Import/Export Commands
createPublicKey        Creates an RSA public key
importPubKey           Imports RSA/DSA/EC Public key
exportPubKey           Exports RSA/DSA/EC Public key
importPrivateKey       Imports RSA/DSA/EC private key
exportPrivateKey       Exports RSA/DSA/EC private key
imSymKey               Imports a Symmetric key
exSymKey               Exports a Symmetric key
wrapKey                Wraps a key from from HSM using the specified handle
unWrapKey              UnWraps a key into HSM using the specified handle

     Key Management Commands
deleteKey              Delete Key
setAttribute           Sets an attribute of an object
getKeyInfo             Get Key Info about shared users/sessions
findKey                Find Key
findSingleKey          Find single Key
getAttribute           Reads an attribute from an object

     Certificate Setup Commands
getCert                Gets Partition Certificates stored on HSM

     Key Transfer Commands
insertMaskedObject     Inserts a masked object
extractMaskedObject    Extracts a masked object

     Management Crypto Commands
sign                   Generates a signature
verify                 Verifies a signature
aesWrapUnwrap          Does NIST AES Wrap/Unwrap

     Helper Commands
Error2String           Converts Error codes to Strings
                       save key handle in fake PEM format
getCaviumPrivKey       Saves an RSA private key handle
                       in fake PEM format
IsValidKeyHandlefile   Checks if private key file has
                       an HSM key handle or a real key
listAttributes         List all attributes for getAttributes
```

```
   listECCCurveIds        List HSM supported ECC CurveIds
```

Parameters

Questo comando non ha parametri.

Argomenti correlati

- loginHSM e logoutHSM

Importa una chiave privata usando AWS CloudHSM KMU

Utilizzate il importPrivateKey comando in AWS CloudHSM key_mgmt_util per importare una chiave privata asimmetrica da un file a un modulo di sicurezza hardware (HSM). L'HSM non consente l'importazione diretta di chiavi in formato cleartext. Il comando crittografa la chiave privata utilizzando una chiave di wrapping AES specificata dall'utente e decrittografa la chiave all'interno dell'HSM. Se state cercando di associare una chiave a un AWS CloudHSM certificato, fate riferimento a questo argomento.

> **Note**
>
> Non è possibile importare una chiave PEM protetta da password utilizzando una chiave simmetrica o privata.

È necessario specificare una chiave di wrapping AES con un valore di attributo 1 `OBJ_ATTR_UNWRAP` e `OBJ_ATTR_ENCRYPT`. Per trovare gli attributi della chiave, utilizza il comando getAttribute.

> **Note**
>
> Questo comando non offre la possibilità di contrassegnare la chiave importata come non esportabile.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
importPrivateKey -h
```

```
importPrivateKey -l <label>
                 -f <key-file>
                 -w <wrapping-key-handle>
                 [-sess]
                 [-id <key-id>]
                 [-m_value <0...8>]
                 [min_srv <minimum-number-of-servers>]
                 [-timeout <number-of-seconds>]
                 [-u <user-ids>]
                 [-wk <wrapping-key-file>]
                 [-attest]
```

Esempi

Questo esempio illustra come utilizzare importPrivateKey per importare una chiave privata in un HSM.

Example: Importare una chiave privata

Questo comando importa la chiave privata da un file denominato `rsa2048.key` con l'etichetta `rsa2048-imported` e una chiave di wrapping con handle `524299`. Quando il comando viene completato, importPrivateKey restituisce un'handle per la chiave importata e un messaggio di successo.

```
Command: importPrivateKey -f rsa2048.key -l rsa2048-imported -w 524299

BER encoded key length is 1216

Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

Private Key Unwrapped.  Key Handle: 524301

Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-l`

Specifica un'etichetta definita dall'utente per la chiave privata.

Obbligatorio: sì

`-f`

Specifica il nome del file della chiave da importare.

Obbligatorio: sì

`-w`

Specifica l'handle di una chiave di wrapping. Questo parametro è obbligatorio. Per trovare le handle della chiave, utilizza il comando findKey.

Per determinare se una chiave può essere utilizzata come chiave di wrapping, utilizzare getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_WRAP` (262). Per creare una chiave di wrapping, utilizza genSymKey per creare una chiave AES (tipo 31).

Se utilizzi il parametro `-wk` per specificare una chiave di unwrapping esterna, la chiave di wrapping `-w` viene utilizzata per eseguire il wrapping della chiave durante l'importazione, ma non per annullarlo.

Obbligatorio: sì

`-sess`

Specifica la chiave importata come una chiave di sessione.

Impostazione predefinita: la chiave importata è detenuta come persistente (token) nel cluster.

Obbligatorio: no

`-id`

Specifica l'ID della chiave da importare.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

`-m_value`

Specifica il numero di utenti che devono approvare le operazioni di cifratura che utilizzano la chiave importata. Inserire un valore da **`0`** a **`8`**.

Questo parametro è valido soltanto quando il parametro `-u` nel comando condivide la chiave con un numero sufficiente di utenti per soddisfare il requisito `m_value`.

Impostazione Predefinita: 0

Obbligatorio: no

`-min_srv`

Speciifica il numero minimo HSMs su cui la chiave importata viene sincronizzata prima della scadenza del valore del `-timeout` parametro. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

`-timeout`

Speciifica il numero di secondi di attesa per la sincronizzazione della chiave HSMs quando il `min-serv` parametro è incluso. Se non viene specificato un numero, il processo prosegue a tempo indefinito.

Impostazione predefinita: nessun limite

Obbligatorio: no

`-u`

Specifica l'elenco degli utenti con cui condividere la chiave privata importata. Questo parametro concede agli altri utenti di crittografia HSM (CUs) il permesso di utilizzare la chiave importata nelle operazioni crittografiche.

Immettete un elenco separato da virgole di utenti HSM, ad esempio. IDs `-u 5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'utente HSM IDs di CUs sull'HSM, usa ListUsers.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave importata.

Obbligatorio: no

`-wk`

Specifica la chiave da utilizzare per eseguire il wrapping della chiave importata. Inserire il percorso e il nome di un file che contiene una chiave AES non crittografata.

Quando si include questo parametro, importPrivateKey utilizza la chiave nel file `-wk` per eseguire il wrapping della chiave importata. Utilizza inoltre la chiave specificata dal parametro `-w` per annullare il wrapping.

Impostazione predefinita: Utilizza il codice di wrapping specificato nel parametro `-w` per eseguire il wrapping e per annullarlo.

Obbligatorio: no

`-attest`

Esegue un controllo di attestazione sulla risposta firmware per assicurare che il firmware su cui viene eseguito il cluster non è stata compromesso.

Obbligatorio: no

Argomenti correlati

- wrapKey
- unWrapKey
- genSymKey
- exportPrivateKey

Importa una chiave pubblica usando AWS CloudHSM KMU

Utilizzate il importPubKey comando in AWS CloudHSM key_mgmt_util per importare una chiave pubblica in formato PEM in un modulo di sicurezza hardware (HSM). È possibile utilizzarlo per importare le chiavi pubbliche generate al di fuori del HSM. È inoltre possibile utilizzare il comando per importare le chiavi esportate da HSM, ad esempio quelle esportate tramite il comando exportPubKey.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
importPubKey -h

importPubKey -l <label>
             -f <key-file>
             [-sess]
             [-id <key-id>]
             [min_srv <minimum-number-of-servers>]
             [-timeout <number-of-seconds>]
```

Esempi

Questo esempio illustra come utilizzare importPubKey per importare una chiave pubblica in un HSM.

Example: Importa una chiave pubblica

Questo comando importa una chiave pubblica da un file con nome `public.pem` con l'etichetta `importedPublicKey`. Quando il comando viene completato, importPubKey restituisce un'handle per la chiave importata e un messaggio di successo.

```
Command: importPubKey -l importedPublicKey -f public.pem

Cfm3CreatePublicKey returned: 0x00 : HSM Return: SUCCESS

Public Key Handle: 262230

        Cluster Error Status
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-l`

Specifica un'etichetta definita dall'utente per la chiave pubblica.

Obbligatorio: sì

`-f`

Specifica il nome del file della chiave da importare.

Obbligatorio: sì

`-sess`

Designa la chiave importata come una chiave di sessione.

Impostazione predefinita: la chiave importata è detenuta come persistente (token) nel cluster.

Obbligatorio: no

`-id`

Specifica l'ID della chiave da importare.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

`-min_srv`

Speciifica il numero minimo con cui la chiave importata viene HSMs sincronizzata prima della scadenza del valore del parametro. `-timeout` Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

`-timeout`

Speciifica il numero di secondi di attesa per la sincronizzazione della chiave HSMs quando il `min-serv` parametro è incluso. Se non viene specificato un numero, il processo prosegue a tempo indefinito.

Impostazione predefinita: nessun limite

Obbligatorio: no

Argomenti correlati

- exportPubKey
- Genera Chiavi

Importa una chiave simmetrica in testo semplice usando KMU AWS CloudHSM

Utilizzate il imSymKey comando dello strumento AWS CloudHSM key_mgmt_util per importare una copia in testo semplice di una chiave simmetrica da un file nel modulo di sicurezza hardware (HSM). È possibile utilizzarlo per importare le chiavi generate con qualsiasi metodo diverso dall'HSM e le chiavi che sono state esportate da un HSM, ad esempio le chiavi che il comando, scrive in un file. exSymKey

Durante il processo di importazione, imSymKey utilizza una chiave AES selezionata (la chiave di wrapping) per effettuare il wrapping (crittografare) e quindi annullare il wrapping (decodificare) della chiave da importare. Tuttavia, imSymKey funziona solo per i file che contengono le chiavi in testo non crittografato. Per esportare e importare chiavi crittografate, usa unWrapKeyWrapKey e i comandi.

Inoltre, il comando imSymKey importa solo le chiavi simmetriche. Per importare le chiavi pubbliche, utilizzare importPubKey. Per importare chiavi private, usa importPrivateKeyo wrapKey.

> **Note**
>
> Non è possibile importare una chiave PEM protetta da password utilizzando una chiave simmetrica o privata.

Le chiavi importate funzionano in modo molto smile alle chiavi generate nell'HSM. Tuttavia, il valore dell'attributo OBJ_ATTR_LOCAL è pari a zero, a indicare che non è stato generato a livello locale. È possibile utilizzare il comando seguente per condividere una chiave simmetrica durante l'importazione. È possibile utilizzare il comando `shareKey` in cloudhsm_mgmt_util per condividere la chiave dopo l'importazione.

```
imSymKey -l aesShared -t 31 -f kms.key -w 3296 -u 5
```

Dopo l'importazione di una chiave, assicurarsi di contrassegnare o eliminare il file della chiave. Questo comando non evita di importare lo stesso materiale chiave più volte. Di conseguenza, più chiavi con distinti handle e lo stesso materiale chiave rendono difficile monitorare l'utilizzo del materiale chiave ed evitare il superamento dei limiti crittografici.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
imSymKey -h

imSymKey -f <key-file>
         -w <wrapping-key-handle>
         -t <key-type>
         -l <label>
         [-id <key-ID>]
         [-sess]
         [-wk <wrapping-key-file> ]
         [-attest]
         [-min_srv <minimum-number-of-servers>]
         [-timeout <number-of-seconds> ]
         [-u <user-ids>]
```

Esempi

Questi esempi mostrano come imSymKey importare chiavi simmetriche nel tuo. HSMs

Example: importazione di una chiave simmetrica AES

Questo esempio utilizza imSymKey per importare una chiave simmetrica AES in. HSMs

Il primo comando utilizza OpenSSL per generare una chiave simmetrica AES a 256 bit casuale. Memorizza la chiave nel file `aes256.key`.

```
$  openssl rand -out aes256.key 32
```

Il secondo comando consente imSymKey di importare la chiave AES dal `aes256.key` file in. HSMs Utilizza la chiave 20, una chiave AES nell'HSM, come chiave di wrapping e specifica un'etichetta di `imported`. A differenza dell'ID, l'etichetta non deve essere univoca nel cluster. Il valore del parametro (tipo) `-t` è `31`, che rappresenta AES.

L'output indica che la chiave nel file è stata sottoposta a wrapping e all'annullamento del wrapping, quindi importata nell'HSM, dove le è stato assegnato l'handle 262180.

```
Command:  imSymKey -f aes256.key -w 20 -t 31 -l imported

        Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Unwrapped.  Key Handle: 262180

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Il comando successivo utilizza getAttribute per ottenere l'attributo OBJ_ATTR_LOCAL (attributo 355) della chiave appena importata e lo scrive sul file `attr_262180`.

```
Command:  getAttribute -o 262180 -a 355 -out attributes/attr_262180
Attributes dumped into attributes/attr_262180_imported file

        Cfm3GetAttribute returned: 0x00 : HSM Return: SUCCESS
```

Quando si analizza il file degli attributi, è possibile verificare che il valore dell'attributo `OBJ_ATTR_LOCAL` è zero, il che indica che il materiale chiave non è stato generato nell'HSM.

```
$  cat attributes/attr_262180_local
OBJ_ATTR_LOCAL
```

```
0x00000000
```

Example: spostamento di una chiave simmetrica tra cluster

Questo esempio illustra come utilizzare exSymKey e imSymKey per spostare una chiave AES non crittografata tra i cluster. È possibile utilizzare un processo come questo per creare un wrapping AES esistente su HSMs entrambi i cluster. Una volta inserita la chiave di wrapping condivisa, è possibile utilizzare WrapKey unWrapKeye spostare le chiavi crittografate tra i cluster.

L'utente CU che esegue questa operazione deve disporre dell'autorizzazione per accedere a entrambi i cluster. HSMs

Il primo comando utilizza exSymKey per esportare la chiave 14, una chiave AES a 32 bit, dal cluster 1 al file `aes.key`. Utilizza la chiave 6, una chiave AES HSMs nel cluster 1, come chiave di wrapping.

```
Command: exSymKey -k 14 -w 6 -out aes.key

        Cfm3WrapKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapHostKey returned: 0x00 : HSM Return: SUCCESS


Wrapped Symmetric Key written to file "aes.key"
```

L'utente accede quindi a key_mgmt_util nel cluster 2 ed esegue un imSymKey comando per importare la chiave nel file nel cluster 2. `aes.key` HSMs Questo comando utilizza la chiave 252152, una chiave AES nel cluster 2, come chiave di wrapping. HSMs

Poiché le chiavi di wrapping utilizzate da exSymKey e imSymKey eseguono il wrapping e immediatamente annullano il wrapping delle chiavi di destinazione, le chiavi di wrapping su cluster diversi non devono essere le stesse.

L'output indica che la chiave è stata importata nel cluster 2 e che le è stato assegnato un handle di 21.

```
Command:  imSymKey -f aes.key -w 262152 -t 31 -l xcluster

        Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS
```

```
        Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Unwrapped.  Key Handle: 21

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Per dimostrare che la chiave 14 del cluster 1 e la chiave 21 del cluster 2 hanno lo stesso materiale chiave, ottieni il valore di controllo della chiave (KCV) di ciascuna chiave. Se i valori KCV sono gli stessi, il materiale chiave è lo stesso.

Il comando seguente utilizza getAttribute nel cluster 1 per scrivere il valore dell'attributo KCV (attributo 371) della chiave 14 sul file `attr_14_kcv`. Poi, utilizza un comando cat per ottenere il contenuto del file `attr_14_kcv`.

```
Command:  getAttribute -o 14 -a 371 -out attr_14_kcv
Attributes dumped into attr_14_kcv file

$  cat attr_14_kcv
OBJ_ATTR_KCV
0xc33cbd
```

Questo comando simile utilizza getAttribute nel cluster 2 per scrivere il valore dell'attributo KCV (attributo 371) della chiave 21 sul file `attr_21_kcv`. Poi, utilizza un comando cat per ottenere il contenuto del file `attr_21_kcv`.

```
Command:  getAttribute -o 21 -a 371 -out attr_21_kcv
Attributes dumped into attr_21_kcv file

$  cat attr_21_kcv
OBJ_ATTR_KCV
0xc33cbd
```

L'output indica che i valori KCV delle due chiavi sono gli stessi, il che dimostra che il materiale chiave è lo stesso.

Poiché lo stesso materiale chiave esiste in entrambi i cluster, ora è possibile condividere le chiavi crittografate tra i cluster senza mai esporre la chiave in chiaro. HSMs Ad esempio, è possibile utilizzare il comando `wrapKey` con la chiave di wrapping 14 per esportare una chiave crittografata

dal cluster 1 e quindi utilizzare `unWrapKey` con la chiave di wrapping 21 per importare la chiave crittografata nel cluster 2.

Example: importazione di una chiave di sessione

Questo comando utilizza i parametri `-sess` di imSymKey per importare una chiave Triple DES a 192 bit valida solo per la sessione corrente.

Il comando utilizza il parametro `-f` per specificare il file che contiene la chiave da importare, il parametro `-t` per specificare il tipo di chiave e il parametro `-w` per specificare la chiave di wrapping. Utilizza il parametro `-l` per specificare un'etichetta che qualifichi la chiave e il parametro `-id` per creare un identificatore intuitivo, ma univoco per la chiave. Viene inoltre utilizzato il parametro `-attest` per verificare il firmware che importa la chiave.

L'output indica che la chiave è stata sottoposta a wrapping e all'annullamento del wrapping, importata nell'HSM e che le è stato assegnato l'handle di 37. Inoltre, il controllo di attestazione ha avuto esito positivo, il che indica che il firmware non è stato danneggiato.

```
Command:  imSymKey -f 3des192.key -w 6 -t 21 -l temp -id test01 -sess -attest

        Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS

        Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS

        Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Unwrapped.  Key Handle: 37

        Attestation Check : [PASS]

        Cluster Error Status
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Quindi, è possibile utilizzare i comandi getAttribute o findKey per verificare gli attributi della chiave appena importata. Il comando seguente utilizza findKey per verificare che la chiave 37 disponga del tipo, dell'etichetta e dell'ID specificati dal comando e che sia una chiave di sessione. Come illustrato alla riga 5 dell'output, findKey indica che l'unica chiave corrispondente a tutti gli attributi è la chiave 37.

```
Command:  findKey -t 21 -l temp -id test01 -sess 1
Total number of keys present 1
```

```
 number of keys matched from start index 0::0
37

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

-f

Specifica il file che contiene la chiave da importare.

Il file deve contenere una copia in testo semplice di una chiave AES o Triple DES della lunghezza specificata. RC4 e le chiavi DES non sono valide in modalità FIPS. HSMs

- AES: 16, 24 o 32 byte
- Triple DES (3DES): 24 byte

Obbligatorio: sì

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-id

Specifica un identificatore definito dall'utente per la chiave. Digita una stringa univoca nel cluster. L'impostazione predefinita è una stringa vuota.

Impostazione predefinita: nessun valore ID.

Obbligatorio: no

-l

Specifica un'etichetta definita dall'utente per la chiave. Digita una stringa.

Puoi usare qualsiasi frase che ti aiuti a identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Obbligatorio: sì

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del parametro. `-timeout` Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-t

Specifica il tipo di chiave simmetrica. Inserisci la costante che rappresenta il tipo di chiave. Ad esempio, per creare una chiave AES, immetti `-t 31`.

Valori validi:

- 21: Triple DES (3DES).
- 31: AES

Obbligatorio: sì

-u

Condivide la chiave importata con utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CUs) il permesso di utilizzare questa chiave nelle operazioni crittografiche.

Digita un ID o un elenco separato da virgole di utenti HSM, ad esempio -u. IDs `5,6` Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare l'ID, è possibile utilizzare il comando listUsers nello strumento a riga di comando cloudhsm_mgmt_util o il comando listUsers nello strumento a riga di comando key_mgmt_util.

Obbligatorio: no

-w

Specifica l'handle di una chiave di wrapping. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Una chiave di wrapping è una chiave nell'HSM che viene utilizzata per crittografare ("eseguire il wrapping") e quindi decodificare ("annullare il wrapping) la chiave durante il processo di importazione. Solo le chiavi AES possono essere utilizzate come chiavi di wrapping.

Puoi usare qualsiasi chiave AES (di qualsiasi dimensione) come chiave di wrapping. Poiché la chiave di wrapping effettua e quindi annulla immediatamente il wrapping della chiave di

destinazione, puoi utilizzare una chiave AES valida solo per la sessione come chiave di wrapping. Per determinare se una chiave può essere utilizzata come una chiave di wrapping, utilizza getAttribute per ottenere il valore dell'attributo OBJ_ATTR_WRAP (262). Per creare una chiave di wrapping, utilizza genSymKey per creare una chiave AES (tipo 31).

Se si utilizza il parametro -wk per specificare una chiave di wrapping esterna, la chiave di wrapping -w viene utilizzata per annullare il wrapping della chiave importata, ma non per eseguirlo.

> **Note**
>
> La chiave 4 è una chiave interna non supportata. Ti consigliamo di utilizzare una chiave AES che crei e gestisci come chiave di wrapping.

Obbligatorio: sì

-wk

Utilizza la chiave AES nel file specificato per eseguire il wrapping della chiave importata. Inserire il percorso e il nome di un file che contiene una chiave AES non crittografata.

Quando includi questo parametro. imSymKey utilizza la chiave nel file -wk per eseguire il wrapping della chiave importata e utilizza la chiave nell'HSM specificato dal parametro -w per annullarne il wrapping. I valori di parametro -w e -wk devono determinare la stessa chiave non crittografata.

Impostazione predefinita: utilizzo della chiave di wrapping sull'HSM per annullare il wrapping.

Obbligatorio: no

Argomenti correlati

- genSymKey
- exSymKey
- wrapKey
- unWrapKey
- exportPrivateKey
- exportPubKey

Inserisci un oggetto mascherato usando AWS CloudHSM KMU

Utilizzate il insertMaskedObject comando in AWS CloudHSM key_mgmt_util per inserire un oggetto mascherato da un file in un modulo di sicurezza hardware (HSM) designato. Gli oggetti nascosti sono oggetti clonati estratti da un HSM utilizzando il comando extractMaskedObject. Possono essere utilizzati solo dopo averli inseriti nel cluster originale. È possibile inserire solo un oggetto mascherato nello stesso cluster da cui è stato generato, o un clone dello stesso cluster. Questo include qualsiasi versione clonata del cluster originale generata dalla copia di un backup tra le regioni e utilizzando tale backup per creare un nuovo cluster.

Gli oggetti mascherati sono un modo efficiente per scaricare e sincronizzare le chiavi, incluse cui le chiavi nonestraibili (ovvero chiavi che hanno un valore `OBJ_ATTR_EXTRACTABLE` di `0`). In questo modo, le chiavi possono essere sincronizzate in modo sicuro tra cluster correlati in diverse regioni senza la necessità di aggiornare il file di configurazione. AWS CloudHSM

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
insertMaskedObject -h

insertMaskedObject -f <filename>
                   [-min_srv <minimum-number-of-servers>]
                   [-timeout <number-of-seconds>]
```

Esempi

Questo esempio illustra come utilizzare insertMaskedObject per inserire un file oggetto nascosto in un HSM.

Example: Insert a masked object

Questo comando inserisce un oggetto nascosto in un HSM da un file denominato `maskedObj`. Quando il comando viene completato, insertMaskedObject restituisce un'handle della chiave per la chiave decrittografata dall'oggetto nascosto e un messaggio di successo.

```
Command: insertMaskedObject -f maskedObj

Cfm3InsertMaskedObject returned: 0x00 : HSM Return: SUCCESS
        New Key Handle: 262433
```

```
        Cluster Error Status
        Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-f`

Specifica il nome del file dell'oggetto nascosto da inserire.

Obbligatorio: sì

`-min_srv`

Specifica il numero minimo di server su cui l'oggetto nascosto inserito è sincronizzato prima che il valore del parametro `-timeout` scada. Se l'oggetto non è sincronizzato sul numero di server specificato nel tempo allocato, non viene inserito.

Impostazione predefinita: 1

Obbligatorio: no

`-timeout`

Specifica il numero di secondi di attesa per la sincronizzazione della chiave tra i server quando viene incluso il parametro `min-serv`. Se non viene specificato un numero, il processo prosegue a tempo indefinito.

Impostazione predefinita: nessun limite

Obbligatorio: no

Argomenti correlati

- extractMaskedObject

- syncKey
- Copiare un backup tra regioni
- Creazione di un AWS CloudHSM cluster da un backup precedente

Convalida il file chiave utilizzando AWS CloudHSM KMU

Usa il IsValidKeyHandlefile comando in AWS CloudHSM key_mgmt_util per scoprire se un file di chiave contiene una vera chiave privata o una falsa chiave RSA PEM. Un file PEM falso non contiene il materiale della chiave privata, ma i riferimenti alla chiave privata nell'HSM. Tale file può essere utilizzato per stabilire l'offload dal tuo server web a. SSL/TLS AWS CloudHSM Per ulteriori informazioni, consulta SSL/TLS Offload su Linux con Tomcat o SSL/TLS Offload su Linux con NGINX o Apache.

> **Note**
>
> IsValidKeyHandlefilefunziona solo con le chiavi RSA.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
IsValidKeyHandlefile -h

IsValidKeyHandlefile -f <rsa-private-key-file>
```

Esempi

Questi esempi mostrano come utilizzare IsValidKeyHandlefile per stabilire se un determinato file chiave contiene il materiale di chiavi reali o il materiale di chiavi PEM false.

Example: Convalida una vera chiave privata

Questo comando conferma che il file chiamato `privateKey.pem` contiene materiale di chiavi reali.

```
Command: IsValidKeyHandlefile -f privateKey.pem
```

```
Input key file has real private key
```

Example: Invalida una chiave PEM falsa

Questo comando conferma che il file chiamato `caviumKey.pem` contiene materiale di chiavi PEM false ottenuto dall'handle della chiave 15.

```
Command: IsValidKeyHandlefile -f caviumKey.pem

Input file has invalid key handle: 15
```

Parameters

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Obbligatorio: sì

`-f`

Speciifica il file di chiave privata RSA da verificare per verificare la presenza di materiale chiave valido.

Campo obbligatorio: sì

Argomenti correlati

- getCaviumPrivChiave
- SSL/TLS Offload su Linux utilizzando Tomcat
- SSL/TLS Offload su Linux utilizzando NGINX o Apache

Elenca gli attributi di una AWS CloudHSM chiave usando KMU

Utilizzate il listAttributes comando in AWS CloudHSM key_mgmt_util per elencare gli attributi di una chiave e le costanti che li rappresentano. AWS CloudHSM Puoi utilizzare queste costanti per identificare gli attributi nei comandi getAttribute e setAttribute. Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

Questo comando non ha parametri.

```
listAttributes
```

Esempio

Questo comando elenca gli attributi della chiave che puoi ottenere e modificare in key_mgmt_util e le costanti che li rappresentano. Per informazioni sull'interpretazione degli attributi delle chiavi, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Per rappresentare tutti gli attributi nel comando getAttribute di key_mgmt_util, utilizza 512.

```
Command: listAttributes

Following are the possible attribute values for getAttributes:

      OBJ_ATTR_CLASS                = 0
      OBJ_ATTR_TOKEN                = 1
      OBJ_ATTR_PRIVATE              = 2
      OBJ_ATTR_LABEL                = 3
      OBJ_ATTR_KEY_TYPE             = 256
      OBJ_ATTR_ENCRYPT              = 260
      OBJ_ATTR_DECRYPT              = 261
      OBJ_ATTR_WRAP                 = 262
      OBJ_ATTR_UNWRAP               = 263
      OBJ_ATTR_SIGN                 = 264
      OBJ_ATTR_VERIFY               = 266
      OBJ_ATTR_LOCAL                = 355
      OBJ_ATTR_MODULUS              = 288
      OBJ_ATTR_MODULUS_BITS         = 289
      OBJ_ATTR_PUBLIC_EXPONENT      = 290
      OBJ_ATTR_VALUE_LEN            = 353
      OBJ_ATTR_EXTRACTABLE          = 354
      OBJ_ATTR_KCV                  = 371
```

Argomenti correlati

- listAttributes in cloudhsm_mgmt_util

- OttieniAttributo
- setAttribute
- Riferimento per l'attributo della chiave

Elenca tutti AWS CloudHSM gli utenti che utilizzano KMU

Utilizzate il listUsers comando in AWS CloudHSM key_mgmt_util per inserire gli utenti nei moduli di sicurezza hardware (HSM), insieme al tipo di utente e ad altri attributi.

In key_mgmt_util, ListUsers restituisce un output che rappresenta HSMs tutto il cluster, anche se non è coerente. Per ottenere informazioni sugli utenti in ciascun HSM, utilizza il comando listUsers in cloudhsm_mgmt_util.

I comandi utente in key_mgmt_util listUsers e getKeyInfo, sono comandi di sola lettura che crypto users () sono autorizzati a eseguire. CUs Gli altri comandi di gestione utenti fanno parte di cloudhsm_mgmt_util. Vengono eseguiti da crypto officer (CO) che dispongono di autorizzazioni di gestione degli utenti.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
listUsers

listUsers -h
```

Esempio

Questo comando elenca gli utenti del cluster e i relativi attributi. HSMs È possibile utilizzare l'`User ID`attributo per identificare gli utenti in altri comandi, ad esempio findKey, getAttribute e. getKeyInfo

```
Command:  listUsers

        Number Of Users found 4

        Index       User ID     User Type       User Name           MofnPubKey
 LoginFailureCnt         2FA
```

```
    1               1       PCO             admin                   NO
0               NO
    2               2       AU              app_user                NO
0               NO
    3               3       CU              alice                   YES
0               NO
    4               4       CU              bob                     NO
0               NO
    5               5       CU              trent                   YES
0               NO

    Cfm3ListUsers returned: 0x00 : HSM Return: SUCCESS
```

L'output include i seguenti attributi degli utenti:

- ID utente: identifica l'utente nei comandi key_mgmt_util e cloudhsm_mgmt_util.
- Tipo utente: stabilisce quali operazioni può eseguire l'utente sull'HSM.
- Nome utente: visualizza il nome intuitivo definito dall'utente.
- MofnPubKey: indica se l'utente ha registrato una coppia di chiavi per la firma dei token di autenticazione del quorum.
- LoginFailureCnt: indica il numero di volte in cui l'utente ha effettuato l'accesso senza successo.
- 2FA: indica che l'utente ha abilitato l'autenticazione a più fattori.

Parameters

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

Argomenti correlati

- listUsers su cloudhsm_mgmt_util
- findKey
- getAttribute
- getKeyInfo

Accedere e disconnettersi da un HSM utilizzando KMU AWS CloudHSM

Utilizzate i logoutHSM comandi loginHSM and in AWS CloudHSM key_mgmt_util per accedere e disconnettervi dai moduli di sicurezza hardware (HSM) in un cluster. Una volta effettuato l'accesso a HSMs, è possibile utilizzare key_mgmt_util per eseguire una serie di operazioni di gestione delle chiavi, tra cui la generazione, la sincronizzazione e il wrapping di chiavi pubbliche e private.

Prima di eseguire qualsiasi comando key_mgmt_util, devi avviare key_mgmt_util. Per gestire le chiavi con key_mgmt_util, devi accedere come utente crittografico (CU). HSMs

> **Note**
>
> Se superi cinque tentativi di accesso errati, il tuo account viene bloccato. Se il cluster è stato creato prima di febbraio 2018, l'account viene bloccato dopo 20 tentativi di accesso errati. Per sbloccare l'account, un responsabile della crittografia (CO) deve reimpostare la password utilizzando il comando ModificaPswd in cloudhsm_mgmt_util.
> Se disponi di più HSM nel cluster, potresti avere consentiti più tentativi di accesso errati prima che l'account venga bloccato. Questo perché il client CloudHSM bilancia il carico tra diversi. HSMs Pertanto, il tentativo di accesso potrebbe non iniziare sullo stesso HSM ogni volta. Se stai testando questa funzionalità, ti consigliamo di farlo su un cluster con un solo HSM attivo.

Sintassi

```
loginHSM -h

loginHSM -u <user type>
         { -p | -hpswd } <password>
         -s <username>
```

Esempio

Questo esempio mostra come accedere e disconnettersi da un cluster con i HSMs `loginHSM` comandi and. `logoutHSM`

Example: Accedere a HSMs

Questo comando consente di accedere HSMs come utente crittografico (`CU`) con nome utente `example_user` e password`aws`. L'output mostra che hai effettuato l'accesso HSMs a tutti gli utenti del cluster.

```
Command:  loginHSM -u CU -s example_user -p aws

Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: accedi con una password nascosta

Questo comando è lo stesso dell'esempio precedente, tranne che questa volta si specifica che il sistema deve nascondere la password.

```
Command:  loginHSM -u CU -s example_user -hpswd
```

Il sistema ti invita a inserire la tua password. Si immette la password, il sistema la nasconde e l'output mostra che il comando ha avuto successo e che ci si è connessi al. HSMs

```
Enter password:

Cfm3LoginHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS

Command:
```

Example: Esci dal HSMs

Questo comando consente di disconnettersi da. HSMs L'output mostra che l'utente si è disconnesso da tutti gli utenti del HSMs cluster.

```
Command: logoutHSM

Cfm3LogoutHSM returned: 0x00 : HSM Return: SUCCESS

Cluster Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

-h

Mostra aiuto per questo comando.

-u

Specifica il tipo di utente di accesso. Per utilizzare key_mgmt_util, è necessario accedere come CU.

Obbligatorio: sì

-s

Specifica il nome utente di accesso.

Obbligatorio: sì

{-p | -hpswd}

Specificare la password di accesso con `-p`. La password viene visualizzata in testo normale quando la digiti. Per nascondere la password, utilizza il parametro opzionale `-hpswd` piuttosto di `-p` e segui le istruzioni.

Campo obbligatorio: sì

Argomenti correlati

- Esci

Imposta gli attributi delle AWS CloudHSM chiavi usando KMU

Utilizzate il setAttribute comando in AWS CloudHSM key_mgmt_util per convertire una chiave valida solo nella sessione corrente in una chiave persistente che esiste fino a quando non la eliminate. L'operazione viene effettuata modificando il valore dell'attributo token della chiave (`OBJ_ATTR_TOKEN`) da falso (`0`) a vero (`1`). Puoi modificare solo gli attributi di chiavi di tua proprietà.

Inoltre, puoi utilizzare il comando setAttribute in cloudhsm_mgmt_util per modificare l'etichetta, eseguire e annullare il wrapping e crittografare e decodificare gli attributi.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
setAttribute -h

setAttribute -o <object handle>
             -a 1
```

Esempio

Questo esempio mostra come convertire una chiave di sessione in una chiave persistente.

Il primo comando utilizza il `-sess` parametro di genSymKeyper creare una chiave AES a 192 bit valida solo nella sessione corrente. L'output indica che l'handle della nuova chiave di sessione è `262154`.

```
Command: genSymKey -t 31 -s 24 -l tmpAES -sess

        Cfm3GenerateSymmetricKey returned: 0x00 : HSM Return: SUCCESS

        Symmetric Key Created.  Key Handle: 262154

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
```

Questo comando utilizza findKey per trovare le chiavi di sessione nella sessione corrente. L'output conferma che la chiave `262154` è una chiave di sessione.

```
Command:  findKey -sess 1

Total number of keys present 1

 number of keys matched from start index 0::0
262154

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

```
        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Questo comando utilizza setAttribute per convertire la chiave 262154 da una chiave di sessione a una chiave persistente. Per farlo, modifica il valore dell'attributo token della chiave (OBJ_ATTR_TOKEN) da 0 (falso) a 1 (vero). Per informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Il comando utilizza il parametro -o per specificare l'handle della chiave (262154) e il parametro -a per specificare la costante che rappresenta l'attributo token (1). Quando si esegue il comando, viene richiesto un valore per l'attributo token. L'unico valore valido è 1 (vero): il valore per una chiave persistente.

```
Command: setAttribute -o 262154 -a 1
         This attribute is defined as a boolean value.
          Enter the boolean attribute value (0 or 1):1

        Cfm3SetAttribute returned: 0x00 : HSM Return: SUCCESS

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Per confermare che la chiave 262154 è ora persistente, questo comando utilizza findKey per cercare le chiavi di sessione (-sess 1) e le chiavi persistenti (-sess 0). Questa volta, il comando non trova alcuna chiave di sessione, ma restituisce 262154 nell'elenco delle chiavi persistenti.

```
Command: findKey -sess 1

Total number of keys present 0

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS



Command: findKey -sess 0
```

```
Total number of keys present 5

 number of keys matched from start index 0::4
6, 7, 524296, 9, 262154

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS

        Cfm3FindKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-o

Specifica l'handle della chiave di destinazione. È possibile specificare una sola chiave in ogni comando. Per individuare l'handle di una chiave, utilizza findKey.

Obbligatorio: sì

-a

Specifica la costante che rappresenta l'attributo da modificare. L'unico valore valido è 1, che rappresenta l'attributo token, `OBJ_ATTR_TOKEN`.

Per ottenere gli attributi e i valori interi, utilizza listAttributes.

Campo obbligatorio: sì

Argomenti correlati

- setAttribute in cloudhsm_mgmt_util
- getAttribute
- listAttributes
- Riferimento per l'attributo della chiave

Genera una firma usando AWS CloudHSM KMU

Utilizzate il sign comando in AWS CloudHSM key_mgmt_util per utilizzare una chiave privata scelta per generare una firma per un file.

Per utilizzare sign, è necessario disporre innanzitutto di una chiave privata nell'HSM. È possibile generare una chiave privata con i comandi genSymKey, genRSAKeyPair o genECCKeyPair. È anche possibile importarne uno con il comando importPrivateKey. Per ulteriori informazioni, vedi Generare chiavi.

Il comando sign utilizza un meccanismo di firma designato dall'utente, rappresentato da un numero intero, per firmare un file di messaggio. Per un elenco dei possibili meccanismi di firma, vedi Parametri.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
sign -h

sign -f <file name>
     -k <private key handle>
     -m <signature mechanism>
     -out <signed file name>
```

Esempio

Questo esempio illustra come utilizzare sign per firmare un file.

Example: firma un file

Questo comando firma un file denominato `messageFile` con una chiave privata con handle `266309`. Utilizza il meccanismo di firma `SHA256_RSA_PKCS` (1) e salva il file firmato risultante come `signedFile`.

```
Command: sign -f messageFile -k 266309 -m 1 -out signedFile

Cfm3Sign returned: 0x00 : HSM Return: SUCCESS

signature is written to file signedFile
```

```
Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
Node id 2 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

Questo comando accetta i parametri seguenti.

`-f`

Nome del file da firmare.

Obbligatorio: sì

`-k`

L'handle della chiave privata da utilizzare per la firma.

Obbligatorio: sì

`-m`

Numero intero che rappresenta il meccanismo di firma da utilizzare per la firma. I possibili meccanismi corrispondono ai seguenti numeri interi:

Meccanismo di firma	Numero intero corrispondente
`SHA1_RSA_PKCS`	0
`SHA256_RSA_PKCS`	1
`SHA384_RSA_PKCS`	2
`SHA512_RSA_PKCS`	3
`SHA224_RSA_PKCS`	4
`SHA1_RSA_PKCS_PSS`	5
`SHA256_RSA_PKCS_PSS`	6
`SHA384_RSA_PKCS_PSS`	7

Meccanismo di firma	Numero intero corrispondente
SHA512_RSA_PKCS_PSS	8
SHA224_RSA_PKCS_PSS	9
ECDSA_SHA1	15
ECDSA_SHA224	16
ECDSA_SHA256	17
ECDSA_SHA384	18
ECDSA_SHA512	19

Obbligatorio: sì

-out

Il nome del file in cui viene salvato il file firmato.

Campo obbligatorio: sì

Argomenti correlati

- Verifica
- importPrivateKey
- Gen. Pair RSAKey
- ECCKeycoppia di generi
- genSymKey
- Genera Chiavi

Scartare una AWS CloudHSM chiave usando KMU

Utilizzate il unWrapKey comando dello strumento AWS CloudHSM key_mgmt_util per importare una chiave simmetrica o privata avvolta (crittografata) da un file nell'HSM. È concepito per importare le chiavi di crittografia su cui è stato eseguito il comando wrapKey nell'interfaccia a riga di comando

key_mgmt_util, ma può essere utilizzato anche per annullare il wrapping delle chiavi effettuato con altri strumenti. Tuttavia, in tali situazioni, ti consigliamo di utilizzare le librerie software PKCS # 11 o JCE per annullare il wrapping della chiave.

Le chiavi AWS CloudHSM importate funzionano come le chiavi generate da. Tuttavia, il valore dell'attributo OBJ_ATTR_LOCAL è zero e indica che non sono state generate localmente.

Dopo aver importato una chiave, assicurati di contrassegnare o eliminare il file della chiave. Questo comando non evita di importare lo stesso materiale chiave più volte. Di conseguenza, più chiavi con distinti handle e lo stesso materiale chiave rendono difficile monitorare l'utilizzo del materiale chiave ed evitare il superamento dei limiti crittografici.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
unWrapKey -h

unWrapKey -f <key-file-name>
          -w <wrapping-key-handle>
          [-sess]
          [-min_srv <minimum-number-of-HSMs>]
          [-timeout <number-of-seconds>]
          [-aad <additional authenticated data filename>]
          [-tag_size <tag size>]
          [-iv_file <IV file>]
          [-attest]
          [-m <wrapping-mechanism>]
          [-t <hash-type>]
          [-nex]
          [-u <user id list>]
          [-m_value <number of users needed for approval>]
          [-noheader]
          [-l <key-label>]
          [-id <key-id>]
          [-kt <key-type>]
          [-kc <key-class>]
          [-i <unwrapping-IV>]
```

Esempio

Questi esempi mostrano come unWrapKey importare una chiave incapsulata da un file in. HSMs Nel primo esempio, abbiamo annullato il wrapping di una chiave eseguito con il comando key_mgmt_util wrapKey, che quindi aveva un'intestazione. Nel secondo esempio, abbiamo annullato il wrapping di una chiave eseguito al di fuori di key_mgmt_util, quindi senza intestazione.

Example: annullare il wrapping di una chiave (con intestazione)

Questo comando importa una copia con wrapping di una chiave simmetrica 3DES in un modulo HSM. Il wrapping della chiave viene annullato da una chiave AES con un'etichetta 6, identica dal punto di vista crittografico a quella utilizzata per eseguire il wrapping della chiave 3DES. L'output indica che è stato effettuato l'annullamento del wrapping della chiave nel file e che è stata importata e che l'handle della chiave importata è 29.

```
        Command:  unWrapKey -f 3DES.key -w 6 -m 4

        Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS

        Key Unwrapped.  Key Handle: 29

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Example: annullare il wrapping di una chiave (senza intestazione)

Questo comando importa una copia con wrapping di una chiave simmetrica 3DES in un modulo HSM. Il wrapping della chiave viene annullato da una chiave AES con un'etichetta 6, identica dal punto di vista crittografico a quella utilizzata per eseguire il wrapping della chiave 3DES. Dal momento che il wrapping di questa chiave 3DES non è stato eseguito con key_mgmt_util, viene specificato il parametro `noheader`, insieme ai parametri richiesti: l'etichetta della chiave (`unwrapped3DES`), la classe della chiave (4) e il tipo di chiave (21). L'output indica che è stato effettuato l'annullamento del wrapping della chiave nel file ed è stata importata e che l'handle della chiave importata è 8.

```
Command: unWrapKey -f 3DES.key -w 6 -noheader -l unwrapped3DES -kc 4 -kt 21 -m 4

      Cfm3CreateUnwrapTemplate2 returned: 0x00 : HSM Return: SUCCESS
       Cfm2UnWrapWithTemplate3 returned: 0x00 : HSM Return: SUCCESS
```

```
        Key Unwrapped. Key Handle: 8

        Cluster Error Status
        Node id 1 and err state 0x00000000 : HSM Return: SUCCESS
        Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-f

Il percorso e il nome del file contenente la chiave su cui è stato eseguito il wrapping.

Obbligatorio: sì

-w

Specifica la chiave di wrapping. Immettere l'handle di una chiave AES o RSA nell'HSM. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Per creare una chiave di wrapping, utilizzare genSymKeyper generare una chiave AES (tipo 31) o gen RSAKey Pair per generare una coppia di chiavi RSA (tipo 0). Se utilizzi una coppia di chiavi RSA, assicurati di eseguire il wrapping della chiave con una e di annullare il wrapping con l'altra. Per determinare se una chiave può essere usata come chiave di wrapping, utilizza getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_WRAP`, che è rappresentato dalla costante `262`.

Obbligatorio: sì

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Utilizza questo parametro quando hai bisogno di una chiave solo per un breve periodo, ad esempio una chiave di wrapping che crittografa e quindi decodifica rapidamente un'altra chiave. Non utilizzare una chiave di sessione per crittografare dati che potresti aver bisogno di decodificare al termine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Obbligatorio: no

-min_srv

Speciifica il numero minimo HSMs su cui la chiave viene sincronizzata prima della scadenza del valore del parametro. `-timeout` Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave con ogni HSM del cluster. Per velocizzare il processo, impostate un valore inferiore `min_srv` al numero di componenti del HSMs cluster e impostate un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Obbligatorio: no

-timeout

Speciifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero HSMs specificato dal parametro. `min_srv`

Questo parametro è valido solo quando il parametro `min_srv` viene utilizzato anche nel comando.

Impostazione Predefinita: No timeout. Il comando attende a tempo indefinito e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Obbligatorio: no

-attestare

Esegue un controllo di integrità per verificare che il firmware su cui viene eseguito il cluster non sia stato manomesso.

Impostazione predefinita: nessun controllo di attestazione.

Obbligatorio: no

-nex

Rende la chiave non estraibile. La chiave generata non può essere esportata dall'HSM.

Impostazione predefinita: la chiave è estraibile.

Obbligatorio: no

-m

Il valore che rappresenta il meccanismo di wrapping. CloudHSM supporta i seguenti meccanismi:

Meccanismo	Valore
`AES_KEY_WRAP_PAD_PKCS5`	4
`NIST_AES_WRAP_NO_PAD`	5
`NIST_AES_WRAP_PAD`	6
`RSA_AES`	7
`RSA_OAEP` (per la dimensione massima dei dati, vedi la nota più avanti in questa sezione)	8
`AES_GCM`	10
`CLOUDHSM_AES_GCM`	11
`RSA_PKCS` (per la dimensione massima dei dati, vedi la nota più avanti in questa sezione). Vedi la nota 1 di seguito per una modifica imminente.	12

Obbligatorio: sì

> **Note**
>
> Quando si utilizza il meccanismo di `RSA_OAEP` wrapping, la dimensione massima della chiave che è possibile avvolgere è determinata dal modulo della chiave RSA e dalla

lunghezza dell'hash specificato nel modo seguente: Dimensione massima della chiave = modulusLengthIn Bytes- (2* Bytes) -2. hashLengthIn
Quando si utilizza il meccanismo di wrapping RSA_PKCS, la dimensione massima della chiave che è possibile avvolgere è determinata dal modulo della chiave RSA come segue: Dimensione massima della chiave = (modulusLengthInByte -11).

-t

Algoritmo hash	Valore
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (valido per i meccanismi RSA_AES e RSA_0AEP)	6

Obbligatorio: no

-no intestazione

Se si sta per eseguire l'annullamento del wrapping di una chiave eseguito all'esterno di key_mgmt_util, è necessario specificare questo parametro e tutti gli altri parametri associati.

Obbligatorio: no

> **Note**
>
> Se si specifica questo parametro, è necessario specificare anche i seguenti parametri -noheader:

- -l

  Specifica l'etichetta da aggiungere alla chiave su cui è stato annullato wrapping.

Obbligatorio: sì

- -kc

  Specifica la classe della chiave su cui annullare il wrapping. Di seguito sono elencati i valori accettabili:

  3 = chiave privata di una coppia di chiavi pubbliche-private

  4 = chiave segreta (simmetrica)

  Obbligatorio: sì

- -kt

  Specifica il tipo di chiave su cui annullare il wrapping. Di seguito sono elencati i valori accettabili:

  0 = `RSA`

  1 = `DSA`

  3 = `ECC`

  16 = `GENERIC_SECRET`

  21 = `DES3`

  31 = `AES`

  Obbligatorio: sì

È inoltre possibile specificare i parametri `-noheader` seguenti:

- -id

  L'ID da aggiungere alla chiave su cui è stato annullato il wrapping.

  Obbligatorio: no

- -i

  Il vettore di inizializzazione (IV) da utilizzare per annullare il wrapping.

  Obbligatorio: no

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Argomenti correlati

- wrapKey
- exSymKey
- imSymKey

Verifica la firma di un file utilizzando AWS CloudHSM KMU

Utilizzate il verify comando in AWS CloudHSM key_mgmt_util per confermare se un file è stato firmato o meno con una determinata chiave. Per farlo, il comando verify confronta un file firmato rispetto a un file di origine e analizza se sono correlati a livello di crittografia in base a un determinato meccanismo di firma e a una chiave pubblica. I file possono essere registrati con l'operazione. AWS CloudHSM sign

I meccanismi di firma sono rappresentati dai numeri interi elencati nella sezione parametri.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
verify -h

verify -f <message-file>
       -s <signature-file>
       -k <public-key-handle>
       -m <signature-mechanism>
```

Esempio

Questi esempi mostrano come utilizzare verify per controllare se una determinata chiave pubblica è stato utilizzata per firmare un determinato file.

Example: verifica della firma di un file

Questo comando tenta di verificare se un file denominato `hardwarCert.crt` è stato firmato dalla chiave pubblica `262276` tramite il meccanismo di firma `SHA256_RSA_PKCS` per produrre il file firmato

`hardwareCertSigned`. Poiché i parametri dati rappresentano una vera e propria relazione di firma, il comando restituisce un messaggio di successo.

```
Command: verify -f hardwareCert.crt -s hardwareCertSigned -k 262276 -m 1

Signature verification successful

Cfm3Verify returned: 0x00 : HSM Return: SUCCESS
```

Example: Dimostrare un rapporto di firma falso

Questo comando verifica se un file denominato `hardwareCert.crt` è stato firmato dalla chiave pubblica 262276 tramite il meccanismo di firma `SHA256_RSA_PKCS` per produrre il file firmato `userCertSigned`. Poiché i parametri dati non rappresentano una vera e propria relazione di firma, il comando restituisce un messaggio di errore.

```
Command: verify -f hardwarecert.crt -s usercertsigned -k 262276 -m 1
Cfm3Verify returned: 0x1b

CSP Error: ERR_BAD_PKCS_DATA
```

Parameters

Questo comando accetta i parametri seguenti.

`-f`

Il nome del file di messaggio originale.

Obbligatorio: sì

`-s`

Il nome del file firmato.

Campo obbligatorio: sì

`-k`

L'handle della chiave pubblica che si pensa sia stata utilizzata per firmare il file.

Obbligatorio: sì

-m

Numero intero che rappresenta il meccanismo di firma suggerito utilizzato per firmare il file. I possibili meccanismi corrispondono ai seguenti numeri interi:

Meccanismo di firma	Numero intero corrispondente
SHA1_RSA_PKCS	0
SHA256_RSA_PKCS	1
SHA384_RSA_PKCS	2
SHA512_RSA_PKCS	3
SHA224_RSA_PKCS	4
SHA1_RSA_PKCS_PSS	5
SHA256_RSA_PKCS_PSS	6
SHA384_RSA_PKCS_PSS	7
SHA512_RSA_PKCS_PSS	8
SHA224_RSA_PKCS_PSS	9
ECDSA_SHA1	15
ECDSA_SHA224	16
ECDSA_SHA256	17
ECDSA_SHA384	18
ECDSA_SHA512	19

Campo obbligatorio: sì

Argomenti correlati

- Firma
- getCert
- Genera Chiave

Esporta qualsiasi AWS CloudHSM chiave usando KMU

Utilizzate il wrapKey comando in AWS CloudHSM key_mgmt_util per esportare una copia crittografata di una chiave simmetrica o privata dal modulo di sicurezza hardware (HSM) in un file. Quando esegui wrapKey, devi specificare la chiave da esportare, una chiave sull'HSM per crittografare la chiave da esportare (eseguirne il wrapping) e il file di output.

Il comando `wrapKey` scrive la chiave crittografata su un file specificato, ma non rimuove la chiave dall'HSM, né ne impedisce l'utilizzo nelle operazioni di crittografia. È possibile esportare la stessa chiave più volte.

Soltanto il proprietario della chiave, ovvero l'utente CU che ha creato la chiave, è in grado di esportarla. Gli utenti che condividono la chiave possono utilizzarla nelle operazioni di crittografia, ma non possono esportarla.

Per reimportare la chiave crittografata nell'HSM, usa. unWrapKey Per esportare una chiave in testo semplice da un HSM, utilizzate exSymKeyo come appropriato. exportPrivateKey Il aesWrapUnwrapcomando non può decrittografare (scartare) le chiavi che eseguono la crittografia. wrapKey

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere a HSM come crypto user (CU).

Sintassi

```
wrapKey -h

wrapKey -k <exported-key-handle>
        -w <wrapping-key-handle>
        -out <output-file>
        [-m <wrapping-mechanism>]
        [-aad <additional authenticated data filename>]
        [-t <hash-type>]
```

```
        [-noheader]
        [-i <wrapping IV>]
        [-iv_file <IV file>]
        [-tag_size <num_tag_bytes>>]
```

Esempio

Example

Questo comando esporta una chiave simmetrica Triple DES (3DES) a 192 bit (handle di chiave 7). Utilizza una chiave AES a 256 bit nell'HSM (handle di chiave 14) per eseguire il wrapping della chiave 7, quindi scrive la chiave 3DES crittografata nel file `3DES-encrypted.key`.

L'output indica che la chiave 7 (la chiave 3DES) è stata sottoposta a wrapping e che è stata scritta sul file specificato. La chiave crittografata è di 307 byte.

```
        Command:  wrapKey -k 7 -w 14 -out 3DES-encrypted.key -m 4

        Key Wrapped.

        Wrapped Key written to file "3DES-encrypted.key length 307

        Cfm2WrapKey returned: 0x00 : HSM Return: SUCCESS
```

Parameters

-h

Visualizza l'aiuto per il comando.

Obbligatorio: sì

-k

L'handle della chiave che si desidera esportare. Digita l'handle della chiave simmetrica o privata posseduta. Per trovare gli handle della chiave, utilizza il comando findKey.

Per verificare che una chiave possa essere esportata, utilizza il comando getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_EXTRACTABLE`, che è rappresentato dalla costante `354`. Per

informazioni sull'interpretazione degli attributi chiave, vedi AWS CloudHSM riferimento agli attributi chiave per KMU.

Puoi esportare solo le chiavi di tua proprietà. Per trovare il proprietario di una chiave, usa il comando. getKeyInfo

Obbligatorio: sì

-w

Specifica la chiave di wrapping. Immettere l'handle di una chiave AES o RSA nell'HSM. Questo parametro è obbligatorio. Per trovare gli handle della chiave, utilizza il comando findKey.

Per creare una chiave di wrapping, utilizzare genSymKeyper generare una chiave AES (tipo 31) o gen RSAKey Pair per generare una coppia di chiavi RSA (tipo 0). Se utilizzi una coppia di chiavi RSA, assicurati di eseguire il wrapping della chiave con una e di annullare il wrapping con l'altra. Per determinare se una chiave può essere usata come chiave di wrapping, utilizza getAttribute per ottenere il valore dell'attributo `OBJ_ATTR_WRAP`, che è rappresentato dalla costante `262`.

Obbligatorio: sì

-output

Il percorso e il nome del file di output. Quando il comando viene completato, questo file contiene una copia crittografata della chiave esportata. Se il file già esiste, il comando lo sovrascrive senza preavviso.

Obbligatorio: sì

-m

Il valore che rappresenta il meccanismo di wrapping. CloudHSM supporta i seguenti meccanismi:

Meccanismo	Valore
`AES_KEY_WRAP_PAD_PKCS5`	4
`NIST_AES_WRAP_NO_PAD`	5
`NIST_AES_WRAP_PAD`	6
`RSA_AES`	7

Meccanismo	Valore
RSA_OAEP (per la dimensione massima dei dati, vedi la nota più avanti in questa sezione)	8
AES_GCM	10
CLOUDHSM_AES_GCM	11
RSA_PKCS (per la dimensione massima dei dati, vedi la nota più avanti in questa sezione). Vedi la nota 1 di seguito per una modifica imminente.	12

Obbligatorio: sì

> **Note**
>
> Quando si utilizza il meccanismo di RSA_OAEP wrapping, la dimensione massima della chiave che è possibile avvolgere è determinata dal modulo della chiave RSA e dalla lunghezza dell'hash specificato, nel modo seguente: Dimensione massima della chiave = (Bytes-2* Bytes-2). modulusLengthIn hashLengthIn
> Quando si utilizza il meccanismo di wrapping RSA_PKCS, la dimensione massima della chiave che è possibile avvolgere è determinata dal modulo della chiave RSA come segue: Dimensione massima della chiave = (modulusLengthInByte -11).

-t

Il valore che rappresenta l'algoritmo hash. CloudHSM supporta i seguenti algoritmi:

Algoritmo hash	Valore
SHA1	2
SHA256	3
SHA384	4

Algoritmo hash	Valore
SHA512	5
SHA224 (valido per i meccanismi RSA_AES e RSA_OAEP)	6

Obbligatorio: no

-aad

Il nome del file contenente AAD.

> **Note**
>
> Valido solo per i meccanismi AES_GCM e CLOUDHSM_AES_GCM.

Obbligatorio: no

-no intestazione

Omette l'intestazione che specifica gli attributi della chiave specifici per CloudHSM. Utilizzare questo parametro solo se prevedi di annullare il wrapping della chiave con strumenti all'esterno di key_mgmt_util.

Obbligatorio: no

-i

Il vettore di inizializzazione (IV) (valore esadecimale).

> **Note**
>
> Valido solo se passato con il parametro -noheader per i meccanismi CLOUDHSM_AES_KEY_WRAP e NIST_AES_WRAP.

Obbligatorio: no

-iv_file

Il file in cui si desidera scrivere il valore IV ottenuto in risposta.

> **Note**
>
> Valido solo se passato con il parametro `-noheader` per il meccanismo `AES_GCM`.

Obbligatorio: no

-tag_size

La dimensione del tag da salvare insieme al blob oggetto del wrapping.

> **Note**
>
> Valido solo se passato con il parametro `-noheader` per i meccanismi `AES_GCM` e `CLOUDHSM_AES_GCM`. La dimensione minima del tag è otto.

Obbligatorio: no

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Argomenti correlati

- exSymKey
- imSymKey
- unWrapKey

AWS CloudHSM riferimento agli attributi chiave per KMU

I comandi AWS CloudHSM key_mgmt_util utilizzano costanti per rappresentare gli attributi delle chiavi in un modulo di sicurezza hardware (HSM). Questo argomento può aiutarti a identificare gli attributi, individuare le costanti che li rappresentano nei comandi e comprenderne i valori.

Puoi impostare gli attributi di una chiave al momento della sua creazione. Per modificare l'attributo token, che indica se una chiave è persistente o se è presente solo nella sessione, utilizza il comando setAttribute in key_mgmt_util. Utilizza il comando `setAttribute` in cloudhsm_mgmt_util per modificare l'etichetta, eseguire e annullare il wrapping o crittografare e decodificare gli attributi.

Per ottenere un elenco degli attributi e delle relative costanti, utilizza listAttributes. Per ottenere i valori degli attributi di una chiave, utilizza getAttribute.

Nella tabella seguente sono elencati gli attributi della chiave, le relative costanti e i valori validi.

Attributo	Costante	Valori
OBJ_ATTR_ALL	512	Rappresenta tutti gli attributi.
OBJ_ATTR_ALWAYS_SENSITIVE	357	0: Falso. 1: Vero.
OBJ_ATTR_CLASS	0	2: chiave pubblica in una coppia di chiavi pubblica-privata. 3: chiave privata in una coppia di chiavi pubblica-privata. 4: chiave segreta (simmetrica).
OBJ_ATTR_DECRYPT	261	0: Falso. 1: Vero. La chiave può essere utilizzata per decodificare i dati.
OBJ_ATTR_DERIVE	268	0: Falso. 1: Vero. La funzione ricava la chiave.
OBJ_ATTR_DESTROYABLE	370	0: Falso. 1: Vero.
OBJ_ATTR_ENCRYPT	260	0: Falso. 1: Vero. La chiave può essere utilizzata per crittografare i dati.

Attributo	Costante	Valori
OBJ_ATTR_EXTRACTABLE	354	0: Falso. 1: Vero. La chiave può essere esportata da. HSMs
OBJ_ATTR_ID	258	Stringa definita dall'uten te. Deve essere univoca nel cluster. L'impostazione predefinita è una stringa vuota.
OBJ_ATTR_KCV	371	Valore di controllo della chiave. Per ulteriori informazi oni, vedi Ulteriori dettagli.
OBJ_ATTR_KEY_TYPE	256	0: RSA. 1: DSA. 3: EC. 16: segreta generica. 18: RC4. 21: Triple DES (3DES). 31: AES.
OBJ_ATTR_LABEL	3	Stringa definita dall'utente. Non è necessario che sia univoca nel cluster.
OBJ_ATTR_LOCAL	355	0. Falso. La chiave è stata importata in HSMs. 1: Vero.

Attributo	Costante	Valori
OBJ_ATTR_MODULUS	288	Il modulo utilizzato per creare una coppia di chiavi RSA. Per le chiavi EC, questo valore rappresenta la codifica DER del valore ANSI X9.62 «Q» in formato ECPoint esadecimale. Per altri tipi di chiavi, questo attributo non esiste.
OBJ_ATTR_MODULUS_BITS	289	La lunghezza del modulo utilizzato per creare una coppia di chiavi RSA. Per le chiavi EC, questo rappresenta l'ID della curva ellittica utilizzata per generare la chiave. Per altri tipi di chiavi, questo attributo non esiste.
OBJ_ATTR_NEVER_EXTRACTABLE	356	0: Falso. 1: Vero. HSMsLa chiave non può essere esportata da.
OBJ_ATTR_PUBLIC_EXPONENT	290	L'esponente pubblico utilizzato per creare una coppia di chiavi RSA. Per altri tipi di chiavi, questo attributo non esiste.

Attributo	Costante	Valori
OBJ_ATTR_PRIVATE	2	0: Falso. 1: Vero. Questo attributo indica se gli utenti non autenticati possono elencare gli attributi della chiave. Poiché il provider CloudHSM PKCS#11 attualmente non supporta le sessioni pubbliche, tutte le chiavi (incluse le chiavi pubbliche di una coppia di chiavi pubblica-privata) hanno l'attributo impostato su 1.
OBJ_ATTR_SENSITIVE	259	0: Falso. Chiave pubblica in una coppia di chiavi pubblica-privata. 1: Vero.
OBJ_ATTR_SIGN	264	0: Falso. 1: Vero. La chiave può essere utilizzata per la firma (chiavi private).
OBJ_ATTR_TOKEN	1	0: Falso. Chiave di sessione. 1: Vero. Chiave persistente.
OBJ_ATTR_TRUSTED	134	0: Falso. 1: Vero.

Attributo	Costante	Valori
OBJ_ATTR_UNWRAP	263	0: Falso. 1: Vero. La chiave può essere utilizzata per decodificare le chiavi.
OBJ_ATTR_UNWRAP_TEMPLATE	1073742354	I valori devono utilizzare il modello di attributo applicato a qualsiasi chiave di cui è stato annullato il wrapping utilizzando questa chiave di wrapping.
OBJ_ATTR_VALUE_LEN	353	Lunghezza della chiave in byte.
OBJ_ATTR_VERIFY	266	0: Falso. 1: Vero. La chiave può essere utilizzata per la verifica (chiavi pubbliche).
OBJ_ATTR_WRAP	262	0: Falso. 1: Vero. La chiave può essere utilizzata per crittografare le chiavi.
OBJ_ATTR_WRAP_TEMPLATE	1073742353	I valori devono utilizzare il modello di attributo per abbinare la chiave sottoposta a wrapping usando questa chiave di wrapping.
OBJ_ATTR_WRAP_WITH_TRUSTED	528	0: Falso. 1: Vero.

Ulteriori dettagli

Valore di controllo della chiave (kcv)

Il valore di controllo chiave (KCV) è un hash o checksum a 3 byte di una chiave che viene generato quando l'HSM importa o genera una chiave. Puoi anche calcolare un KCV al di fuori dell'HSM, ad esempio dopo aver esportato una chiave. È quindi possibile confrontare i valori KCV per confermare l'identità e l'integrità della chiave. Per ottenere il KCV di una chiave, usa getAttribute.

AWS CloudHSM utilizza il seguente metodo standard per generare un valore di controllo della chiave:

- Chiavi simmetriche: primi 3 byte del risultato della crittografia a blocchi zero con la chiave.
- Coppie di chiavi asimmetriche: primi 3 byte dell'hash SHA-1 della chiave pubblica.
- Chiavi HMAC: KCV per le chiavi HMAC non è attualmente supportato.

Offload delle operazioni con AWS CloudHSM SDK del client

Utilizza un SDK del client per l'offload delle operazioni crittografiche da applicazioni basate su linguaggi o piattaforme ai moduli di sicurezza hardware (HSM).

AWS CloudHSM offre due versioni principali e Client SDK 5 è l'ultima. Offre una serie di vantaggi rispetto a Client SDK 3 (la serie precedente). Per ulteriori informazioni, consulta la pagina sui vantaggi di Client SDK 5. Per ulteriori informazioni sulle piattaforme supportate, consulta la pagina AWS CloudHSM Piattaforme supportate da Client SDK 5.

I seguenti argomenti descrivono come lavorare con AWS CloudHSM Client SDK.

AWS CloudHSM supporta i seguenti componenti:

the section called "Libreria PKCS #11"

PKCS #11 è uno standard per l'esecuzione di operazioni di crittografia su moduli di sicurezza hardware (HSM). AWS CloudHSM offre implementazioni della libreria PKCS #11 conformi a PKCS #11 versione 2.40.

the section called "OpenSSL Dynamic Engine"

L' AWS CloudHSM OpenSSL Dynamic Engine consente di trasferire le operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

the section called "Provider OpenSSL"

L' AWS CloudHSM OpenSSL Dynamic Engine Provider consente di trasferire le operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL Provider.

the section called "Provider JCE"

Il provider AWS CloudHSM JCE è conforme alla Java Cryptographic Architecture (JCA). Il provider consente di eseguire operazioni crittografiche sull'HSM.

the section called "Provider di archiviazione delle chiavi (KSP)"

Il AWS CloudHSM client per Windows include provider CNG e KSP.

Argomenti

- Controlla il tuo AWS CloudHSM Versione Client SDK
- Confronta AWS CloudHSM Supporto per i componenti Client SDK

- Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5
- Utilizzo di Client SDK 5 per lavorare con AWS CloudHSM
- Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM

Controlla il tuo AWS CloudHSM Versione Client SDK

Usa i seguenti comandi per verificare la versione di Client SDK con cui stai utilizzando. AWS CloudHSM

Amazon Linux

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

Amazon Linux 2

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 6

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 7

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

CentOS 8

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 6

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 7

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

RHEL 8

Utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

Ubuntu 16.04 LTS

Utilizza il seguente comando:

```
apt list --installed | grep ^cloudhsm
```

Ubuntu 18.04 LTS

Utilizza il seguente comando:

```
apt list --installed | grep ^cloudhsm
```

Ubuntu 20.04 LTS

Utilizza il seguente comando:

```
apt list --installed | grep ^cloudhsm
```

Windows Server

Utilizza il seguente comando:

```
wmic product get name,version
```

Confronta AWS CloudHSM Supporto per i componenti Client SDK

Oltre agli strumenti da riga di comando, il Client SDK 3 contiene componenti che consentono di scaricare le operazioni crittografiche sull'HSM da varie applicazioni basate su piattaforme o linguaggi. Client SDK 5 è uguale a Client SDK 3, tranne per il fatto che non supporta ancora i provider CNG e KSP. La tabella seguente mette a confronto la disponibilità dei componenti nel Client SDK 3 e nel Client SDK 5.

Componente	Client SDK 5	Client SDK 3
Libreria PKCS #11	Sì	Sì
Provider JCE	Sì	Sì
OpenSSL Dynamic Engine	Sì	Sì
Key Storage Provider (KSP)	Sì	Sì
Utility di gestione CloudHSM (CMU) [1]	Sì	Sì
Utility di gestione delle chiavi (KMU) [1]	Sì	Sì
Strumento di configurazione	Sì	Sì

[1] I componenti CMU e KMU sono inclusi nella CLI di CloudHSM con Client SDK 5.

Le sezioni seguenti descrivono i componenti.

Libreria PKCS #11

PKCS #11 è uno standard per l'esecuzione di operazioni crittografiche su moduli di sicurezza hardware (HSM). AWS CloudHSM offre implementazioni della libreria PKCS #11 conformi alla versione 2.40 di PKCS #11.

- Per Client SDK 3, la libreria PKCS #11 è un componente solo per Linux che corrisponde al supporto di base Linux. Per ulteriori informazioni, consulta the section called “Supporto Linux per Client SDK 3”.

- Per Client SDK 5, la libreria PKCS #11 è un componente multipiattaforma che corrisponde al supporto di base di Linux e Windows Client SDK 5. Per ulteriori informazioni, consultare the section called "Supporto Linux per Client SDK 5" e the section called "Supporto Windows per Client SDK 5".

Utility di gestione CloudHSM (CMU)

Lo strumento a riga di comando CloudHSM Management Utility (CMU) aiuta i responsabili delle criptovalute a gestire gli utenti negli HSM. Include strumenti che consentono di creare, eliminare ed elencare utenti e modificare le password degli utenti. Per ulteriori informazioni, consulta AWS CloudHSM Utilità di gestione (CMU).

Utility di gestione delle chiavi (KMU)

La Key Management Utility (KMU) è uno strumento a riga di comando che aiuta gli utenti crittografici (CU) a gestire le chiavi sui moduli di sicurezza hardware (HSM). Per ulteriori informazioni, consulta AWS CloudHSM Utilità di gestione delle chiavi (KMU).

Provider JCE

Il provider AWS CloudHSM JCE è conforme alla Java Cryptographic Architecture (JCA). Il provider consente di eseguire operazioni crittografiche sull'HSM.

Il provider JCE è un componente esclusivamente per Linux che corrisponde al supporto di base di Linux. Per ulteriori informazioni, consulta the section called "Supporto Linux per Client SDK 3".

- Per Client SDK 3 richiede OpenJDK 1.8

OpenSSL Dynamic Engine

L' AWS CloudHSM OpenSSL Dynamic Engine consente di trasferire le operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

- Per Client SDK 3, OpenSSL Dynamic Engine è un componente solo per Linux che non corrisponde al supporto di base di Linux. Consulta le esclusioni riportate di seguito.
  - Richiede OpenSSL 1.0.2[f+]

  Piattaforme non supportate:

- CentOS 8
- Red Hat Enterprise Linux (RHEL) 8
- Ubuntu 18.04 LTS

  Queste piattaforme vengono fornite con una versione di OpenSSL non compatibile con il motore dinamico OpenSSL for Client SDK 3. AWS CloudHSM supporta queste piattaforme con OpenSSL Dynamic Engine per Client SDK 5.
- Per Client SDK 5, OpenSSL Dynamic Engine è un componente solo per Linux che richiede OpenSSL 1.0.2, 1.1.1 o 3.x.

Key Storage Provider (KSP)

Key Storage Provider (KSP) è un'API crittografica specifica per il sistema operativo Microsoft Windows.

Per Client SDK 3, i provider CNG e KSP sono un componente solo per Windows che corrisponde al supporto di base di Windows. Per ulteriori informazioni, consulta Supporto Windows per AWS CloudHSM Client SDK 3.

Per Client SDK 5, il Key Storage Provider (KSP) è un componente solo per Windows che corrisponde al supporto di base di Windows. Per ulteriori informazioni, consulta Supporto Windows per AWS CloudHSM Client SDK 5.

Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Nel AWS CloudHSM, le applicazioni del cliente eseguono operazioni crittografiche utilizzando il AWS CloudHSM Client Software Development Kit (SDK). Client SDK 5 è l'SDK principale che continua ad avere nuove funzionalità e supporto per la piattaforma.

Client SDK 3 include due strumenti a riga di comando separati: la CMU per la gestione degli utenti e la KMU per la gestione delle chiavi e l'esecuzione di operazioni con le chiavi. Client SDK 5 consolida le funzioni di CMU e KMU (strumenti offerti con Client SDK 3) in un unico strumento, il. AWS CloudHSM Interfaccia a riga di comando (CLI) Le operazioni di gestione degli utenti sono disponibili nei sottocomandi e. La categoria di utenti nella CLI di CloudHSM La categoria quorum nella CLI di CloudhSM Le operazioni di gestione delle chiavi sono disponibili nel sottocomando key, mentre le operazioni di crittografia sono disponibili nel sottocomando crypto. Riferimento per i comandi della CLI di CloudHSMPer un elenco completo dei comandi, vedere.

Per i vantaggi della migrazione, consultaVantaggi di AWS CloudHSM Client SDK 5.

Consultate i seguenti argomenti per istruzioni dettagliate sulla migrazione da Client SDK 3 a Client SDK 5. L'ultima versione di AWS CloudHSM Client SDK è la 5.16.

- Migra la tua libreria AWS CloudHSM PKCS #11 da Client SDK 3 a Client SDK 5
- Migra il tuo OpenSSL Dynamic Engine AWS CloudHSM da Client SDK 3 a Client SDK 5
- Esegui la migrazione del tuo Key Storage Provider (KSP) da AWS CloudHSM Client SDK 3 a Client SDK 5
- Esegui la migrazione del tuo provider JCE da AWS CloudHSM Client SDK 3 a Client SDK 5

Per funzionalità o casi d'uso non supportati dalla CLI di CloudHSM, contatta. Supporto AWS

Migra la tua libreria AWS CloudHSM PKCS #11 da Client SDK 3 a Client SDK 5

Utilizzate questo argomento per migrare la libreria AWS CloudHSM PKCS #11 da Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi di AWS CloudHSM Client SDK 5

Nel AWS CloudHSM, le applicazioni dei clienti eseguono operazioni crittografiche utilizzando il AWS CloudHSM Client Software Development Kit (SDK). Client SDK 5 è l'SDK principale che continua ad avere nuove funzionalità e supporto per la piattaforma.

Per consultare le istruzioni di migrazione per tutti i provider, consulta. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Preparati affrontando le modifiche più importanti

Rivedi queste modifiche sostanziali e aggiorna di conseguenza l'applicazione nel tuo ambiente di sviluppo.

I meccanismi di avvolgimento sono cambiati

Meccanismo Client SDK 3	Meccanismo Client SDK 5 equivalente
`CKM_AES_KEY_WRAP`	`CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD`

Meccanismo Client SDK 3	Meccanismo Client SDK 5 equivalente
CKM_AES_KEY_WRAP_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD
CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD	CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD

ECDH

In Client SDK 3, è possibile utilizzare ECDH e specificare un KDF. Questa funzionalità non è attualmente disponibile in Client SDK 5. Se la tua applicazione necessita di questa funzionalità, contatta l'assistenza.

Gli handle dei tasti ora sono specifici della sessione

Per utilizzare correttamente gli handle delle chiavi in Client SDK 5, è necessario ottenere gli handle delle chiavi ogni volta che si esegue un'applicazione. Se disponi di applicazioni esistenti che utilizzeranno gli stessi key handle in sessioni diverse, devi modificare il codice per ottenere l'handle di chiave ogni volta che esegui l'applicazione. Per informazioni sul recupero degli handle dei tasti, vedete questo esempio AWS CloudHSM PKCS #11. Questa modifica è conforme alla specifica PKCS #11 2.40.

Esegui la migrazione a Client SDK 5

Segui le istruzioni in questa sezione per migrare da Client SDK 3 a Client SDK 5.

> **Note**
>
> Amazon Linux, Ubuntu 16.04, Ubuntu 18.04, CentOS 6, CentOS 8 e RHEL 6 non sono attualmente supportati con Client SDK 5. Se attualmente utilizzi una di queste piattaforme con Client SDK 3, dovrai scegliere una piattaforma diversa durante la migrazione a Client SDK 5.

1. Disinstalla la libreria PKCS #11 per Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-client-pkcs11
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-client-pkcs11
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-client-pkcs11
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-client-pkcs11
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-pkcs11
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-pkcs11
   ```

2. Arresta il Client Daemon per Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo service cloudhsm-client stop
   ```

   CentOS 7

   ```
   $ sudo service cloudhsm-client stop
   ```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

3. Disinstalla il Client Daemon per Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client
```

CentOS 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

4. Installa la libreria Client SDK PKCS #11 seguendo la procedura riportata di seguito. Installa la libreria PKCS #11 per AWS CloudHSM Client SDK 5
5. Client SDK 5 introduce un nuovo formato di file di configurazione e uno strumento di avvio da riga di comando. Per avviare la libreria Client SDK 5 PKCS #11, segui le istruzioni elencate nella guida per l'utente riportata di seguito. Esegui il bootstrap di Client SDK
6. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Le migliori pratiche per AWS CloudHSM

Migra il tuo OpenSSL Dynamic Engine AWS CloudHSM da Client SDK 3 a Client SDK 5

Usa questo argomento per migrare il tuo OpenSSL Dynamic Engine AWS CloudHSM da Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi di AWS CloudHSM Client SDK 5

Nel AWS CloudHSM, le applicazioni dei clienti eseguono operazioni crittografiche utilizzando il AWS CloudHSM Client Software Development Kit (SDK). Client SDK 5 è l'SDK principale che continua ad avere nuove funzionalità e supporto per la piattaforma.

> **Note**
>
> La generazione di numeri casuali non è attualmente supportata in Client SDK 5 con OpenSSL Dynamic Engine.

Per consultare le istruzioni di migrazione per tutti i provider, consulta. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Esegui la migrazione a Client SDK 5

Segui le istruzioni in questa sezione per migrare da Client SDK 3 a Client SDK 5.

> **Note**
>
> Amazon Linux, Ubuntu 16.04, Ubuntu 18.04, CentOS 6, CentOS 8 e RHEL 6 non sono attualmente supportati con Client SDK 5. Se attualmente utilizzi una di queste piattaforme con Client SDK 3, dovrai scegliere una piattaforma diversa durante la migrazione a Client SDK 5.

1. Disinstalla OpenSSL Dynamic Engine for Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-client-dyn
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-client-dyn
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-client-dyn
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-client-dyn
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-dyn
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-dyn
   ```

2. Arresta il Client Daemon per Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo service cloudhsm-client stop
   ```

   CentOS 7

   ```
   $ sudo service cloudhsm-client stop
   ```

   RHEL 7

   ```
   $ sudo service cloudhsm-client stop
   ```

   RHEL 8

   ```
   $ sudo service cloudhsm-client stop
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo systemctl stop cloudhsm-client
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo systemctl stop cloudhsm-client
   ```

3. Disinstalla il Client Daemon per Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-client
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-client
   ```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

4. Installa il Client SDK OpenSSL Dynamic Engine seguendo i passaggi riportati di seguito. Installa il motore dinamico OpenSSL per AWS CloudHSM Client SDK 5
5. Client SDK 5 introduce un nuovo formato di file di configurazione e uno strumento di avvio da riga di comando. Per avviare il tuo Client SDK 5 OpenSSL Dynamic Engine, segui le istruzioni elencate nella guida per l'utente riportata di seguito. Esegui il bootstrap di Client SDK
6. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Le migliori pratiche per AWS CloudHSM

Esegui la migrazione del tuo Key Storage Provider (KSP) da AWS CloudHSM Client SDK 3 a Client SDK 5

Questo argomento spiega come migrare il Key Storage Provider (KSP) da AWS CloudHSM Client SDK 3 a Client SDK 5. L'ultima versione di AWS CloudHSM Client SDK è la 5.16. Per informazioni sui vantaggi della migrazione, consulta. Vantaggi di AWS CloudHSM Client SDK 5

In AWS CloudHSM, si utilizza il AWS CloudHSM Client Software Development Kit (SDK) per eseguire operazioni crittografiche. Client SDK 5 è l'SDK principale che riceve nuove funzionalità e aggiornamenti del supporto della piattaforma.

Per le istruzioni sulla migrazione per tutti i provider, consulta. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Esegui la migrazione a Client SDK 5

1. Arresta il Client Daemon per Client SDK 3.

```
PS C:\> Stop-Service "AWS CloudHSM Client"
```

2. Installa Client SDK 5 Key Storage Provider (KSP) sulla tua istanza di Windows Server. Per istruzioni, consulta Installa il Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5.
3. Configura il tuo Client SDK 5 Key Storage Provider (KSP) utilizzando il nuovo formato di file di configurazione e lo strumento di avvio da riga di comando. Per istruzioni, consulta Esegui il bootstrap di Client SDK.
4. Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5 include la modalità di SDK3 compatibilità per supportare i file di riferimento chiave generati in. SDK3 Per ulteriori informazioni, consulta SDK3 modalità di compatibilità per Key Storage Provider (KSP) per AWS CloudHSM.

> **Note**
>
> È necessario abilitare la modalità di SDK3 compatibilità quando si utilizzano i file di riferimento chiave generati da Client SDK 3 con Client SDK 5.

Esegui la migrazione a nuove istanze di Windows Server

1. Completa tutti i passaggi descritti in Migrazione a Client SDK 5 sulle tue nuove istanze di Windows Server.
2. Verifica la presenza di file di riferimento chiave esistenti

   Nell'istanza originale di Windows Server, verifica la presenza di file di riferimento chiave in`C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`.

   - Se esistono file di riferimento chiave, copia tutti i contenuti in `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP GlobalPartition` Includi nello stesso percorso di directory sulla nuova istanza di Windows Server. Crea la directory se non esiste.
   - Se i file di riferimento chiave non esistono, utilizzali `cloudhsm-cli key generate-file --encoding ksp-key-reference` sulla nuova istanza di Windows Server per crearli. Per istruzioni, consulta Generazione di riferimenti chiave KSP (Windows).
3. Verifica il certificato principale

   Controlla il tuo certificato principale presso le autorità di certificazione root affidabili:

```
PS C:\Users\Administrator\Desktop> certutil -store Root

Root "Trusted Root Certification Authorities"
================ Certificate 0 ================
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
 NotBefore: 2/5/2020 1:38 PM
 NotAfter: 2/5/2021 1:48 PM
 Issuer: CN=MYRootCA
Signature matches Public Key
Root Certificate: Subject matches Issuer
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

> **Note**
>
> Annota il numero di serie del certificato da utilizzare nel passaggio successivo.

4. Esporta il certificato principale

   Esporta il certificato principale in un file:

   ```
   certutil -store Root certificate-serial-number root-certificate-name.cer
   ```

5. Verifica il certificato HSM-backend

   Controlla il tuo certificato HSM-Backend nel Personal Certificate Store:

   ```
   PS C:\Users\Administrator\Desktop> certutil -store My

   my "Personal"
   ================ Certificate 0 ================
   Serial Number: certificate-serial-number
   Issuer: CN=MYRootCA
    NotBefore: 2/5/2020 1:38 PM
    NotAfter: 2/5/2021 1:48 PM
   Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
    L=Houston, S=Texas, C=US
   Non-root Certificate
   Cert Hash(sha1): cert-hash
     Key Container = key-container-name
     Provider = Cavium Key Storage Provider
   Private key is NOT exportable
   Encryption test passed
   CertUtil: -store command completed successfully.
   ```

   > **Note**
   >
   > Annota il numero di serie del certificato da utilizzare nel passaggio successivo.

6. Esporta il certificato HSM-Backend

   Esporta il certificato HSM-Backend in un file:

   ```
   certutil -store My certificate-serial-number signed-certificate-name.cer
   ```

7. Importa il certificato principale

   Sulla tua nuova istanza di Windows:

1. Copia il file CA principale nella tua nuova istanza di Windows
2. Importa il certificato:

```
certutil -addstore Root root-certificate-name.cer
```

8. Verifica l'installazione del certificato principale

   Conferma che il certificato principale sia installato correttamente:

```
PS C:\Users\Administrator\Desktop> certutil -store Root

Root "Trusted Root Certification Authorities"
================ Certificate 0 ================
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
 NotBefore: 2/5/2020 1:38 PM
 NotAfter: 2/5/2021 1:48 PM
 Issuer: CN=MYRootCA
Signature matches Public Key
Root Certificate: Subject matches Issuer
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

9. Importa il certificato HSM-backend

   Sulla tua nuova istanza di Windows:

   1. Copia il certificato HSM-Backend nella tua nuova istanza di Windows
   2. Importa il certificato:

```
certutil -addstore My signed-certificate-name.cer
```

10. Verifica l'installazione del certificato HSM-backend

    Conferma che il certificato HSM-Backend sia installato correttamente:

```
PS C:\Users\Administrator\Desktop> certutil -store My

my "Personal"
```

```
================ Certificate 0 ================
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
 NotBefore: 2/5/2020 1:38 PM
 NotAfter: 2/5/2021 1:48 PM
Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
 L=Houston, S=Texas, C=US
Non-root Certificate
Cert Hash(sha1): cert-hash
No key provider information
Cannot find the certificate and private key for decryption.
CertUtil: -store command completed successfully.
```

> **Note**
>
> Annota il numero di serie del certificato da utilizzare nei passaggi successivi.

11. Crea un file di riferimento chiave (opzionale)

    Completate questo passaggio solo se dovete creare un nuovo file di riferimento chiave. Altrimenti, passa alla fase successiva.

    > **Note**
    >
    > Questa funzionalità è disponibile solo nella versione SDK 5.16.0 e successive.

    1. Installa OpenSSL ed estrai il modulo:

       ```
       openssl x509 -in signed-certificate-name.cer -modulus -noout
       ```

       > **Note**
       >
       > Il comando OpenSSL restituisce il modulo nel formato:. `Modulus=modulus-value`
       > Nota che deve *`modulus-value`* essere utilizzato nel comando successivo.

    2. Crea un file di riferimento chiave con la CLI di CloudHSM, vedi: Generazione di riferimenti chiave KSP (Windows)

```
& "C:\Program Files\Amazon\CloudHSM\bin\cloudhsm-cli.exe" key generate-
file --encoding ksp-key-reference --filter attr.class=private-key
 attr.modulus=0xmodulus-value
```

> **Note**
>
> Gli *modulus-value* argomenti del comando CLI di CloudHSM devono avere il `0x` prefisso per indicare il formato esadecimale.
> I file di riferimento chiave vengono creati in. `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`

12. Crea una configurazione di riparazione

    Crea un file denominato `repair.txt` con i seguenti contenuti:

```
[Properties]
11 = "" ; Add friendly name property
2 = "{text}" ; Add Key Provider Information property
_continue_="Container=key-container-name&"
_continue_="Provider=Cavium Key Storage Provider&"
_continue_="Flags=0&"
_continue_="KeySpec=2"
```

> **Note**
>
> Sostituisci *key-container-name* con il nome del file di riferimento della chiave da. `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`

13. Ripara l'archivio dei certificati

    Esegui il comando di riparazione:

```
certutil -repairstore My certificate-serial-number repair.txt
```

> **Note**
>
> Il numero di serie del certificato viene ottenuto dai passaggi precedenti durante la verifica dell'installazione del certificato HSM-Backend.

14. Verifica l'associazione dei certificati

    Conferma che il certificato sia associato correttamente:

```
PS C:\Users\Administrator\Desktop> certutil -store My

my "Personal"
================ Certificate 0 ================
Serial Number: certificate-serial-number
Issuer: CN=MYRootCA
 NotBefore: 2/5/2020 1:38 PM
 NotAfter: 2/5/2021 1:48 PM
Subject: CN=www.mydomain.com, OU=Certificate Management, O=Information Technology,
 L=Houston, S=Texas, C=US
Non-root Certificate
Cert Hash(sha1): cert-hash
  Key Container = key-container-name
  Provider = Cavium Key Storage Provider
Private key is NOT exportable
ERROR: Could not verify certificate public key against private key
CertUtil: -store command completed successfully.
```

    Verifica che l'output mostri:

    - Il nome corretto del contenitore delle chiavi
    - Il provider di archiviazione delle chiavi Cavium
    - `ERROR: Could not verify certificate public key against private key`È un problema noto, vedi Problema: la verifica di un archivio di certificati non riesce

15. Testa la tua applicazione

    Prima di completare la migrazione:

    1. Testa la tua applicazione nel tuo ambiente di sviluppo
    2. Aggiorna il codice per risolvere eventuali modifiche sostanziali

3. Per indicazioni specifiche sull'applicazione, consulta Integrazione di applicazioni di terze parti con AWS CloudHSM

Verifica la migrazione

Dopo aver completato i passaggi di migrazione, verifica che:

- I certificati sono installati correttamente negli archivi di certificati corretti
- I file di riferimento chiave sono presenti nella posizione corretta
- L'applicazione può eseguire operazioni crittografiche utilizzando i certificati migrati

Risoluzione dei problemi

Se riscontri problemi durante la migrazione, verifica:

- Tutti i certificati vengono esportati correttamente dal sistema di origine
- I numeri di serie dei certificati corrispondono tra i sistemi
- I nomi dei contenitori delle chiavi nel file repair.txt corrispondono ai file di riferimento delle chiavi
- SDK3 la modalità di compatibilità è abilitata se si utilizzano SDK3 file di riferimento chiave generati

Argomenti correlati

- Le migliori pratiche per AWS CloudHSM

Esegui la migrazione del tuo provider JCE da AWS CloudHSM Client SDK 3 a Client SDK 5

Utilizzate questo argomento per migrare il vostro provider JCE da AWS CloudHSM Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi di AWS CloudHSM Client SDK 5

Nel AWS CloudHSM, le applicazioni dei clienti eseguono operazioni crittografiche utilizzando il AWS CloudHSM Client Software Development Kit (SDK). Client SDK 5 è l'SDK principale che continua ad avere nuove funzionalità e supporto per la piattaforma.

Il provider Client SDK 3 JCE utilizza classi personalizzate APIs che non fanno parte delle specifiche JCE standard. Client SDK 5 per il provider JCE è conforme alla specifica JCE ed è retrocompatibile

con Client SDK 3 in alcune aree. Le applicazioni del cliente potrebbero richiedere modifiche come parte della migrazione a Client SDK 5. Questa sezione descrive le modifiche necessarie per una migrazione corretta.

Per consultare le istruzioni di migrazione per tutti i provider, consultaMigrazione da AWS CloudHSM Client SDK 3 a Client SDK 5.

Argomenti

- Preparati affrontando le modifiche più importanti
- Esegui la migrazione a Client SDK 5
- Argomenti correlati

Preparati affrontando le modifiche più importanti

Rivedi queste modifiche sostanziali e aggiorna di conseguenza l'applicazione nel tuo ambiente di sviluppo.

La classe e il nome del Provider sono cambiati

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Classe e nome del provider	La classe del provider JCE in Client SDK 3 viene chiamata `CaviumProvider` e ha il nome Provider. `Cavium`	In Client SDK 5, la classe Provider viene chiamata `CloudHsmProvider` e ha il nome Provider. `CloudHSM`	Un esempio di come inizializzare l'`CloudHsmProvider` oggetto è disponibile nel repository di AWS CloudHSM GitHub esempio.

L'accesso esplicito è cambiato, quello implicito no

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Login esplicito	Client SDK 3 utilizza la `LoginManager` classe per l'accesso esplicito. [1]	In Client SDK 5, il `CloudHSM` provider implementa l'accesso `AuthProvider` esplicito. `AuthProvider` è una classe Java standard e segue il modo idiomatico di Java per accedere a un provider. Grazie alla migliore gestione dello stato di accesso in Client SDK 5, le applicazioni non devono più monitorare ed eseguire l'accesso durante le riconness ioni. [2]	Per un esempio su come utilizzare l'accesso esplicito con Client SDK 5, consulta l' LoginRunn er esempio nel repository di esempio di AWS GitHub CloudHSM.
Accesso implicito	Non sono richieste modifiche per l'accesso implicito. Lo stesso file di proprietà e tutte le variabili di ambiente continueranno a funzionar e per l'accesso implicito durante la migrazione da Client SDK 3 a Client SDK 5.		Per un esempio su come utilizzare l'accesso implicito con Client SDK 5, consulta l'LoginRu nner esempio nel repository di esempio. AWS CloudHSM GitHub

- [1] Frammento di codice Client SDK 3:

```
LoginManager lm = LoginManager.getInstance();

lm.login(partition, user, pass);
```

- [2] Frammento di codice Client SDK 5:

```
// Construct or get the existing provider object
AuthProvider provider = new CloudHsmProvider();

// Call login method on the CloudHsmProvider object
// Here loginHandler is a CallbackHandler
provider.login(null, loginHandler);
```

Per un esempio su come utilizzare l'accesso esplicito con Client SDK 5, consulta l'LoginRunner esempio nel repository di esempio. AWS CloudHSM GitHub

La generazione delle chiavi è cambiata

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Generazione delle chiavi	In Client SDK 3, `Cavium[Key-type]AlgorithmParameterSpec` viene utilizzato per specificare i parametri di generazione delle chiavi. Per un frammento di codice, consulta la nota a piè di pagina. 1	In Client SDK 5, `KeyAttributesMap` viene utilizzato per specificare gli attributi di generazione delle chiavi. Per un frammento di codice, consulta la nota a piè di pagina. 2	Per un esempio su come generare una chiave simmetrica, consulta l'`KeyAttributesMap` esempio nel repository di SymmetricKeys esempio di AWS GitHub CloudHSM.
Generazione di coppie di chiavi	In Client SDK 3, `Cavium[Key-type]Al`	In Client SDK 5, `KeyPairAttributesMap`	Per un esempio su come `KeyAttributesMap` generare

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
	`gorithmpa rameterSpec` viene utilizzato per specificare i parametri di generazione delle key pair. Per un frammento di codice, consulta la nota a piè di pagina. 3	viene utilizzato per specificare questi parametri. Per un frammento di codice, consulta la nota a piè di pagina. 4	una chiave asimmetrica, consultate l'esempio nel AsymmetricKeys repository di esempio. AWS CloudHSM GitHub

- [1] Frammento di codice per la generazione di chiavi Client SDK 3:

```
KeyGenerator keyGen = KeyGenerator.getInstance("AES", "Cavium");
CaviumAESKeyGenParameterSpec aesSpec = new CaviumAESKeyGenParameterSpec(
keySizeInBits,
keyLabel,
isExtractable,
isPersistent);
keyGen.init(aesSpec);
SecretKey aesKey = keyGen.generateKey();
```

- [2] Frammento di codice per la generazione di chiavi Client SDK 5:

```
KeyGenerator keyGen = KeyGenerator.getInstance("AES",
CloudHsmProvider.PROVIDER_NAME);

final KeyAttributesMap aesSpec = new KeyAttributesMap();
aesSpec.put(KeyAttribute.LABEL, keyLabel);
aesSpec.put(KeyAttribute.SIZE, keySizeInBits);
aesSpec.put(KeyAttribute.EXTRACTABLE, isExtractable);
aesSpec.put(KeyAttribute.TOKEN, isPersistent);

keyGen.init(aesSpec);
SecretKey aesKey = keyGen.generateKey();
```

- [3] Frammento di codice di generazione di key pair Client SDK 3:

```
KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("rsa", "Cavium");
CaviumRSAKeyGenParameterSpec spec = new CaviumRSAKeyGenParameterSpec(
keySizeInBits,
new BigInteger("65537"),
label + ":public",
label + ":private",
isExtractable,
isPersistent);

keyPairGen.initialize(spec);

keyPairGen.generateKeyPair();
```

- [4] Frammento di codice di generazione di key pair Client SDK 5:

```
KeyPairGenerator keyPairGen =
KeyPairGenerator.getInstance("RSA", providerName);

// Set attributes for RSA public key
final KeyAttributesMap publicKeyAttrsMap = new KeyAttributesMap();
publicKeyAttrsMap.putAll(additionalPublicKeyAttributes);
publicKeyAttrsMap.put(KeyAttribute.LABEL, label + ":Public");
publicKeyAttrsMap.put(KeyAttribute.MODULUS_BITS, keySizeInBits);
publicKeyAttrsMap.put(KeyAttribute.PUBLIC_EXPONENT,
new BigInteger("65537").toByteArray());

// Set attributes for RSA private key
final KeyAttributesMap privateKeyAttrsMap = new KeyAttributesMap();
privateKeyAttrsMap.putAll(additionalPrivateKeyAttributes);
privateKeyAttrsMap.put(KeyAttribute.LABEL, label + ":Private");

// Create KeyPairAttributesMap and use that to initialize the
// keyPair generator
KeyPairAttributesMap keyPairSpec =
new KeyPairAttributesMapBuilder()
.withPublic(publicKeyAttrsMap)
.withPrivate(privateKeyAttrsMap)
.build();

keyPairGen.initialize(keyPairSpec);
keyPairGen.generateKeyPair();
```

La ricerca, l'eliminazione e il riferimento alle chiavi sono cambiati

La ricerca di una chiave già generata con AWS CloudHSM comporta l'utilizzo di. KeyStore Client SDK 3 è di due KeyStore tipi: `Cavium` e. `CloudHSM` Client SDK 5 ha solo un KeyStore tipo:. `CloudHSM`

Il passaggio da `Cavium` KeyStore a `CloudHSM` KeyStore richiede un cambio di KeyStore tipo. Inoltre, Client SDK 3 utilizza le maniglie dei tasti per fare riferimento alle chiavi, mentre Client SDK 5 utilizza le etichette delle chiavi. Le modifiche comportamentali risultanti sono elencate di seguito.

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Riferimenti chiave	Con Client SDK 3, le applicazioni utilizzano etichette o maniglie di tasti per fare riferimento alle chiavi nell'HSM. Utilizzano etichette KeyStore per trovare una chiave oppure usano maniglie per creare `CaviumKey` oggetti.	In Client SDK 5, le applicazioni possono utilizzare il comando AWS CloudHSM KeyStore Classe Java per Client SDK 5 per trovare le chiavi per etichetta. Per trovare le chiavi in base alla maniglia, usa il comando AWS CloudHSM `KeyStoreWithAttributes` with AWS CloudHSM `KeyReferenceSpec` .	
Ricerca di voci multiple	Quando si cerca una chiave utilizzando `getEntry` o `getCertificate` in scenari in cui sono presenti più elementi con gli stessi criteri	Con AWS CloudHSM `KeyStore` and`KeyStoreWithAttributes` , questo stesso scenario comporterà la generazione di	

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
	in `Cavium` KeyStore, verrà restituita solo la prima voce trovata. `getKey`	un'eccezione. Per risolvere questo problema, si consiglia di impostare etichette univoche per le chiavi utilizzando il Imposta gli attributi delle chiavi con CloudhSM CLI comando nella CLI di CloudHSM. Oppure usa `KeyStoreWithAttributes#getKeys` per restituire tutte le chiavi che corrispondono ai criteri.	

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Trova tutte le chiavi	In Client SDK 3 è possibile trovare tutte le chiavi nell'HSM utilizzando. `Util.findAllKeys()`	Client SDK 5 rende la ricerca delle chiavi più semplice ed efficiente utilizzando la classe. `KeyStoreWithAttributes` Quando possibile, memorizza nella cache le chiavi per ridurre al minimo la latenza. Per ulteriori informazioni, consulta Gestisci in modo efficace le chiavi nell'applicazione. Se devi recuperare tutte le chiavi dall'HSM, usale `KeyStoreWithAttributes#getKeys` con una chiave vuota. `KeyAttributesMap`	Un esempio che utilizza la `KeyStoreWithAttributes` classe per trovare una chiave è disponibile nel repository di AWS CloudHSM GitHub esempio e un frammento di codice è mostrato in. 1

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Eliminazione della chiave	Client SDK 3 utilizza `Util.deleteKey()` per eliminare una chiave.	L'`Key`oggetto in Client SDK 5 implementa l'`Destroyable` interfaccia che consente di eliminare le chiavi utilizzando il `destroy()` metodo di questa interfaccia.	Un codice di esempio che mostra la funzionalità di eliminazione delle chiavi è disponibile nel repository di esempio GitHub CloudHSM. Un frammento di esempio per ogni SDK è mostrato in. 2

- [1] uno snippet è mostrato di seguito:

```
KeyAttributesMap findSpec = new KeyAttributesMap();
findSpec.put(KeyAttribute.LABEL, label);
findSpec.put(KeyAttribute.KEY_TYPE, keyType);
KeyStoreWithAttributes keyStore = KeyStoreWithAttributes.getInstance("CloudHSM");

keyStore.load(null, null);
keyStore.getKey(findSpec);
```

- [2] Eliminazione di una chiave in Client SDK 3:

```
Util.deleteKey(key);
```

  Eliminazione di una chiave in Client SDK 5:

```
((Destroyable) key).destroy();
```

Le operazioni di cancellazione di cifratura sono cambiate, le altre operazioni di cifratura no

> **Note**
>
> Non sono necessarie modifiche per le operazioni di cifratura. encrypt/decrypt/wrap

Le operazioni Unwrap richiedono la sostituzione della `CaviumUnwrapParameterSpec` classe Client SDK 3 con una delle seguenti classi specifiche per le operazioni crittografiche elencate.

- `GCMUnwrapKeySpec`per unwrap `AES/GCM/NoPadding`
- `IvUnwrapKeySpec`per `AESWrap unwrap` e `AES/CBC/NoPadding unwrap`
- `OAEPUnwrapKeySpec` per `RSA OAEP unwrap`

Frammento di esempio per: `OAEPUnwrapkeySpec`

```
OAEPParameterSpec oaepParameterSpec =
new OAEPParameterSpec(
        "SHA-256",
        "MGF1",
        MGF1ParameterSpec.SHA256,
        PSpecified.DEFAULT);

KeyAttributesMap keyAttributesMap =
        new KeyAttributesMap(KeyAttributePermissiveProfile.KEY_CREATION);
keyAttributesMap.put(KeyAttribute.TOKEN, true);
keyAttributesMap.put(KeyAttribute.EXTRACTABLE, false);

OAEPUnwrapKeySpec spec = new OAEPUnwrapKeySpec(oaepParameterSpec,
        keyAttributesMap);

Cipher hsmCipher =
        Cipher.getInstance(
                "RSA/ECB/OAEPPadding",
                CloudHsmProvider.PROVIDER_NAME);
hsmCipher.init(Cipher.UNWRAP_MODE, key, spec);
```

Le operazioni di firma non sono cambiate

Non sono necessarie modifiche per le operazioni di firma.

Esegui la migrazione a Client SDK 5

Segui le istruzioni in questa sezione per migrare da Client SDK 3 a Client SDK 5.

> **Note**
>
> Amazon Linux, Ubuntu 16.04, Ubuntu 18.04 CentOS 6, CentOS 8 e RHEL 6 non sono attualmente supportati con Client SDK 5. Se attualmente utilizzi una di queste piattaforme con Client SDK 3, dovrai scegliere una piattaforma diversa durante la migrazione a Client SDK 5.

1. Disinstalla il provider JCE per Client SDK 3.

   Amazon Linux 2

   ```
   $ sudo yum remove cloudhsm-client-jce
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-client-jce
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-client-jce
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-client-jce
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-jce
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo apt remove cloudhsm-client-jce
   ```

2. Arresta il Client Daemon per Client SDK 3.

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl stop cloudhsm-client
```

3. Disinstalla il Client Daemon per Client SDK 3.

Amazon Linux 2

```
$ sudo yum remove cloudhsm-client
```

CentOS 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 7

```
$ sudo yum remove cloudhsm-client
```

RHEL 8

```
$ sudo yum remove cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo apt remove cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo apt remove cloudhsm-client
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

4. Installa il provider Client SDK JCE seguendo la procedura riportata di seguito. Installa il provider JCE per AWS CloudHSM Client SDK 5
5. Client SDK 5 introduce un nuovo formato di file di configurazione e uno strumento di avvio da riga di comando. Per avviare il provider Client SDK 5 JCE, segui le istruzioni elencate nella guida per l'utente riportata di seguito. Esegui il bootstrap di Client SDK
6. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Le migliori pratiche per AWS CloudHSM

Utilizzo di Client SDK 5 per lavorare con AWS CloudHSM

AWS CloudHSM include due versioni principali di Client SDK:

- Client SDK 5: questo è il nostro Client SDK più recente e quello predefinito. Per informazioni sui benefici e i vantaggi che offre, vedi Vantaggi di AWS CloudHSM Client SDK 5.

- Client SDK 3: Questo è il nostro vecchio Client SDK. Include un set completo di componenti per la compatibilità delle applicazioni basate su piattaforme e linguaggi e strumenti di gestione.

Per istruzioni sulla migrazione da Client SDK 3 a Client SDK 5, vedere. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Questo argomento descrive Client SDK 5. Per verificare quale versione di Client SDK stai utilizzando, consulta

Argomenti

- Vantaggi di AWS CloudHSM Client SDK 5
- AWS CloudHSM Piattaforme supportate da Client SDK 5
- Libreria PKCS #11 per AWS CloudHSM Client SDK 5
- OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5
- Provider OpenSSL per AWS CloudHSM Client SDK 5
- Provider di archiviazione delle chiavi (KSP) per AWS CloudHSM Client SDK 5
- Provider JCE per AWS CloudHSM Client SDK 5

Vantaggi di AWS CloudHSM Client SDK 5

Rispetto a AWS CloudHSM Client SDK 3, Client SDK 5 è più facile da gestire, offre una configurabilità superiore e una maggiore affidabilità. Il Client SDK 5 offre inoltre alcuni vantaggi chiave aggiuntivi al Client SDK 3.

Progettato per l'architettura serverless

Il Client SDK 5 non richiede un client daemon, quindi devi più gestire un servizio in background. Questo aiuta gli utenti in alcuni modi importanti:

- Semplifica il processo di avvio dell'applicazione. Tutto ciò che devi fare per iniziare a usare CloudHSM è configurare l'SDK prima di eseguire l'applicazione.
- Non è necessario un processo in esecuzione costante, il che semplifica l'integrazione con componenti serverless come Lambda ed Elastic Container Service (ECS).

Integrazioni migliori con terze parti e portabilità semplificata

Il Client SDK 5 segue da vicino le specifiche JCE e offre una portabilità più semplice tra diversi provider JCE e integrazioni migliori con terze parti

Esperienza utente e configurabilità migliorate

Il Client SDK 5 migliora la leggibilità dei messaggi di log e fornisce eccezioni e meccanismi di gestione degli errori più chiari, il che semplifica notevolmente l'autodiagnostica per gli utenti. SDK 5 offre anche una varietà di configurazioni, elencate nella pagina Configurazione dello strumento.

Supporto più ampio per la piattaforma

Il Client SDK 5 offre un maggiore supporto per le moderne piattaforme operative. Ciò include il supporto per le tecnologie ARM e un maggiore supporto per JCE, PKCS #11 e OpenSSL. Per ulteriori informazioni, consulta Piattaforme supportate.

IPv6 supporto per la connessione

Client SDK 5.14+ supporta connessioni a dual-stack utilizzando. HSMs IPv6

Funzionalità e meccanismi aggiuntivi

Il Client SDK 5 include funzionalità e meccanismi aggiuntivi che non sono disponibili nel Client SDK 3 e il Client SDK 5 continuerà ad aggiungere altri meccanismi in futuro.

AWS CloudHSM Piattaforme supportate da Client SDK 5

Il supporto di base è diverso per ogni versione di Client SDK. AWS CloudHSM Il supporto della piattaforma per i componenti di un SDK in genere corrisponde al supporto di base, ma non sempre. Per determinare il supporto della piattaforma per un determinato componente, assicurati innanzitutto che la piattaforma desiderata appaia nella sezione base dell'SDK, quindi controlla eventuali esclusioni o altre informazioni pertinenti nella sezione dei componenti.

AWS CloudHSM supporta solo sistemi operativi a 64 bit.

Le piattaforme supportate cambiano nel tempo. Le versioni precedenti del CloudHSM Client SDK potrebbero non supportare tutti i sistemi operativi elencati qui. Verifica se il sistema operativo

supporta le versioni precedenti del Client SDK di CloudHSM consultando le note di rilascio. Per ulteriori informazioni, vedi Download per AWS CloudHSM Client SDK.

Per le piattaforme supportate per il precedente Client SDK, vedi AWS CloudHSM Piattaforme supportate da Client SDK 3

Client SDK 5 non richiede un client daemon.

Argomenti

- Supporto Linux per AWS CloudHSM Client SDK 5
- Supporto Windows per AWS CloudHSM Client SDK 5
- Supporto serverless per AWS CloudHSM Client SDK 5
- Compatibilità HSM per AWS CloudHSM Client SDK 5

Supporto Linux per AWS CloudHSM Client SDK 5

AWS CloudHSM Client SDK 5 supporta i seguenti sistemi operativi e piattaforme Linux.

Piattaforme supportate	Architettura: x86_64	Architettura ARM
Amazon Linux 2	Sì	Sì
Amazon Linux 2023	Sì	Sì
Red Hat Enterprise Linux 8 (8.3+)	Sì	Sì
Red Hat Enterprise Linux 9 (9.2+)	Sì	Sì
Red Hat Enterprise Linux 10 (versione 10.0+)	Sì	Sì
Ubuntu 22.04 LTS	Sì	Sì
Ubuntu 24.04 LTS	Sì	Sì
Ubuntu 26.04 LTS	Sì	Sì

- SDK 5.16 è stata l'ultima versione a fornire il supporto per la piattaforma Ubuntu 20.04 LTS. Per ulteriori informazioni, consulta il sito Web di Ubuntu. https://ubuntu.com/blog/ubuntu-20-04-lts-end-of-life-standard-support-is-coming-to-an-end-heres-how-to-prepare
- SDK 5.12 è stata l'ultima versione a fornire il supporto per la piattaforma CentOS 7 (7.8+). Per ulteriori informazioni, vedi il sito web CentOS.
- SDK 5.12 è stata l'ultima versione a fornire il supporto per la piattaforma Red Hat Enterprise Linux 7 (7.8+). Per ulteriori informazioni, consultate il sito Web di Red Hat.
- SDK 5.4.2 è stata l'ultima versione a fornire il supporto per la piattaforma CentOS 8. Per ulteriori informazioni, vedi il sito web CentOS.

Supporto Windows per AWS CloudHSM Client SDK 5

AWS CloudHSM Client SDK 5 supporta le seguenti versioni di Windows Server.

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019
- Microsoft Windows Server 2022
- Microsoft Windows Server 2025

Supporto serverless per AWS CloudHSM Client SDK 5

AWS CloudHSM Client SDK 5 supporta i seguenti servizi AWS serverless.

- AWS Lambda
- Docker/ECS

Compatibilità HSM per AWS CloudHSM Client SDK 5

La tabella seguente descrive la compatibilità di AWS CloudHSM Client SDK 5 per gli HSM.

hsm1.medium	hsm2m. medio
Compatibile con Client SDK versione 5.0.0 e successive.	Compatibile con Client SDK versione 5.9.0 e successive.

Libreria PKCS #11 per AWS CloudHSM Client SDK 5

PKCS #11 è uno standard per l'esecuzione di operazioni di crittografia su moduli di sicurezza hardware (HSM). AWS CloudHSM offre implementazioni della libreria PKCS #11 conformi a PKCS #11 versione 2.40.

Per informazioni sul processo di bootstrap, consulta la pagina Connessione al cluster. Per la risoluzione dei problemi, consulta. Problemi noti della libreria PKCS #11 per AWS CloudHSM

Per informazioni sull'utilizzo di Client SDK 3, consulta la pagina Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM.

Argomenti

- Installa la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Effettua l'autenticazione alla libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Tipi di chiave supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Meccanismi supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Operazioni API supportate per la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Attributi chiave nella libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Esempi di codice per la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Configurazioni avanzate per la libreria PKCS #11 per AWS CloudHSM
- Archiviazione dei certificati con la libreria PKCS #11

Installa la libreria PKCS #11 per AWS CloudHSM Client SDK 5

Questo argomento fornisce le istruzioni per l'installazione della versione più recente della libreria PKCS #11 per la serie di versioni AWS CloudHSM Client SDK 5. Per ulteriori informazioni sull'SDK del client o sulla libreria PKCS #11, consulta la pagina sull'utilizzo dell'SDK del client e la pagina sulla libreria PKCS #11.

Con Client SDK 5, non è necessario installare o eseguire un daemon del client.

Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Per ulteriori informazioni sulla libreria PKCS #11 in Client SDK 5, consulta la pagina sulla libreria PKCS #11.

> **Note**
>
> Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Come installare e configurare la libreria PKCS #11

1. Utilizza i seguenti comandi per scaricare e installare la libreria PKCS #11.

   Amazon Linux 2023

   Installa la libreria PKCS #11 per Amazon Linux 2023 sull'architettura X86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-pkcs11-latest.amzn2023.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-pkcs11-latest.amzn2023.x86_64.rpm
   ```

   Installa la libreria PKCS #11 per Amazon Linux 2023 sull'architettura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-pkcs11-latest.amzn2023.aarch64.rpm
   ```

   Amazon Linux 2

   Installa la libreria PKCS #11 per Amazon Linux 2 sull'architettura X86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-
   pkcs11-latest.el7.x86_64.rpm
   ```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el7.x86_64.rpm
```

Installa la libreria PKCS #11 per Amazon Linux 2 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-pkcs11-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

Installa la libreria PKCS #11 per RHEL 10 su architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-pkcs11-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el10.x86_64.rpm
```

Installa la libreria PKCS #11 per RHEL 10 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/cloudhsm-pkcs11-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

Installa la libreria PKCS #11 per RHEL 9 su architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-pkcs11-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el9.x86_64.rpm
```

Installa la libreria PKCS #11 per RHEL 9 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Installa la libreria PKCS #11 per RHEL 8 su architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el8.x86_64.rpm
```

Installa la libreria PKCS #11 per RHEL 8 su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-pkcs11-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-pkcs11-latest.el8.aarch64.rpm
```

Ubuntu 26.04 LTS

Installa la libreria PKCS #11 per Ubuntu 26.04 LTS su architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-pkcs11_latest_u26.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u26.04_amd64.deb
```

Installa la libreria PKCS #11 per Ubuntu 26.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-pkcs11_latest_u26.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u26.04_arm64.deb
```

Ubuntu 24.04 LTS

Installa la libreria PKCS #11 per Ubuntu 24.04 LTS su architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-pkcs11_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u24.04_amd64.deb
```

Installa la libreria PKCS #11 per Ubuntu 24.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-pkcs11_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Installa la libreria PKCS #11 per Ubuntu 22.04 LTS sull'architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-pkcs11_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u22.04_amd64.deb
```

Installa la libreria PKCS #11 per Ubuntu 22.04 LTS su architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-pkcs11_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u22.04_arm64.deb
```

Windows Server

Installa la libreria PKCS #11 per Windows Server sull'architettura X86_64:

1. Scarica la libreria PKCS #11 per Client SDK 5.
2. Esegui il programma di installazione della libreria PKCS #11 () con privilegi amministrativi di Windows AWSCloudHSMPKCS11-latest.msi.

2. Utilizza lo strumento di configurazione per specificare la posizione del certificato emittente. Per istruzioni, consulta Specifica la posizione del certificato di emissione.
3. Per connetterti al cluster, consulta la pagina Esegui il bootstrap di Client SDK.
4. Puoi trovare i file della libreria PKCS #11 nelle seguenti posizioni:

   - File binari, script di configurazione e file di log Linux:

     ```
     /opt/cloudhsm
     ```

     File binari Windows:

     ```
     C:\Program Files\Amazon\CloudHSM
     ```

     Script di configurazione e file di log Windows:

     ```
     C:\ProgramData\Amazon\CloudHSM
     ```

Effettua l'autenticazione alla libreria PKCS #11 per AWS CloudHSM Client SDK 5

Quando si utilizza la libreria PKCS #11, l'applicazione viene eseguita come un particolare utente crittografico (CU) negli HSM in. AWS CloudHSM L'applicazione è in grado di visualizzare e gestire solo le chiavi di proprietà e condivise dall'utente di crittografia. È possibile usare un utente di crittografia esistente nei moduli HSM o creare un nuovo utente di crittografia per l'applicazione. Per informazioni su come gestire gli CU, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM e la pagina sulla gestione degli utenti HSM con CloudHSM Management Utility (CMU)

Per specificare il CU nella libreria PKCS #11, utilizza il parametro pin della funzione C_Login di PKCS #11. Per AWS CloudHSM, il parametro pin ha il seguente formato:

```
<CU_user_name>:<password>
```

Ad esempio, il comando seguente imposta il pin della libreria PKCS #11 sul CU con il nome utente `CryptoUser` e la password `CUPassword123!`.

```
CryptoUser:CUPassword123!
```

Tipi di chiave supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

La libreria PKCS #11 per AWS CloudHSM Client SDK 5 supporta i seguenti tipi di chiave.

Tipo di chiavi	Description
AES	Genera chiavi AES a 128, 192 e 256 bit.
Triple DES (3DES, DESede)	Genera chiavi Triple DES a 192 bit. Vedi la nota 1 di seguito per una modifica imminente.
EC	Genera chiavi con le curve secp224r1 (P-224), secp256r1 (), secp256k1 (Blockchain), secp384r1 (P-256), secp521r1 () ed ed25519. P-384 P-521 [2]
ML-DSA	Genera coppie ML-DSA-65 di ML-DSA [3] chiavi con set di parametri, e. ML-DSA-44 ML-DSA-87
GENERIC_SECRET	Genera segreti generici da 1 a 800 byte.
RSA	General chiavi RSA da 2048-bit a 4096-bit, con incrementi di 256 bit.

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

[2] Supportato solo su istanze hsm2m.medium in modalità non FIPS.

[3] A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

Meccanismi supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

La libreria PKCS #11 è conforme alla versione 2.40 della specifica PKCS #11. Per richiamare una funzione di crittografia utilizzando PKCS #11, chiamare una funzione con un determinato meccanismo. Le sezioni seguenti riassumono le combinazioni di funzioni e meccanismi supportati da AWS CloudHSM Client SDK 5.

La libreria PKCS #11 supporta i seguenti algoritmi:

- Crittografia e decrittografia: AES-CBC,, AES-CTR, AES-ECB, AES-GCM, e DES3-CBC DES3-ECB RSA-OAEP RSA-PKCS
- Firma e verifica: RSA, HMAC, ECDSA (con e senza hashing), EdDSA (Ed25519 e Ed25519ph) e (con modalità mu pura ed esterna) ML-DSA
- Hash/digest— SHA1, SHA224, SHA256, SHA384 e SHA512
- Key wrap: AES Key Wrap,, e [1] AES-GCM RSA-AES RSA-OAEP
- Derivazione della chiave: SP800-108 contatore KDF ed ECDH con KDF (gli algoritmi KDF supportati sono X9.63 con SHA1, SHA224, SHA256, SHA384, SHA512)

Argomenti

- Funzioni di generazione chiavi e coppie di chiavi
- Funzioni di firma e verifica
- Funzioni Sign recover e Verify recover
- Funzioni di digest
- Funzioni di crittografia e decrittografia
- Funzioni di derivazione della chiave
- Funzioni di wrapping e annullamento del wrapping
- Dimensione massima dei dati per ogni meccanismo
- Annotazioni sui meccanismi

Funzioni di generazione chiavi e coppie di chiavi

La libreria AWS CloudHSM software per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni Generate Key e Key Pair.

- `CKM_RSA_PKCS_KEY_PAIR_GEN`
- `CKM_RSA_X9_31_KEY_PAIR_GEN`: il funzionamento di questo meccanismo è identico a quello del meccanismo `CKM_RSA_PKCS_KEY_PAIR_GEN`, ma offre maggiori garanzie per la generazione di p e q.
- `CKM_EC_KEY_PAIR_GEN`
- `CKM_CLOUDHSM_ML_DSA_KEY_PAIR_GEN`[3]
- `CKM_GENERIC_SECRET_KEY_GEN`
- `CKM_AES_KEY_GEN`
- `CKM_DES3_KEY_GEN`: modifica imminente indicata nella nota a piè di pagina 5.

Funzioni di firma e verifica

La libreria AWS CloudHSM software per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni Sign and Verify. Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non ci sono limiti alla dimensione dei dati che l'SDK può sottoporre a hash.

Con gli algoritmi RSA e ECDSA di Client SDK 5, l'hashing viene effettuato in locale, quindi non ci sono limiti di dati. Con HMAC e PureEd DSA (Ed25519), c'è un limite di dati. Per maggiori informazioni, consulta la nota a piè di pagina 2.

RSA

- `CKM_RSA_X_509`
- `CKM_RSA_PKCS`: solo operazioni a parte singola.
- `CKM_RSA_PKCS_PSS`: solo operazioni a parte singola.
- `CKM_SHA1_RSA_PKCS`
- `CKM_SHA224_RSA_PKCS`
- `CKM_SHA256_RSA_PKCS`
- `CKM_SHA384_RSA_PKCS`
- `CKM_SHA512_RSA_PKCS`

- CKM_SHA512_RSA_PKCS
- CKM_SHA1_RSA_PKCS_PSS
- CKM_SHA224_RSA_PKCS_PSS
- CKM_SHA256_RSA_PKCS_PSS
- CKM_SHA384_RSA_PKCS_PSS
- CKM_SHA512_RSA_PKCS_PSS

ECDSA

- CKM_ECDSA: solo operazioni a parte singola.
- CKM_ECDSA_SHA1
- CKM_ECDSA_SHA224
- CKM_ECDSA_SHA256
- CKM_ECDSA_SHA384
- CKM_ECDSA_SHA512

ML-DSA

- CKM_CLOUDHSM_ML_DSA_KEY_PAIR_GEN[3]— Genera coppie di chiavi. ML-DSA
- CKM_CLOUDHSM_ML_DSA[3]— ML-DSA Firma e verifica puramente.
- CKM_CLOUDHSM_EXT_MU_ML_DSA[3]— ML-DSA firma e verifica con mu esterno.

EdDSA

- CKM_CLOUDHSM_EDDSA[6, 7] — con parametri NULL (PureEdDSA), solo operazioni in una sola parte.

HMAC

- CKM_SHA_1_HMAC[2]
- CKM_SHA224_HMAC[2]
- CKM_SHA256_HMAC[2]

- CKM_SHA384_HMAC[2]
- CKM_SHA512_HMAC[2]

CMAC

- CKM_AES_CMAC

Funzioni Sign recover e Verify recover

Client SDK 5 non supporta le funzioni Sign Recover e Verify Recover.

Funzioni di digest

La libreria AWS CloudHSM software per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni Digest. Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non ci sono limiti alla dimensione dei dati che l'SDK può sottoporre a hash.

- CKM_SHA_1
- CKM_SHA224
- CKM_SHA256
- CKM_SHA384
- CKM_SHA512

Funzioni di crittografia e decrittografia

La libreria AWS CloudHSM software per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni Encrypt e Decrypt.

- CKM_RSA_X_509
- CKM_RSA_PKCS: solo operazioni a parte singola. Modifica imminente indicata nella nota a piè di pagina 5.
- CKM_RSA_PKCS_OAEP: solo operazioni a parte singola.
- CKM_AES_ECB
- CKM_AES_CTR
- CKM_AES_CBC
- CKM_AES_CBC_PAD

- `CKM_DES3_CBC`: modifica imminente indicata nella nota a piè di pagina 5.
- `CKM_DES3_ECB`: modifica imminente indicata nella nota a piè di pagina 5.
- `CKM_DES3_CBC_PAD`: modifica imminente indicata nella nota a piè di pagina 5.
- `CKM_AES_GCM` [1, 2]
- `CKM_CLOUDHSM_AES_GCM`[3]

Funzioni di derivazione della chiave

La libreria AWS CloudHSM software per la libreria PKCS #11 supporta i seguenti meccanismi di derivazione delle chiavi:

- `CKM_SP800_108_COUNTER_KDF`
- `CKM_ECDH1_DERIVE`- Supporta la derivazione delle chiavi ECDH con i seguenti tipi di KDF definiti dal fornitore: [6]
  - `CKD_CLOUDHSM_X963_SHA1_KDF` X9.63 - KDF con SHA1 [8]
  - `CKD_CLOUDHSM_X963_SHA224_KDF`- KDF con SHA224 X9.63 [8]
  - `CKD_CLOUDHSM_X963_SHA256_KDF`- KDF con SHA256 X9.63 [8]
  - `CKD_CLOUDHSM_X963_SHA384_KDF`- KDF con SHA384 X9.63 [8]
  - `CKD_CLOUDHSM_X963_SHA512_KDF`- X9.63 KDF con SHA512 [8]

Funzioni di wrapping e annullamento del wrapping

La libreria AWS CloudHSM software per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni Wrap e Unwrap.

Per ulteriori informazioni sul wrapping delle chiavi AES, consulta la pagina sul wrapping delle chiavi AES.

- `CKM_RSA_PKCS`: solo operazioni a parte singola. Una modifica imminente è indicata nella nota a piè di pagina 5.
- `CKM_RSA_PKCS_OAEP`[4]
- `CKM_AES_GCM`[1, 3]
- `CKM_CLOUDHSM_AES_GCM`[3]

- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD[3]
- CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD[3]
- CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD[3]

Dimensione massima dei dati per ogni meccanismo

La tabella seguente elenca le dimensioni massime dei dati per ciascun meccanismo:

Dimensione massima del set di dati

Meccanismo	Dimensione massima dei dati in byte
CKM_SHA_1_HMAC	16288
CKM_SHA224_HMAC	16256
CKM_SHA256_HMAC	16288
CKM_SHA384_HMAC	16224
CKM_SHA512_HMAC	16224
CKM_AES_CBC	16272
CKM_AES_GCM	16224
CKM_CLOUDHSM_AES_GCM	16224
CKM_DES3_CBC	16280
CKM_CLOUDHSM_EDDSA (parametri PureEd DSA, NULL)	16000

Annotazioni sui meccanismi

- [1] Quando si esegue la AES-GCM crittografia, l'HSM non accetta i dati del vettore di inizializzazione (IV) dall'applicazione. È necessario utilizzare un IV generato dall'HSM. L'IV da 12 byte fornito dall'HSM viene scritto nel riferimento della memoria indicato dall'elemento pIV della

struttura di parametri CK_GCM_PARAMS fornita dall'utente. Per evitare confusione tra gli utenti, PKCS #11 SDK nella versione 1.1.1 e successive garantisce che PiV punti a un buffer azzerato quando la crittografia viene inizializzata. AES-GCM

- [2] Quando si opera sui dati utilizzando uno dei seguenti meccanismi, se il buffer dati supera la dimensione massima dei dati, l'operazione genera un errore. Per questi meccanismi, tutta l'elaborazione dei dati deve avvenire all'interno dell'HSM. Per informazioni sui set di dimensioni massime dei dati per ciascun meccanismo, fare riferimento a. Dimensione massima dei dati per ogni meccanismo
- Vendor-defined meccanismo [3]. Per utilizzare i meccanismi definiti dal fornitore CloudHSM, le applicazioni PKCS #11 devono includere /opt/cloudhsm/include/pkcs11/pkcs11t.h durante la compilazione.

  CKM_CLOUDHSM_AES_GCM: questo meccanismo proprietario è un'alternativa programmaticamente più sicura allo standard CKM_AES_GCM. Antepone il IV generato dall'HSM al testo cifrato invece di scriverlo nuovamente nella struttura CK_GCM_PARAMS fornita durante l'inizializzazione del codice. È possibile utilizzare questo meccanismo con le funzioni C_Encrypt, C_WrapKey, C_Decrypt e C_UnwrapKey. Quando si utilizza questo meccanismo, la variabile pIV nella struttura CK_GCM_PARAMS deve essere impostata su NULL. Quando si utilizza questo meccanismo con C_Decrypt e C_UnwrapKey, il IV dovrebbe essere anteposto al testo cifrato che viene scartato.

  CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD: AES Key Wrap con riempimento PKCS #5.

  CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD: AES Key Wrap con riempimento a zeri.

  CKM_CLOUDHSM_ML_DSA_KEY_PAIR_GEN: Genera una coppia di ML-DSA chiavi. Usa l'CKA_CLOUDHSM_PARAMETER_SETattributo (tipoCK_CLOUDHSM_ML_DSA_PARAMETER_SET_TYPE) per specificare il set di parametri. Valori validi: CKP_CLOUDHSM_ML_DSA_44, CKP_CLOUDHSM_ML_DSA_65, CKP_CLOUDHSM_ML_DSA_87.

  CKM_CLOUDHSM_ML_DSA: ML-DSA Firma e verifica puramente. Firma o verifica i dati grezzi utilizzando una coppia di ML-DSA chiavi.

  CKM_CLOUDHSM_EXT_MU_ML_DSA: ML-DSA firma e verifica con mu esterno. Firma o verifica i dati che sono stati preelaborati in un valore mu esterno.

  CKM_CLOUDHSM_EDDSA: meccanismo di firma EdDSA. Con parametri NULL: PureEd DSA (Ed25519), solo parte singola, massimo 16.000 byte. Con CK_EDDSA_PARAMS (phFlag =

`CK_TRUE`,`ulContextDataLen = 0`): ED25519ph, componente singolo e multiparte, nessun limite di dimensione. La `CK_EDDSA_PARAMS` struttura segue le specifiche PKCS #11 v3.0.

- [4] I seguenti `CK_MECHANISM_TYPE` e `CK_RSA_PKCS_MGF_TYPE` sono supportati come `CK_RSA_PKCS_OAEP_PARAMS` per `CKM_RSA_PKCS_OAEP`:
  - `CKM_SHA_1` tramite `CKG_MGF1_SHA1`
  - `CKM_SHA224` tramite `CKG_MGF1_SHA224`
  - `CKM_SHA256` tramite `CKG_MGF1_SHA256`
  - `CKM_SHA384` tramite `CKM_MGF1_SHA384`
  - `CKM_SHA512` tramite `CKM_MGF1_SHA512`
- [5] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.
- [6] Tipi definiti dal fornitore. Per poter utilizzare i tipi definiti dal fornitore di CloudHSM, le applicazioni PKCS #11 devono includerli durante la compilazione. `cloudhsm_pkcs11_vendor_defs.h` Questo si trova in `/opt/cloudhsm/include/pkcs11/cloudhsm_pkcs11_vendor_defs.h` per piattaforme basate su Linux e per piattaforme basate su Windows `C:\Program Files\Amazon\CloudHSM\include\pkcs11\cloudhsm_pkcs11_vendor_defs.h`
- [7] Supportato solo su istanze hsm2m.medium in modalità non FIPS.
- [8] Le funzioni di derivazione delle chiavi (KDF) sono specificate nella pubblicazione speciale 800-56A del NIST, revisione 3. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf

Operazioni API supportate per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

La libreria PKCS #11 supporta le seguenti operazioni API PKCS #11 per AWS CloudHSM Client SDK 5.

- `C_CloseAllSessions`
- `C_CloseSession`
- `C_CreateObject`
- `C_Decrypt`
- `C_DecryptFinal`
- `C_DecryptInit`

- C_DecryptUpdate
- C_DeriveKey
- C_DestroyObject
- C_Digest
- C_DigestFinal
- C_DigestInit
- C_DigestUpdate
- C_Encrypt
- C_EncryptFinal
- C_EncryptInit
- C_EncryptUpdate
- C_Finalize
- C_FindObjects
- C_FindObjectsFinal
- C_FindObjectsInit
- C_GenerateKey
- C_GenerateKeyPair
- C_GenerateRandom
- C_GetAttributeValue
- C_GetFunctionList
- C_GetInfo
- C_GetMechanismInfo
- C_GetMechanismList
- C_GetSessionInfo
- C_GetSlotInfo
- C_GetSlotList
- C_GetTokenInfo
- C_Initialize
- C_Login

- `C_Logout`
- `C_OpenSession`
- `C_Sign`
- `C_SignFinal`
- `C_SignInit`
- `C_SignUpdate`
- `C_UnWrapKey`
- `C_Verify`
- `C_VerifyFinal`
- `C_VerifyInit`
- `C_VerifyUpdate`
- `C_WrapKey`

Attributi chiave nella libreria PKCS #11 per AWS CloudHSM Client SDK 5

Un oggetto AWS CloudHSM chiave può essere una chiave pubblica, privata o segreta. Le azioni consentite su un oggetto chiave sono specificate tramite gli attributi. Gli attributi sono definiti quando l'oggetto chiave viene creato. Quando si utilizza la libreria PKCS #11 per AWS CloudHSM, assegniamo i valori predefiniti come specificato dallo standard PKCS #11.

AWS CloudHSM non supporta tutti gli attributi elencati nella specifica PKCS #11. Siamo conformi alla specifica per tutti gli attributi supportati. Questi attributi sono elencati nelle rispettive tabelle.

Le funzioni di crittografia, ad esempio `C_CreateObject`, `C_GenerateKey`, `C_GenerateKeyPair`, `C_UnwrapKey`, e `C_DeriveKey` che creano, modificano o copiano gli oggetti utilizzano un modello di attributo come uno dei loro parametri. Per ulteriori informazioni sul trasferimento di un modello di attributo durante la creazione di un oggetto, consulta la pagina sulla generazione di chiavi attraverso la libreria PKCS #11 per vedere degli esempi.

I seguenti argomenti forniscono ulteriori informazioni sugli attributi AWS CloudHSM chiave e sulla lettura degli attributi degli oggetti chiave e certificato.

Argomenti

- Recupera gli attributi con la libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Tabelle degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 5

- Modifica degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 5
- Interpretazione dei codici di errore della libreria PKCS #11 per AWS CloudHSM Client SDK 5

Recupera gli attributi con la libreria PKCS #11 per AWS CloudHSM Client SDK 5

Quando l'applicazione legge gli attributi da una chiave o da un oggetto certificato, la libreria PKCS #11 invia una richiesta al cluster per ogni chiamata. `C_GetAttributeValue` Il numero di attributi nel modello non modifica il numero di richieste. Una chiamata che richiede dieci attributi invia una richiesta. Dieci chiamate che richiedono un attributo inviano dieci richieste.

Ogni chiamata legge gli attributi dell'oggetto dal cluster in un'unica richiesta e quindi copia i valori richiesti dal modello. La libreria PKCS #11 non memorizza i valori nella cache, quindi ogni chiamata invia una nuova richiesta.

Per ridurre il numero di richieste inviate dall'applicazione, segui questi consigli:

- Richiedi tutti gli attributi di cui hai bisogno in un unico modello. Crea un singolo modello che contenga tutti gli attributi che desideri e passalo a una `C_GetAttributeValue` chiamata. L'esempio seguente legge tre attributi di una chiave con una chiamata.

```
CK_OBJECT_CLASS obj_class;
CK_BYTE label[128];
CK_BYTE id[128];

CK_ATTRIBUTE attrs[] = {
        {CKA_CLASS, &obj_class, sizeof(obj_class)},
        {CKA_LABEL, label,       sizeof(label)},
        {CKA_ID,    id,          sizeof(id)},
};

rv = funcs->C_GetAttributeValue(session, object, attrs,
                                sizeof(attrs) / sizeof(CK_ATTRIBUTE));
```

  Per ulteriori informazioni, vedere Elencare gli attributi chiave sul GitHub sito Web.

- Richiedi solo gli attributi supportati dal tuo oggetto e che puoi leggere. Quando il modello nomina un attributo che l'oggetto non ha, `C_GetAttributeValue` imposta un valore `ulValueLen` `CK_UNAVAILABLE_INFORMATION` per quell'attributo e restituisce`CKR_ATTRIBUTE_TYPE_INVALID`. Un attributo sensibile si comporta allo stesso modo e restituisce`CKR_ATTRIBUTE_SENSITIVE`. In entrambi i casi, gli altri attributi del modello

ricevono comunque i loro valori. Controlla `ulValueLen` ogni attributo anziché solo il valore restituito dalla chiamata. Per gli attributi supportati da ogni tipo di oggetto e per gli attributi sensibili, consultaTabelle degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 5.

- Controlla quante chiamate effettua l'associazione linguistica. Un'associazione PKCS #11 per un linguaggio come Python o Java decide quante `C_GetAttributeValue` chiamate inviare. Alcuni binding inviano una chiamata per ogni attributo, mentre altri ne inviano due. Utilizzate l'interfaccia fornita dall'associazione per leggere più attributi contemporaneamente e misurate le chiamate inviate dall'applicazione.
- Memorizza nella cache i valori degli attributi nell'applicazione. Leggi gli attributi di un oggetto una sola volta e memorizza i valori necessari all'applicazione. È possibile riutilizzare un valore memorizzato nella cache finché qualcosa non modifica l'oggetto. Quando altre applicazioni o utenti possono modificare i tuoi oggetti, aggiorna la cache dopo averlo fatto.
- Allocate i buffer prima di leggere un attributo. Lo standard PKCS #11 descrive uno schema a due passaggi per valori di lunghezza variabile. L'applicazione chiama `pValue` set `C_GetAttributeValue` to per `NULL_PTR` conoscere la lunghezza, quindi lo chiama nuovamente per leggere il valore. Entrambe le chiamate inviano una richiesta. Quando conosci una dimensione del buffer che contiene i valori che ti aspetti, alloca prima i buffer e leggi i valori con una chiamata.

  Un buffer troppo piccolo restituisce tale attributo e `CKR_BUFFER_TOO_SMALL` la chiamata invia comunque una richiesta. Scegliete dimensioni del buffer sufficientemente generose per i valori previsti. Gli attributi che contengono un modello, ad esempio`CKA_WRAP_TEMPLATE`, necessitano comunque del modello a due passaggi.

> **Note**
>
> Questi consigli si applicano sia agli oggetti chiave che agli oggetti certificati e a ciascuno di essi si applicano limiti diversi. L'archiviazione dei certificati ha un limite di velocità di lettura fisso e ogni `C_GetAttributeValue` chiamata a un oggetto certificato utilizza un'operazione di lettura da tale oggetto. Per gli oggetti chiave, l'attributo throttle degli HSM viene letto quando raggiungono la capacità anziché a una velocità fissa. La libreria PKCS #11 riprova queste richieste automaticamente. L'invio di un minor numero di richieste aiuta l'applicazione a rimanere entro entrambi i limiti. Per ulteriori informazioni, consulta Limiti di archiviazione dei certificati e Limitazione HSM.

Tabelle degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 5

Le tabelle della libreria PKCS #11 AWS CloudHSM contengono un elenco di attributi che differiscono in base al tipo di chiave. Indica se un determinato attributo è supportato per un particolare tipo di chiave quando si utilizza una funzione crittografica specifica con. AWS CloudHSM

Legenda:

- ✓ indica che CloudHSM supporta l'attributo per il tipo di chiave specifico.
- ✖ indica che CloudHSM non supporta l'attributo per il tipo di chiave specifico.
- R indica che il valore dell'attributo è di sola lettura per il tipo di chiave specifico.
- S indica che l'attributo non può essere letto da `GetAttributeValue` poiché è sensibile.
- Una cella vuota nella colonna Valore predefinito indica che non vi è alcun valore predefinito specifico assegnato all'attributo.

GenerateKeyPair

Attributo	Tipo di chiavi						Valore predefinito
	EC privato	EC pubblico	RSA privato	RSA pubblico	ML-DSA privato	ML-DSA pubblico	
`CKA_CLASS`	✓	✓	✓	✓	✓	✓	
`CKA_KEY_TYPE`	✓	✓	✓	✓	✓	✓	
`CKA_LABEL`	✓	✓	✓	✓	✓	✓	
`CKA_ID`	✓	✓	✓	✓	✓	✓	
`CKA_LOCAL`	R	R	R	R	R	R	True

Attributo	Tipo di chiavi						Valore predefinito
CKA_TOKEN	✓	✓	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✖	✓[3]	✖	✓	✖	✖	False
CKA_DECRYPT	✓[3]	✖	✓	✖	✖	✖	False
CKA_DERIVE	✓	✓	✓	✓	✖	✖	False
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	True
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	✓	✖	False
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	✖	
CKA_VERIFY	✖	✓	✖	✓	✖	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✓[3]	✖	✓	✖	✖	False

Attributo	Tipo di chiavi						Valore predefinito
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✖	✖	
CKA_TRUSTED	✖	✓	✖	✓	✖	✓	False
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✖	False
CKA_UNWRAP	✓[3]	✖	✓	✖	✖	✖	False
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✖	✖	
CKA_SENSITIVE	✓[1]	✖	✓[1]	✖	✓	✖	True
CKA_ALWAYS_SENSITIVE	R	✖	R	✖	✖	✖	
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✖	True
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	✖	✖	
CKA_MODULUS	✖	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi						Valore predefinito
CKA_MODULUS_BITS	✖	✖	✖	✓[2]	✖	✖	
CKA_PRIME_1	✖	✖	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✓[2]	✖	✖	
CKA_EC_PARAMS	✖	✓[2]	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	✖	
CKA_VALUE	✖	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi						Valore predefinito
CKA_VALUE_LEN	✖	✖	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	✖	✖	
CKA_CLOUDHSM_PARAMETER_SET	✖	✖	✖	✖	✓	✓	

GenerateKey

Attributo	Tipo di chiavi			Valore predefinito
	AES	DES3	Segreto generico	
CKA_CLASS	✓	✓	✓	
CKA_KEY_TYPE	✓	✓	✓	
CKA_LABEL	✓	✓	✓	
CKA_ID	✓	✓	✓	
CKA_LOCAL	R	R	R	True
CKA_TOKEN	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓	✓	✓	True
CKA_DESTROYABLE	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	True
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	True
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_WRAP_TEMPLATE	✓	✓	✖	
CKA_TRUSTED	✓	✓	✖	False

Attributo	Tipo di chiavi			Valore predefinito
CKA_WRAP_WITH_TRUSTED	✓	✓	✓	False
CKA_UNWRAP	✓	✓	✖	False
CKA_UNWRAP_TEMPLATE	✓	✓	✖	
CKA_SENSITIVE	✓	✓	✓	True
CKA_ALWAYS_SENSITIVE	✖	✖	✖	
CKA_EXTRACTABLE	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
CKA_VALUE	✖	✖	✖	
CKA_VALUE_LEN	✓[2]	✖	✓[2]	
CKA_CHECK_VALUE	R	R	R	

CreateObject

> **ML-DSA e CreateObject**
>
> ML-DSA l'importazione di chiavi tramite non `C_CreateObject` è supportata. Per utilizzare ML-DSA le chiavi, generale direttamente sull'HSM utilizzando`C_GenerateKeyPair`.

Attributo	Tipo di chiavi							Valore predefinito
	EC privato	EC pubblico	RSA privato	RSA pubblico	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	R	R	False
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖	False

Attributo	Tipo di chiavi							Valore predefinito
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓	True
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	✖	✖	False
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖	False
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖	
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✖	False
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓	False

Attributo	Tipo di chiavi							Valore predefinito
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖	False
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖	
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓	True
CKA_ALWAYS_SENSITIVE	R	✖	R	✖	R	R	R	
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	R	R	R	
CKA_MODULUS	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✓	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✓	✖	✖	✖	✖	

Attributo	Tipo di chiavi							Valore predefinito
CKA_EXPONENT_1	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✓[2]	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_EC_PARAMS	✓[2]	✓[2]	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✓[2]	✖	✖	✖	✖	✖	
CKA_VALUE	✓[2]	✖	✖	✖	✓[2]	✓[2]	✓[2]	
CKA_VALUE_LEN	✖	✖	✖	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	R	R	R	

UnwrapKey

Attributo	Tipo di chiavi					Valore predefinito
	EC privato	RSA privato	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	False
CKA_TOKEN	✓	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✖	✖	✓	✓	✖	False
CKA_DECRYPT	✖	✓	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓	✓	✓	✓	✓	True

Attributo	Tipo di chiavi					Valore predefinito
CKA_DESTROYABLE	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	False
CKA_VERIFY	✖	✖	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✖	✓	✓	✖	False
CKA_UNWRAP	✖	✓	✓	✓	✖	False
CKA_SENSITIVE	✓	✓	✓	✓	✓	True
CKA_EXTRACTABLE	✓	✓	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	

Attributo	Tipo di chiavi					Valore predefinito
CKA_MODULUS	✖	✖	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi					Valore predefinito
CKA_VALUE	✖	✖	✖	✖	✖	
CKA_VALUE_LEN	✖	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	R	

DeriveKey

Attributo	Tipo di chiavi			Valore predefinito
	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	
CKA_ID	✓	✓	✓	
CKA_LOCAL	R	R	R	True
CKA_TOKEN	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓	✓	✓	True
CKA_DESTROYABLE	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_UNWRAP	✓	✓	✖	False
CKA_SENSITIVE	R	R	R	True
CKA_EXTRACTABLE	✓	✓	✓	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
CKA_VALUE	✖	✖	✖	
CKA_VALUE_LEN	✓[2]	✖	✓[2]	
CKA_CHECK_VALUE	R	R	R	

GetAttributeValue

Attributo	Tipo di chiavi								
	EC privato	EC pubblico	RSA privato	RSA pubblico	ML-DSA privato	ML-DSA pubblico	AES	DES3	Segreto generico
CKA_CLASS	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	R	R	✓	✓	✓

Attributo	Tipo di chiavi								
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓	✓	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✖	✖	✖	✓	✖	✖	✓	✓	✖
CKA_DECRYPT	✖	✖	✓	✖	✖	✖	✓	✓	✖
CKA_DERIVE	✓	✓	✓	✓	✖	✖	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✖	✓	✖	✓	✖	✓	✓	✓
CKA_SIGN_RECOVER	✖	✖	✓	✖	✖	✖	✖	✖	✖
CKA_VERIFY	✖	✓	✖	✓	✖	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✖	✖	✖	✓	✖	✖	✖	✖	✖
CKA_WRAP	✖	✖	✖	✓	✖	✖	✓	✓	✖
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✖	✖	✓	✓	✖

Attributo	Tipo di chiavi								
CKA_TRUSTED	✖	✓	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✖	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✖	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✖	✓	✓	✓
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✖	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✖	✓	✖	R	R	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	R	R	R	R
CKA_MODULUS	✖	✖	✓	✓	✖	✖	✖	✖	✖
CKA_MODULUS_BITS	✖	✖	✖	✓	✖	✖	✖	✖	✖
CKA_PRIME_1	✖	✖	S	✖	✖	✖	✖	✖	✖

Attributo	Tipo di chiavi								
CKA_PRIME_2	✖	✖	S	✖	✖	✖	✖	✖	✖
CKA_COEFFICIENT	✖	✖	S	✖	✖	✖	✖	✖	✖
CKA_EXPONENT_1	✖	✖	S	✖	✖	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖	✖	✖
CKA_PUBLIC_EXPONENT	✖	✖	✓	✓	✖	✖	✖	✖	✖
CKA_EC_PARAMS	✓	✓	✖	✖	✖	✖	✖	✖	✖
CKA_EC_POINT	✖	✓	✖	✖	✖	✖	✖	✖	✖
CKA_VALUE	S	✖	✖	✖	✖	✖	✓	✓	✓
CKA_VALUE_LEN	✖	✖	✖	✖	✖	✖	✓	✖	✓
CKA_CHECK_VALUE	✓	✓	✓	✓	✖	✖	✓	✓	✖

Attributo	Tipo di chiavi								
CKA_CLOUDHSM_PARAMETER_SET	✖	✖	✖	✖	R	R	✖	✖	✖

Annotazioni degli attributi

- [1] Questo attributo è parzialmente supportato dal firmware e deve essere impostato esplicitamente solo sul valore predefinito.
- [2] Attributo obbligatorio.
- [3] Per le chiavi Ed25519 (chiavi EC con curva ed25519): CKA_ENCRYPT, CKA_DECRYPT, CKA_WRAP, CKA_UNWRAP e CKA_DERIVE non sono funzionali. Le chiavi Ed25519 sono di sola firma e non supportano le operazioni di crittografia, wrapping o derivazione delle chiavi.

Modifica degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 5

Alcuni attributi della libreria PKCS #11 di un AWS CloudHSM oggetto possono essere modificati dopo che l'oggetto è stato creato, mentre altri no. Per modificare gli attributi, utilizzate il comando key set-attribute di CloudHSM CLI. È inoltre possibile ricavare un elenco di attributi utilizzando il comando key list di CloudHSM CLI.

L'elenco seguente mostra gli attributi modificabili dopo la creazione dell'oggetto:

- `CKA_LABEL`
- `CKA_TOKEN`

> **Note**
>
> La modifica è consentita solo per la modifica di una chiave di sessione in una chiave di token. Usa il comando key set-attribute di CloudHSM CLI per modificare il valore dell'attributo.

- `CKA_ENCRYPT`
- `CKA_DECRYPT`
- `CKA_SIGN`

- `CKA_VERIFY`
- `CKA_WRAP`
- `CKA_UNWRAP`
- `CKA_LABEL`
- `CKA_SENSITIVE`
- `CKA_DERIVE`

> **Note**
>
> Questo attributo supporta la derivazione della chiave. Deve essere `False` per tutte le chiavi pubbliche e non può essere impostato su `True`. Per le chiavi segrete ed EC private, può essere impostato su `True` o `False`.

- `CKA_TRUSTED`

> **Note**
>
> Questo attributo può essere impostato su `True` o su `False` solo da Responsabile della crittografia (CO).

- `CKA_WRAP_WITH_TRUSTED`

> **Note**
>
> Applica questo attributo a una chiave di dati esportabile per specificare che è possibile eseguire il wrapping della chiave solo con chiavi contrassegnate come `CKA_TRUSTED`. Una volta impostato l'attributo `CKA_WRAP_WITH_TRUSTED` su true, questo diventa di sola lettura e non è possibile modificarlo o rimuoverlo.

Interpretazione dei codici di errore della libreria PKCS #11 per AWS CloudHSM Client SDK 5

Specificando nel modello un attributo della libreria PKCS #11 che non è supportato da una chiave specifica si genera un errore. La tabella riportata di seguito contiene i codici di errore generati quando si violano le specifiche:

Codice di errore	Descrizione
`CKR_TEMPLATE_INCONSISTENT`	Si riceve questo errore quando si specifica un attributo nel modello di attributo, in cui l'attributo è conforme alla specifica PKCS #11, ma non è supportato da CloudHSM.
`CKR_ATTRIBUTE_TYPE_INVALID`	Si riceve questo errore quando si recupera il valore di un attributo, che è conforme alla specifica PKCS #11, ma non è supportato da CloudHSM.
`CKR_ATTRIBUTE_INCOMPLETE`	Si riceve questo errore quando non si specifica l'attributo obbligatorio nel modello di attributo.
`CKR_ATTRIBUTE_READ_ONLY`	Si riceve questo errore quando si specifica un attributo di sola lettura nel modello di attributo.

Esempi di codice per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

Gli esempi di codice riportati GitHub mostrano come eseguire attività di base utilizzando la libreria PKCS #11 per AWS CloudHSM Client SDK 5.

Prerequisiti

Prima di eseguire gli esempi, attieniti alla seguente procedura per configurare l'ambiente:

- Installa e configura la libreria PKCS #11 per Client SDK 5.
- Configura un crypto user (CU). L'applicazione utilizza questo account HSM per eseguire i codici di esempio sull'HSM.

Esempi di codice

Gli esempi di codice per la libreria AWS CloudHSM software per PKCS #11 sono disponibili su. GitHub Questo repository include esempi su come eseguire operazioni comuni utilizzando PKCS #11, tra cui crittografia, decrittografia, firma e verifica.

- Generazione di chiavi (AES, RSA, EC)

- Genera coppie di chiavi Ed25519
- Genera coppie di chiavi ML-DSA
- Elenco degli attributi chiave
- Crittografia e decodifica dei dati con AES GCM
- Crittografia e decrittografia dei dati con AES_CTR
- Crittografia e decrittografia dei dati con 3DES
- Firma e verifica dei dati con RSA
- Firma e verifica i dati con ECDSA
- Firma e verifica i dati con EdDSA
- Firma e verifica i dati con ML-DSA
- Derivazione delle chiavi utilizzando HMAC KDF
- Wrapping e annullamento del wrapping delle chiavi mediante riempimento PKCS #5
- Wrapping e annullamento del wrapping delle chiavi con AES utilizzando senza riempimento
- Wrapping e annullamento del wrapping delle chiavi con AES mediante riempimento a zeri
- Avvolgi e scatta le chiavi con AES-GCM
- Wrapping e annullamento del wrapping delle chiavi con RSA

Configurazioni avanzate per la libreria PKCS #11 per AWS CloudHSM

Il provider AWS CloudHSM PKCS #11 include la seguente configurazione avanzata, che non fa parte delle configurazioni generali utilizzate dalla maggior parte dei clienti. Queste configurazioni offrono funzionalità aggiuntive.

- Connessione a più slot con PKCS #11
- Ripetizione del tentativo di configurazione per PKCS #11

Configurazione a più slot con libreria PKCS #11 per AWS CloudHSM

Un singolo slot nella libreria PKCS #11 di Client SDK 5 rappresenta una singola connessione a un cluster in AWS CloudHSM. Con Client SDK 5, è possibile configurare la libreria PKCS11 per consentire a più slot di connettere gli utenti a più cluster CloudHSM da una singola applicazione PKCS #11.

Utilizza le istruzioni riportate in questo argomento per fare in modo che l'applicazione utilizzi la funzionalità multislot per connettersi a più cluster.

Argomenti

- Multi-slot prerequisiti per la libreria PKCS #11 per AWS CloudHSM
- Configurare la libreria PKCS #11 per la funzionalità multi-slot per AWS CloudHSM
- Aggiungi un cluster con funzionalità multi-slot per AWS CloudHSM
- Rimuovi un cluster con funzionalità multi-slot per AWS CloudHSM

Multi-slot prerequisiti per la libreria PKCS #11 per AWS CloudHSM

Prima di configurare più slot per la libreria PKCS #11 per AWS CloudHSM, completare i seguenti prerequisiti.

- Due o più AWS CloudHSM cluster a cui desideri connetterti, insieme ai relativi certificati di cluster.
- Un'istanza EC2 con i gruppi di sicurezza configurata correttamente per connettersi a tutti i cluster riportati sopra. Per ulteriori informazioni su come configurare un cluster e l'istanza del client, consulta Guida introduttiva AWS CloudHSM.
- Per configurare la funzionalità multislot, è necessario aver già scaricato e installato la libreria PKCS #11. Se non lo hai ancora fatto, consulta le istruzioni riportate in ???.

Configurare la libreria PKCS #11 per la funzionalità multi-slot per AWS CloudHSM

Per configurare la libreria PKCS #11 per la funzionalità multi-slot per AWS CloudHSM, segui questi passaggi:

1. Individua i cluster a cui desideri connetterti utilizzando la funzionalità multislot.
2. Aggiungi tali cluster alla configurazione PKCS #11 seguendo le istruzioni riportate in ???
3. La prossima volta che l'applicazione PKCS #11 verrà eseguita, la funzionalità multislot sarà abilitata.

Aggiungi un cluster con funzionalità multi-slot per AWS CloudHSM

Quando ti connetti a più slot con PKCS #11 for AWS CloudHSM, usa il configure-pkcs11 add-cluster comando per aggiungere un cluster alla tua configurazione.

Sintassi

```
configure-pkcs11 add-cluster [OPTIONS]
        --cluster-id <CLUSTER ID>
        [--region <REGION>]
        [--endpoint <ENDPOINT>]
        [--hsm-ca-cert <HSM CA CERTIFICATE FILE>]
        [--client-cert-hsm-tls-file <CLIENT CERTIFICATE FILE>]
        [--client-key-hsm-tls-file <CLIENT KEY FILE>]
        [-h, --help]
```

Esempi

Aggiungi un cluster utilizzando il parametro **`cluster-id`**

Example

Utilizza il parametro configure-pkcs11 add-cluster insieme a `cluster-id` per aggiungere un cluster (con l'ID del `cluster-1234567`) alla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 add-cluster --cluster-id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" add-cluster --
cluster-id <cluster-1234567>
```

> **Tip**
>
> Se l'utilizzo del parametro configure-pkcs11 add-cluster con `cluster-id` non dà come risultato l'aggiunta del cluster, consulta l'esempio seguente per una versione più lunga del comando che richiede anche i parametri `--region` e `--endpoint` per identificare il cluster che si sta aggiungendo. Se, ad esempio, la regione del cluster è diversa da quella configurata come impostazione predefinita per la CLI di AWS, è necessario impiegare il parametro `--region` per utilizzare la regione corretta. Inoltre, è possibile specificare l'endpoint AWS CloudHSM API da utilizzare per la chiamata, operazione che può essere necessaria per varie

configurazioni di rete, ad esempio l'utilizzo di endpoint di interfaccia VPC per i quali non viene utilizzato il nome host DNS predefinito. AWS CloudHSM

Aggiungi un cluster utilizzando i parametri cluster-id, endpoint e region

Example

Utilizza configure-pkcs11 add-cluster insieme ai parametri `cluster-id`, `endpoint` e `region` per aggiungere un cluster (con l'ID del `cluster-1234567`) alla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 add-cluster --cluster-id <cluster-1234567>
 --region <us-east-1> --endpoint <https://cloudhsmv2.us-east-1.amazonaws.com>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" add-cluster --
cluster-id <cluster-1234567> --region <us-east-1> --endpoint <https://cloudhsmv2.us-
east-1.amazonaws.com>
```

Per ulteriori informazioni sui parametri `--cluster-id`, `--region` e `--endpoint`, vedi the section called “Parameters”.

Parameters

--cluster-id ***<Cluster ID>***

Effettua una chiamata `DescribeClusters` per trovare tutti gli indirizzi IP a interfaccia di rete elastica (ENI) dell'HSM nel cluster associati all'ID del cluster. Il sistema aggiunge gli indirizzi IP ENI ai file di configurazione. AWS CloudHSM

> **Note**
>
> Se si utilizza il `--cluster-id` parametro di un'istanza EC2 all'interno di un VPC che non ha accesso alla rete Internet pubblica, è necessario creare un endpoint VPC di interfaccia con cui connettersi. AWS CloudHSM Per ulteriori informazioni sugli endpoint VPC, consulta la pagina ???.

Obbligatorio: sì

--endpoint ***<Endpoint>***

Specifica l'endpoint AWS CloudHSM API utilizzato per effettuare la chiamata. `DescribeClusters` È necessario impostare questa opzione insieme a `--cluster-id`.

Obbligatorio: no

--hsm-ca-cert ***<HsmCA Certificate Filepath>***

Specifica il percorso del file del certificato HSM CA.

Obbligatorio: no

--regione ***<Region>***

Specifica la regione del cluster. È necessario impostare questa opzione insieme a `--cluster-id`.

Se non indichi il parametro `--region`, il sistema sceglie la regione tentando di leggere le variabili di ambiente `AWS_DEFAULT_REGION` o `AWS_REGION`. Se queste variabili non sono impostate, il sistema controlla la regione associata al tuo profilo indicata nel tuo file di AWS Config (generalmente `~/.aws/config`) a meno che non sia stato specificato un file diverso nella variabile di ambiente `AWS_CONFIG_FILE`. Se non è stata impostata nessuna delle variabili precedenti, il sistema utilizza la regione `us-east-1` per impostazione predefinita.

Obbligatorio: no

--client-cert-hsm-tls-file ***<client certificate hsm tls path>***

Percorso del certificato client utilizzato per l'autenticazione reciproca TLS Client-HSM.

Usa questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-key-hsm-tls-file`.

Obbligatorio: no

--client-key-hsm-tls-file ***<client key hsm tls path>***

Percorso della chiave client utilizzata per l'autenticazione reciproca TLS Client-HSM.

Usa questa opzione solo se hai registrato almeno un trust anchor su HSM con CloudHSM CLI. È necessario impostare questa opzione insieme a `--client-cert-hsm-tls-file`.

Obbligatorio: no

Rimuovi un cluster con funzionalità multi-slot per AWS CloudHSM

Quando ti connetti a più slot con PKCS #11, utilizza il comando configure-pkcs11 remove-cluster per rimuovere un cluster dagli slot PKCS #11 disponibili.

Sintassi

```
configure-pkcs11 remove-cluster [OPTIONS]
        --cluster-id <CLUSTER ID>
        [-h, --help]
```

Esempi

Rimuovi un cluster utilizzando il **parametro cluster-id**

Example

Utilizza il parametro configure-pkcs11 remove-cluster insieme a `cluster-id` per rimuovere un cluster (con l'ID del `cluster-1234567`) dalla tua configurazione.

Linux

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 remove-cluster --cluster-
id <cluster-1234567>
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" remove-cluster
 --cluster-id <cluster-1234567>
```

Per ulteriori informazioni sul parametro `--cluster-id`, vedi the section called "Parameters".

Parametro

--cluster-id ***<Cluster ID>***

L'ID del cluster da rimuovere dalla configurazione

Obbligatorio: sì

Riprova i comandi per la libreria PKCS #11 per AWS CloudHSM

AWS CloudHSM Client SDK 5.8.0 e versioni successive dispongono di una strategia di riprova automatica integrata che riproverà le operazioni con limitazione HSM dal lato client. Quando un HSM rallenta le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, il client SDKs tenterà di riprovare le operazioni limitate fino a 3 volte, effettuando un backup esponenziale. Questa strategia automatica può essere impostata su una delle due modalità: off e standard.

- off: il Client SDK non eseguirà alcun ulteriore tentativo per le operazioni limitate da parte dell'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e successive. In questa modalità, il client SDKs riproverà automaticamente le operazioni limitate effettuando un backup esponenziale.

Per ulteriori informazioni, consulta Limitazione HSM.

Disattiva i comandi per l'esecuzione di ulteriori tentativi

Linux

Come impostare i comandi Nuovo tentativo su off per Client SDK 5 su Linux

- Per impostare la configurazione in modalità off, utilizza i seguenti comandi:

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --default-retry-mode off
```

Windows

Per impostare i comandi Nuovo Tentativo su off per Client SDK 5 su Windows

- Per impostare la configurazione in modalità off, utilizza i seguenti comandi:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --default-
retry-mode off
```

Archiviazione dei certificati con la libreria PKCS #11

La libreria AWS CloudHSM PKCS #11 supporta l'archiviazione di certificati a chiave pubblica come «oggetti pubblici» (come definiti in PKCS #11 2.40) su cluster hsm2m.medium. Questa funzionalità

consente alle sessioni PKCS #11 pubbliche e private di creare, recuperare, modificare ed eliminare certificati a chiave pubblica.

Per utilizzare l'archiviazione dei certificati con la libreria PKCS #11, è necessario abilitarla nella configurazione del client. Una volta abilitata, è possibile gestire gli oggetti dei certificati dalle applicazioni PKCS #11. Le operazioni che si applicano sia al certificato che agli oggetti chiave, come C_ FindObjects, restituiranno risultati dall'archiviazione delle chiavi e dei certificati.

> **Limiti di archiviazione dei certificati**
>
> L'archiviazione dei certificati limita il numero di certificati archiviati per ogni cluster e la velocità delle operazioni di lettura e scrittura per ogni HSM. Per ulteriori informazioni, consulta Limiti di archiviazione dei certificati.

Argomenti

- Abilitazione dell'archiviazione dei certificati
- Operazioni API di archiviazione dei certificati
- Attributi di archiviazione dei certificati
- Registri di controllo dell'archiviazione dei certificati
- Limiti di archiviazione dei certificati

Abilitazione dell'archiviazione dei certificati

È possibile abilitare l'archiviazione dei certificati sui cluster hsm2m.medium utilizzando lo strumento di configurazione della libreria PKCS #11. Questa funzionalità è disponibile nelle versioni SDK 5.13 e successive. Per un elenco delle operazioni che supportano il tipo di oggetto del certificato, vedere. Operazioni API di archiviazione dei certificati

Per abilitare l'archiviazione dei certificati, segui questi passaggi per il tuo sistema operativo:

Linux

- Abilita l'archiviazione dei certificati

  Esegui il comando seguente:

  ```
  $ sudo /opt/cloudhsm/bin/configure-pkcs11 --enable-certificate-storage
  ```

Windows

- Abilita l'archiviazione dei certificati

  Aprire un prompt dei comandi ed eseguire il comando seguente:

  ```
  PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" --enable-certificate-storage
  ```

Operazioni API di archiviazione dei certificati

Le seguenti operazioni PKCS #11 supportano il tipo di oggetto del certificato (`CKO_CERTIFICATE`):

Operazioni generali relative ai certificati

`C_CreateObject`

Crea un nuovo oggetto certificato.

`C_DestroyObject`

Elimina un oggetto certificato esistente.

`C_GetAttributeValue`

Ottiene il valore di uno o più attributi di un oggetto certificato.

`C_SetAttributeValue`

Aggiorna il valore di uno o più attributi di un oggetto certificato.

Operazioni di ricerca degli oggetti certificati

`C_FindObjectsInit`

Avvia una ricerca di oggetti del certificato.

`C_FindObjects`

Continua la ricerca degli oggetti del certificato.

`C_FindObjectsFinal`

Termina la ricerca degli oggetti del certificato.

Limitazione dei codici di restituzione

Quando l'archiviazione dei certificati limita un'operazione in caso di superamento del limite di velocità di lettura o scrittura, il codice restituito dipende dall'operazione. Operazioni di scrittura (`C_CreateObject`C_SetAttributeValue, e`C_DestroyObject`) e restituzione dell'operazione di `C_FindObjectsInit` lettura. `CKR_FUNCTION_FAILED` `C_GetAttributeValue`attualmente restituisce`CKR_DEVICE_ERROR`. In tutti i casi, segui le migliori pratiche per gestire il throttling in. Limitazione HSM

Attributi di archiviazione dei certificati

La tabella seguente elenca gli attributi degli oggetti certificati supportati e i relativi valori:

Attributo	Valore predefinito	Description
`CKA_CLASS`	Richiesto	Deve essere `CKO_CERTIFICATE` .
`CKA_TOKEN`	True	Deve essere `True`.
`CKA_MODIFIABLE`	True	Deve essere `True`.
`CKA_PRIVATE`	False	Deve essere `False`.
`CKA_LABEL`	Empty	Limite 127 caratteri.
`CKA_COPYABLE`	False	Deve essere `False`.
`CKA_DESTROYABLE`	True	Deve essere `True`.
`CKA_CERTIFICATE_TYPE`	Richiesto	Deve essere `CKC_X_509` .
`CKA_TRUSTED`	False	Deve essere `False`.
`CKA_CERTIFICATE_CATEGORY`	`CK_CERTIFICATE_CATEGORY_UNSPECIFIED`	Deve essere `CK_CERTIFICATE_CATEGORY_UNSPECIFIED` .

Attributo	Valore predefinito	Description
CKA_CHECK_VALUE	Derivato da CKA_VALUE	Impostato automaticamente in base aCKA_VALUE .
CKA_START_DATE	Empty	La data «non precedente» del certificato.
CKA_END_DATE	Empty	La data «non successiva» del certificato.
CKA_PUBLIC_KEY_INFO	Empty	La dimensione massima è di 16 kilobyte.
CKA_SUBJECT	Richiesto	L'oggetto del certificato.
CKA_ID	Empty	La dimensione massima è 128 byte. L'unicità non viene applicata.
CKA_ISSUER	Empty	L'emittente del certificato.
CKA_SERIAL_NUMBER	Empty	Il numero di serie del certificato.
CKA_VALUE	Richiesto	La dimensione massima è 32 kilobyte.

Registri di controllo dell'archiviazione dei certificati

AWS CloudHSM scrive log di controllo per le operazioni di archiviazione dei certificati che modificano i dati in un flusso di log Amazon CloudWatch Events separato all'interno del gruppo di CloudWatch log del cluster. Questo flusso di log prende il nome dal cluster, non da un HSM specifico all'interno del cluster.

Per informazioni sull'accesso ai log di controllo CloudWatch, vedereUtilizzo di Amazon CloudWatch Logs e AWS CloudHSM Audit Logs.

Campi di immissione dei log

`object_handle`

L'identificatore univoco dell'oggetto del certificato.

`op_code`

L'operazione eseguita o tentata. Valori possibili:

- `CreateObject`
- `DestroyObject`
- `SetAttributeValues`

`response`

`OK`se l'operazione è riuscita o uno dei seguenti tipi di errore:

- `DuplicateAttribute`
- `InvalidAttributeValue`
- `ObjectNotFound`
- `MaxObjectsReached`
- `InternalFailure`

`attributes`

Gli eventuali attributi modificati.

`timestamp`

L'ora in cui è avvenuta l'operazione, in millisecondi dall'epoca Unix.

Esempi di registro di controllo

CreateObject esempio

```
{
    "object_handle": 463180677312929947,
    "op_code": "CreateObject",
    "response": "OK",
    "attributes": null,
    "timestamp": 1725482483671
}
```

DestroyObject esempio

```
{
    "object_handle": 463180677312929947,
    "op_code": "DestroyObject",
```

```
    "response": "OK",
    "attributes": null,
    "timestamp": 1725482484559
}
```

SetAttributeValues esempio

```
{
    "object_handle": 463180678453346687,
    "op_code": "SetAttributeValues",
    "response": "OK",
    "attributes": [
        "Label"
    ],
    "timestamp": 1725482488004
}
```

Esempio non riuscito CreateObject

```
{
    "object_handle": null,
    "op_code": "CreateObject",
    "response": "MaxObjectsReached",
    "attributes": null,
    "timestamp": 1726084937125
}
```

Limiti di archiviazione dei certificati

I seguenti limiti si applicano all'archiviazione dei certificati. Non puoi modificare questi limiti. Il numero massimo di certificati archiviati si applica a ciascun cluster. I limiti di velocità di lettura e scrittura si applicano a ciascun HSM, pertanto la velocità totale supportata da un cluster aumenta man mano che si aggiungono HSM.

Limite	Valore	Scope
Numero massimo di certificati archiviati	60	Ogni cluster
Operazioni di lettura al secondo	10	Ogni HSM
Operazioni di scrittura al secondo	1	Ogni HSM

Le operazioni di lettura includono `C_FindObjectsInit` e. `C_GetAttributeValue` Le operazioni di scrittura includono `C_CreateObject``C_SetAttributeValue`, e`C_DestroyObject`.

I seguenti comportamenti si applicano quando si superano questi limiti:

- Se si supera il numero massimo di certificati archiviati, l'`C_CreateObject`operazione viene `CKR_FUNCTION_FAILED` ripristinata e l'archiviazione dei certificati registra un `MaxObjectsReached` errore nei registri di controllo dell'archiviazione dei certificati. I certificati esistenti rimangono leggibili. Per archiviare certificati aggiuntivi, elimina i certificati che non ti servono più.
- Se si supera il limite di velocità di lettura o scrittura, l'archiviazione dei certificati limita la restituzione delle richieste e delle operazioni`CKR_FUNCTION_FAILED`, ad eccezione di`C_GetAttributeValue`, che restituisce `CKR_DEVICE_ERROR` (vedere[Problema: C_GetAttributeValue restituisce CKR_DEVICE_ERROR quando è limitato]). La libreria PKCS #11 non ritenta le operazioni limitate di archiviazione dei certificati (vedi[Problema: la libreria PKCS #11 non riprova le operazioni di archiviazione dei certificati con limitazioni]), quindi l'applicazione deve riprovarle con un backoff esponenziale. Per ulteriori informazioni, consulta Limitazione HSM.

Per rimanere entro il limite di velocità di lettura, riduci il numero di operazioni di lettura inviate dall'applicazione. Ogni `C_GetAttributeValue` chiamata conta come un'operazione di lettura, indipendentemente dal numero di attributi nel modello. Un'applicazione che richiede tutti gli attributi necessari in una chiamata utilizza un'operazione di lettura. Un'applicazione che effettua una chiamata separata per ogni attributo utilizza un'operazione di lettura per ciascuno di essi. L'aggiunta di HSM aumenta la velocità totale supportata dal cluster, ma non modifica questo rapporto. Per ulteriori informazioni, consulta Recupera gli attributi con la libreria PKCS #11 per AWS CloudHSM Client SDK 5.

OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5

L' AWS CloudHSM OpenSSL Dynamic Engine consente di trasferire le operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

AWS CloudHSM fornisce un motore dinamico OpenSSL, di cui puoi leggere in o. AWS CloudHSM Offload SSL/TLS su Linux utilizzando Tomcat con JSSE AWS CloudHSM SSL/TLS offload su Linux usando NGINX o Apache con OpenSSL Per un esempio sull'utilizzo AWS CloudHSM con OpenSSL, consulta questo blog sulla sicurezza di AWS. Per ulteriori informazioni sulle piattaforme supportate per gli SDK, consulta la pagina the section called “Piattaforme supportate”. Per la risoluzione dei problemi, consulta. Problemi noti di OpenSSL Dynamic Engine per AWS CloudHSM

Usa le seguenti sezioni per installare e configurare il motore AWS CloudHSM dinamico per OpenSSL, usando Client SDK 5.

Per informazioni sull'utilizzo di Client SDK 3, consulta la pagina Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM.

Argomenti

- Installa il motore dinamico OpenSSL per AWS CloudHSM Client SDK 5
- Tipi di chiave supportati per OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5
- Meccanismi supportati per OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5
- Configurazioni avanzate per OpenSSL per AWS CloudHSM

Installa il motore dinamico OpenSSL per AWS CloudHSM Client SDK 5

Usa le seguenti sezioni per installare OpenSSL Dynamic Engine for AWS CloudHSM Client SDK 5.

> **Note**
>
> Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Come installare e configurare OpenSSL Dynamic Engine

1. Utilizzare i comandi seguenti per scaricare e installare il motore OpenSSL.

   Amazon Linux 2023

   Installa OpenSSL Dynamic Engine per Amazon Linux 2023 su architettura x86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-dyn-latest.amzn2023.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-dyn-latest.amzn2023.x86_64.rpm
   ```

Installa l'OpenSSL Dynamic Engine per Amazon Linux 2023 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
cloudhsm-dyn-latest.amzn2023.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.amzn2023.aarch64.rpm
```

Amazon Linux 2

Installa l'OpenSSL Dynamic Engine per Amazon Linux 2 sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-
dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el7.x86_64.rpm
```

Installa l'OpenSSL Dynamic Engine per Amazon Linux 2 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-
dyn-latest.el7.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el7.aarch64.rpm
```

RHEL 9 (9.2+)

Installa l'OpenSSL Dynamic Engine per RHEL 9 sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
dyn-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el9.x86_64.rpm
```

Installa l'OpenSSL Dynamic Engine per RHEL 9 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
dyn-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Installa l'OpenSSL Dynamic Engine per RHEL 8 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-dyn-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el8.x86_64.rpm
```

Installa l'OpenSSL Dynamic Engine per RHEL 8 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-dyn-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-dyn-latest.el8.aarch64.rpm
```

Ubuntu 26.04 LTS

Installa l'OpenSSL Dynamic Engine per Ubuntu 26.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-dyn_latest_u26.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u26.04_amd64.deb
```

Installa l'OpenSSL Dynamic Engine per Ubuntu 26.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-dyn_latest_u26.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u26.04_arm64.deb
```

Ubuntu 24.04 LTS

Installa l'OpenSSL Dynamic Engine per Ubuntu 24.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-dyn_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u24.04_amd64.deb
```

Installa l'OpenSSL Dynamic Engine per Ubuntu 24.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-dyn_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Installa l'OpenSSL Dynamic Engine per Ubuntu 22.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-dyn_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u22.04_amd64.deb
```

Installa l'OpenSSL Dynamic Engine per Ubuntu 22.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-dyn_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u22.04_arm64.deb
```

Hai installato la libreria condivisa per il motore dinamico su `/opt/cloudhsm/lib/libcloudhsm_openssl_engine.so`.

2. Esegui il bootstrap di Client SDK 5. Per ulteriori informazioni sulle operazioni di bootstrap, consulta la pagina Esegui il bootstrap di Client SDK.
3. Imposta una variabile di ambiente con le credenziali di un crypto user (CU). Per ulteriori informazioni sulla creazione di CU, consulta la pagina the section called "creazione".

```
$ export CLOUDHSM_PIN=<HSM user name>:<password>
```

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU. In Client SDK 3 le credenziali del CU si archiviano nella variabile di ambiente `n3fips_password`. Client SDK 5 supporta entrambe le variabili di ambiente, ma si consiglia di utilizzare `CLOUDHSM_PIN`.
> Quando si impostano le variabili di `CLOUDHSM_PIN` ambiente, è necessario evitare qualsiasi carattere speciale che possa essere interpretato dalla shell.

4. Collega l'installazione di OpenSSL Dynamic Engine al cluster. Per ulteriori informazioni, consulta la pagina Connessione al cluster.
5. Esegui il bootstrap di Client SDK 5. Per ulteriori informazioni, consulta the section called "Esegui il bootstrap di Client SDK".

Verifica di OpenSSL Dynamic Engine per Client SDK 5

Usa il comando seguente per verificare l'installazione di OpenSSL Dynamic Engine.

```
$ openssl engine -t cloudhsm
```

Il seguente output verifica la configurazione:

```
(cloudhsm) CloudHSM OpenSSL Engine
     [ available ]
```

Tipi di chiave supportati per OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5

L' AWS CloudHSM OpenSSL Dynamic Engine supporta i seguenti tipi di chiave con Client SDK 5.

Tipo di chiavi	Description
EC	Tipi di chiave ECDSA sign/verify per P-256, e P-384 secp256k1. Per generare chiavi EC interoperabili con il motore OpenSSL, consulta la pagina Esporta una chiave asimmetrica con CloudHSM CLI.
RSA	Generazione di chiavi RSA per chiavi a 2048, 3072 e 4096 bit.RSA. sign/verify Viene eseguito l'offload della verifica sul software OpenSSL.

Meccanismi supportati per OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5

L' AWS CloudHSM OpenSSL Dynamic Engine supporta i seguenti meccanismi per le funzioni Sign and Verify con Client SDK 5.

Funzioni di firma e verifica

Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non ci sono limiti alla dimensione dei dati che possono essere sottoposti a hashing.

Tipi di firma RSA

- SHA1 con RSA
- SHA224 con RSA
- SHA256 con RSA
- SHA384 con RSA
- SHA512 con RSA

Tipi di firma ECDSA

- SHA1 con ECDSA
- SHA224 con ECDSA
- SHA256 con ECDSA

- SHA384 con ECDSA
- SHA512 con ECDSA

Configurazioni avanzate per OpenSSL per AWS CloudHSM

Il provider AWS CloudHSM OpenSSL include la seguente configurazione avanzata, che non fa parte delle configurazioni generali utilizzate dalla maggior parte dei clienti. Queste configurazioni offrono funzionalità aggiuntive.

- Comandi Nuovo tentativo per OpenSSL

Riprova i comandi per OpenSSL per AWS CloudHSM

AWS CloudHSM Client SDK 5.8.0 e versioni successive dispongono di una strategia di riprova automatica integrata che riproverà le operazioni dal lato client. HSM-throttled Quando un HSM limita le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, gli SDK client tenteranno di ripetere le operazioni limitate fino a 3 volte, effettuando un backup esponenziale. Questa strategia automatica può essere impostata su una delle due modalità: off e standard.

- off: il Client SDK non eseguirà alcun ulteriore tentativo per le operazioni limitate da parte dell'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e successive. In questa modalità, gli SDK del client ripeteranno automaticamente i tentativi per le operazioni con un backoff esponenziale.

Per ulteriori informazioni, vedi Limitazione HSM.

Disattiva i comandi per l'esecuzione di ulteriori tentativi

È possibile utilizzare i seguenti comandi per impostare i comandi di ripetizione dei tentativi sulla modalità off:

```
$ sudo /opt/cloudhsm/bin/configure-dyn --default-retry-mode off
```

Provider OpenSSL per AWS CloudHSM Client SDK 5

L' AWS CloudHSM OpenSSL Provider ti consente di trasferire le operazioni crittografiche TLS sul tuo cluster CloudHSM tramite l'API OpenSSL Provider. L'interfaccia Provider è l'approccio consigliato per le nuove implementazioni che utilizzano OpenSSL 3.2 e versioni successive.

Usa le seguenti sezioni per installare e configurare il provider AWS CloudHSM OpenSSL, utilizzando Client SDK 5.

> **Requisiti di versione OpenSSL per ML-DSA**
>
> ML-DSA i tipi di chiave richiedono OpenSSL 3.5 o successivo.

Piattaforme supportate

Il provider OpenSSL richiede OpenSSL 3.2 o successivo, disponibile su EL9+, EL10+, Ubuntu 26.04 LTS e Amazon Linux 2023+.

Verifica la compatibilità: `openssl version`

Argomenti

- Installa il provider OpenSSL per AWS CloudHSM Client SDK 5
- Tipi di chiave supportati per OpenSSL Provider per AWS CloudHSM Client SDK 5
- Meccanismi supportati dal provider OpenSSL
- Configurazione avanzata del provider OpenSSL

Installa il provider OpenSSL per AWS CloudHSM Client SDK 5

Utilizza le seguenti sezioni per installare l'OpenSSL Provider for AWS CloudHSM Client SDK 5.

> **Note**
>
> Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Requisiti

Il provider OpenSSL richiede i tipi di cluster hsm2m.medium e la versione minima di CloudHSM Client SDK 5.17.0 o successiva.

Installa il provider OpenSSL

Per installare il provider OpenSSL

1. Usa i seguenti comandi per scaricare e installare il provider OpenSSL.

   Amazon Linux 2023

   Installa il provider OpenSSL per Amazon Linux 2023 sull'architettura x86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-openssl-provider-latest.amzn2023.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-openssl-provider-latest.amzn2023.x86_64.rpm
   ```

   Installa il provider OpenSSL per Amazon Linux 2023 sull'architettura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-openssl-provider-latest.amzn2023.aarch64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-openssl-provider-latest.amzn2023.aarch64.rpm
   ```

   RHEL 9 (9.2+)

   Installa il provider OpenSSL per RHEL 9 sull'architettura x86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
   openssl-provider-latest.el9.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-openssl-provider-latest.el9.x86_64.rpm
   ```

   Installa il provider OpenSSL per RHEL 9 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
openssl-provider-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el9.aarch64.rpm
```

RHEL 10 (10.0+)

Installa il provider OpenSSL per RHEL 10 sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/
cloudhsm-openssl-provider-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el10.x86_64.rpm
```

Installa il provider OpenSSL per RHEL 10 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/
cloudhsm-openssl-provider-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-openssl-provider-latest.el10.aarch64.rpm
```

Ubuntu 26.04 LTS

Installa il provider OpenSSL per Ubuntu 26.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-openssl-provider_latest_u26.04_amd64.deb
```

```
$ sudo dpkg -i ./cloudhsm-openssl-provider_latest_u26.04_amd64.deb
```

Installa il provider OpenSSL per Ubuntu 26.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/
cloudhsm-openssl-provider_latest_u26.04_arm64.deb
```

```
$ sudo dpkg -i ./cloudhsm-openssl-provider_latest_u26.04_arm64.deb
```

Hai installato la libreria condivisa per il provider OpenSSL all'indirizzo. `/opt/cloudhsm/lib/licloudhsm_openssl_provider.so`

2. Esegui il bootstrap di Client SDK 5. Per ulteriori informazioni sulle operazioni di bootstrap, consulta la pagina Esegui il bootstrap di Client SDK.
3. Imposta la variabile di `CLOUDHSM_PIN` ambiente con le tue credenziali di utente crittografico (CU):

```
$ export CLOUDHSM_PIN=<username>:<password>
```

4. Connetti la tua installazione di OpenSSL Provider al cluster. Per ulteriori informazioni, consulta la pagina Connessione al cluster.

Verifica dell'installazione

Verifica che il provider OpenSSL sia installato correttamente:

```
$ CLOUDHSM_PIN=<username>:<password> openssl list -providers -provider cloudhsm
```

Dovresti vedere un risultato simile a:

```
Providers:
  cloudhsm
    name: AWS CloudHSM OpenSSL Provider
    version: 5.17.0
    status: active
  default
    name: OpenSSL Default Provider
    version: 3.2.2
    status: active
```

Tipi di chiave supportati per OpenSSL Provider per AWS CloudHSM Client SDK 5

Il provider AWS CloudHSM OpenSSL supporta i seguenti tipi di chiave con Client SDK 5.

Tipo di chiavi	Description
RSA	Operazioni di crittografia RSA sign/verify e asimmetrica. Viene eseguito l'offload della verifica sul software OpenSSL. Per generare chiavi RSA interoperabili con il provider OpenSSL, vedere. Esporta una chiave asimmetrica con CloudHSM CLI
EC	ECDSA per, sign/verify e curve. P-256 P-384 P-521 Viene eseguito l'offload della verifica sul software OpenSSL. Per generare chiavi EC interoperabili con il provider OpenSSL, vedere. Esporta una chiave asimmetrica con CloudHSM CLI
Ed 25519	sign/verify Operazioni EdDSA utilizzando Curve25519 (RFC 8032). Viene eseguito l'offload della verifica sul software OpenSSL. Ed25519 è disponibile solo su cluster non FIPS. Per generare chiavi Ed25519 interoperabili con il provider OpenSSL, vedere. Esporta una chiave asimmetrica con CloudHSM CLI
ML-DSA-44, ML-DSA-65, ML-DSA-87	Post-quantum sign/verify operazioni di firma digitale come definite in FIPS 204. La verifica viene trasferita al software OpenSSL. ML-DSA[1] richiede OpenSSL 3.5 o successivo. Per generare ML-DSA chiavi interoperabili con il provider OpenSSL, consulta. Esporta una chiave asimmetrica con CloudHSM CLI

[1] A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

Meccanismi supportati dal provider OpenSSL

L'SDK AWS CloudHSM OpenSSL Provider supporta un set completo di meccanismi crittografici per varie operazioni, tra cui firme digitali, crittografia asimmetrica, crittografia simmetrica, scambio di chiavi e altro ancora.

Tipi di firma RSA

Il provider OpenSSL supporta le firme digitali RSA con diversi algoritmi di hash e schemi di riempimento:

SHA1 con RSA

SHA-1 firme RSA con algoritmo hash

- Imbottitura PKCS #1 v1.5
- Imbottitura PSS (Probabilistic Signature Scheme)

SHA224 con RSA

Firme RSA con algoritmo hash SHA-224

- Imbottitura PKCS #1 v1.5
- Imbottitura PSS

SHA256 con RSA

Firme RSA con algoritmo hash SHA-256

- Imbottitura PKCS #1 v1.5
- Imbottitura PSS

SHA384 con RSA

Firme RSA con algoritmo hash SHA-384

- Imbottitura PKCS #1 v1.5
- Imbottitura PSS

SHA512 con RSA

Firme RSA con algoritmo hash SHA-512

- Imbottitura PKCS #1 v1.5
- Imbottitura PSS

Tipi di firma ECDSA

Il provider OpenSSL supporta le firme digitali ECDSA con più algoritmi di hash:

SHA1 con ECDSA

Firme SHA-1 ECDSA con algoritmo hash

SHA224 con ECDSA

Firme ECDSA con algoritmo hash SHA-224

SHA256 con ECDSA

Firme ECDSA con algoritmo hash SHA-256

SHA384 con ECDSA

Firme ECDSA con algoritmo hash SHA-384

SHA512 con ECDSA

Firme ECDSA con algoritmo hash SHA-512

Tipi di firma EdDSA

Il provider OpenSSL supporta le firme digitali EdDSA. Ed25519 è disponibile solo su cluster non FIPS.

Ed25519

Ed25519 esegue le firme EdDSA utilizzando Curve25519 (RFC 8032) con firma one-shot e un hash interno. SHA-512 Non è possibile selezionare un algoritmo di digest esterno. Ed25519 supporta la negoziazione delle firme TLS 1.3.

ML-DSA tipi di firma

Il provider OpenSSL supporta ML-DSA (Module-Lattice Digital Signature Algorithm) le firme digitali post-quantistiche come definite in FIPS 204. ML-DSA richiede OpenSSL 3.5 o successivo.

ML-DSA-44, ML-DSA-65, ML-DSA-87

ML-DSA fornisce firme post-quantistiche a NIST security levels 2 (ML-DSA-4 4, 128 bit), 3 (, 192 bit) e 5 (ML-DSA-65, 256 bit). ML-DSA-87 Tutte le varianti utilizzano la firma in modalità

pura one-shot. Non è possibile selezionare un algoritmo di digest esterno. ML-DSA supporta la negoziazione delle firme TLS 1.3.

Configurazione avanzata del provider OpenSSL

L'SDK AWS CloudHSM OpenSSL Provider offre opzioni di configurazione avanzate per personalizzarne il comportamento per diversi casi d'uso e ambienti. Utilizza lo strumento di configurazione per impostare queste opzioni.

- Riprova la configurazione per OpenSSL Provider

Riprova i comandi per OpenSSL Provider per AWS CloudHSM

AWS CloudHSM Client SDK 5.8.0 e versioni successive dispongono di una strategia di riprova automatica integrata che riproverà le operazioni con limitazione HSM dal lato client. Quando un HSM rallenta le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, il client SDKs tenterà di riprovare le operazioni limitate fino a 3 volte, effettuando un backup esponenziale. Questa strategia automatica può essere impostata su una delle due modalità: off e standard.

- off: il Client SDK non eseguirà alcun ulteriore tentativo per le operazioni limitate da parte dell'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e successive. In questa modalità, il client SDKs riproverà automaticamente le operazioni limitate effettuando un backup esponenziale.

Per ulteriori informazioni, consulta Limitazione HSM.

Disattiva i comandi per l'esecuzione di ulteriori tentativi

Linux

Come impostare i comandi Nuovo tentativo su off per Client SDK 5 su Linux

- Per impostare la configurazione in modalità off, utilizza i seguenti comandi:

```
$ sudo /opt/cloudhsm/bin/configure-openssl-provider --default-retry-mode off
```

Provider di archiviazione delle chiavi (KSP) per AWS CloudHSM Client SDK 5

Key Storage Provider (KSP) è un'API crittografica specifica per il sistema operativo Microsoft Windows. Key Storage Provider (KSP) consente agli sviluppatori di utilizzare tecniche crittografiche per proteggere le applicazioni basate su Windows.

Per informazioni sul processo di bootstrap, consulta la pagina Connessione al cluster.

Per informazioni sull'utilizzo di Client SDK 3, vedi Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM.

Argomenti

- Installa il Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5
- Autenticazione al Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5
- Tipi di chiavi supportati per Key Storage Provider (KSP) for AWS CloudHSM Client SDK 5
- Operazioni API supportate Key storage provider (KSP) per AWS CloudHSM Client SDK 5
- Configurazioni avanzate per KSP per AWS CloudHSM

Installa il Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5

Utilizza le seguenti sezioni per installare il Key storage provider (KSP) per AWS CloudHSM Client SDK 5.

> **Note**
>
> Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Per installare e configurare il Key Storage Provider (KSP)

1. Installa il Key Storage Provider (KSP) per Windows Server sull'architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Windows/
AWSCloudHSMKSP-latest.msi -Outfile C:\AWSCloudHSMKSP-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMKSP-latest.msi /
quiet /norestart /log C:\client-install.txt' -Wait
```

2. Utilizza lo strumento di configurazione per specificare la posizione del certificato emittente. Per istruzioni, consulta Specifica la posizione del certificato di emissione.
3. Per connetterti al cluster, consulta la pagina Esegui il bootstrap di Client SDK.
4. È possibile trovare i file Key Storage Provider (KSP) nelle seguenti posizioni:

   - File binari Windows:

     ```
     C:\Program Files\Amazon\CloudHSM
     ```

     Script di configurazione e file di log Windows:

     ```
     C:\ProgramData\Amazon\CloudHSM
     ```

Autenticazione al Key Storage Provider (KSP) per AWS CloudHSM Client SDK 5

Prima di utilizzare il Key storage provider (KSP) per AWS CloudHSM Client SDK 5, è necessario impostare le credenziali di accesso per l'HSM sul sistema. Sono disponibili due opzioni:

- Windows Credentials Manager (consigliato per una maggiore sicurezza)
- Variabili di ambiente di sistema (configurazione più semplice)

Gestione credenziali di Windows

È possibile configurare le credenziali utilizzando l'`set_cloudhsm_credentials`utilità o l'interfaccia di Windows Credentials Manager.

- Utilizzo dell'utility **`set_cloudhsm_credentials`**:

  Il programma di installazione di Windows include l'utilità. `set_cloudhsm_credentials` Puoi utilizzare questa utility per trasferire le credenziali di accesso HSM a Gestione credenziali di

Windows. Se si desidera compilare questa utilità dai sorgenti, è possibile utilizzare il codice Python incluso nell'installatore.

1. Accedi a `C:\Program Files\Amazon\CloudHSM\tools\`.
2. Esegui il comando seguente:

```
set_cloudhsm_credentials.exe --username <CU USER> --password <CU PASSWORD>
```

- Utilizzo dell'interfaccia di gestione delle credenziali:
  1. Apri Credential Manager:
     - Inserisci `credential manager` nella casella di ricerca della barra delle applicazioni
     - Seleziona Credential Manager
  2. Seleziona Credenziali di Windows per gestire le credenziali di Windows.
  3. Seleziona Aggiungi una credenziale generica
  4. Inserisci i seguenti dettagli:
     - Indirizzo Internet o di rete:`CLOUDHSM_PIN`.
     - Nome utente:*<CU USER>*.
     - Password:*<CU PASSWORD>*.
  5. Seleziona OK

Variabili di ambiente del sistema

Puoi impostare variabili di ambiente di sistema per identificare il tuo HSM e il tuo utente crittografico (CU).

> **Warning**
>
> L'impostazione delle credenziali tramite le variabili di ambiente di sistema memorizza la password in testo non crittografato sul sistema. Per una maggiore sicurezza, utilizzate invece Windows Credential Manager.

È possibile impostare le variabili di ambiente utilizzando:

- Tipo setx.
- Pannello di controllo delle proprietà del sistema di Windows (scheda Avanzate).

- imposta variabili permanenti di ambiente di sistema Metodi programmatici.

Per impostare la variabile di ambiente di sistema:

CLOUDHSM_PIN=*<CU USERNAME>*:*<CU PASSWORD>*

Identifica un crypto user (CU) nell'HSM e fornisce tutte le informazioni di login richieste. La tua applicazione viene autenticata ed eseguita come questo CU. L'applicazione dispone delle autorizzazioni di questo CU e può visualizzare e gestire solo le chiavi di proprietà del CU e quelle con questi condivise. Per creare una nuova CU, utilizzate il comando user create nella CLI di CloudHSM. Per trovare quelli esistenti CUs, usa il comando user list nella CLI di CloudHSM.

Esempio:

```
setx /m CLOUDHSM_PIN test_user:password123
```

> **Note**
>
> Quando imposti le variabili di ambiente CLOUDHSM_PIN, devi evitare qualsiasi carattere speciale che possa essere interpretato dalla tua shell.

Tipi di chiavi supportati per Key Storage Provider (KSP) for AWS CloudHSM Client SDK 5

Il AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti tipi di chiavi con Client SDK 5.

Tipo di chiavi	Description
EC	Genera chiavi con le curve secp256r1 (P-256), secp384r1 (P-384) e secp521r1 (P-521).
RSA	Genera chiavi RSA a 2048, 3072 e 4096 bit.

Operazioni API supportate Key storage provider (KSP) per AWS CloudHSM Client SDK 5

I parametri del KSP sono definiti da Microsoft KSP. Per ulteriori informazioni, consulta la documentazione Microsoft.

Il Key Storage Provider (KSP) supporta le seguenti operazioni dell'API KSP per AWS CloudHSM Client SDK 5.

- `NCryptOpenStorageProvider`
- NCryptOpenKey
- NCryptCreatePersistedKey
- NCryptGetProperty
- NCryptSetProperty
- NCryptFinalizeKey
- NCryptDeleteKey
- NCryptFreeObject
- NCryptFreeBuffer
- NCryptIsAlgSupported
- NCryptEnumAlgorithms
- NCryptEnumKeys
- NCryptExportKey
- NCryptSignHash
- NCryptVerifySignature

NCryptOpenStorageProvider funzione con Key Storage Provider (KSP)

La `NCryptOpenStorageProvider` funzione carica e inizializza il Key Storage Provider (KSP).

Parameters

`phProvider`[fuori]

Un puntatore a una `NCRYPT_PROV_HANDLE` variabile che memorizza l'handle del provider.

`pszProviderName`[in]

Un puntatore a una stringa Unicode con terminazione nulla che identifica il provider di archiviazione delle chiavi. AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti valori:

Valore	Significato
«Provider di archiviazione chiavi CloudHSM»	Identifica il nome del provider Client SDK 5. Si consiglia di utilizzare questo nome per impostazione predefinita.
«Cavium Key Storage Provider»	Identifica il nome del provider Client SDK 3. Supporto della compatibilità con le versioni precedenti.

> **Note**
>
> I valori sono stringhe letterali a caratteri larghi, come indicato da L prima del valore letterale.

`dwFlags`[in]

Bandiere che modificano il comportamento della funzione. Nessun flag è definito per questa funzione.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.

Codice di restituzione	Description
NOTE_FAIL	L'operazione non è stata completata.

NCryptOpenKey con Key Storage Provider (KSP)

La `NCryptOpenKey` funzione apre una chiave esistente nel Key Storage Provider (KSP).

Parameters

`hProvider`[in]

L'handle KSP che contiene la chiave. Usa `NCryptOpenStorageProvider`per ottenere la maniglia.

`phKey`[fuori]

Un puntatore a una `NCRYPT_KEY_HANDLE` variabile che memorizza la maniglia della chiave.

`pszKeyName`[in]

Un puntatore a una stringa Unicode con terminazione nulla contenente il nome della chiave.

`dwLegacyKeySpec`[in, non utilizzato]

AWS CloudHSM Key Storage Provider (KSP) non utilizza questo parametro.

`dwFlags`[nel]

Bandiere che modificano il comportamento della funzione. Nessun flag è definito per questa funzione.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.

Codice di restituzione	Description
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hProvider`
NET_BAD_KEYSET	Il nome chiave fornito non ha restituito un risultato univoco.

NCryptCreatePersistedKey con Key Storage Provider (KSP)

La `NCryptCreatePersistedKey` funzione crea una nuova chiave e la memorizza nel Key Storage Provider (KSP). È possibile utilizzare la `NCryptSetProperty`funzione per impostarne le proprietà dopo la creazione. È necessario chiamare `NCryptFinalizeKey`prima di poter utilizzare la chiave.

Parameters

`hProvider`[in]

L'handle del provider di archiviazione delle chiavi in cui verrà creata la chiave. `NCryptOpenStorageProvider`Usalo per ottenere questa maniglia.

`phKey`[fuori]

L'indirizzo di una `NCRYPT_KEY_HANDLE` variabile che memorizza l'handle della chiave.

`pszAlgId`[in]

Un puntatore a una stringa Unicode con terminazione nulla che specifica l'identificatore dell'algoritmo crittografico per la creazione della chiave.

AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti algoritmi:

Costante/valore	Description
BCRYPT_RSA_ALGORITHM «RSA»	L'algoritmo a chiave pubblica RSA.
BCRYPT_ECDSA_P256_ALGORITHM	L'algoritmo di firma digitale a curva ellittica primaria a 256 bit (FIPS 186-2).

Costante/valore	Description
«ECDSA_P256"	
BCRYPT_ECDSA_P384_ALGORITHM «ECDSA_P384"	L'algoritmo di firma digitale a curva ellittica primaria a 384 bit (FIPS 186-2).
BCRYPT_ECDSA_P521_ALGORITHM «ECDSA_P521"	L'algoritmo di firma digitale a curva ellittica primaria a 521 bit (FIPS 186-2).

`pszKeyName`[in, opzionale]

Un puntatore a una stringa Unicode con terminazione nulla che contiene il nome della chiave. Se questo parametro è NULL, questa funzione creerà una chiave effimera che non è persistente.

`dwLegacyKeySpec`[in, non utilizzato]

AWS CloudHSM Key Storage Provider (KSP) non utilizza questo parametro.

`dwFlags`[nel]

Bandiere per modificare il comportamento della funzione. Utilizzate zero o più dei seguenti valori:

Valore	Significato
NCRYPT_MACHINE_KEY_FLAG	Questa bandiera non ha effetto.
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.
NCRYPT_OVERWRITE_KEY_FLAG	Specificando questo flag si sovrascrive qualsiasi chiave esistente con lo stesso nome nell'HSM. Senza questo flag, la funzione ritorna. NTE_EXISTS

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	La funzione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOT_NOT_SUPPORTED	Il `pszAlgId` parametro contiene un valore non supportato.
NTE_EXISTS	Esiste già una chiave con il nome specifica to e l'operazione non è stata utilizzata. `NCRYPT_OVERWRITE_KEY_FLAG`

NCryptGetProperty con Key Storage Provider (KSP)

La `NCryptGetProperty` funzione recupera i valori delle proprietà per un oggetto di archiviazione delle chiavi.

Parameters

`hObject`[in]

La maniglia dell'oggetto di cui si desidera recuperare la proprietà. È possibile utilizzare:

- Un provider handle () `NCRYPT_PROV_HANDLE`
- Una maniglia chiave (`NCRYPT_KEY_HANDLE`)

`pszProperty` [in]

Un puntatore a una stringa Unicode con terminazione nulla contenente il nome della proprietà da recuperare.

Quando si utilizza`NCRYPT_PROV_HANDLE`, AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti identificatori KSP:

Identificatore/valore	Description
NCRYPT_IMPL_TYPE_PROPERTY «Tipo Impl»	Un DWORD contenente flag che definiscono i dettagli di implementazione del provider
NCRYPT_MAX_NAME_LENGTH_PROPERTY «Lunghezza massima del nome»	Un DWORD contenente la lunghezza massima (in caratteri) per un nome di chiave persistente.
NCRYPT_NAME_PROPERTY «Nome»	Un puntatore a una stringa Unicode con terminazione nulla contenente il nome KSP.
NCRYPT_VERSION_PROPERTY «Versione»	Un DWORD contenente la versione del provider (parola alta: versione principale, parola bassa: versione minore).
NCRYPT_USE_CONTEXT_PROPERTY «Usa contesto»	Un puntatore a una stringa Unicode con terminazione nulla che descrive il contesto dell'operazione.
NCRYPT_SECURITY_DESCR_SUPPORT_PROPERTY «Security Descor Support»	Indica se il provider supporta i descrittori di sicurezza per le chiavi.

Quando viene utilizzato`NCRYPT_KEY_HANDLE`, AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti identificatori KSP:

Identificatore/valore	Description
NCRYPT_ALGORITHM_PROPERTY «Nome algoritmo»	Una stringa Unicode con terminazione nulla contenente il nome dell'algoritmo della chiave.
NCRYPT_BLOCK_LENGTH_PROPERTY	Un DWORD contenente la lunghezza del blocco di crittografia in byte.

Identificatore/valore	Description
«Lunghezza del blocco»	
NCRYPT_EXPORT_POLICY_PROPERTY «Politica di esportazione»	Un DWORD contenente flag che specifica no la politica di esportazione della chiave persistente.
NCRYPT_KEY_USAGE_PROPERTY «Utilizzo delle chiavi»	Un DWORD contenente flag che definiscono i dettagli di utilizzo delle chiavi.
NCRYPT_KEY_TYPE_PROPERTY «Tipo di chiave»	Un DWORD contenente flag che definiscono il tipo di chiave.
NCRYPT_LENGTH_PROPERTY «Lunghezza»	Un DWORD contenente la lunghezza della chiave in bit.
NCRYPT_LENGTHS_PROPERTY «Lunghezze»	Un puntatore a una struttura NCRYPT_SUPPORTED_LENGTHS contenente le dimensioni delle chiavi supportate.
NCRYPT_NAME_PROPERTY «Nome»	Un puntatore a una stringa Unicode con terminazione nulla contenente il nome della chiave.
NCRYPT_SECURITY_DESCR_PROPERTY «Descrizione di sicurezza»	Un puntatore a una struttura SECURITY_DESCRIPTOR contenente informazioni chiave sul controllo degli accessi.
NCRYPT_ALGORITHM_GROUP_PROPERTY «Gruppo di algoritmi»	Una stringa Unicode con terminazione nulla contenente il nome del gruppo di algoritmi dell'oggetto.
NCRYPT_UNIQUE_NAME_PROPERTY «Nome univoco»	Un puntatore a una stringa Unicode con terminazione nulla contenente il nome univoco della chiave.

> **Note**
>
> I valori sono stringhe letterali a caratteri larghi, come indicato da L prima del valore letterale.

`pbOutput`[fuori]

L'indirizzo di un buffer per memorizzare il valore della proprietà. Specificare la dimensione del buffer utilizzando. `cbOutput`

Per determinare la dimensione del buffer richiesta, impostate questo parametro su NULL. La funzione memorizza la dimensione richiesta (in byte) nella posizione indicata da. `pcbResult`

`cbOutput`[in]

La dimensione del `pbOutput` buffer in byte.

`pcbResult`[fuori]

Un puntatore a una variabile DWORD che memorizza il numero di byte copiati nel buffer. `pbOutput`

Se `pbOutput` è NULL, memorizza la dimensione richiesta (in byte).

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. Puoi usare zero o:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Quando pszProperty è`NCRYPT_SECURITY_DESCR_PROPERTY`, usa uno o una combinazione di:

Valore	Significato
INFORMAZIONI_DI_SICUREZZA_DEL PROPRIETARIO	Questa bandiera non ha effetto.

Valore	Significato
GROUP_SECURITY_INFORMATION	Questa bandiera non ha effetto.
DACL_SECURITY_INFORMATION	Questa bandiera non ha effetto.
LABEL_SECURITY_INFORMATION	Questa bandiera non ha effetto.
SACL_SECURITY_INFORMATION	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOT_NOT_SUPPORTED	Il `pszAlgId` parametro contiene un valore che non è supportato.
NTE_INVALID_HANDLE	L'handle in non è valido. `hObject`
NET_BUFFER_TOO_SMALL	Il `cbOutput` parametro è troppo piccolo per i valori restituiti.

NCryptSetProperty con Key Storage Provider (KSP)

La `NCryptSetProperty` funzione imposta i valori delle proprietà per un oggetto di archiviazione delle chiavi.

Parameters

`hObject`[in]

La maniglia dell'oggetto di cui si desidera impostare la proprietà. È possibile utilizzare:

- Un provider handle (`NCRYPT_PROV_HANDLE`)
- Una maniglia chiave (`NCRYPT_KEY_HANDLE`)

`pszProperty` [in]

Un puntatore a una stringa Unicode con terminazione nulla contenente il nome della proprietà da recuperare.

Quando si utilizza`NCRYPT_PROV_HANDLE`, AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti identificatori KSP:

Identificatore/valore	Description
NCRYPT_USE_CONTEXT_PROPERTY «Usa contesto»	Un puntatore a una stringa Unicode con terminazione nulla che descrive il contesto dell'operazione.

Quando viene utilizzato`NCRYPT_KEY_HANDLE`, AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti identificatori KSP:

Identificatore/valore	Description
NCRYPT_KEY_USAGE_PROPERTY «Utilizzo delle chiavi»	Un DWORD contenente una serie di flag che definiscono i dettagli di utilizzo delle chiavi. Questa proprietà si applica solo alle chiavi. Può contenere zero o una combinazione di uno o più dei seguenti valori. NCRYPT_ALLOW_DECRYPT_FLAG (0x00000001) NCRYPT_ALLOW_SIGNING_FLAG (0x00000002)

Identificatore/valore	Description
NCRYPT_LENGTH_PROPERTY «Lunghezza»	Un DWORD contenente la lunghezza della chiave in bit.
NCRYPT_EXPORT_POLICY_PROPERTY «Politica di esportazione»	Un DWORD contenente flag che specifica no la politica di esportazione della chiave persistente. Questo può contenere zero o una combinazione di uno o più dei seguenti valori. NCRYPT_ALLOW_EXPORT_FLAG (0x00000001)

> **Note**
>
> I valori sono stringhe letterali a caratteri larghi, come indicato da L prima del valore letterale.

`pbInput`[in]

L'indirizzo di un buffer che contiene il nuovo valore della proprietà. `cbInput`contiene la dimensione del buffer.

`cbInput`[in]

La dimensione del `pbInput` buffer in byte.

`dwFlags`[in]

Bandiere che modificano il comportamento della funzione. Nessun flag è definito per questa funzione.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOT_NOT_SUPPORTED	Il `pszProperty` parametro contiene un valore che non è supportato.
NTE_INVALID_HANDLE	L'handle in non è valido. `hObject`
NET_BAD_DATA	I dati indicati da `pbInput` e non sono validi. `cbInput`

NCryptFinalizeKey con Key Storage Provider (KSP)

La `NCryptFinalizeKey` funzione completa una chiave KSP. È necessario chiamare questa funzione prima di poter utilizzare il tasto.

Parameters

`hKey`[in]

La maniglia della chiave da completare. Ottieni questo handle chiamando la NCryptCreatePersistedKeyfunzione.

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. Puoi usare zero o questi valori:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.
NCRYPT_NO_KEY_VALIDATION	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NET_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hKey`
NOT_NOT_SUPPORTED	Il `dwFlags` parametro contiene un valore non supportato.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.

NCryptDeleteKey con Key Storage Provider (KSP)

La `NCryptDeleteKey` funzione elimina una chiave KSP dal Key Storage Provider (KSP).

Parameters

`hKey`[in]

La maniglia della chiave da eliminare.

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. È possibile utilizzare zero o più dei seguenti valori:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	La funzione ha avuto successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hKey`
NTE_INTERNAL_ERROR	Si è verificato un errore interno durante l'elimina zione della chiave.

NCryptFreeObject con Key Storage Provider (KSP)

La `NCryptFreeObject` funzione rilascia il provider o il key handle dal Key Storage Provider (KSP).

Parameters

`hObject`[in]

La maniglia dell'oggetto da rilasciare. È possibile utilizzare:

- Un provider handle (`NCRYPT_PROV_HANDLE`)
- Una maniglia chiave (`NCRYPT_KEY_HANDLE`)

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_HANDLE	L'handle in non è valido. `hObject`

NCryptFreeBuffer con Key Storage Provider (KSP)

La `NCryptFreeBuffer` funzione rilascia un blocco di memoria allocato dal Key Storage Provider (KSP).

Parameters

`pvInput`[in]

L'indirizzo della memoria da rilasciare.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NET_FAIL	L'operazione non è stata completata.

NCryptIsAlgSupported con Key Storage Provider (KSP)

NCryptIsAlgSupported la funzione determina se Key Storage Provider (KSP) supporta uno specifico algoritmo crittografico.

Parameters

`hProvider`[in]

L'handle del provider di archiviazione delle chiavi. `NCryptOpenStorageProvider`Usalo per prendere il manico.

`pszAlgId`[nel]

Un puntatore a una stringa Unicode con terminazione nulla che contiene l'identificatore dell'algoritmo crittografico per creare la chiave. AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti algoritmi:

Costante/valore	Description
BCRYPT_RSA_ALGORITHM «RSA»	L'algoritmo a chiave pubblica RSA.
BCRYPT_ECDSA_P256_ALGORITHM «ECDSA_P256"	L'algoritmo di firma digitale a curva ellittica primaria a 256 bit (FIPS 186-2).
BCRYPT_ECDSA_P384_ALGORITHM «ECDSA_P384"	L'algoritmo di firma digitale a curva ellittica primaria a 384 bit (FIPS 186-2).
BCRYPT_ECDSA_P521_ALGORITHM «ECDSA_P521"	L'algoritmo di firma digitale a curva ellittica primaria a 521 bit (FIPS 186-2).

`dwFlags`[nel]

Bandiere che modificano il comportamento delle funzioni. Può essere zero o il seguente valore:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOT_NOT_SUPPORTED	Il `pszAlgId` parametro contiene un valore non supportato.
NTE_INVALID_HANDLE	L'handle in non è valido. `hProvider`

NCryptEnumAlgorithms con Key Storage Provider (KSP)

La `NCryptEnumAlgorithms` funzione recupera i nomi degli algoritmi supportati dal Key Storage Provider (KSP).

Parameters

`hProvider`[in]

L'handle del provider di archiviazione delle chiavi per il quale enumerare gli algoritmi. Utilizzate la `NCryptOpenStorageProvider`funzione per ottenere questo handle.

`dwAlgOperations`[in]

Un insieme di valori che specificano quali classi di algoritmi enumerare. È possibile utilizzare zero per enumerare tutti gli algoritmi o combinare uno o più di questi valori:

Valore	Significato
NCRYPT_ASYMMETRIC_ENCRYPTIO N_OPERATION 0x00000004	Elenca gli algoritmi di crittografia asimmetrica.
NCRYPT_SIGNATURE_OPERATION	Elenca gli algoritmi di firma digitale.

Valore	Significato
0x00000010	

`pdwAlgCount`[fuori]

L'indirizzo di un DWORD che memorizza il numero di elementi nell'`ppAlgList`array.

`ppAlgList`[fuori]

L'indirizzo di un puntatore a `NCryptAlgorithmName` struttura che memorizza una matrice di nomi di algoritmi registrati. Il `pdwAlgCount` parametro indica il numero di elementi in questa matrice.

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. Usa zero o il seguente valore:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.

Codice di restituzione	Description
NOT_NOT_SUPPORTED	Il `dwAlgOperations` parametro contiene un valore non supportato.

NCryptEnumKeys con Key Storage Provider (KSP)

NCryptEnumKeys function elenca le chiavi memorizzate nel Key Storage Provider (KSP).

Parameters

`hProvider`[in]

L'handle del provider di archiviazione delle chiavi. `NCryptOpenStorageProvider`Usalo per ottenere questo handle.

`pszScope`[in, inutilizzato]

Imposta questo parametro su NULL.

`ppKeyName`[fuori]

Un indirizzo puntatore a una `NCryptKeyName` struttura che memorizza il nome della chiave. Per liberare questa memoria dopo l'uso, chiama`NCryptFreeBuffer`.

`ppEnumState`[dentro, fuori]

Un indirizzo puntatore VOID che tiene traccia dell'avanzamento dell'enumerazione. Il provider di archiviazione delle chiavi utilizza queste informazioni internamente per gestire la sequenza di enumerazione. Per iniziare una nuova enumerazione dall'inizio, imposta questo puntatore su NULL.

Per liberare questa memoria dopo aver completato l'enumerazione, passate questo puntatore a. `NCryptFreeBuffer`

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. Questa funzione non ha flag.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hProvider`
NOTE_NO_MORE_ITEMS	L'enumerazione ha elencato tutte le chiavi disponibili.

NCryptExportKey con Key Storage Provider (KSP)

La `NCryptExportKey` funzione esporta una chiave KSP in una memoria. BLOB Questa funzione supporta solo l'esportazione di chiavi pubbliche.

Parameters

`hKey`[in]

La maniglia della chiave da esportare.

`hExportKey`[in, inutilizzato]

AWS CloudHSM Key Storage Provider (KSP) non utilizza questo parametro.

`pszBlobType`[nel]

Una stringa Unicode con terminazione nulla che specifica il tipo da esportare. BLOB AWS CloudHSM Key Storage Provider (KSP) supporta i seguenti valori:

Valore	Significato
BCRYPT_RSAPUBLIC_BLOB	Esporta una chiave pubblica RSA. Il `pbOutput` buffer contiene una `BCRYPT_RSAKEY_BLOB` struttura seguita dai dati chiave.

Valore	Significato
BCRYPT_ECCPUBLIC_BLOB	Esporta una chiave pubblica. ECC Il `pbOutput` buffer contiene una `BCRYPT_ECCKEY_BLOB` struttura seguita dai dati chiave.

`pParameterList`[in, non utilizzato]

AWS CloudHSM Key Storage Provider (KSP) non utilizza questo parametro.

`pbOutput`[fuori, opzionale]

Un indirizzo buffer per memorizzare la chiave BLOB. Specificare la dimensione del buffer utilizzando. `cbOutput` Se impostata su NULL, la funzione memorizza la dimensione richiesta (in byte) nel DWORD a cui fa riferimento. `pcbResult`

`cbOutput`[in]

La dimensione del `pbOutput` buffer in byte.

`pcbResult`[fuori]

Un indirizzo variabile DWORD che memorizza il numero di byte copiati nel buffer. `pbOutput` Se `pbOutput` è NULL, la funzione memorizza la dimensione del buffer richiesta in byte.

`dwFlags`[in]

Bandiere che modificano il funzionamento della funzione. È possibile utilizzare zero o quanto segue:

Valore	Significato
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hProvider`
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOTE_BAD_KEY_STATE	Lo stato della chiave non è valido.
NOT_NOT_SUPPORTED	Il `dwFlags` parametro `pszBlobType` or contiene un valore non supportato.
STATUS_INTERNAL_ERROR	Si è verificato un errore interno durante l'operazione.

NCryptSignHash con Key Storage Provider (KSP)

La `NCryptSignHash` funzione crea una firma di un valore hash.

Parameters

`hKey`[in]

L'handle della chiave da usare per firmare l'hash.

`pPaddingInfo`[in, opzionale]

Un puntatore a una struttura contenente informazioni di riempimento. Il tipo di struttura dipende dal valore. `dwFlags` Utilizzate questo parametro solo con chiavi asimmetriche; impostatelo su NULL per altri tipi di chiavi.

`pbHashValue`[in]

Un puntatore a un buffer contenente il valore hash da firmare. Specificate la dimensione del buffer utilizzando. `cbHashValue`

`cbHashValue`[in]

La dimensione, in byte, del `pbHashValue` buffer da firmare.

`pbSignature`[fuori]

L'indirizzo di un buffer per memorizzare la firma. Specificare la dimensione del buffer utilizzando. `cbSignature`

Per determinare la dimensione del buffer richiesta, impostate questo parametro su NULL. La funzione memorizza la dimensione richiesta (in byte) nella posizione indicata da. `pcbResult`

`cbSignature`[in]

La dimensione del `pbSignature` buffer in byte. La funzione ignora questo parametro se `pbSignature` è NULL.

`pcbResult`[fuori]

Un puntatore a una variabile DWORD che memorizza il numero di byte copiati nel buffer. `pbSignature`

Se `pbSignature` è NULL, memorizza la dimensione del buffer richiesta, in byte.

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. I flag consentiti dipendono dal tipo di chiave. Usa uno di questi valori:

Valore	Significato
BCRYPT_PAD_ PKCS1	Utilizza lo schema di imbottitura. PKCS1 Impostato `pPaddingInfo` in modo che punti a una `BCRYPT_PKCS1_PADDING_INFO` struttura.
BCRYPT_PAD_PSS	Utilizza lo schema di imbottitura Probabilistic Signature Scheme (PSS). Imposta il `pPaddingInfo` parametro in modo che punti a una struttura. `BCRYPT_PSS_PADDING_INFO`
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hKey`
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NET_BUFFER_TOO_SMALL	Il `pcbOutput` parametro è troppo piccolo per i valori restituiti.
NET_BAD_KEY_STATE	Lo stato della chiave non è valido.
NTE_INTERNAL_ERROR	Si è verificato un errore interno durante la firma dell'hash.

NCryptVerifySignature con Key Storage Provider (KSP)

La `NCryptVerifySignature` funzione conferma se una firma corrisponde a un hash specificato.

Parameters

`hKey`[in]

L'handle della chiave da usare per decrittografare la firma. È necessario utilizzare la parte della chiave pubblica della coppia di chiavi utilizzata per firmare i dati con `NCryptSignHash`.

`pPaddingInfo`[in, opzionale]

Un puntatore a una struttura contenente informazioni di riempimento. Il tipo di struttura dipende dal valore. `dwFlags` Utilizzate questo parametro solo con chiavi asimmetriche; impostatelo su NULL per altri tipi di chiavi.

`pbHashValue`[in]

Un puntatore a un buffer contenente il valore hash da firmare. Specificate la dimensione del buffer utilizzando. `cbHashValue`

`cbHashValue`[in]

La dimensione del `pbHashValue` buffer in byte.

`pbSignature`[fuori]

L'indirizzo di un buffer contenente l'hash firmato dei dati. Usa `NCryptSignHash`per creare questa firma. Specificare la dimensione del buffer utilizzando`cbSignature`.

`cbSignature`[in]

La dimensione del `pbSignature` buffer in byte. Usa `NCryptSignHash`per creare la firma.

`dwFlags`[in]

Bandiere per modificare il comportamento della funzione. I flag consentiti dipendono dal tipo di chiave. Usa uno di questi valori:

Valore	Significato
NCRYPT_PAD_ _FLAG PKCS1	Indica il padding utilizzato per la firma. PKCS1 Impostato `pPaddingInfo` per puntare a una `BCRYPT_PKCS1_PADDING_INFO` struttura.
NCRYPT_PAD_PSS_FLAG	Indica il padding PSS (Probabilistic Signature Scheme) utilizzato per la firma. Impostato in `pPaddingInfo` modo che punti a una struttura. `BCRYPT_PSS_PADDING_INFO`
NCRYPT_SILENT_FLAG	Questa bandiera non ha effetto.

Valore restituito

La funzione restituisce un codice di stato per indicare l'esito positivo o negativo.

I codici di restituzione comuni includono:

Codice di restituzione	Description
ERROR_SUCCESS	L'operazione è stata completata con successo.
NTE_INVALID_PARAMETER	Uno o più parametri non sono validi.
NOTE_FAIL	L'operazione non è stata completata.
NTE_INVALID_HANDLE	L'handle in non è valido. `hKey`
NET_BAD_FLAGS	Il `dwFlags` parametro contiene un valore non valido.
NOTE_BAD_SIGNATURE	La firma non è stata verificata.
NET_BAD_KEY_STATE	Lo stato della chiave non è valido.
NTE_INTERNAL_ERROR	Si è verificato un errore interno durante la verifica della firma.

Configurazioni avanzate per KSP per AWS CloudHSM

Il AWS CloudHSM Key Storage Provider (KSP) include la seguente configurazione avanzata, che non fa parte delle configurazioni generali utilizzate dalla maggior parte dei clienti. Queste configurazioni offrono funzionalità aggiuntive.

- SDK3 modalità di compatibilità per KSP

SDK3 modalità di compatibilità per Key Storage Provider (KSP) per AWS CloudHSM

Key Storage Provider (KSP) implementa diversi approcci per l'interazione con le chiavi HSM:

- Client SDK 5: fornisce una comunicazione diretta con le chiavi memorizzate nell'HSM, eliminando la necessità di file di riferimento locali

- Client SDK 3: mantiene i file locali sul server Windows che fungono da riferimenti alle chiavi archiviate nell'HSM, utilizzando questi file per facilitare le operazioni chiave

Per i clienti che migrano da Client SDK 3 a Client SDK 5, l'attivazione dell'opzione della modalità di SDK3 compatibilità supporta le operazioni che utilizzano i file di riferimento chiave esistenti preservando al contempo l'architettura di archiviazione delle chiavi HSM sottostante.

Abilita la SDK3 modalità di compatibilità

Windows

Per abilitare la modalità di SDK3 compatibilità per Key Storage Provider (KSP) for Client SDK 5 in Windows

- È possibile utilizzare il seguente comando per abilitare la modalità di SDK3 compatibilità:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --enable-sdk3-compatibility-mode
```

Disabilita la modalità di SDK3 compatibilità

Windows

Per disabilitare la modalità di SDK3 compatibilità per Key Storage Provider (KSP) for Client SDK 5 in Windows

- È possibile utilizzare il seguente comando per disabilitare la modalità di SDK3 compatibilità:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-ksp.exe" --disable-sdk3-compatibility-mode
```

Provider JCE per AWS CloudHSM Client SDK 5

Il provider AWS CloudHSM JCE è un'implementazione del provider basata sul framework del provider Java Cryptographic Extension (JCE). JCE consente di eseguire operazioni di crittografia utilizzando Java Development Kit (JDK). In questa guida, il provider AWS CloudHSM JCE viene talvolta definito provider JCE. Utilizza il provider JCE e il JDK per trasferire le operazioni crittografiche sull'HSM. Per la risoluzione dei problemi, vedere. Problemi noti dell'SDK JCE per AWS CloudHSM

Per informazioni sull'utilizzo di Client SDK 3, vedi Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM.

Argomenti

- Installa il provider JCE per AWS CloudHSM Client SDK 5
- Tipi di chiavi supportati per il provider JCE per AWS CloudHSM Client SDK 5
- Nozioni di base sulla gestione delle chiavi nel provider JCE per AWS CloudHSM Client SDK 5
- Meccanismi supportati per il provider JCE per AWS CloudHSM Client SDK 5
- Attributi chiave Java supportati per AWS CloudHSM Client SDK 5
- Esempi di codice per AWS CloudHSM libreria software per Java for Client SDK 5
- AWS CloudHSM Fornitore JCE Javadocs
- AWS CloudHSM KeyStore Classe Java per Client SDK 5
- Configurazioni avanzate per AWS CloudHSM JCE for Client SDK 5

Installa il provider JCE per AWS CloudHSM Client SDK 5

Il provider JCE per AWS CloudHSM Client SDK 5 è compatibile con OpenJDK 17, OpenJDK 21 e OpenJDK 25. È possibile scaricare OpenJDK dal sito Web di OpenJDK.

> **Note**
>
> - Client SDK 5.17.0 è l'ultima versione a supportare OpenJDK 8.
> - Client SDK 5.17.1 è l'ultima versione a supportare OpenJDK 11.

Utilizza le seguenti sezioni per installare e fornire le credenziali al provider.

> **Note**
>
> Per eseguire un singolo cluster HSM con Client SDK 5, è necessario gestire innanzitutto le impostazioni di durabilità delle chiavi del client impostando `disable_key_availability_check` su `True`. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi e la pagina sullo strumento di configurazione di Client SDK 5.

Argomenti

- Fase 1: Installare il provider JCE
- Passaggio 2: fornire le credenziali al provider JCE

Fase 1: Installare il provider JCE

1. Utilizza i seguenti comandi per scaricare e installare il provider JCE.

   Amazon Linux 2023

   Installa il provider JCE per Amazon Linux 2023 sull'architettura x86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-jce-latest.amzn2023.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-jce-latest.amzn2023.x86_64.rpm
   ```

   Installa il provider JCE per Amazon Linux 2023 sull'architettura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Amzn2023/
   cloudhsm-jce-latest.amzn2023.aarch64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-jce-latest.amzn2023.aarch64.rpm
   ```

   Amazon Linux 2

   Installa il provider JCE per Amazon Linux 2 su architettura x86_64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-
   jce-latest.el7.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-jce-latest.el7.x86_64.rpm
   ```

   Installa il provider JCE per Amazon Linux 2 sull'architettura ARM64:

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-
   jce-latest.el7.aarch64.rpm
   ```

```
$ sudo yum install ./cloudhsm-jce-latest.el7.aarch64.rpm
```

RHEL 10 (10.0+)

Installa il provider JCE per RHEL 10 su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/
cloudhsm-jce-latest.el10.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el10.x86_64.rpm
```

Installa il provider JCE per RHEL 10 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL10/
cloudhsm-jce-latest.el10.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el10.aarch64.rpm
```

RHEL 9 (9.2+)

Installa il provider JCE per RHEL 9 (9.2+) sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
jce-latest.el9.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el9.x86_64.rpm
```

Installa il provider JCE per RHEL 9 (9.2+) sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL9/cloudhsm-
jce-latest.el9.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el9.aarch64.rpm
```

RHEL 8 (8.3+)

Installa il provider JCE per RHEL 8 sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el8.x86_64.rpm
```

Installa il provider JCE per RHEL 8 sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-jce-latest.el8.aarch64.rpm
```

```
$ sudo yum install ./cloudhsm-jce-latest.el8.aarch64.rpm
```

Ubuntu 26.04 LTS

Installa il provider JCE per Ubuntu 26.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-jce_latest_u26.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u26.04_amd64.deb
```

Installa il provider JCE per Ubuntu 26.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Resolute/cloudhsm-jce_latest_u26.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u26.04_arm64.deb
```

Ubuntu 24.04 LTS

Installa il provider JCE per Ubuntu 24.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-jce_latest_u24.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u24.04_amd64.deb
```

Installa il provider JCE per Ubuntu 24.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Noble/
cloudhsm-jce_latest_u24.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u24.04_arm64.deb
```

Ubuntu 22.04 LTS

Installa il provider JCE per Ubuntu 22.04 LTS sull'architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-jce_latest_u22.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u22.04_amd64.deb
```

Installa il provider JCE per Ubuntu 22.04 LTS sull'architettura ARM64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Jammy/
cloudhsm-jce_latest_u22.04_arm64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u22.04_arm64.deb
```

Windows Server

Installa il provider JCE per Windows Server sull'architettura x86_64, apri PowerShell come amministratore ed esegui il seguente comando:

```
PS C:\> wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/
Windows/AWSCloudHSMJCE-latest.msi -Outfile C:\AWSCloudHSMJCE-latest.msi
```

```
PS C:\> Start-Process msiexec.exe -ArgumentList '/i C:\AWSCloudHSMJCE-
latest.msi /quiet /norestart /log C:\client-install.txt' -Wait
```

2. Esegui il bootstrap di Client SDK 5. Per ulteriori informazioni sulle operazioni di bootstrap, consulta la pagina Esegui il bootstrap di Client SDK.
3. Individua i seguenti file del provider JCE:

   Linux

   - `/opt/cloudhsm/java/cloudhsm-<version>.jar`
   - `/opt/cloudhsm/bin/configure-jce`
   - `/opt/cloudhsm/bin/jce-info`

   Windows

   - `C:\Program Files\Amazon\CloudHSM\java\cloudhsm-<version>.jar>`
   - `C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe`
   - `C:\Program Files\Amazon\CloudHSM\bin\jce_info.exe`

Passaggio 2: fornire le credenziali al provider JCE

I moduli HSM necessitano di autenticare l'applicazione Java, prima che l'applicazione sia in grado di utilizzarli. I moduli HSM autenticano una sessione utilizzando un accesso esplicito o un metodo di accesso implicito.

Accesso esplicito questo metodo consente di fornire le credenziali AWS CloudHSM direttamente nell'applicazione. Utilizza il metodo dal `AuthProvider`, in cui si passa il nome utente CU, la password e l'ID nel modello di pin. Per ulteriori informazioni, vedi codice di esempio di Accesso a un HSM.

Accesso implicito questo metodo consente di impostare le credenziali AWS CloudHSM in un nuovo file di proprietà, proprietà del sistema, oppure come variabili di ambiente.

- Proprietà di sistema Imposta le credenziali attraverso le proprietà di sistema durante l'esecuzione di un'applicazione. I seguenti esempi mostrano due modi differenti con cui è possibile eseguire questa operazione:

Linux

```
$ java -DHSM_USER=<HSM user name> -DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_USER","<HSM user name>");
System.setProperty("HSM_PASSWORD","<password>");
```

Windows

```
PS C:\> java -DHSM_USER=<HSM user name> -DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_USER","<HSM user name>");
System.setProperty("HSM_PASSWORD","<password>");
```

- Variabili di ambiente Imposta le credenziali come variabili di ambiente.

> **Note**
>
> Quando impostate le variabili di ambiente, dovete evitare qualsiasi carattere speciale che possa essere interpretato dalla shell.

Linux

```
$ export HSM_USER=<HSM user name>
$ export HSM_PASSWORD=<password>
```

Windows

```
PS C:\> $Env:HSM_USER="<HSM user name>"
PS C:\> $Env:HSM_PASSWORD="<password>"
```

Le credenziali potrebbero non essere disponibili se l'applicazione non le fornisce o se viene eseguita un'operazione prima che l'HSM autentichi la sessione. In questi casi, la libreria software CloudHSM per Java cerca le credenziali nel seguente ordine:

1. Proprietà di sistema

2. Variabili di ambiente

Tipi di chiavi supportati per il provider JCE per AWS CloudHSM Client SDK 5

La libreria AWS CloudHSM software per Java consente di generare i seguenti tipi di chiavi.

Tipo di chiavi	Description
AES	Genera chiavi AES a 128, 192 e 256 bit.
Triple DES (3DES, DESede)	Genera una chiave Triple DES a 192 bit. [1]
EC	Genera coppie di chiavi EC: curve NIST secp224r1 (P-224), secp256r1 (), secp256k1 (Blockchain), secp384r1 (P-256), secp521r1 () e ed25519. P-384 P-521 [2]
GENERIC_SECRET	Genera segreti generici da 1 a 800 byte.
HMAC	Supporto hash per SHA1, SHA224, SHA256, SHA384, SHA512.
ML-DSA	ML-DSA[3]Genera coppie ML-DSA-65 di chiavi con set di parametri, e. ML-DSA-44 ML-DSA-87
RSA	General chiavi RSA da 2048-bit a 4096-bit, con incrementi di 256 bit.

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

[2] Supportato solo su istanze hsm2m.medium in modalità non FIPS.

[3] A partire dal 1° settembre 2026, ML-DSA è disponibile in modalità FIPS per i cluster hsm2m.medium.

Nozioni di base sulla gestione delle chiavi nel provider JCE per AWS CloudHSM Client SDK 5

Le nozioni di base sulla gestione delle chiavi nel provider JCE implicano l'importazione, l'esportazione, il caricamento tramite handle, oppure l'eliminazione di chiavi. Per ulteriori informazioni su come gestire le chiavi, vedi l'esempio di codice Gestire le chiavi.

Puoi inoltre trovare ulteriori esempi di codice del provider JCE all'indirizzoEsempi di codice.

Meccanismi supportati per il provider JCE per AWS CloudHSM Client SDK 5

Questo argomento fornisce informazioni sui meccanismi supportati per il provider JCE con AWS CloudHSM Client SDK 5. Per informazioni sulle interfacce Java Cryptography Architecture (JCA) e sulle classi di motore supportate da AWS CloudHSM, consulta i seguenti argomenti.

Argomenti

- Funzioni di generazione chiavi e coppie di chiavi
- Funzioni di cifratura
- Funzioni di firma e verifica
- Funzioni di digest
- Hash-based funzioni del codice di autenticazione dei messaggi (HMAC)
- Cipher-based funzioni del codice di autenticazione dei messaggi (CMAC)
- Funzioni chiave dell'accordo
- Converti le chiavi in specifiche chiave utilizzando le principali fabbriche
- Annotazioni sui meccanismi

Funzioni di generazione chiavi e coppie di chiavi

La libreria AWS CloudHSM software per Java consente di utilizzare le seguenti operazioni per generare funzioni di chiave e coppia di chiavi.

- `RSA`
- `EC`
- `EdDSA (Ed25519)`[vedi nota2]
- `ML-DSA`

- AES
- DESede (Triple DES)[vedi nota1]
- GenericSecret

Funzioni di cifratura

La libreria AWS CloudHSM software per Java supporta le seguenti combinazioni di algoritmo, modalità e spaziatura interna.

Algoritmo	Modalità	Padding	Note
AES	CBC	AES/CBC/NoPadding AES/CBC/PKCS5Padding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE . Implementa Cipher.UNWRAP_MODE for AES/CBC NoPadding
AES	ECB	AES/ECB/PKCS5Padding AES/ECB/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE .
AES	CTR	AES/CTR/NoPadding	Implementa Cipher.ENCRYPT_MODE e Cipher.DECRYPT_MODE .
AES	GCM	AES/GCM/NoPadding	Implementa Cipher.WR

Algoritmo	Modalità	Padding	Note
			`AP_MODE` , `Cipher.UN WRAP_MODE` , `Cipher.EN CRYPT_MODE` e `Cipher.DE CRYPT_MODE` . Quando esegue la AES-GCM crittogra fia, l'HSM ignora il vettore di inizializ zazione (IV) nella richiesta e utilizza un IV generato. Al termine dell'oper azione, è necessario chiamare `Cipher.ge tIV()` per ottenere il IV.

AESWrap	ECB	`AESWrap/ECB/ NoPadding`
		
		`AESWrap/ECB/ PKCS5Padding`
		
		`AESWrap/ECB/ ZeroPadding`
			Implementa `Cipher.WR AP_MODE` e `Cipher.UN WRAP_MODE` .

Algoritmo	Modalità	Padding	Note

DESede (Triple DES)	CBC	`DESede/CBC/PKCS5Padding`
		
		`DESede/CBC/NoPadding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`. Vedi la nota 1 di seguito per una modifica imminente.

DESede (Triple DES)	ECB	`DESede/ECB/NoPadding`
		
		`DESede/ECB/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`. Vedi la nota 1 di seguito per una modifica imminente.

Algoritmo	Modalità	Padding	Note

RSA	ECB	`RSA/ECB/PKCS1Padding` vedi nota 1
		
		`RSA/ECB/OAEPPadding`
		
		`RSA/ECB/OAEPWithSHA-1ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-224ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-256ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-384ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-512ANDMGF1Padding`
			Implementa `Cipher.WRAP_MODE`, `Cipher.UNWRAP_MODE`, `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.

Algoritmo	Modalità	Padding	Note
RSA	ECB	`RSA/ECB/NoPadding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`.

RSAAESWrap	ECB	`RSAAESWrap/ECB/OAEPPadding`
		
		`RSAAESWrap/ECB/OAEPWithSHA-1ANDMGF1Padding`
		
		`RSAAESWrap/ECB/OAEPWithSHA-224ANDMGF1Padding`
		
		`RSAAESWrap/ECB/OAEPWithSHA-256ANDMGF1Padding`
		
		`RSAAESWrap/ECB/OAEPWithSHA-384ANDMGF1Padding`
		
		`RSAAESWrap/ECB/OAEPWithSHA-512ANDMGF1Padding`
			Implementa `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`.

Funzioni di firma e verifica

La libreria AWS CloudHSM software per Java supporta i seguenti tipi di firma e verifica. Con Client SDK 5 e gli algoritmi di firma con hashing, i dati vengono sottoposti a hashing localmente nel software prima di essere inviati all'HSM per. signature/verification Ciò significa che non ci sono limiti alla dimensione dei dati che possono essere sottoposti a hash dall'SDK.

Tipi di firma RSA

- `NONEwithRSA`
- `RSASSA-PSS`
- `SHA1withRSA`
- `SHA1withRSA/PSS`
- `SHA1withRSAandMGF1`
- `SHA224withRSA`
- `SHA224withRSAandMGF1`
- `SHA224withRSA/PSS`
- `SHA256withRSA`
- `SHA256withRSAandMGF1`
- `SHA256withRSA/PSS`
- `SHA384withRSA`
- `SHA384withRSAandMGF1`
- `SHA384withRSA/PSS`
- `SHA512withRSA`
- `SHA512withRSAandMGF1`
- `SHA512withRSA/PSS`

Tipi di firma ECDSA

- `NONEwithECDSA`
- `SHA1withECDSA`
- `SHA224withECDSA`
- `SHA256withECDSA`
- `SHA384withECDSA`

- SHA512withECDSA

ML-DSA tipi di firma

- ML-DSA
- ML-DSA-44
- ML-DSA-65
- ML-DSA-87
- ML-DSA-EXTERNAL-MU

Tipi di firma EdDSA vedi nota [2]

- Ed25519o EdDSA (Dimensione massima dei dati: 16.000 byte)
- Ed25519ph

Funzioni di digest

La libreria AWS CloudHSM software per Java supporta i seguenti riassunti dei messaggi. Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non ci sono limiti alla dimensione dei dati che possono essere sottoposti a hash dall'SDK.

- SHA-1
- SHA-224
- SHA-256
- SHA-384
- SHA-512

Hash-based funzioni del codice di autenticazione dei messaggi (HMAC)

La libreria AWS CloudHSM software per Java supporta i seguenti algoritmi HMAC.

- HmacSHA1(Dimensione massima dei dati in byte: 16288)
- HmacSHA224(Dimensione massima dei dati in byte: 16256)
- HmacSHA256(Dimensione massima dei dati in byte: 16288)
- HmacSHA384(Dimensione massima dei dati in byte: 16224)

- `HmacSHA512`(Dimensione massima dei dati in byte: 16224)

Cipher-based funzioni del codice di autenticazione dei messaggi (CMAC)

I CMAC (codici di autenticazione dei Cipher-based messaggi) creano codici di autenticazione dei messaggi (MAC) utilizzando un codice a blocchi e una chiave segreta. Differiscono dagli HMAC in quanto utilizzano un metodo a chiave simmetrica a blocchi per i MAC anziché un metodo di hashing.

La libreria AWS CloudHSM software per Java supporta i seguenti algoritmi CMAC.

- `AESCMAC`

Funzioni chiave dell'accordo

La libreria AWS CloudHSM software per Java supporta ECDH con Key Derivation Functions (KDF). Sono supportati i seguenti tipi di KDF:

- `ECDHwithX963SHA1KDF`Supporta l'algoritmo X9.63 KDF SHA1 [3]
- `ECDHwithX963SHA224KDF`Supporta l'algoritmo X9.63 KDF SHA224 [3]
- `ECDHwithX963SHA256KDF`Supporta l'algoritmo KDF SHA256 X9.63 [3]
- `ECDHwithX963SHA384KDF`Supporta l'algoritmo KDF SHA384 X9.63 [3]
- `ECDHwithX963SHA512KDF`Supporta l'algoritmo X9.63 KDF SHA512 [3]

Converti le chiavi in specifiche chiave utilizzando le principali fabbriche

È possibile utilizzare le fabbriche di chiavi per convertire le chiavi in specifiche chiave. AWS CloudHSM ha due tipi di fabbriche chiave per JCE:

SecretKeyFactory: Utilizzato per importare o derivare chiavi simmetriche. Usando SecretKeyFactory, puoi passare una chiave supportata o una supportata in cui importare o KeySpec derivare chiavi simmetriche. AWS CloudHSM Di seguito sono riportate le specifiche supportate per: KeyFactory

- Le seguenti KeySpec classi SecretKeyFactory del `generateSecret` metodo For sono supportate:
  - KeyAttributesMappuò essere utilizzato per importare byte di chiave con attributi aggiuntivi come chiave CloudHSM. Un esempio può essere trovato qui.

- SecretKeySpecpuò essere utilizzato per importare una specifica di chiave simmetrica come chiave CloudHSM.
- AesCmacKdfParameterSpecpuò essere utilizzato per derivare chiavi simmetriche utilizzando un'altra chiave AES CloudHSM.

> **Note**
>
> SecretKeyFactoryil `translateKey` metodo accetta qualsiasi chiave che implementa l'interfaccia chiave. https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/security/Key.html

KeyFactory: Utilizzato per importare chiavi asimmetriche. Usando KeyFactory, puoi passare una chiave supportata o supportata in KeySpec cui importare una chiave asimmetrica. AWS CloudHSM Per ulteriori informazioni, fai riferimento a queste risorse:

- KeyFactoryIl `generatePublic` metodo di For, sono supportate le seguenti KeySpec classi:
  - CloudHSM KeyAttributesMap per RSA, EC ed EdDSA. KeyTypes Un esempio può essere trovato qui
  - X509EncodedKeySpecper RSA, EC ed EdDSA Public Key
  - RSAPublicKeySpecper RSA Public Key
  - ECPublicKeySpecper EC Public Key (non applicabile per Ed25519)
- Per il `generatePrivate` metodo KeyFactory For, sono supportate le seguenti KeySpec classi:
  - CloudHSM KeyAttributesMap per RSA, EC ed EdDSA. KeyTypes Un esempio può essere trovato qui
  - PKCS8EncodedKeySpecper chiavi private RSA, EC ed EdDSA
  - RSAPrivateCrtKeySpecper RSA Private Key
  - ECPrivateKeySpecper EC Private Key (non applicabile per Ed25519)

> **ML-DSA limitazione dell'importazione delle chiavi**
>
> AWS CloudHSM non supporta l'importazione di ML-DSA chiavi tramite `KeyFactory` (`generatePublicegeneratePrivate`). Per utilizzare ML-DSA le chiavi, generale direttamente sull'HSM utilizzando. `KeyPairGenerator` ML-DSA

`KeyFactory`Supporta l'esportazione di chiavi pubbliche utilizzando `getKeySpec` (restituisce`X509EncodedKeySpec`).

KeyFactoryIl `translateKey` metodo di For, accetta qualsiasi chiave che implementa l'interfaccia https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/security/Key.html chiave.

Annotazioni sui meccanismi

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

[2] Supportato solo su istanze hsm2m.medium in modalità non FIPS.

[3] Le funzioni di derivazione chiave (KDF) sono specificate nella pubblicazione speciale 800-56A del NIST, revisione 3. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf

Attributi chiave Java supportati per AWS CloudHSM Client SDK 5

Questo argomento fornisce informazioni sugli attributi chiave Java supportati per AWS CloudHSM Client SDK 5. In questo argomento viene descritto come utilizzare un'estensione proprietaria per il provider JCE per impostare attributi chiave. Utilizzare questa estensione per impostare gli attributi della chiave supportati e i relativi valori durante le operazioni seguenti:

- Generazione delle chiavi
- Importazione delle chiavi

Per esempi di utilizzo degli attributi chiave, vedi the section called "Esempi di codice".

Argomenti

- Comprensione degli attributi
- Attributi supportati
- Impostazione attributi per una chiave

Comprensione degli attributi

Gli attributi chiave vengono utilizzati per specificare le operazioni consentite su oggetti chiave, incluse le chiavi pubbliche, private o segrete. Gli attributi e i valori della chiave vengono definiti durante le operazioni di creazione degli oggetti chiave.

Tuttavia, la Java Cryptography Extension (JCE) non specifica come impostare i valori sugli attributi chiave, pertanto la maggior parte delle operazioni erano consentite per impostazione predefinita. Al contrario, lo standard PKCS# 11 definisce un set completo di attributi con valori predefiniti più restrittivi. A partire dal provider JCE 3.1, AWS CloudHSM fornisce un'estensione proprietaria che consente di impostare valori più restrittivi per gli attributi di uso comune.

Attributi supportati

Puoi impostare i valori per gli attributi elencati nella tabella sottostante. Come best practice, imposta i valori solo per gli attributi che desideri rendere restrittivi. Se non si specifica un valore, AWS CloudHSM utilizza il valore predefinito specificato nella tabella seguente. Una cella vuota nella colonna Valore predefinito indica che all'attributo non è stato assegnato alcun valore predefinito specifico.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
DECRYPT	TRUE		TRUE[1]	Il valore Vero indica che è possibile utilizzare la chiave per decodificare qualsiasi buffer. Questo attributo è in genere impostato su FALSO per una chiave il cui WRAP è

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
				impostato su vero.
DERIVE				Consente di utilizzare una chiave per derivare altre chiavi.
ENCRYPT	TRUE	TRUE[1]		Il valore Vero indica che è possibile utilizzare la chiave per crittografare qualsiasi buffer.
EXTRACTABLE	TRUE		TRUE	Il valore Vero indica che è possibile esportare questa chiave dall'HSM.
ID				Un valore definito dall'utente utilizzato per identificare la chiave.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
KEY_TYPE				Utilizzato per identificare il tipo di chiave (AES, DeSede, general secret, EC, RSA o). ML-DSA
LABEL				Una stringa definita dall'utente che consente di identificare comodamente le chiavi sull'HSM. Per seguire le best practice, utilizza un'etichetta univoca per ogni chiave in modo che sia più facile trovarla in seguito.
LOCAL				Indica una chiave generata dall'HSM.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
MLDSA_ALGORITHM				Specifica il set di ML-DSA parametri utilizzato per la generazione di coppie di ML-DSA chiavi. Valori validi: ML-DSA-44 ML-DSA-65, e ML-DSA-87.
OBJECT_CLASS				Utilizzato per identificare la classe oggetto di una chiave (SecretKey, PublicKey o PrivateKey).

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
PRIVATE	TRUE	TRUE	TRUE	Il valore Vero indica che un utente potrebbe non poter accedere alla chiave finché l'utente non viene autentica to. Per maggiore chiarezza , gli utenti non possono accedere a nessuna chiave AWS CloudHSM finché non vengono autenticati, anche se questo attributo è impostato su FALSE.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
SIGN	TRUE		TRUE	Il valore Vero indica che è possibile utilizzare la chiave per firmare un messaggio di digest. In genere viene impostato su FALSO per le chiavi pubbliche e per le chiavi private archiviate.
SIZE				Un attributo che definisce la dimensione di una chiave. Per maggiori dettagli sulle dimensioni delle chiavi supportate, vedi Meccanismi supportati per Client SDK 5.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
TOKEN	FALSE	FALSE	FALSE	Una chiave permanent e replicata in tutti i moduli HSM nel cluster e inclusa nei backup. TOKEN = FALSO implica una chiave effimera, che viene cancellat a automatic amente quando la connessione al HSM viene interrotta o ci si disconnette.
UNWRAP	TRUE		TRUE[1]	Il valore Vero indica che è possibile utilizzar e la chiave per annullare il wrapping (importazione) di un'altra chiave.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
VERIFY	TRUE	TRUE		Il valore Vero indica che è possibile utilizzare la chiave per verificare una firma. In genere è impostato su FALSO per chiavi private.
WRAP	TRUE	TRUE[1]		Il valore Vero indica che è possibile utilizzare la chiave per eseguire il wrapping di un'altra chiave. In genere viene impostato su FALSO per chiavi private.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
`WRAP_WITH_TRUSTED`	`FALSE`		`FALSE`	Il valore Vero indica che una chiave può essere soggetta a wrapping e ad annullamento del wrapping con chiavi con l'attributo `TRUSTED` impostato su vero. Una volta che una chiave ha il valore `WRAP_WITH_TRUSTED` impostato su vero, tale attributo è di sola lettura e non può essere impostato su falso. Per saperne di più sul wrapping di fiducia, vedi Utilizzo di chiavi fidate per controllare l'annullamento

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
				del wrapping delle chiavi.

[1] Per le chiavi Ed25519, ENCRYPT, WRAP, DECRYPT e UNWRAP non funzionano. Le chiavi Ed25519 sono di sola firma e non supportano le operazioni di crittografia o wrapping. Quando si importano le chiavi Ed25519, questi attributi sono impostati su FALSE.

> **Note**
>
> È possibile ottenere un supporto più ampio per gli attributi nella libreria PKCS #11. Per ulteriori informazioni, vedi Attributi PKCS #11 supportati.

Impostazione attributi per una chiave

`KeyAttributesMap`è un Map-like oggetto Java, che è possibile utilizzare per impostare i valori degli attributi per gli oggetti chiave. I metodi per la funzione `KeyAttributesMap` sono simili a quelli utilizzati per la manipolazione della mappa Java.

Per impostare valori personalizzati sugli attributi, sono disponibili due opzioni:

- Utilizzare i metodi elencati nella tabella seguente
- Utilizzare i modelli di generatore illustrati più avanti in questo documento

Gli oggetti della mappa attributi supportano i seguenti metodi per impostare gli attributi:

Operation	Valore restituito	Metodo **KeyAttributesMap**
Ottenere il valore di un attributo chiave per una chiave esistente	Oggetto (contenente il valore) o nulla	get(keyAttribute)
Inserire il valore di un attributo chiave	Il valore precedente associato all'attributo chiave o nulla se non esiste alcuna mappatura per un attributo chiave	put(keyAttribute, value)
Compilare i valori per più attributi chiave	N/A	putAll (chiaveAttributesMap)
Rimuovere una coppia chiave-valore dalla mappa degli attributi	Il valore precedente associato all'attributo chiave o nulla se non esiste alcuna mappatura per un attributo chiave	remove(keyAttribute)

> **Note**
>
> Eventuali attributi non specificati in modo esplicito vengono impostati sui valori predefiniti elencati nella tabella precedente in the section called "Attributi supportati".

Impostazione di attributi per una coppia di chiavi

Utilizza la classe Java `KeyPairAttributesMap` per gestire gli attributi chiave per una coppia di chiavi. `KeyPairAttributesMap` incapsula due oggetti `KeyAttributesMap`; uno per una chiave pubblica e uno per una chiave privata.

Per impostare singoli attributi per la chiave pubblica e la chiave privata separatamente, puoi utilizzare il metodo `put()` sull'oggetto mappa `KeyAttributes` corrispondente per tale chiave. Utilizza il metodo `getPublic()` per recuperare la mappa degli attributi per la chiave pubblica e utilizza `getPrivate()` per recuperare la mappa degli attributi per la chiave privata. Compila il valore di più

attributi chiave insieme per coppie di chiavi pubbliche e private utilizzando la `putAll()` con una mappa degli attributi della coppia di chiavi come relativo argomento.

Esempi di codice per AWS CloudHSM libreria software per Java for Client SDK 5

Questo argomento fornisce risorse e informazioni sugli esempi di codice Java per AWS CloudHSM Client SDK 5.

Prerequisiti

Prima di eseguire gli esempi, devi configurare l'ambiente:

- Installa e configura il provider Java Cryptographic Extension (JCE).
- Configura un nome utente e una password HSM validi. Le autorizzazioni per l'utente di crittografia (CU) sono sufficienti per queste attività. L'applicazione utilizza queste credenziali per accedere all'HSM in ciascun esempio.
- Decidi come fornire le credenziali al provider JCE.

Esempi di codice

I seguenti esempi di codice mostrano come utilizzare il provider JCE AWS CloudHSM per eseguire attività di base. Altri esempi di codice sono disponibili su. GitHub

- Esegui l'accesso a un modulo HSM
- Gestisci chiavi
- Genera di chiavi simmetriche
- Genera chiavi asimmetriche
- Crittografa e decrittografa con AES-GCM
- Crittografa e decrittografa con AES-CTR
- Crittografa e decrittografa con vedi nota DESede-ECB [1]
- Firma e verifica con chiavi RSA
- Firma e verifica con Chiavi EC
- Firma e verifica con le chiavi EdDSA (vedi nota) [2]
- Firma e verifica con le chiavi ML-DSA
- Usa attributi chiave supportati

- Utilizzo dell'archivio delle chiavi CloudHSM

[1] In conformità alle linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

[2] Supportato solo su istanze hsm2m.medium in modalità non FIPS.

AWS CloudHSM Fornitore JCE Javadocs

Utilizza il provider JCE Javadocs per ottenere informazioni sull'utilizzo dei tipi e dei metodi Java definiti nell'SDK JCE di AWS CloudHSM. Per scaricare la versione più recente di Javadoc per AWS CloudHSM, consultate la AWS CloudHSM ultima versione di Client SDK sezione della pagina Download.

È possibile importare Javadocs in un ambiente di sviluppo integrato (IDE) o visualizzarli in un Web browser.

AWS CloudHSM KeyStore Classe Java per Client SDK 5

La AWS CloudHSM `KeyStore` classe fornisce un archivio di chiavi per scopi speciali PKCS12 . Questo archivio chiavi può archiviare i certificati insieme ai dati della chiave e correlarli ai dati della chiave memorizzati su AWS CloudHSM. La AWS CloudHSM `KeyStore` classe implementa la `KeyStore` Service Provider Interface (SPI) della Java Cryptography Extension (JCE). Per ulteriori informazioni sull'utilizzo`KeyStore`, vedete Class. KeyStore

> **Note**
>
> Poiché i certificati sono informazioni pubbliche e, per massimizzare la capacità di archiviazione delle chiavi crittografiche, non AWS CloudHSM supporta l'archiviazione dei certificati su. HSMs

Scegli l'archivio di chiavi appropriato per AWS CloudHSM Client SDK 5

Il provider AWS CloudHSM Java Cryptographic Extension (JCE) offre un AWS CloudHSM per scopi speciali. KeyStore La AWS CloudHSM `KeyStore` classe supporta l'offload delle operazioni chiave sull'HSM, l'archiviazione locale dei certificati e le operazioni basate sui certificati.

Caricate il CloudHSM per scopi speciali come segue: KeyStore

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inizializza il Client SDK 5 AWS CloudHSM KeyStore

Effettua AWS CloudHSM KeyStore l'accesso nello stesso modo in cui accedi al provider JCE. È possibile utilizzare le variabili di ambiente o il file delle proprietà del sistema ed è necessario effettuare il login prima di iniziare a utilizzare CloudHSM KeyStore. Per un esempio di accesso a un HSM utilizzando JCE, vedi Accedi a un HSM.

Se lo desideri, puoi specificare una password per crittografare il PKCS12 file locale che contiene i dati dell'archivio delle chiavi. Quando si crea il AWS CloudHSM Keystore, si imposta la password e la si fornisce quando si utilizzano i metodi load, set e get.

Crea un'istanza di un nuovo oggetto CloudHSM come segue KeyStore :

```
ks.load(null, null);
```

Scrivi i dati dell'archivio chiavi in un file utilizzando il metodo `store`. Da quel momento in poi, puoi caricare l'archivio chiavi esistente utilizzando il metodo `load` con il file sorgente e la password come segue:

```
ks.load(inputStream, password);
```

Usa il nostro Client SDK 5 AWS CloudHSM KeyStore AWS CloudHSM

AWS CloudHSM KeyStore è conforme alle KeyStore specifiche della classe JCE e fornisce le seguenti funzioni.

- `load`

  Carica l'archivio chiavi dal flusso di input specificato. Se durante il salvataggio dell'archivio chiavi è stata impostata una password, è necessario fornire questa stessa password affinché il caricamento abbia esito positivo. Impostare entrambi i parametri su nulla per inizializzare un nuovo archivio di chiavi vuoto.

  ```
  KeyStore ks = KeyStore.getInstance("CloudHSM");
  ks.load(inputStream, password);
  ```

- `aliases`

  Restituisce un'enumerazione dei nomi alias di tutte le voci nell'istanza dell'archivio chiavi considerato. I risultati includono oggetti archiviati localmente nel PKCS12 file e oggetti residenti nell'HSM.

  Esempio di codice:

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
for(Enumeration<String> entry = ks.aliases(); entry.hasMoreElements();) {
    String label = entry.nextElement();
    System.out.println(label);
}
```

- `containsalias`

  Restituisce vero se l'archivio chiavi ha accesso ad almeno un oggetto con l'alias specificato. L'archivio delle chiavi controlla gli oggetti archiviati localmente nel PKCS12 file e gli oggetti che risiedono sull'HSM.

- `deleteEntry`

  Elimina una voce di certificato dal file locale PKCS12 . L'eliminazione dei dati chiave memorizzati in un HSM non è supportata utilizzando. AWS CloudHSM KeyStore È possibile eliminare le chiavi utilizzando il metodo `destroy` dell'interfaccia Distruggibile.

```
((Destroyable) key).destroy();
```

- `getCertificate`

  Restituisce il certificato associato a un alias, se disponibile. Se l'alias non esiste o fa riferimento a un oggetto che non è un certificato, la funzione restituisce NULLA.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
Certificate cert = ks.getCertificate(alias);
```

- `getCertificateAlias`

  Restituisce il nome (alias) della prima voce dell'archivio chiavi i cui dati corrispondono al certificato specificato.

```
KeyStore ks = KeyStore.getInstance("CloudHSM");
```

```
String alias = ks.getCertificateAlias(cert);
```

- `getCertificateChain`

  Restituisce la catena di certificati associata all'alias specificato. Se l'alias non esiste o fa riferimento a un oggetto che non è un certificato, la funzione restituisce NULLA.

- `getCreationDate`

  Restituisce la data di creazione della voce identificata dall'alias specificato. Se una data di creazione non è disponibile, la funzione restituisce la data in cui il certificato è diventato valido.

- `getKey`

  GetKey viene passato all'HSM e restituisce un oggetto chiave corrispondente all'etichetta specificata. Poiché interroga `getKey` direttamente l'HSM, può essere utilizzato per qualsiasi chiave sull'HSM indipendentemente dal fatto che sia stata generata da. KeyStore

  ```
  Key key = ks.getKey(keyLabel, null);
  ```

- `isCertificateEntry`

  Controlla se la voce con l'alias specificato rappresenta una voce di certificato.

- `isKeyEntry`

  Controlla se la voce con l'alias specificato rappresenta una voce chiave. L'azione cerca l'alias sia nel PKCS12 file che nell'HSM.

- `setCertificateEntry`

  Assegna il certificato dato all'alias specificato. Se l'alias specificato è già in uso per identificare una chiave o un certificato, viene generata una `KeyStoreException`. È possibile utilizzare il codice JCE per ottenere l'oggetto chiave e quindi utilizzare il KeyStore `SetKeyEntry` metodo per associare il certificato alla chiave.

- `setKeyEntry` con chiave `byte[]`

  Questa API non è attualmente supportata con Client SDK 5.

- `setKeyEntry` con oggetto `Key`

  Assegna la chiave considerata all'alias specificato e la memorizza all'interno dell'HSM. Se la chiave non esiste già all'interno dell'HSM, verrà importata nell'HSM come chiave di sessione estraibile.

Se l'oggetto `Key` è di tipo `PrivateKey`, deve essere accompagnato da una catena di certificati corrispondente.

Se l'alias esiste già, la chiamata `SetKeyEntry` genera un `KeyStoreException` e impedisce la sovrascrittura della chiave. Se la chiave deve essere sovrascritta, utilizza KMU o JCE a tale scopo.

- `engineSize`

  Restituisce il numero di voci nell'archivio chiavi.

- `store`

  Memorizza l'archivio delle chiavi nel flusso di output specificato come PKCS12 file e lo protegge con la password specificata. Inoltre, persiste tutte le chiavi caricate (che sono impostate usando le chiamate `setKey`).

Configurazioni avanzate per AWS CloudHSM JCE for Client SDK 5

Il provider AWS CloudHSM JCE include le seguenti configurazioni avanzate, che non fanno parte delle configurazioni generali utilizzate dalla maggior parte dei clienti.

- Connessione a più cluster
- Estrazione delle chiavi tramite JCE
- Riprova la configurazione per JCE

Connessione a più AWS CloudHSM cluster con il provider JCE

Questa configurazione consente a una singola istanza client di comunicare con più AWS CloudHSM cluster. Rispetto al fatto che una singola istanza comunica solo con un singolo cluster, questa può essere una funzionalità che in alcuni casi consente di risparmiare sui costi. La `CloudHsmProvider` classe è l'implementazione AWS CloudHSM della classe Provider di Java Security. Ogni istanza di questa classe rappresenta una connessione all'intero AWS CloudHSM cluster. Si crea un'istanza di questa classe e la si aggiunge all'elenco del provider di sicurezza Java in modo da poter interagire con essa utilizzando classi JCE standard.

L'esempio seguente crea un'istanza di questa classe e la aggiunge all'elenco del provider di sicurezza Java:

```
if (Security.getProvider(CloudHsmProvider.PROVIDER_NAME) == null) {
    Security.addProvider(new CloudHsmProvider());
}
```

`CloudHsmProvider` può essere configurato in due modi:

1. Configura con file (configurazione predefinita)
2. Configura usando il codice

I seguenti argomenti descrivono queste configurazioni e come connettersi a più cluster.

Argomenti

- Configura la AWS CloudHSM CloudHsmProvider classe con un file (configurazione predefinita)
- Configura la AWS CloudHSM CloudHsmProvider classe usando il codice
- Connect a più AWS CloudHSM cluster

Configura la AWS CloudHSM **`CloudHsmProvider`** classe con un file (configurazione predefinita)

Il modo predefinito per configurare la AWS CloudHSM `CloudHsmProvider` classe è con un file.

Quando si crea un'istanza `CloudHsmProvider` utilizzando il costruttore predefinito, per impostazione predefinita cercherà il file di configurazione nel percorso `/opt/cloudhsm/etc/cloudhsm-jce.cfg` in Linux. Questo file di configurazione può essere configurato utilizzando `configure-jce`.

Un oggetto creato utilizzando il costruttore predefinito utilizzerà il nome del provider CloudHSM predefinito `CloudHSM`. Il nome del provider è utile per interagire con JCE e fargli sapere quale provider utilizzare per varie operazioni. Di seguito è riportato un esempio di utilizzo del nome del provider CloudHSM per il funzionamento della crittografia:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", "CloudHSM");
```

Configura la AWS CloudHSM **`CloudHsmProvider`** classe usando il codice

A partire dalla versione 5.8.0 di Client SDK, puoi anche configurare la AWS CloudHSM `CloudHsmProvider` classe utilizzando il codice Java. Il modo per farlo è usare un

oggetto di classe `CloudHsmProviderConfig`. È possibile creare l'oggetto utilizzando `CloudHsmProviderConfigBuilder`.

`CloudHsmProvider` ha un altro costruttore che accetta l'oggetto `CloudHsmProviderConfig`, come mostra l'esempio seguente.

Example

```
CloudHsmProviderConfig config = CloudHsmProviderConfig.builder()
                                    .withCluster(
                                        CloudHsmCluster.builder()
                                            .withHsmCAFilePath(hsmCAFilePath)

 .withClusterUniqueIdentifier("CloudHsmCluster1")
        .withServer(CloudHsmServer.builder().withHostIP(hostName).build())
                        .build())
        .build();
CloudHsmProvider provider = new CloudHsmProvider(config);
```

In questo esempio, il nome del provider JCE è. `CloudHsmCluster1` Questo è il nome che l'applicazione può quindi utilizzare per interagire con JCE:

Example

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", "CloudHsmCluster1");
```

In alternativa, le applicazioni possono anche utilizzare l'oggetto provider creato sopra per far sapere a JCE di utilizzare quel provider per l'operazione:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider);
```

Se con il metodo `withClusterUniqueIdentifier` non viene specificato un identificatore univoco, viene creato automaticamente un nome casuale. Per ottenere questo identificatore casuale, le applicazioni possono chiamare `provider.getName()` per ottenere l'identificatore.

Connect a più AWS CloudHSM cluster

Ciascuno `CloudHsmProvider` rappresenta una connessione al tuo AWS CloudHSM Cluster. Se si desidera comunicare con un altro cluster dalla stessa applicazione, è possibile creare un altro oggetto `CloudHsmProvider` con configurazioni per l'altro cluster e interagire con quest'altro cluster utilizzando l'oggetto provider o utilizzando il nome del provider, come illustrato nell'esempio seguente.

Example

```
CloudHsmProviderConfig config = CloudHsmProviderConfig.builder()
                                    .withCluster(
                                        CloudHsmCluster.builder()
                                            .withHsmCAFilePath(hsmCAFilePath)

 .withClusterUniqueIdentifier("CloudHsmCluster1")
        .withServer(CloudHsmServer.builder().withHostIP(hostName).build())
                        .build())
        .build();
CloudHsmProvider provider1 = new CloudHsmProvider(config);

if (Security.getProvider(provider1.getName()) == null) {
    Security.addProvider(provider1);
}

CloudHsmProviderConfig config2 = CloudHsmProviderConfig.builder()
                                    .withCluster(
                                        CloudHsmCluster.builder()
                                            .withHsmCAFilePath(hsmCAFilePath2)

 .withClusterUniqueIdentifier("CloudHsmCluster2")
        .withServer(CloudHsmServer.builder().withHostIP(hostName2).build())
                        .build())
        .build();
CloudHsmProvider provider2 = new CloudHsmProvider(config2);

if (Security.getProvider(provider2.getName()) == null) {
    Security.addProvider(provider2);
}
```

Dopo aver configurato entrambi i provider (entrambi i cluster) sopra, è possibile interagire con essi utilizzando l'oggetto provider o utilizzando il nome del provider.

Ampliando questo esempio che mostra come parlare con`cluster1`, è possibile utilizzare il seguente esempio per un'AES/GCM/NoPaddingoperazione:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider1);
```

E nella stessa applicazione per eseguire la generazione di chiavi "AES" sul secondo cluster utilizzando il nome del provider, è possibile utilizzare anche il seguente esempio:

```
Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding", provider2.getName());
```

Estrazione delle chiavi tramite JCE per AWS CloudHSM

La Java Cryptography Extension (JCE) utilizza un'architettura che consente di collegare diverse implementazioni di crittografia. AWS CloudHSM fornisce uno di questi provider JCE che trasferisce le operazioni crittografiche all'HSM. Affinché la maggior parte degli altri provider JCE lavori con le chiavi archiviate in AWS CloudHSM, devono estrarre i byte chiave dai moduli HSM in testo non crittografato nella memoria della macchina per utilizzarli. I moduli HSM in genere consentono di estrarre le chiavi solo come oggetti soggetti a wrapping, non come testo in chiaro. Tuttavia, per supportare i casi d'uso di integrazione tra provider, AWS CloudHSM consente un'opzione di configurazione opt-in per consentire l'estrazione dei byte chiave in chiaro.

> **Important**
>
> JCE trasferisce le operazioni AWS CloudHSM ogni volta che viene specificato il provider AWS CloudHSM o viene utilizzato un oggetto chiave. AWS CloudHSM Non è necessario estrarre le chiavi in chiaro se si prevede che l'operazione avvenga all'interno dell'HSM. L'estrazione delle chiavi in testo non crittografato è necessaria solo quando l'applicazione non può utilizzare meccanismi sicuri come eseguire e annullare il wrapping di una chiave a causa delle restrizioni imposte da una libreria di terze parti o da un provider JCE.

Per impostazione predefinita, il provider AWS CloudHSM JCE consente l'estrazione di chiavi pubbliche per funzionare con provider JCE esterni. I seguenti metodi sono sempre consentiti:

Classe	Metodo	Formato (getEncoded)
EcPublicKey	getEncoded()	X.509
	getW()	N/A
RSAPublicKey	getEncoded()	X.509
	get PublicExponent ()	N/A
CloudHsmEdDSAPublicKey	getEncoded()	X.509
CloudHsmMldsaPublicKey	getEncoded()	X.509

Per impostazione predefinita, il provider AWS CloudHSM JCE non consente l'estrazione di byte di chiave in chiaro per le chiavi private o segrete. Se il tuo caso d'uso lo richiede, puoi abilitare l'estrazione dei byte di chiave in chiaro per le chiavi private o segrete alle seguenti condizioni:

1. L'attributo `EXTRACTABLE` per le chiavi private e segrete è impostato su true.
   - Per impostazione predefinita, l'attributo `EXTRACTABLE` per le chiavi private e segrete è impostato su true. Le chiavi `EXTRACTABLE` sono chiavi che possono essere esportate dall'HSM. Per ulteriori informazioni, vedi Attributi Java supportati per Client SDK 5.
2. L'attributo `WRAP_WITH_TRUSTED` per le chiavi private e segrete è impostato su falso.
   - `getEncoded`, `getPrivateExponent` e `getS` non possono essere utilizzate con chiavi private che non possono essere esportate in chiaro. `WRAP_WITH_TRUSTED` non consente l'esportazione delle chiavi private dall'HSM in chiaro. Per maggiori informazioni, vedi Using trusted keys to control key unwraps.

Consenti al provider JCE di estrarre i segreti delle chiavi private da AWS CloudHSM

Usa i seguenti passaggi per consentire al provider AWS CloudHSM JCE di estrarre i segreti delle tue chiavi private.

> **Important**
>
> Questa modifica alla configurazione consente l'estrazione di tutti i byte chiave `EXTRACTABLE` in chiaro dal cluster HSM. Per una maggiore sicurezza, è consigliabile prendere in considerazione l'utilizzo di metodi di wrapping di chiavi per estrarre la chiave dall'HSM in modo sicuro. Ciò impedisce l'estrazione involontaria dei byte della chiave dall'HSM.

1. Utilizza i seguenti comandi per consentire l'estrazione delle chiavi private o segrete in JCE:

   Linux

   ```
   $ /opt/cloudhsm/bin/configure-jce --enable-clear-key-extraction-in-software
   ```

   Windows

   ```
   PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --enable-clear-key-extraction-in-software
   ```

2. Una volta abilitata l'estrazione delle chiavi in chiaro, vengono abilitati i seguenti metodi per estrarre le chiavi private in memoria.

Classe	Metodo	Formato (getEncoded)
Chiave	getEncoded()	RAW
ECPrivateKey	getEncoded()	PKCS #8
	getS()	N/A
RSAPrivateCrtKey	getEncoded()	X.509
	get PrivateExponent ()	N/A
	getPrimeP()	N/A
	getPrimeQ()	N/A
	ottieni PrimeExponent P ()	N/A
	ottieni PrimeExponent Q ()	N/A
	ottieni CrtCoefficient ()	N/A
CloudHsmEdDSAPrivateKey	getEncoded()	PKCS #8
CloudHsmMldsaPrivateKey	getEncoded()	N/A

Se desideri ripristinare il comportamento predefinito e non consentire a JCE di esportare le chiavi in chiaro, esegui il seguente comando:

Linux

```
$ /opt/cloudhsm/bin/configure-jce --disable-clear-key-extraction-in-software
```

Windows

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --disable-clear-key-extraction-in-software
```

Riprova i comandi per JCE per AWS CloudHSM

AWS CloudHSM Client SDK 5.8.0 e versioni successive dispongono di una strategia di riprova automatica integrata che riproverà le operazioni con limitazione HSM dal lato client. Quando un HSM rallenta le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, il client SDKs tenterà di riprovare le operazioni limitate fino a 3 volte, effettuando un backup esponenziale. Questa strategia automatica può essere impostata su una delle due modalità: off e standard.

- off: il Client SDK non eseguirà alcun ulteriore tentativo per le operazioni limitate da parte dell'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e successive. In questa modalità, il client SDKs riproverà automaticamente le operazioni limitate effettuando un backup esponenziale.

Per ulteriori informazioni, consulta Limitazione HSM.

Disattiva i comandi per l'esecuzione di ulteriori tentativi

Linux

Come impostare i comandi Nuovo tentativo su off per Client SDK 5 su Linux

- Per impostare la configurazione in modalità off, utilizza i seguenti comandi:

```
$ sudo /opt/cloudhsm/bin/configure-jce --default-retry-mode off
```

Windows

Per impostare i comandi Nuovo Tentativo su off per Client SDK 5 su Windows

- Per impostare la configurazione in modalità off, utilizza i seguenti comandi:

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" --default-
retry-mode off
```

Utilizzo della versione SDK precedente con cui lavorare AWS CloudHSM

⚠ Le versioni SDK 5.8.0 e precedenti hanno raggiunto la fine del supporto. Dopo il 31 marzo 2025 la documentazione per le versioni SDK 3.4.4 e precedenti non sarà più disponibile.

AWS CloudHSM include due versioni principali di Client SDK:

- Client SDK 5: questo è il nostro Client SDK più recente e quello predefinito. Per informazioni sui benefici e i vantaggi che offre, vedi Vantaggi di AWS CloudHSM Client SDK 5.
- Client SDK 3: Questo è il nostro vecchio Client SDK. Include un set completo di componenti per la compatibilità delle applicazioni basate su piattaforme e linguaggi e strumenti di gestione.

Per istruzioni sulla migrazione da Client SDK 3 a Client SDK 5, consulta. Migrazione da AWS CloudHSM Client SDK 3 a Client SDK 5

Questo argomento descrive Client SDK 3. Per vedere quale versione di Client SDK stai utilizzando, consulta. Controlla il tuo AWS CloudHSM Versione Client SDK

Per scaricarla, vedi Download.

Argomenti

- Aggiorna AWS CloudHSM Client SDK 3 su Linux
- AWS CloudHSM Piattaforme supportate da Client SDK 3
- Libreria PKCS #11 per AWS CloudHSM Client SDK 3
- AWS CloudHSM Motore dinamico OpenSSL per Client SDK 3
- Provider JCE per AWS CloudHSM Client SDK 3
- API di crittografia: Next Generation (CNG) e provider di archiviazione delle chiavi (KSP) per AWS CloudHSM

Aggiorna AWS CloudHSM Client SDK 3 su Linux

⚠ Le versioni SDK 5.8.0 e precedenti hanno raggiunto la fine del supporto. Dopo il 31 marzo 2025 la documentazione per le versioni SDK 3.4.4 e precedenti non sarà più disponibile.

Con AWS CloudHSM Client SDK 3.1 e versioni successive, la versione del daemon client e tutti i componenti installati devono corrispondere per l'aggiornamento. Per tutti i sistemi basati su Linux, devi utilizzare un singolo comando per aggiornare in batch il client daemon con la stessa versione della libreria PKCS #11, il provider Java Cryptographic Extension (JCE) o OpenSSL Dynamic Engine. Questo requisito non si applica ai sistemi basati su Windows perché i file per i provider KSP e CNG sono già inclusi nel pacchetto del client daemon.

Per verificare la versione del client daemon

- Su un sistema Linux basato su Red Hat (inclusi Amazon Linux e CentOS), utilizza il seguente comando:

```
rpm -qa | grep ^cloudhsm
```

- In un sistema Linux basato su Debian, utilizza il seguente comando:

```
apt list --installed | grep ^cloudhsm
```

- In un sistema Windows, utilizza il seguente comando:

```
wmic product get name,version
```

Argomenti

- Prerequisiti
- Fase 1: interruzione del client daemon
- Fase 2: Aggiornamento del Client SDK
- Fase 3: avvio del client daemon

Prerequisiti

Scarica l'ultima versione del daemon AWS CloudHSM client e scegli i tuoi componenti.

> **Note**
>
> Non devi installare tutte le librerie. Per ogni componente che installi, devi aggiornare questo componente in modo che corrisponda alla versione del client daemon.

Client daemon di Linux più recente

Amazon Linux

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

CentOS 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

RHEL 7

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-latest.el7.x86_64.rpm
```

RHEL 8

```
sudo yum install wget
```

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client_latest_u18.04_amd64.deb
```

Libreria PKCS #11 più recente

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-pkcs11_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```

OpenSSL Dynamic Engine più recente

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-dyn_latest_amd64.deb
```

Ultimo provider JCE

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-jce-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-jce_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-jce_latest_u18.04_amd64.deb
```

Fase 1: interruzione del client daemon

Utilizzare il seguente comando per arrestare il client daemon.

Amazon Linux

```
$ sudo stop cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client stop
```

CentOS 7

```
$ sudo service cloudhsm-client stop
```

CentOS 8

```
$ sudo service cloudhsm-client stop
```

RHEL 7

```
$ sudo service cloudhsm-client stop
```

RHEL 8

```
$ sudo service cloudhsm-client stop
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Fase 2: Aggiornamento del Client SDK

Il comando seguente mostra la sintassi richiesta per aggiornare il client daemon e i componenti. Prima di eseguire il comando, rimuovi gli eventuali componenti che non intendi aggiornare.

Amazon Linux

```
$ sudo yum install ./cloudhsm-client-latest.el6.x86_64.rpm \
              <./cloudhsm-client-pkcs11-latest.el6.x86_64.rpm> \
              <./cloudhsm-client-dyn-latest.el6.x86_64.rpm> \
              <./cloudhsm-client-jce-latest.el6.x86_64.rpm>
```

Amazon Linux 2

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \
              <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \
              <./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \
              <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

CentOS 7

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \
```

```
                <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \
                <./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \
                <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

CentOS 8

```
$ sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm \
                <./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm> \
                <./cloudhsm-client-jce-latest.el8.x86_64.rpm>
```

RHEL 7

```
$ sudo yum install ./cloudhsm-client-latest.el7.x86_64.rpm \
                <./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm> \
                <./cloudhsm-client-dyn-latest.el7.x86_64.rpm> \
                <./cloudhsm-client-jce-latest.el7.x86_64.rpm>
```

RHEL 8

```
$ sudo yum install ./cloudhsm-client-latest.el8.x86_64.rpm \
                <./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm> \
                <./cloudhsm-client-jce-latest.el8.x86_64.rpm>
```

Ubuntu 16.04 LTS

```
$ sudo apt install ./cloudhsm-client_latest_amd64.deb \
                <cloudhsm-client-pkcs11_latest_amd64.deb> \
                <cloudhsm-client-dyn_latest_amd64.deb> \
                <cloudhsm-client-jce_latest_amd64.deb>
```

Ubuntu 18.04 LTS

```
$ sudo apt install ./cloudhsm-client_latest_u18.04_amd64.deb \
                <cloudhsm-client-pkcs11_latest_amd64.deb> \
                <cloudhsm-client-jce_latest_amd64.deb>
```

Fase 3: avvio del client daemon

Utilizza il seguente comando per avviare il client daemon.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

AWS CloudHSM Piattaforme supportate da Client SDK 3

⚠ Le versioni SDK 5.8.0 e precedenti hanno raggiunto la fine del supporto. Dopo il 31 marzo 2025 la documentazione per le versioni SDK 3.4.4 e precedenti non sarà più disponibile.

AWS CloudHSM Client SDK 3 richiede un daemon client e offre strumenti da riga di comando tra cui CloudHSM Management Utility (CMU), KMU (key management utility) e lo strumento di configurazione.

Il supporto di base è diverso per ogni versione di Client SDK. AWS CloudHSM Generalmente le piattaforme supportate per i componenti di un SDK corrispondono al supporto di base, ma non è sempre così. Per determinare il supporto della piattaforma per un determinato componente, assicurati innanzitutto che la piattaforma desiderata compaia nella sezione base dell'SDK, quindi controlla eventuali esclusioni o altre informazioni pertinenti nella sezione del componente.

Le piattaforme supportate cambiano nel tempo. Le versioni precedenti del CloudHSM Client SDK potrebbero non supportare tutti i sistemi operativi elencati qui. Verifica se il sistema operativo supporta le versioni precedenti del Client SDK di CloudHSM consultando le note di rilascio. Per ulteriori informazioni, consulta Download per AWS CloudHSM Client SDK.

AWS CloudHSM supporta solo sistemi operativi a 64 bit.

Argomenti

- Supporto Linux per AWS CloudHSM Client SDK 3
- Supporto Windows per AWS CloudHSM Client SDK 3
- Compatibilità HSM per AWS CloudHSM Client SDK 3

Supporto Linux per AWS CloudHSM Client SDK 3

AWS CloudHSM Client SDK 3 supporta i seguenti sistemi operativi e piattaforme Linux.

- Amazon Linux
- Amazon Linux 2
- CentOS 6.10+ [2]

- CentOS 7.3+
- CentOS 8 [1,4]
- Red Hat Enterprise Linux (RHEL) 6.10+ [2]
- Red Hat Enterprise Linux (RHEL) 7.3+
- Red Hat Enterprise Linux (RHEL) 8 [1]
- Ubuntu 16.04 LTS [3]
- Ubuntu 18.04 LTS [1]

[1] Nessun supporto per OpenSSL Dynamic Engine. Per ulteriori informazioni, consulta la pagina su OpenSSL Dynamic Engine.

[2] Nessun supporto per Client SDK 3.3.0 e versioni successive.

[3] SDK 3.4 è l'ultima versione supportata su Ubuntu 16.04.

[4] SDK 3.4 è l'ultima versione supportata su CentOS 8.3+.

Supporto Windows per AWS CloudHSM Client SDK 3

AWS CloudHSM Client SDK 3 supporta le seguenti versioni di Windows Server.

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Compatibilità HSM per AWS CloudHSM Client SDK 3

La tabella seguente descrive la compatibilità di AWS CloudHSM Client SDK 3 per. HSMs

hsm1.medium	hsm2m. medio
Compatibile con la versione Client SDK 3.1.0 e successive.	Non supportato.

Libreria PKCS #11 per AWS CloudHSM Client SDK 3

PKCS #11 è uno standard per l'esecuzione di operazioni crittografiche su moduli di sicurezza hardware (HSM) in. AWS CloudHSM

Per informazioni sul processo di bootstrap, consulta la pagina Connessione al cluster.

Argomenti

- Installa la libreria PKCS #11 per AWS CloudHSM Client SDK 3
- Effettua l'autenticazione alla libreria PKCS #11 per Client SDK 3 AWS CloudHSM
- Tipi di chiave supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 3
- Meccanismi supportati per Client SDK 3 AWS CloudHSM
- Operazioni API supportate per Client SDK 3 AWS CloudHSM
- Attributi chiave nella libreria PKCS #11 per Client SDK 3 AWS CloudHSM
- Esempi di codice per la libreria PKCS #11 per AWS CloudHSM Client SDK 3

Installa la libreria PKCS #11 per AWS CloudHSM Client SDK 3

Questo argomento fornisce istruzioni per l'installazione della libreria PKCS #11 per la serie di versioni AWS CloudHSM Client SDK 3. Per ulteriori informazioni sull'SDK del client o sulla libreria PKCS #11, consulta la pagina sull'utilizzo dell'SDK del client e la pagina sulla libreria PKCS #11.

Prerequisiti per Client SDK 3

La libreria PKCS #11 richiede il client. AWS CloudHSM

Se non hai installato e configurato il AWS CloudHSM client, fallo ora seguendo i passaggi riportati di seguito. Installazione del client (Linux) Dopo aver installato e configurato il client, esegui questo comando per avviarlo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Installa la libreria PKCS #11 per Client SDK 3

Il seguente comando consente di scaricare e installare la libreria PKCS #11.

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-pkcs11-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-pkcs11_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-pkcs11_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-pkcs11_latest_u18.04_amd64.deb
```

- Se sull'istanza EC2 su cui è stata installata la libreria PKCS #11 non sono installati altri componenti di Client SDK 3, devi fare il bootstrap di Client SDK 3. È necessario eseguire questa operazione una sola volta su ogni istanza con un componente Client SDK 3.
- I file della libreria PKCS #11 sono disponibili nelle seguenti posizioni:

  File binari Linux, script di configurazione, certificati e file di log:

  ```
  /opt/cloudhsm/lib
  ```

Effettua l'autenticazione alla libreria PKCS #11 per Client SDK 3 AWS CloudHSM

Quando utilizzate la libreria PKCS #11, l'applicazione viene eseguita come un particolare utente crittografico (CU) nel vostro computer. HSMs AWS CloudHSM L'applicazione è in grado di visualizzare e gestire solo le chiavi di proprietà e condivise dall'utente di crittografia. È possibile utilizzare una CU esistente nel proprio CU HSMs o crearne una nuova. Per informazioni sulla gestione CUs, consulta Gestione degli utenti HSM con CloudHSM CLI e Gestione degli utenti HSM con CloudHSM Management Utility (CMU).

Per specificare il CU nella libreria PKCS #11, utilizza il parametro pin della funzione C_Login di PKCS #11. Infatti AWS CloudHSM, il parametro pin ha il seguente formato:

```
<CU_user_name>:<password>
```

Ad esempio, il comando seguente imposta il pin della libreria PKCS #11 sul CU con il nome utente `CryptoUser` e la password `CUPassword123!`.

```
CryptoUser:CUPassword123!
```

Tipi di chiave supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 3

La libreria PKCS #11 supporta i seguenti tipi di chiavi con AWS CloudHSM Client SDK 3.

Tipo di chiavi	Description
RSA	Genera chiavi RSA da 2048 bit a 4096 bit, con incrementi di 256 bit.
EC	Genera chiavi con le curve secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384) e secp521r1 (P-521).
AES	Genera chiavi AES a 128, 192 e 256 bit.
DES3 (Triplo DES)	Genera chiavi a 192 bit DES3 . Vedi la nota 1 di seguito per una modifica imminente.
GENERIC_SECRET	Genera segreti generici da 1 a 64 byte.

- [1] In conformità alle linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Meccanismi supportati per Client SDK 3 AWS CloudHSM

La libreria PKCS #11 supporta i seguenti algoritmi per AWS CloudHSM Client SDK 3:

- Crittografia e decrittografia: AES-CBC, AES-CTR, AES-ECB, AES-GCM, -CBC, -ECB, RSA-OAEP e RSA-PKCS DES3 DES3
- Firma e verifica: RSA, HMAC e ECDSA; con e senza hashing
- SHA1 SHA256 SHA384Hash/digest SHA224 —,,, e SHA512

- Wrapping della chiave: AES Key Wrap,[4] AES-GCM, RSA-AES e RSA-OAEP
- Derivazione chiave: ECDH, -108 CTR KDF [5] SP800

Tabella dei meccanismi e delle funzioni della libreria PKCS #11

La libreria PKCS #11 è conforme alla versione 2.40 della specifica PKCS #11. Per richiamare una funzione di crittografia utilizzando PKCS #11, chiamare una funzione con un determinato meccanismo. La seguente tabella riassume le combinazioni di funzioni e meccanismi supportati da AWS CloudHSM.

Interpretazione della tabella delle funzioni del meccanismo PKCS #11

Un segno ✓ indica che AWS CloudHSM supporta il meccanismo della funzione. Non supportiamo tutte le funzioni elencate nella specifica PKCS #11. Un segno ✖ indica che AWS CloudHSM non supporta ancora il meccanismo per una determinata funzione, anche se lo standard PKCS #11 lo consente. Le celle vuote indicano che lo standard PKCS #11 non supporta il meccanismo per una determinata funzione.

Meccanismi e funzioni PKCS #11 supportati

Meccanismo	Funzioni						
	Generate chiavi o coppie di chiavi	Sign & Verify (Firma e verifica)	SR & VR	Digest	Encrypt & Decrypt (Crittografia e decrittografia)	Derive Key (Deriva chiave)	Avvolgi & UnWrap
CKM_RSA_PKCS_KEY_PAIR_GEN	✓						
CKM_RSA_X9_31_KEY_PAIR_GEN	✓[2]						

Meccanismo	Funzioni						
CKM_RSA_X_509		✓			✓		
CKM_RSA_PKCS see note 8		✓[1]	✖		✓[1]		✓[1]
CKM_RSA_PKCS_OAEP					✓[1]		✓[6]
CKM_SHA1_RSA_PKCS		✓[3.2]					
CKM_SHA224_RSA_PKCS		✓[3.2]					
CKM_SHA256_RSA_PKCS		✓[3.2]					
CKM_SHA384_RSA_PKCS		✓[2,3.2]					
CKM_SHA512_RSA_PKCS		✓[3.2]					
CKM_RSA_PKCS_PSS		✓[1]					
CKM_SHA1_RSA_PKCS_PSS		✓[3.2]					

Meccanismo	Funzioni						
CKM_SHA224_RSA_PKCS_PSS		✓[3.2]					
CKM_SHA256_RSA_PKCS_PSS		✓[3.2]					
CKM_SHA384_RSA_PKCS_PSS		✓[2,3.2]					
CKM_SHA512_RSA_PKCS_PSS		✓[3.2]					
CKM_EC_KEY_PAIR_GEN	✓						
CKM_ECDSA		✓[1]					
CKM_ECDSA_SHA1		✓[3.2]					
CKM_ECDSA_SHA224		✓[3.2]					
CKM_ECDSA_SHA256		✓[3.2]					
CKM_ECDSA_SHA384		✓[3.2]					

Meccanismo	Funzioni						
CKM_ECDSA_SHA512		✓[3.2]					
CKM_ECDH1_DERIVE						✓[5]	
CKM_SP800_108_COUNTER_KDF						✓	
CKM_GENERIC_SECRET_KEY_GEN	✓						
CKM_AES_KEY_GEN	✓						
CKM_AES_ECB					✓		✖
CKM_AES_CTR					✓		✖
CKM_AES_CBC					✓[3.3]		✖
CKM_AES_CBC_PAD					✓		✖
CKM_DES3_KEY_GEN see note 8	✓						

Meccanismo	Funzioni						
CKM_DES3_CBC see note 8					✓[3.3]		✖
CKM_DES3_CBC_PAD see note 8					✓		✖
CKM_DES3_ECB see note 8					✓		✖
CKM_AES_GCM					✓[3.3, 4]		✓[7.1]
CKM_CLOUDHSM_AES_GCM					✓[7.1]		✓[7.1]
CKM_SHA_1				✓[3.1]			
CKM_SHA_1_HMAC		✓[3.3]					
CKM_SHA224				✓[3.1]			
CKM_SHA224_HMAC		✓[3.3]					
CKM_SHA256				✓[3.1]			

Meccanismo	Funzioni						
CKM_SHA256_HMAC		✓[3.3]					
CKM_SHA384				✓[3.1]			
CKM_SHA384_HMAC		✓[3.3]					
CKM_SHA512				✓[3.1]			
CKM_SHA512_HMAC		✓[3.3]					
CKM_RSA_AES_KEY_WRAP							✓
CKM_AES_KEY_WRAP							✓
CKM_AES_KEY_WRAP_PAD							✓
CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD							✓[7.1]
CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD							✓[7.1]

Meccanismo	Funzioni						
CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD							✓[7.1]

Annotazioni sui meccanismi

- [1] Solo operazioni a parte singola.
- [2] Il meccanismo è identico al meccanismo CKM_RSA_PKCS_KEY_PAIR_GEN dal punto di vista funzionale, ma offre maggiori garanzie per le generazioni p e q.
- [3.1] AWS CloudHSM approccia l'hashing in modo diverso in base al Client SDK. Per Client SDK 3, la posizione in cui viene eseguito l'hashing varia a seconda della dimensione dei dati e del fatto che si utilizzino operazioni a parte singola o in più parti.

  Operazioni a parte singola in Client SDK 3

  La Tabella 3.1 elenca la dimensione massima del set di dati per ciascun meccanismo per Client SDK 3. L'intero hash viene calcolato all'interno dell'HSM. Nessun supporto per dimensioni di dati superiori a 16 KB.

  Tabella 3.1, Dimensione massima del set di dati per operazioni a parte singola

Meccanismo	Dimensione massima dei dati
CKM_SHA_1	16296
CKM_SHA224	16264
CKM_SHA256	16296
CKM_SHA384	16232
CKM_SHA512	16232

  Operazioni in più parti per Client SDK 3

Supporto per dimensioni di dati superiori a 16 KB, ma la dimensione dei dati determina dove avviene l'hashing. I buffer di dati inferiori a 16 KB sono sottoposti a hash all'interno dell'HSM. I buffer di dimensione compresa tra 16 KB e la dimensione massima dei dati del sistema sono sottoposti a hash in locale nel software. Ricorda: le funzioni hash non richiedono segreti crittografici, quindi puoi calcolarle in sicurezza al di fuori dell'HSM.

- [3.2] AWS CloudHSM approccia l'hashing in modo diverso in base al Client SDK. Per Client SDK 3, la posizione in cui viene eseguito l'hashing varia a seconda della dimensione dei dati e del fatto che si utilizzino operazioni a parte singola o in più parti.

  Operazioni a parte singola per Client SDK 3

  La Tabella 3.2 elenca la dimensione massima del set di dati per ciascun meccanismo per Client SDK 3. Nessun supporto per dimensioni di dati superiori a 16 KB.

  Tabella 3.2, Dimensione massima del set di dati per operazioni a parte singola

Meccanismo	Dimensione massima dei dati
CKM_SHA1_RSA_PKCS	16296
CKM_SHA224_RSA_PKCS	16264
CKM_SHA256_RSA_PKCS	16296
CKM_SHA384_RSA_PKCS	16232
CKM_SHA512_RSA_PKCS	16232
CKM_SHA1_RSA_PKCS_PSS	16296
CKM_SHA224_RSA_PKCS_PSS	16264
CKM_SHA256_RSA_PKCS_PSS	16296
CKM_SHA384_RSA_PKCS_PSS	16232
CKM_SHA512_RSA_PKCS_PSS	16232
CKM_ECDSA_SHA1	16296

Meccanismo	Dimensione massima dei dati
CKM_ECDSA_SHA224	16264
CKM_ECDSA_SHA256	16296
CKM_ECDSA_SHA384	16232
CKM_ECDSA_SHA512	16232

Operazioni in più parti per Client SDK 3

Supporto per dimensioni di dati superiori a 16 KB, ma la dimensione dei dati determina dove avviene l'hashing. I buffer di dati inferiori a 16 KB sono sottoposti a hash all'interno dell'HSM. I buffer di dimensione compresa tra 16 KB e la dimensione massima dei dati del sistema sono sottoposti a hash in locale nel software. Ricorda: le funzioni hash non richiedono segreti crittografici, quindi puoi calcolarle in sicurezza al di fuori dell'HSM.

- [3.3] Quando si opera sui dati utilizzando uno dei seguenti meccanismi, se il buffer dati supera la dimensione massima dei dati, l'operazione genera un errore. Per questi meccanismi, tutta l'elaborazione dei dati deve avvenire all'interno dell'HSM. La tabella seguente elenca la dimensione massima dei dati per ciascun meccanismo:

  Tabella 3.3, Dimensione massima del set di dati

Meccanismo	Dimensione massima dei dati
CKM_SHA_1_HMAC	16288
CKM_SHA224_HMAC	16256
CKM_SHA256_HMAC	16288
CKM_SHA384_HMAC	16224
CKM_SHA512_HMAC	16224
CKM_AES_CBC	16272
CKM_AES_GCM	16224

Meccanismo	Dimensione massima dei dati
CKM_CLOUDHSM_AES_GCM	16224
CKM_DES3_CBC	16280

- [4] Quando si esegue la crittografia AES-GCM, l'HSM non accetta i dati del vettore di inizializzazione (IV) dall'applicazione. È necessario utilizzare un IV generato dall'HSM. L'IV da 12 byte fornito dall'HSM viene scritto nel riferimento della memoria indicato dall'elemento pIV della struttura di parametri CK_GCM_PARAMS fornita dall'utente. Per evitare confusione, l'SDK PKCS #11 nella versione 1.1.1 e successive assicura che pIV punti a un buffer azzerato quando viene inizializzata la crittografia AES-GCM.
- [5] Solo Client SDK 3. Questo meccanismo viene implementato per supportare i casi di offload SSL/TLS e viene eseguito solo in parte all'interno dell'HSM. Prima di utilizzare il meccanismo, consulta "(Problema) la deviazione della chiave ECDH viene eseguita parzialmente all'interno dell'HSM" in Problemi noti della libreria PKCS #11 per AWS CloudHSM. CKM_ECDH1_DERIVE non supporta la curva secp521r1 (P-521).
- [6] I seguenti CK_MECHANISM_TYPE e CK_RSA_PKCS_MGF_TYPE sono supportati come CK_RSA_PKCS_OAEP_PARAMS per CKM_RSA_PKCS_OAEP:
  - CKM_SHA_1 tramite CKG_MGF1_SHA1
  - CKM_SHA224 tramite CKG_MGF1_SHA224
  - CKM_SHA256 tramite CKG_MGF1_SHA256
  - CKM_SHA384 tramite CKM_MGF1_SHA384
  - CKM_SHA512 tramite CKM_MGF1_SHA512
- [7.1] Meccanismo definito dal fornitore. Per utilizzare i meccanismi definiti dal fornitore CloudHSM, le applicazioni PKCS #11 devono includere /opt/cloudhsm/include/pkcs11/cloudhsm_pkcs11_vendor_defs.h durante la compilazione.

  CKM_CLOUDHSM_AES_GCM: questo meccanismo proprietario è un'alternativa programmaticamente più sicura allo standard CKM_AES_GCM. Antepone il IV generato dall'HSM al testo cifrato invece di scriverlo nuovamente nella struttura CK_GCM_PARAMS fornita durante l'inizializzazione del codice. È possibile utilizzare questo meccanismo con le funzioni C_Encrypt, C_WrapKey, C_Decrypt e C_UnwrapKey. Quando si utilizza questo meccanismo, la variabile pIV nella struttura CK_GCM_PARAMS deve essere impostata su NULL. Quando si utilizza questo meccanismo con C_Decrypt e C_UnwrapKey, il IV dovrebbe essere anteposto al testo cifrato che viene scartato.

`CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD`: AES Key Wrap con riempimento PKCS #5

`CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD`: AES Key Wrap con riempimento a zeri

Per ulteriori informazioni sul wrapping delle chiavi AES, consulta la pagina sul wrapping delle chiavi AES.

- [8] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Operazioni API supportate per Client SDK 3 AWS CloudHSM

La libreria PKCS #11 supporta le seguenti operazioni API PKCS #11 per AWS CloudHSM Client SDK 3.

- `C_CloseAllSessions`
- `C_CloseSession`
- `C_CreateObject`
- `C_Decrypt`
- `C_DecryptFinal`
- `C_DecryptInit`
- `C_DecryptUpdate`
- `C_DeriveKey`
- `C_DestroyObject`
- `C_Digest`
- `C_DigestFinal`
- `C_DigestInit`
- `C_DigestUpdate`
- `C_Encrypt`
- `C_EncryptFinal`
- `C_EncryptInit`
- `C_EncryptUpdate`
- `C_Finalize`

- C_FindObjects
- C_FindObjectsFinal
- C_FindObjectsInit
- C_GenerateKey
- C_GenerateKeyPair
- C_GenerateRandom
- C_GetAttributeValue
- C_GetFunctionList
- C_GetInfo
- C_GetMechanismInfo
- C_GetMechanismList
- C_GetSessionInfo
- C_GetSlotInfo
- C_GetSlotList
- C_GetTokenInfo
- C_Initialize
- C_Login
- C_Logout
- C_OpenSession
- C_Sign
- C_SignFinal
- C_SignInit
- C_SignRecover (supporto solo per Client SDK 3)
- C_SignRecoverInit (supporto solo per Client SDK 3)
- C_SignUpdate
- C_UnWrapKey
- C_Verify
- C_VerifyFinal
- C_VerifyInit
- C_VerifyRecover (supporto solo per Client SDK 3)

- `C_VerifyRecoverInit` (supporto solo per Client SDK 3)
- `C_VerifyUpdate`
- `C_WrapKey`

Attributi chiave nella libreria PKCS #11 per Client SDK 3 AWS CloudHSM

Un oggetto può essere una chiave pubblica, privata o una chiave segreta. Le azioni consentite su un oggetto chiave sono specificate tramite gli attributi. Gli attributi sono definiti quando l'oggetto chiave viene creato. Quando si utilizza la libreria PKCS #11 per AWS CloudHSM, assegniamo valori predefiniti come specificato dallo standard PKCS #11.

AWS CloudHSM non supporta tutti gli attributi elencati nella specifica PKCS #11. Siamo conformi alla specifica per tutti gli attributi supportati. Questi attributi sono elencati nelle rispettive tabelle.

Le funzioni di crittografia, ad esempio `C_CreateObject`, `C_GenerateKey`, `C_GenerateKeyPair`, `C_UnwrapKey`, e `C_DeriveKey` che creano, modificano o copiano gli oggetti utilizzano un modello di attributo come uno dei loro parametri. Per ulteriori informazioni sul trasferimento di un modello di attributo durante la creazione di un oggetto, consulta l'esempio su come generare chiavi attraverso la libreria PKCS #11.

I seguenti argomenti forniscono ulteriori informazioni sugli attributi AWS CloudHSM chiave di Client SDK 3.

Argomenti

- Tabella degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 3
- Modifica degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 3
- Interpretazione dei codici di errore della libreria PKCS #11 per Client SDK 3 AWS CloudHSM

Tabella degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 3

La tabella della libreria PKCS #11 per AWS CloudHSM Client SDK 3 contiene un elenco di attributi che differiscono in base al tipo di chiave. Indica se un determinato attributo è supportato per un particolare tipo di chiave quando si utilizza una funzione crittografica specifica con. AWS CloudHSM

Legenda:

- ✓ indica che CloudHSM supporta l'attributo per il tipo di chiave specifico.
- ✖ indica che CloudHSM non supporta l'attributo per il tipo di chiave specifico.

- R indica che il valore dell'attributo è di sola lettura per il tipo di chiave specifico.
- S indica che l'attributo non può essere letto da `GetAttributeValue` poiché è sensibile.
- Una cella vuota nella colonna Valore predefinito indica che non vi è alcun valore predefinito specifico assegnato all'attributo.

GenerateKeyPair

Attributo	Tipo di chiavi				Valore predefinito
	EC privato	EC pubblico	RSA privato	RSA pubblico	
`CKA_CLASS`	✓	✓	✓	✓	
`CKA_KEY_TYPE`	✓	✓	✓	✓	
`CKA_LABEL`	✓	✓	✓	✓	
`CKA_ID`	✓	✓	✓	✓	
`CKA_LOCAL`	R	R	R	R	True
`CKA_TOKEN`	✓	✓	✓	✓	False
`CKA_PRIVATE`	✓[1]	✓[1]	✓[1]	✓[1]	True
`CKA_ENCRYPT`	✖	✓	✖	✓	False
`CKA_DECRYPT`	✓	✖	✓	✖	False

Attributo	Tipo di chiavi				Valore predefinito
CKA_DERIVE	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	False
CKA_SIGN_RECOVER	✖	✖	✓[3]	✖	
CKA_VERIFY	✖	✓	✖	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✓[4]	
CKA_WRAP	✖	✓	✖	✓	False
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	
CKA_TRUSTED	✖	✓	✖	✓	False
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	False
CKA_UNWRAP	✓	✖	✓	✖	False

Attributo	Tipo di chiavi				Valore predefinito
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	
CKA_SENSITIVE	✓	✖	✓	✖	True
CKA_ALWAYS_SENSITIVE	R	✖	R	✖	
CKA_EXTRACTABLE	✓	✖	✓	✖	True
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	
CKA_MODULUS	✖	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✓[2]	
CKA_PRIME_1	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	

Attributo	Tipo di chiavi				Valore predefinito
CKA_EXPONENT_2	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✓[2]	
CKA_EC_PARAMS	✖	✓[2]	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	
CKA_VALUE	✖	✖	✖	✖	
CKA_VALUE_LEN	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	

GenerateKey

Attributo	Tipo di chiavi			Valore predefinito
	AES	DES3	Segreto generico	
CKA_CLASS	✓	✓	✓	

Attributo	Tipo di chiavi			Valore predefinito
CKA_KEY_TYPE	✓	✓	✓	
CKA_LABEL	✓	✓	✓	
CKA_ID	✓	✓	✓	
CKA_LOCAL	R	R	R	True
CKA_TOKEN	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	True
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_WRAP_TEMPLATE	✓	✓	✖	
CKA_TRUSTED	✓	✓	✖	False
CKA_WRAP_WITH_TRUSTED	✓	✓	✓	False
CKA_UNWRAP	✓	✓	✖	False
CKA_UNWRAP_TEMPLATE	✓	✓	✖	
CKA_SENSITIVE	✓	✓	✓	True
CKA_ALWAYS_SENSITIVE	✖	✖	✖	
CKA_EXTRACTABLE	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	

Attributo	Tipo di chiavi			Valore predefinito
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
CKA_VALUE	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
CKA_VALUE_LEN		✓[2]	✖	✓[2]	
CKA_CHECK_VALUE		R	R	R	

CreateObject

Attributo	Tipo di chiavi							Valore predefinito
	EC privato	EC pubblico	RSA privato	RSA pubblico	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	R	R	False
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True

Attributo	Tipo di chiavi							Valore predefinito
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖	False
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✓[3]	✖	✖	✖	✖	False
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✓[4]	✖	✖	✖	
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖	False
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖	
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✖	False

Attributo	Tipo di chiavi							Valore predefinito
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓	False
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖	False
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖	
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓	True
CKA_ALWAYS_SENSITIVE	R	✖	R	✖	R	R	R	
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	R	R	R	
CKA_MODULUS	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✓	✖	✖	✖	✖	

Attributo	Tipo di chiavi							Valore predefinito
CKA_PRIME_2	✖	✖	✓	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✓[2]	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_EC_PARAMS	✓[2]	✓[2]	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✓[2]	✖	✖	✖	✖	✖	
CKA_VALUE	✓[2]	✖	✖	✖	✓[2]	✓[2]	✓[2]	
CKA_VALUE_LEN	✖	✖	✖	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	R	R	R	

UnwrapKey

Attributo	Tipo di chiavi					Valore predefinito
	EC privato	RSA privato	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	✓	✓	
CKA_ID	✓	✓	✓	✓	✓	
CKA_LOCAL	R	R	R	R	R	False
CKA_TOKEN	✓	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✖	✖	✓	✓	✖	False
CKA_DECRYPT	✖	✓	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True

Attributo	Tipo di chiavi					Valore predefinito
CKA_DESTROYABLE	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✓[3]	✖	✖	✖	False
CKA_VERIFY	✖	✖	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✖	✓	✓	✖	False
CKA_UNWRAP	✖	✓	✓	✓	✖	False
CKA_SENSITIVE	✓	✓	✓	✓	✓	True
CKA_EXTRACTABLE	✓	✓	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	

Attributo	Tipo di chiavi					Valore predefinito
CKA_MODULUS	✖	✖	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi					Valore predefinito
CKA_VALUE	✖	✖	✖	✖	✖	
CKA_VALUE_LEN	✖	✖	✖	✖	✖	
CKA_CHECK_VALUE	R	R	R	R	R	

DeriveKey

Attributo	Tipo di chiavi			Valore predefinito
	AES	DES3	Segreto generico	
CKA_CLASS	✓[2]	✓[2]	✓[2]	
CKA_KEY_TYPE	✓[2]	✓[2]	✓[2]	
CKA_LABEL	✓	✓	✓	
CKA_ID	✓	✓	✓	
CKA_LOCAL	R	R	R	True
CKA_TOKEN	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓[1]	✓[1]	✓[1]	True
CKA_SIGN	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_UNWRAP	✓	✓	✖	False
CKA_SENSITIVE	✓	✓	✓	True
CKA_EXTRACTABLE	✓	✓	✓	True

Attributo	Tipo di chiavi			Valore predefinito
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	
CKA_PRIME_1	✖	✖	✖	
CKA_PRIME_2	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
CKA_EC_PARAMS	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	
CKA_VALUE	✖	✖	✖	
CKA_VALUE_LEN	✓[2]	✖	✓[2]	
CKA_CHECK_VALUE	R	R	R	

GetAttributeValue

Attributo	Tipo di chiavi						
	EC privato	EC pubblico	RSA privato	RSA pubblico	AES	DES3	Segreto generico
CKA_CLASS	✓	✓	✓	✓	✓	✓	✓
CKA_KEY_TYPE	✓	✓	✓	✓	✓	✓	✓
CKA_LABEL	✓	✓	✓	✓	✓	✓	✓
CKA_ID	✓	✓	✓	✓	✓	✓	✓
CKA_LOCAL	✓	✓	✓	✓	✓	✓	✓

Attributo	Tipo di chiavi						
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✗	✗	✗	✓	✓	✓	✗
CKA_DECRYPT	✗	✗	✓	✗	✓	✓	✗
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✗	✓	✗	✓	✓	✓
CKA_SIGN_RECOVER	✗	✗	✓	✗	✗	✗	✗
CKA_VERIFY	✗	✓	✗	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✗	✗	✗	✓	✗	✗	✗
CKA_WRAP	✗	✗	✗	✓	✓	✓	✗
CKA_WRAP_TEMPLATE	✗	✓	✗	✓	✓	✓	✗

Attributo	Tipo di chiavi						
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	R	R
CKA_MODULUS	✖	✖	✓	✓	✖	✖	✖
CKA_MODULUS_BITS	✖	✖	✖	✓	✖	✖	✖
CKA_PRIME_1	✖	✖	S	✖	✖	✖	✖

Attributo	Tipo di chiavi						
CKA_PRIME_2	✖	✖	S	✖	✖	✖	✖
CKA_COEFFICIENT	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_1	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖
CKA_PUBLIC_EXPONENT	✖	✖	✓	✓	✖	✖	✖
CKA_EC_PARAMS	✓	✓	✖	✖	✖	✖	✖
CKA_EC_POINT	✖	✓	✖	✖	✖	✖	✖
CKA_VALUE	S	✖	✖	✖	✓[2]	✓[2]	✓[2]
CKA_VALUE_LEN	✖	✖	✖	✖	✓	✖	✓
CKA_CHECK_VALUE	✓	✓	✓	✓	✓	✓	✖

Annotazioni degli attributi

- [1] Questo attributo è parzialmente supportato dal firmware e deve essere impostato esplicitamente solo sul valore predefinito.
- [2] Attributo obbligatorio.
- [3] Solo Client SDK 3. L'attributo `CKA_SIGN_RECOVER` deriva dall'attributo `CKA_SIGN`. Se impostato, può essere impostato solo sullo stesso valore impostato per `CKA_SIGN`. Se non è impostato, ricava il valore predefinito di `CKA_SIGN`. Poiché CloudHSM supporta solo i meccanismi di firma recuperabili basati su RSA, questo attributo è attualmente applicabile solo alle chiavi pubbliche RSA.
- [4] Solo Client SDK 3. L'attributo `CKA_VERIFY_RECOVER` deriva dall'attributo `CKA_VERIFY`. Se impostato, può essere impostato solo sullo stesso valore impostato per `CKA_VERIFY`. Se non è impostato, ricava il valore predefinito di `CKA_VERIFY`. Poiché CloudHSM supporta solo i meccanismi di firma recuperabili basati su RSA, questo attributo è attualmente applicabile solo alle chiavi pubbliche RSA.

Modifica degli attributi della libreria PKCS #11 per AWS CloudHSM Client SDK 3

Alcuni attributi di un oggetto possono essere modificati dopo la creazione dell'oggetto, mentre altri no. Per modificare gli attributi, utilizza il comando setAttribute da cloudhsm_mgmt_util. È inoltre possibile ottenere un elenco di attributi e costanti che li rappresentano utilizzando il comando listAttribute da cloudhsm_mgmt_util.

L'elenco seguente mostra gli attributi modificabili dopo la creazione dell'oggetto:

- `CKA_LABEL`
- `CKA_TOKEN`

> **Note**
>
> La modifica è consentita solo per la modifica di una chiave di sessione in una chiave di token. Utilizza il comando setAttribute da key_mgmt_util per modificare il valore dell'attributo.

- `CKA_ENCRYPT`
- `CKA_DECRYPT`
- `CKA_SIGN`
- `CKA_VERIFY`

- CKA_WRAP
- CKA_UNWRAP
- CKA_LABEL
- CKA_SENSITIVE
- CKA_DERIVE

> **Note**
>
> Questo attributo supporta la derivazione della chiave. Deve essere False per tutte le chiavi pubbliche e non può essere impostato su True. Per le chiavi segrete ed EC private, può essere impostato su True o False.

- CKA_TRUSTED

> **Note**
>
> Questo attributo può essere impostato su True o su False solo da Responsabile della crittografia (CO).

- CKA_WRAP_WITH_TRUSTED

> **Note**
>
> Applica questo attributo a una chiave di dati esportabile per specificare che è possibile eseguire il wrapping della chiave solo con chiavi contrassegnate come CKA_TRUSTED. Una volta impostato l'attributo CKA_WRAP_WITH_TRUSTED su true, questo diventa di sola lettura e non è possibile modificarlo o rimuoverlo.

Interpretazione dei codici di errore della libreria PKCS #11 per Client SDK 3 AWS CloudHSM

Se si specifica nel modello un attributo della libreria PKCS #11 che non è supportato da una chiave specifica, viene generato un errore. La tabella riportata di seguito contiene i codici di errore generati quando si violano le specifiche:

Codice di errore	Descrizione
`CKR_TEMPLATE_INCONSISTENT`	Si riceve questo errore quando si specifica un attributo nel modello di attributo, in cui l'attributo è conforme alla specifica PKCS #11, ma non è supportato da CloudHSM.
`CKR_ATTRIBUTE_TYPE_INVALID`	Si riceve questo errore quando si recupera il valore di un attributo, che è conforme alla specifica PKCS #11, ma non è supportato da CloudHSM.
`CKR_ATTRIBUTE_INCOMPLETE`	Si riceve questo errore quando non si specifica l'attributo obbligatorio nel modello di attributo.
`CKR_ATTRIBUTE_READ_ONLY`	Si riceve questo errore quando si specifica un attributo di sola lettura nel modello di attributo.

Esempi di codice per la libreria PKCS #11 per AWS CloudHSM Client SDK 3

Gli esempi di codice riportati GitHub mostrano come eseguire attività di base utilizzando la libreria PKCS #11 per. AWS CloudHSM

Prerequisiti del codice di esempio

Prima di eseguire gli esempi, attieniti alla seguente procedura per configurare l'ambiente:

- Installa e configura la libreria PKCS #11 per Client SDK 3.
- Configura un crypto user (CU). L'applicazione utilizza questo account HSM per eseguire i codici di esempio sull'HSM.

Esempi di codice

Esempi di codice per la libreria AWS CloudHSM software per PKCS #11 sono disponibili su. GitHub Questo repository include esempi su come eseguire operazioni comuni utilizzando PKCS #11, tra cui crittografia, decrittografia, firma e verifica.

- Generazione di chiavi (AES, RSA, EC)

- Elenco degli attributi chiave
- Crittografia e decodifica dei dati con AES GCM
- Crittografia e decrittografia dei dati con AES_CTR
- Crittografia e decrittografia dei dati con 3DES
- Firma e verifica dei dati con RSA
- Derivazione delle chiavi utilizzando HMAC KDF
- Wrapping e annullamento del wrapping delle chiavi mediante riempimento PKCS #5
- Wrapping e annullamento del wrapping delle chiavi con AES utilizzando senza riempimento
- Wrapping e annullamento del wrapping delle chiavi con AES mediante riempimento a zeri
- Wrapping e annullamento del wrapping delle chiavi con AES-GCM
- Wrapping e annullamento del wrapping delle chiavi con RSA

AWS CloudHSM Motore dinamico OpenSSL per Client SDK 3

L' AWS CloudHSM OpenSSL Dynamic Engine consente di trasferire le operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

AWS CloudHSM Client SDK 3 richiede un demone client per connettersi al cluster. Supporta:

- Generazione di chiavi RSA per chiavi a 2048, 3072 e 4096 bit.
- Firma/verifica RSA.
- Crittografia/decrittografia RSA.
- Generazione di numeri casuali sicuro a livello crittografico e con convalida FIPS.

Usa le seguenti sezioni per installare e configurare il motore AWS CloudHSM dinamico per OpenSSL.

Argomenti

- Prerequisiti per OpenSSL Dynamic Engine con Client SDK 3 AWS CloudHSM
- Installa il motore AWS CloudHSM dinamico OpenSSL per Client SDK 3
- Usa il motore AWS CloudHSM dinamico OpenSSL per Client SDK 3

Prerequisiti per OpenSSL Dynamic Engine con Client SDK 3 AWS CloudHSM

Per ulteriori informazioni sulle piattaforme supportate, consulta la pagina AWS CloudHSM Piattaforme supportate da Client SDK 3.

Prima di poter utilizzare il motore AWS CloudHSM dinamico per OpenSSL con Client SDK 3, è necessario il client. AWS CloudHSM

Il client è un demone che stabilisce una comunicazione end-to-end crittografata con il cluster e il HSMs motore OpenSSL comunica localmente con il client. Per installare e configurare il client, vedere. AWS CloudHSM Installazione del client (Linux) Poi utilizza il seguente comando per avviarlo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 6

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 6

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Installa il motore AWS CloudHSM dinamico OpenSSL per Client SDK 3

I passaggi seguenti descrivono come installare e configurare il motore AWS CloudHSM dinamico per OpenSSL con Client SDK 3. Per ulteriori informazioni sull'upgrade, consulta la pagina Aggiorna Client SDK 3.

Per installare e configurare il motore OpenSSL

1. Utilizzare i comandi seguenti per scaricare e installare il motore OpenSSL.

   Amazon Linux

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
   ```

   Amazon Linux 2

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
   ```

   CentOS 6

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
   ```

   CentOS 7

   ```
   $ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
   ```

   ```
   $ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
   ```

RHEL 6

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el6.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-dyn-latest.el7.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-dyn_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-dyn_latest_amd64.deb
```

Il motore OpenSSL è installato in `/opt/cloudhsm/lib/libcloudhsm_openssl.so`.

2. Utilizzare il comando seguente per impostare una variabile di ambiente denominata `n3fips_password` che contiene le credenziali di un crypto user (CU).

```
$ export n3fips_password=<HSM user name>:<password>
```

Usa il motore AWS CloudHSM dinamico OpenSSL per Client SDK 3

Per utilizzare il motore AWS CloudHSM dinamico per OpenSSL da un'applicazione integrata in OpenSSL, assicurati che l'applicazione utilizzi il motore dinamico OpenSSL denominato. `cloudhsm` La libreria condivisa per il motore dinamico si trova in `/opt/cloudhsm/lib/libcloudhsm_openssl.so`.

Per utilizzare il motore AWS CloudHSM dinamico per OpenSSL dalla riga di comando OpenSSL, usa l'`-engine`opzione per specificare il motore dinamico OpenSSL denominato. `cloudhsm` Esempio:

```
$ openssl s_server -cert <server.crt> -key <server.key> -engine cloudhsm
```

Provider JCE per AWS CloudHSM Client SDK 3

Il provider AWS CloudHSM JCE è un'implementazione del provider basata sul framework del provider Java Cryptographic Extension (JCE). JCE consente di eseguire operazioni di crittografia utilizzando Java Development Kit (JDK). In questa guida, il provider AWS CloudHSM JCE viene talvolta definito provider JCE. Utilizza il provider JCE e il JDK per trasferire le operazioni crittografiche sull'HSM.

Argomenti

- Installare il provider JCE per AWS CloudHSM Client SDK 3
- Nozioni di base sulla gestione delle chiavi nel provider JCE per Client SDK 3 AWS CloudHSM
- Meccanismi supportati per Client SDK 3 per AWS CloudHSM Client SDK 3
- Attributi chiave Java supportati per AWS CloudHSM Client SDK 3
- Esempi di codice per la libreria AWS CloudHSM software per Java for Client SDK 3
- AWS CloudHSM KeyStore Classe Java per Client SDK 3

Installare il provider JCE per AWS CloudHSM Client SDK 3

Prima di poter utilizzare il provider JCE, è necessario il AWS CloudHSM client.

Il client è un demone che stabilisce una comunicazione end-to-end crittografata con i membri del HSMs cluster. Il provider JCE comunica localmente con il client. Se non hai installato e configurato il pacchetto AWS CloudHSM client, fallo ora seguendo la procedura riportata qui. Installazione del client (Linux) Dopo aver installato e configurato il client, esegui questo comando per avviarlo.

È supportato solo sui sistemi operativi Linux e altri sistemi operativi compatibili.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl start cloudhsm-client
```

CentOS 8

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl start cloudhsm-client
```

RHEL 8

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 16.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 18.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Ubuntu 20.04 LTS

```
$ sudo systemctl start cloudhsm-client
```

Utilizza le seguenti sezioni per installare, convalidare e fornire credenziali al provider.

Argomenti

- Fase 1: Installare il provider JCE
- Fase 2: Convalidare l'installazione
- Fase 3: Fornire le credenziali al provider JCE

Fase 1: Installare il provider JCE

Utilizza i seguenti comandi per scaricare e installare il provider JCE. Il provider è supportato solo sui sistemi operativi Linux e altri sistemi operativi compatibili.

> **Note**
>
> Per l'aggiornamento, vedi Aggiorna Client SDK 3.

Amazon Linux

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL6/cloudhsm-client-jce-latest.el6.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el6.x86_64.rpm
```

Amazon Linux 2

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

CentOS 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el8.x86_64.rpm
```

RHEL 7

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL7/cloudhsm-client-jce-latest.el7.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el7.x86_64.rpm
```

RHEL 8

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/EL8/cloudhsm-client-jce-latest.el8.x86_64.rpm
```

```
$ sudo yum install ./cloudhsm-client-jce-latest.el8.x86_64.rpm
```

Ubuntu 16.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Xenial/cloudhsm-client-jce_latest_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-jce_latest_amd64.deb
```

Ubuntu 18.04 LTS

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Bionic/cloudhsm-client-jce_latest_u18.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-client-jce_latest_u18.04_amd64.deb
```

Dopo aver eseguito i comandi precedenti, è possibile individuare i seguenti file della del provider JCE:

- `/opt/cloudhsm/java/cloudhsm-<version>.jar`
- `/opt/cloudhsm/java/cloudhsm-test-<version>.jar`
- `/opt/cloudhsm/java/hamcrest-all-1.3.jar`
- `/opt/cloudhsm/java/junit.jar`
- `/opt/cloudhsm/java/log4j-api-2.17.1.jar`
- `/opt/cloudhsm/java/log4j-core-2.17.1.jar`

- `/opt/cloudhsm/lib/libcaviumjca.so`

Fase 2: Convalidare l'installazione

Esecuzione di operazioni di base sull'HSM per convalidare l'installazione.

Per convalidare l'installazione del provider JCE

1. (Facoltativo) Se non hai già installato Java nell'ambiente, utilizza il comando seguente per installarlo.

   Linux (and compatible libraries)

   ```
   $ sudo yum install java-1.8.0-openjdk
   ```

   Ubuntu

   ```
   $ sudo apt-get install openjdk-8-jre
   ```

2. Utilizza i seguenti comandi per impostare le variabili di ambiente necessarie. Sostituisci *<HSM user name>* e *<password>* con le credenziali di un utente crittografico (CU).

   ```
   $ export LD_LIBRARY_PATH=/opt/cloudhsm/lib
   ```

   ```
   $ export HSM_PARTITION=PARTITION_1
   ```

   ```
   $ export HSM_USER=<HSM user name>
   ```

   ```
   $ export HSM_PASSWORD=<password>
   ```

3. Utilizza il seguente comando per eseguire il test di funzionalità di base. Se il comando viene eseguito correttamente, verrà visualizzato un output simile al seguente.

   ```
   $ java8 -classpath "/opt/cloudhsm/java/*" org.junit.runner.JUnitCore
    TestBasicFunctionality

   JUnit version 4.11
   .2018-08-20 17:53:48,514 DEBUG [main] TestBasicFunctionality
    (TestBasicFunctionality.java:33) - Adding provider.
   ```

```
2018-08-20 17:53:48,612 DEBUG [main] TestBasicFunctionality
 (TestBasicFunctionality.java:42) - Logging in.
2018-08-20 17:53:48,612 INFO [main] cfm2.LoginManager (LoginManager.java:104) -
 Looking for credentials in HsmCredentials.properties
2018-08-20 17:53:48,612 INFO [main] cfm2.LoginManager (LoginManager.java:122) -
 Looking for credentials in System.properties
2018-08-20 17:53:48,613 INFO [main] cfm2.LoginManager (LoginManager.java:130) -
 Looking for credentials in System.env
 SDK Version: 2.03
2018-08-20 17:53:48,655 DEBUG [main] TestBasicFunctionality
 (TestBasicFunctionality.java:54) - Generating AES Key with key size 256.
2018-08-20 17:53:48,698 DEBUG [main] TestBasicFunctionality
 (TestBasicFunctionality.java:63) - Encrypting with AES Key.
2018-08-20 17:53:48,705 DEBUG [main] TestBasicFunctionality
 (TestBasicFunctionality.java:84) - Deleting AES Key.
2018-08-20 17:53:48,707 DEBUG [main] TestBasicFunctionality
 (TestBasicFunctionality.java:92) - Logging out.

Time: 0.205

OK (1 test)
```

Fase 3: Fornire le credenziali al provider JCE

HSMs è necessario autenticare l'applicazione Java prima che l'applicazione possa utilizzarla. Ogni applicazione può utilizzare una sessione. HSMs autentica una sessione utilizzando il metodo di accesso esplicito o il metodo di accesso implicito.

Accesso Esplicito – questo metodo consente di fornire le credenziali CloudHSM direttamente nell'applicazione. Utilizza il metodo `LoginManager.login()`, in cui si passa il nome utente, la password e l'ID della partizione HSM. Per ulteriori informazioni sull'utilizzo del metodo di accesso esplicito, vedi l'esempio di codice Accesso a un HSM.

Accesso Implicito – questo metodo consente di impostare le credenziali di CloudHSM in un nuovo file di proprietà, proprietà del sistema, oppure come variabili di ambiente.

- Nuovo file di proprietà Crea un nuovo file con nome `HsmCredentials.properties` e aggiungilo al `CLASSPATH` della tua applicazione. Il file deve contenere il testo seguente:

```
HSM_PARTITION = PARTITION_1
HSM_USER = <HSM user name>
```

```
HSM_PASSWORD = <password>
```

- Proprietà di sistema – Imposta le credenziali attraverso le proprietà di sistema durante l'esecuzione di un'applicazione. I seguenti esempi mostrano due modi differenti con cui è possibile eseguire questa operazione:

```
$ java -DHSM_PARTITION=PARTITION_1 -DHSM_USER=<HSM user name> -
DHSM_PASSWORD=<password>
```

```
System.setProperty("HSM_PARTITION","PARTITION_1");
System.setProperty("HSM_USER","<HSM user name>");
System.setProperty("HSM_PASSWORD","<password>");
```

- Variabili di ambiente Imposta le credenziali come variabili di ambiente.

```
$ export HSM_PARTITION=PARTITION_1
$ export HSM_USER=<HSM user name>
$ export HSM_PASSWORD=<password>
```

Le credenziali potrebbero non essere disponibili se l'applicazione non le fornisce o se viene eseguita un'operazione prima che l'HSM autentichi la sessione. In questi casi, la libreria software CloudHSM per Java cerca le credenziali nel seguente ordine:

1. `HsmCredentials.properties`
2. Proprietà di sistema
3. Variabili di ambiente

Gestione degli errori

La gestione degli errori è più facile con l'accesso esplicito rispetto al metodo di login implicito. Quando si utilizza la classe `LoginManager`, si dispone di un maggiore controllo sulla modalità con cui l'applicazione gestisce gli errori. Il metodo di accesso implicito rende difficile la comprensione della gestione degli errori quando le credenziali non sono valide o si verificano problemi nell'autenticazione della HSMs sessione.

Nozioni di base sulla gestione delle chiavi nel provider JCE per Client SDK 3 AWS CloudHSM

Le nozioni di base sulla gestione delle chiavi nel provider JCE implicano l'importazione, l'esportazione, il caricamento tramite handle, oppure l'eliminazione di chiavi. Per ulteriori informazioni su come gestire le chiavi, vedi l'esempio di codice Gestire le chiavi.

Puoi inoltre trovare ulteriori esempi di codice del provider JCE all'indirizzoEsempi di codice.

Meccanismi supportati per Client SDK 3 per AWS CloudHSM Client SDK 3

Questo argomento fornisce informazioni sui meccanismi supportati per il provider JCE con AWS CloudHSM Client SDK 3. Per informazioni sulle interfacce e le classi di motori Java Cryptography Architecture (JCA) supportate da AWS CloudHSM, consultate i seguenti argomenti.

Argomenti

- Chiavi supportate
- Cifrature supportate
- Digest supportati
- Algoritmi codice di autenticazione dei messaggi basato su hash (HMAC) supportati
- Meccanismi supportati sign/verify
- Annotazioni sui meccanismi

Chiavi supportate

La libreria AWS CloudHSM software per Java consente di generare i seguenti tipi di chiavi.

- AES - chiavi AES a 128, 192 e 256 bit.
- DESede — Chiave 3DES a 92 bit. Vedi la nota 1 di seguito per una modifica imminente.
- Coppie di chiavi ECC per curve NIST secp256r1 (P-256), secp384r1 (P-384), and secp256k1 (Blockchain).
- RSA - chiavi RSA da 2048-bit a 4096-bit, con incrementi di 256 bit.

Oltre ai parametri standard, supportiamo i seguenti parametri per ogni chiave generata.

- Etichetta: un'etichetta della chiave che è possibile utilizzare per cercare le chiavi.
- è esportabile: indica se la chiave può essere esportata dall'HSM.

- è Persistente: indica se la chiave rimane sull'HSM quando la sessione corrente termina.

> **Note**
>
> La libreria Java versione 3.1 offre la possibilità di specificare i parametri in modo più dettagliato. Per ulteriori informazioni, vedi la sezione relativa agli attributi Java supportati.

Cifrature supportate

La libreria AWS CloudHSM software per Java supporta le seguenti combinazioni di algoritmi, modalità e padding.

Algoritmo	Modalità	Padding	Note
AES	CBC	`AES/CBC/NoPadding` `AES/CBC/PKCS5Padding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
AES	ECB	`AES/ECB/NoPadding` `AES/ECB/PKCS5Padding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` . Usare AES di trasformazione.
AES	CTR	`AES/CTR/NoPadding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
AES	GCM	`AES/GCM/NoPadding`	Implementa `Cipher.ENCRYPT_MODE`

Algoritmo	Modalità	Padding	Note
			e `Cipher.DECRYPT_MODE`, `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`. Quando si esegue la crittografia AES-GCM, l'HSM ignora il vettore di inizializzazione (IV) nella richiesta e utilizza un IV da lui generato. Al termine dell'operazione, è necessario chiamare `Cipher.getIV()` per ottenere il IV.

AESWrap	ECB	`AESWrap/ECB/ZeroPadding`
		
		`AESWrap/ECB/NoPadding`
		
		`AESWrap/ECB/PKCS5Padding`
			Implementa `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`, Usare AES di trasformazione.

Algoritmo	Modalità	Padding	Note

DESede (Triplo DES)	CBC	`DESede/CBC/NoPadding`
		
		`DESede/CBC/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
			
			Le routine di generazione della chiave accettano una dimensione di 168 o 192 bit. Tuttavia, internamente, tutte le DESede chiavi sono a 192 bit.
			
			Vedi la nota 1 di seguito per una modifica imminente.

Algoritmo	Modalità	Padding	Note

DESede (Triplo DES)	ECB	`DESede/ECB/NoPadding`
		
		`DESede/ECB/PKCS5Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
			
			Le routine di generazione della chiave accettano una dimensione di 168 o 192 bit. Tuttavia, internamente, tutte le DESede chiavi sono a 192 bit.
			
			Vedi la nota 1 di seguito per una modifica imminente.

RSA	ECB	`RSA/ECB/NoPadding`
		
		`RSA/ECB/PKCS1Padding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .
			
			Vedi la nota 1 di seguito per una modifica imminente.

Algoritmo	Modalità	Padding	Note

RSA	ECB	`RSA/ECB/OAEPPadding`
		
		`RSA/ECB/OAEPWithSHA-1ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-224ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-256ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-384ANDMGF1Padding`
		
		`RSA/ECB/OAEPWithSHA-512ANDMGF1Padding`
			Implementa `Cipher.ENCRYPT_MODE`, `Cipher.DECRYPT_MODE`, `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`.
			
			`OAEPPadding` è OAEP con il tipo di padding SHA-1.

RSAAESWrap	ECB	`OAEPPADDING`	Implementa `Cipher.WRAP_Mode` e `Cipher.UNWRAP_MODE`.

Digest supportati

La libreria AWS CloudHSM software per Java supporta i seguenti digest di messaggi.

- `SHA-1`
- `SHA-224`
- `SHA-256`
- `SHA-384`
- `SHA-512`

> **Note**
>
> I dati di lunghezza inferiore a 16 KB vengono sottoposti a hashing nell'HSM, mentre i dati di dimensioni maggiori vengono sottoposti a hashing nel software.

Algoritmi codice di autenticazione dei messaggi basato su hash (HMAC) supportati

La libreria AWS CloudHSM software per Java supporta i seguenti algoritmi HMAC.

- `HmacSHA1`
- `HmacSHA224`
- `HmacSHA256`
- `HmacSHA384`
- `HmacSHA512`

Meccanismi supportati sign/verify

La libreria AWS CloudHSM software per Java supporta i seguenti tipi di firma e verifica.

Tipi di firma RSA

- `NONEwithRSA`
- `SHA1withRSA`
- `SHA224withRSA`

- SHA256withRSA
- SHA384withRSA
- SHA512withRSA
- SHA1withRSA/PSS
- SHA224withRSA/PSS
- SHA256withRSA/PSS
- SHA384withRSA/PSS
- SHA512withRSA/PSS

Tipi di firma ECDSA

- NONEwithECDSA
- SHA1withECDSA
- SHA224withECDSA
- SHA256withECDSA
- SHA384withECDSA
- SHA512withECDSA

Annotazioni sui meccanismi

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Attributi chiave Java supportati per AWS CloudHSM Client SDK 3

Questo argomento descrive come utilizzare un'estensione proprietaria per la versione 3.1 della libreria Java per impostare gli attributi chiave per AWS CloudHSM Client SDK 3. Utilizzare questa estensione per impostare gli attributi della chiave supportati e i relativi valori durante le operazioni seguenti:

- Generazione delle chiavi
- Importazione delle chiavi
- Annullamento del wrapping delle chiavi

> **Note**
>
> L'estensione per l'impostazione degli attributi della chiave personalizzati è una funzionalità facoltativa. Se disponi già di un codice che funziona nella libreria Java versione 3.0, non è necessario modificare tale codice. Le chiavi create continueranno a contenere gli stessi attributi di prima.

Argomenti

- Comprensione degli attributi
- Attributi supportati
- Impostazione attributi per una chiave
- Mettere tutto insieme

Comprensione degli attributi

Gli attributi chiave vengono utilizzati per specificare le operazioni consentite su oggetti chiave, incluse le chiavi pubbliche, private o segrete. Gli attributi e i valori della chiave vengono definiti durante le operazioni di creazione degli oggetti chiave.

Tuttavia, la Java Cryptography Extension (JCE) non specifica come impostare i valori sugli attributi della chiave, pertanto la maggior parte delle operazioni erano consentite per impostazione predefinita. Al contrario, lo standard PKCS# 11 definisce un set completo di attributi con valori predefiniti più restrittivi. A partire dalla libreria Java versione 3.1, CloudHSM fornisce un'estensione proprietaria che ti consente di impostare valori più restrittivi per gli attributi utilizzati più di frequente.

Attributi supportati

Puoi impostare i valori per gli attributi elencati nella tabella sottostante. Come best practice, imposta i valori solo per gli attributi che desideri rendere restrittivi. Se non specifichi un valore, CloudHSM utilizza il valore predefinito specificato nella tabella sottostante. Una cella vuota nella colonna Valore predefinito indica che all'attributo non è stato assegnato alcun valore predefinito specifico.

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
CKA_TOKEN	FALSE	FALSE	FALSE	Una chiave permanente che viene replicata in tutto il cluster e inclusa HSMs nei backup. CKA_TOKEN = FALSO implica una chiave di sessione, che viene caricata solo su un HSM e cancellat a automatic amente quando la connessione al HSM viene interrotta.
CKA_LABEL				Una stringa definita dall'uten te. Consente di identificare comodamente le chiavi sull'HSM.
CKA_EXTRA CTABLE	TRUE		TRUE	Il valore Vero indica che è possibile esportare questa chiave dall'HSM.

Attributo	Valore predefinito			Note
CKA_ENCRYPT	TRUE	TRUE		Il valore Vero indica che è possibile utilizzare la chiave per crittografare qualsiasi buffer.
CKA_DECRYPT	TRUE		TRUE	Il valore Vero indica che è possibile utilizzare la chiave per decodificare qualsiasi buffer. Questo attributo è in genere impostato su FALSO per una chiave il cui CKA_WRAP è impostato su vero.
CKA_WRAP	TRUE	TRUE		Il valore Vero indica che è possibile utilizzare la chiave per eseguire il wrapping di un'altra chiave. In genere viene impostato su FALSO per chiavi private.

Attributo	Valore predefinito			Note
CKA_UNWRAP	TRUE		TRUE	Il valore Vero indica che è possibile utilizzare la chiave per annullare il wrapping (importare) di un'altra chiave.
CKA_SIGN	TRUE		TRUE	Il valore Vero indica che è possibile utilizzare la chiave per firmare messaggio di digest. In genere viene impostato su FALSO per le chiavi pubbliche e per le chiavi private archiviate.
CKA_VERIFY	TRUE	TRUE		Il valore Vero indica che è possibile utilizzare la chiave per verificare una firma. In genere è impostato su FALSO per chiavi private.

Attributo	Valore predefinito			Note
CKA_PRIVATE	TRUE	TRUE	TRUE	Il valore Vero indica che un utente potrebbe non avere accesso alla chiave finché l'utente non viene autentica to. Per chiarezza , gli utenti non possono accedere alle chiavi in CloudHSM fino a quando non vengono autenticati, anche se questo attributo è impostato su FALSO.

> **Note**
>
> È possibile ottenere un supporto più ampio per gli attributi nella libreria PKCS #11. Per ulteriori informazioni, vedi Attributi PKCS #11 supportati.

Impostazione attributi per una chiave

`CloudHsmKeyAttributesMap` è un oggetto simile a Java Map che puoi utilizzare per impostare i valori degli attributi per gli oggetti chiave. I metodi per la funzione `CloudHsmKeyAttributesMap` sono simili a quelli utilizzati per la manipolazione della mappa Java.

Per impostare valori personalizzati sugli attributi, sono disponibili due opzioni:

- Utilizzare i metodi elencati nella tabella seguente
- Utilizzare i modelli di generatore illustrati più avanti in questo documento

Gli oggetti della mappa attributi supportano i seguenti metodi per impostare gli attributi:

Operation	Valore restituito	Metodo **CloudHSMKeyAttributesMap**
Ottenere il valore di un attributo chiave per una chiave esistente	Oggetto (contenente il valore) o nulla	get(keyAttribute)
Inserire il valore di un attributo chiave	Il valore precedente associato all'attributo chiave o nulla se non esiste alcuna mappatura per un attributo chiave	put(keyAttribute, value)
Compilare i valori per più attributi chiave	N/D	PutAll () keyAttributesMap
Rimuovere una coppia chiave-valore dalla mappa degli attributi	Il valore precedente associato all'attributo chiave o nulla se non esiste alcuna mappatura per un attributo chiave	remove(keyAttribute)

> **Note**
>
> Eventuali attributi non specificati in modo esplicito vengono impostati sui valori predefiniti elencati nella tabella precedente in the section called "Attributi supportati".

Esempio di modello di generatore

Gli sviluppatori troveranno generalmente più conveniente utilizzare le classi tramite il modello di generatore. Come esempi:

```
import com.amazonaws.cloudhsm.CloudHsmKeyAttributes;
```

```
import com.amazonaws.cloudhsm.CloudHsmKeyAttributesMap;
import com.amazonaws.cloudhsm.CloudHsmKeyPairAttributesMap;

CloudHsmKeyAttributesMap keyAttributesSessionDecryptionKey =
   new CloudHsmKeyAttributesMap.Builder()
      .put(CloudHsmKeyAttributes.CKA_LABEL, "ExtractableSessionKeyEncryptDecrypt")
      .put(CloudHsmKeyAttributes.CKA_WRAP, false)
      .put(CloudHsmKeyAttributes.CKA_UNWRAP, false)
      .put(CloudHsmKeyAttributes.CKA_SIGN, false)
      .put(CloudHsmKeyAttributes.CKA_VERIFY, false)
      .build();

CloudHsmKeyAttributesMap keyAttributesTokenWrappingKey =
   new CloudHsmKeyAttributesMap.Builder()
      .put(CloudHsmKeyAttributes.CKA_LABEL, "TokenWrappingKey")
      .put(CloudHsmKeyAttributes.CKA_TOKEN, true)
      .put(CloudHsmKeyAttributes.CKA_ENCRYPT, false)
      .put(CloudHsmKeyAttributes.CKA_DECRYPT, false)
      .put(CloudHsmKeyAttributes.CKA_SIGN, false)
      .put(CloudHsmKeyAttributes.CKA_VERIFY, false)
      .build();
```

Gli sviluppatori possono inoltre utilizzare set di attributi predefiniti come un modo conveniente per applicare le best practice in modelli chiave. Ad esempio:

```
//best practice template for wrapping keys

CloudHsmKeyAttributesMap commonKeyAttrs = new CloudHsmKeyAttributesMap.Builder()
    .put(CloudHsmKeyAttributes.CKA_EXTRACTABLE, false)
    .put(CloudHsmKeyAttributes.CKA_DECRYPT, false)
    .build();

// initialize a new instance of CloudHsmKeyAttributesMap by copying commonKeyAttrs
// but with an appropriate label

CloudHsmKeyAttributesMap firstKeyAttrs = new CloudHsmKeyAttributesMap(commonKeyAttrs);
firstKeyAttrs.put(CloudHsmKeyAttributes.CKA_LABEL, "key label");

// alternatively, putAll() will overwrite existing values to enforce conformance

CloudHsmKeyAttributesMap secondKeyAttrs = new CloudHsmKeyAttributesMap();
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_DECRYPT, true);
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_ENCRYPT, true);
```

```
secondKeyAttrs.put(CloudHsmKeyAttributes.CKA_LABEL, “safe wrapping key”);
secondKeyAttrs.putAll(commonKeyAttrs); // will overwrite CKA_DECRYPT to be FALSE
```

Impostazione di attributi per una coppia di chiavi

Utilizza la classe Java `CloudHsmKeyPairAttributesMap` per gestire gli attributi chiave per una coppia di chiavi. `CloudHsmKeyPairAttributesMap` incapsula due oggetti `CloudHsmKeyAttributesMap`; uno per una chiave pubblica e uno per una chiave privata.

Per impostare singoli attributi per la chiave pubblica e la chiave privata separatamente, puoi utilizzare il metodo `put()` sull'oggetto mappa `CloudHsmKeyAttributes` corrispondente per tale chiave. Utilizza il metodo `getPublic()` per recuperare la mappa degli attributi per la chiave pubblica e utilizza `getPrivate()` per recuperare la mappa degli attributi per la chiave privata. Compila il valore di più attributi chiave insieme per coppie di chiavi pubbliche e private utilizzando la `putAll()` con una mappa degli attributi della coppia di chiavi come relativo argomento.

Esempio di modello di generatore

Gli sviluppatori troveranno generalmente più comodo impostare gli attributi chiave tramite il modello di generatore. Esempio:

```
import com.amazonaws.cloudhsm.CloudHsmKeyAttributes;
import com.amazonaws.cloudhsm.CloudHsmKeyAttributesMap;
import com.amazonaws.cloudhsm.CloudHsmKeyPairAttributesMap;

//specify attributes up-front
CloudHsmKeyAttributesMap keyAttributes =
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_SIGN, false)
        .put(CloudHsmKeyAttributes.CKA_LABEL, "PublicCertSerial12345")
        .build();

CloudHsmKeyPairAttributesMap keyPairAttributes =
    new CloudHsmKeyPairAttributesMap.Builder()
        .withPublic(keyAttributes)
        .withPrivate(
            new CloudHsmKeyAttributesMap.Builder() //or specify them inline
                .put(CloudHsmKeyAttributes.CKA_LABEL, "PrivateCertSerial12345")
                .put (CloudHSMKeyAttributes.CKA_WRAP, FALSE)
                .build()
        )
        .build();
```

> **Note**
>
> Per ulteriori informazioni su questa estensione proprietaria, vedete l'archivio Javadoc e l'esempio su. GitHub Per esplorare Javadoc, scarica ed espandi l'archivio.

Mettere tutto insieme

Per specificare gli attributi chiave con le operazioni chiave, attenersi alla seguente procedura:

1. Creare un'istanza `CloudHsmKeyAttributesMap` per chiavi simmetriche o `CloudHsmKeyPairAttributesMap` per coppie di chiavi.
2. Definire l'oggetto attributi dalla fase 1 con gli attributi e i valori chiave richiesti.
3. Creare un'istanza di una classe `Cavium*ParameterSpec`, corrispondente al tipo di chiave specifico e passare al costruttore questo oggetto attributi configurato.
4. Passare questo oggetto `Cavium*ParameterSpec` in una classe o metodo crittografico corrispondente.

Per riferimento, la tabella seguente contiene le classi `Cavium*ParameterSpec` e i metodi che supportano gli attributi chiave personalizzati.

Tipo di chiavi	Classe specifiche del parametro	Esempio Costruttori
Classe di base	`CaviumKeyGenAlgorithmParameterSpec`	`CaviumKeyGenAlgorithmParameterSpec(CloudHsmKeyAttributesMap keyAttributesMap)`
DES	`CaviumDESKeyGenParameterSpec`	`CaviumDESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)`

Tipo di chiavi	Classe specifiche del parametro	Esempio Costruttori
RSA	`CaviumRSAKeyGenParameterSpec`	`CaviumRSAKeyGenParameterSpec(int keysize, BigInteger publicExponent, CloudHsmKeyPairAttributesMap keyPairAttributesMap)`
Segreto	`CaviumGenericSecretKeyGenParameterSpec`	`CaviumGenericSecretKeyGenParameterSpec(int size, CloudHsmKeyAttributesMap keyAttributesMap)`
AES	`CaviumAESKeyGenParameterSpec`	`CaviumAESKeyGenParameterSpec(int keySize, byte[] iv, CloudHsmKeyAttributesMap keyAttributesMap)`
EC	`CaviumECGenParameterSpec`	`CaviumECGenParameterSpec(String stdName, CloudHsmKeyPairAttributesMap keyPairAttributesMap)`

Esempio di codice: generare ed eseguire il wrapping di una chiave

Questi brevi codici di esempio illustrano le fasi per due diverse operazioni: Generazione chiave e Wrapping della chiave:

```
// Set up the desired key attributes

KeyGenerator keyGen = KeyGenerator.getInstance("AES", "Cavium");
CaviumAESKeyGenParameterSpec keyAttributes = new CaviumAESKeyGenParameterSpec(
    256,
    new CloudHsmKeyAttributesMap.Builder()
        .put(CloudHsmKeyAttributes.CKA_LABEL, "MyPersistentAESKey")
        .put(CloudHsmKeyAttributes.CKA_EXTRACTABLE, true)
        .put(CloudHsmKeyAttributes.CKA_TOKEN, true)
        .build()
);

// Assume we already have a handle to the myWrappingKey
// Assume we already have the wrappedBytes to unwrap

// Unwrap a key using Custom Key Attributes

CaviumUnwrapParameterSpec unwrapSpec = new
 CaviumUnwrapParameterSpec(myInitializationVector, keyAttributes);

Cipher unwrapCipher = Cipher.getInstance("AESWrap", "Cavium");
unwrapCipher.init(Cipher.UNWRAP_MODE, myWrappingKey, unwrapSpec);
Key unwrappedKey = unwrapCipher.unwrap(wrappedBytes, "AES", Cipher.SECRET_KEY);
```

Esempi di codice per la libreria AWS CloudHSM software per Java for Client SDK 3

Questo argomento fornisce risorse e informazioni sugli esempi di codice Java per AWS CloudHSM Client SDK 3.

Prerequisiti

Prima di eseguire gli esempi, è necessario configurare l'ambiente:

- Installa e configura il provider Java Cryptographic Extension (JCE) e il pacchetto client AWS CloudHSM.
- Configura un nome utente e una password HSM validi. Le autorizzazioni per l'utente di crittografia (CU) sono sufficienti per queste attività. L'applicazione utilizza queste credenziali per accedere all'HSM in ciascun esempio.
- Decidi come fornire le credenziali al provider JCE.

Esempi di codice

I seguenti esempi di codice mostrano come utilizzare il provider JCE AWS CloudHSM per eseguire attività di base. Altri esempi di codice sono disponibili su. GitHub

- Esegui l'accesso a un modulo HSM
- Gestisci chiavi
- Generazione di una chiave AES
- Crittografia e decodifica con AES-GCM
- Crittografia e decodifica con AES-CTR
- Crittografia e decodifica con D3DES-ECB vedi nota 1
- Wrapping e annullamento del wrapping delle chiavi con AES-GCM
- Wrapping e annullamento del wrapping delle chiavi con AES
- Wrapping e annullamento del wrapping delle chiavi con RSA
- Usa attributi chiave supportati
- Enumerazione delle chiavi nell'archivio delle chiavi
- Utilizzo dell'archivio delle chiavi CloudHSM
- Firma di messaggi in un esempio multi-thread
- Firma e verifica con chiavi EC

[1] In conformità alle linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

AWS CloudHSM KeyStore Classe Java per Client SDK 3

La AWS CloudHSM **`KeyStore`** classe fornisce un archivio di PKCS12 chiavi per scopi speciali che consente l'accesso alle AWS CloudHSM chiavi tramite applicazioni come keytool e jarsigner. Questo archivio chiavi può archiviare i certificati insieme ai dati della chiave e correlarli ai dati della chiave memorizzati su AWS CloudHSM.

> **Note**
>
> Poiché i certificati sono informazioni pubbliche e, per massimizzare la capacità di archiviazione delle chiavi crittografiche, AWS CloudHSM non supporta l'archiviazione dei certificati su. HSMs

La AWS CloudHSM `KeyStore` classe implementa la `KeyStore` Service Provider Interface (SPI) della Java Cryptography Extension (JCE). Per ulteriori informazioni sull'utilizzo`KeyStore`, vedete Class. KeyStore

Scegli l'archivio di chiavi appropriato per AWS CloudHSM Client SDK 3

Il provider AWS CloudHSM Java Cryptographic Extension (JCE) è dotato di un key store pass-through predefinito di sola lettura che trasferisce tutte le transazioni all'HSM. Questo archivio di chiavi predefinito è distinto da quello per scopi speciali. AWS CloudHSM KeyStore Nella maggior parte delle situazioni, è possibile ottenere prestazioni di runtime e velocità effettiva migliori utilizzando l'impostazione predefinita. È consigliabile utilizzarlo solo AWS CloudHSM KeyStore per le applicazioni in cui è necessario il supporto per i certificati e le operazioni basate sui certificati oltre a trasferire le operazioni chiave sull'HSM.

Sebbene entrambi gli archivi di chiavi utilizzino il provider JCE per le operazioni, sono entità indipendenti e non scambiano informazioni tra loro.

Carica l'archivio chiavi predefinito per l'applicazione Java come segue:

```
KeyStore ks = KeyStore.getInstance("Cavium");
```

Caricate il CloudHSM per scopi speciali come segue: KeyStore

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inizializza for Client SDK 3 AWS CloudHSM KeyStore

Effettua AWS CloudHSM KeyStore l'accesso nello stesso modo in cui accedi al provider JCE. È possibile utilizzare le variabili di ambiente o il file delle proprietà del sistema ed è necessario effettuare il login prima di iniziare a utilizzare CloudHSM KeyStore. Per un esempio di accesso a un HSM utilizzando JCE, vedi Accedi a un HSM.

Se lo desideri, puoi specificare una password per crittografare il PKCS12 file locale che contiene i dati dell'archivio delle chiavi. Quando si crea il AWS CloudHSM Keystore, si imposta la password e la si fornisce quando si utilizzano i metodi load, set e get.

Crea un'istanza di un nuovo oggetto CloudHSM come segue KeyStore :

```
ks.load(null, null);
```

Scrivi i dati dell'archivio chiavi in un file utilizzando il metodo `store`. Da quel momento in poi, puoi caricare l'archivio chiavi esistente utilizzando il metodo `load` con il file sorgente e la password come segue:

```
ks.load(inputStream, password);
```

Usa for Client SDK 3 AWS CloudHSM KeyStore AWS CloudHSM

Un oggetto KeyStore CloudHSM viene generalmente utilizzato tramite un'applicazione di terze parti come jarsigner o keytool. Puoi anche accedere direttamente all'oggetto con il codice.

AWS CloudHSM KeyStore è conforme alle specifiche della classe KeyStore JCE e fornisce le seguenti funzioni.

- `load`

  Carica l'archivio chiavi dal flusso di input specificato. Se durante il salvataggio dell'archivio chiavi è stata impostata una password, è necessario fornire questa stessa password affinché il caricamento abbia esito positivo. Impostare entrambi i parametri su nulla per inizializzare un nuovo archivio di chiavi vuoto.

  ```
  KeyStore ks = KeyStore.getInstance("CloudHSM");
  ks.load(inputStream, password);
  ```

- `aliases`

  Restituisce un'enumerazione dei nomi alias di tutte le voci nell'istanza dell'archivio chiavi considerato. I risultati includono oggetti archiviati localmente nel PKCS12 file e oggetti residenti nell'HSM.

  Esempio di codice:

  ```
  KeyStore ks = KeyStore.getInstance("CloudHSM");
  ```

```
for(Enumeration<String> entry = ks.aliases(); entry.hasMoreElements();)
 {
  String label = entry.nextElement();
  System.out.println(label);
 }
```

- `ContainsAlias`

  Restituisce vero se l'archivio chiavi ha accesso ad almeno un oggetto con l'alias specificato. L'archivio delle chiavi controlla gli oggetti archiviati localmente nel PKCS12 file e gli oggetti che risiedono sull'HSM.

- `DeleteEntry`

  Elimina una voce di certificato dal file locale PKCS12 . L'eliminazione dei dati chiave archiviati in un HSM non è supportata utilizzando. AWS CloudHSM KeyStore È possibile eliminare le chiavi con lo strumento key_mgmt_util di CloudHSM.

- `GetCertificate`

  Restituisce il certificato associato a un alias, se disponibile. Se l'alias non esiste o fa riferimento a un oggetto che non è un certificato, la funzione restituisce NULLA.

  ```
  KeyStore ks = KeyStore.getInstance("CloudHSM");
  Certificate cert = ks.getCertificate(alias)
  ```

- `GetCertificateAlias`

  Restituisce il nome (alias) della prima voce dell'archivio chiavi i cui dati corrispondono al certificato specificato.

  ```
  KeyStore ks = KeyStore.getInstance("CloudHSM");
  String alias = ks.getCertificateAlias(cert)
  ```

- `GetCertificateChain`

  Restituisce la catena di certificati associata all'alias specificato. Se l'alias non esiste o fa riferimento a un oggetto che non è un certificato, la funzione restituisce NULLA.

- `GetCreationDate`

  Restituisce la data di creazione della voce identificata dall'alias specificato. Se una data di creazione non è disponibile, la funzione restituisce la data in cui il certificato è diventato valido.

- `GetKey`

  GetKey viene passato all'HSM e restituisce un oggetto chiave corrispondente all'etichetta specificata. Poiché interroga `getKey` direttamente l'HSM, può essere utilizzato per qualsiasi chiave sull'HSM indipendentemente dal fatto che sia stata generata da. KeyStore

  ```
  Key key = ks.getKey(keyLabel, null);
  ```

- `IsCertificateEntry`

  Controlla se la voce con l'alias specificato rappresenta una voce di certificato.

- `IsKeyEntry`

  Controlla se la voce con l'alias specificato rappresenta una voce chiave. L'azione cerca l'alias sia nel PKCS12 file che nell'HSM.

- `SetCertificateEntry`

  Assegna il certificato dato all'alias specificato. Se l'alias specificato è già in uso per identificare una chiave o un certificato, viene generata una `KeyStoreException`. È possibile utilizzare il codice JCE per ottenere l'oggetto chiave e quindi utilizzare il KeyStore `SetKeyEntry` metodo per associare il certificato alla chiave.

- `SetKeyEntry` con chiave `byte[]`

  Questa API non è attualmente supportata con Client SDK 3.

- `SetKeyEntry` con oggetto `Key`

  Assegna la chiave considerata all'alias specificato e la memorizza all'interno dell'HSM. Se l'oggetto `Key` non è di tipo `CaviumKey`, la chiave viene importata nell'HSM come chiave di sessione estraibile.

  Se l'oggetto `Key` è di tipo `PrivateKey`, deve essere accompagnato da una catena di certificati corrispondente.

  Se l'alias esiste già, la chiamata `SetKeyEntry` genera un `KeyStoreException` e impedisce la sovrascrittura della chiave. Se la chiave deve essere sovrascritta, utilizza KMU o JCE a tale scopo.

- `EngineSize`

  Restituisce il numero di voci nell'archivio chiavi.

- `Store`

Memorizza l'archivio delle chiavi nel flusso di output specificato come PKCS12 file e lo protegge con la password specificata. Inoltre, persiste tutte le chiavi caricate (che sono impostate usando le chiamate `setKey`).

API di crittografia: Next Generation (CNG) e provider di archiviazione delle chiavi (KSP) per AWS CloudHSM

Il AWS CloudHSM client per Windows include provider CNG e KSP.

I provider di archiviazione delle chiavi (KSPs) consentono l'archiviazione e il recupero delle chiavi. Ad esempio, se aggiungi i Servizi certificati Active Directory (AD CS) di Microsoft al server Windows e decidi di creare una nuova chiave privata per l'autorità di certificazione (CA), potrai scegliere il provider KSP che gestirà l'archiviazione delle chiavi. Quando configuri il ruolo AD CS, puoi scegliere questo KSP. Per ulteriori informazioni, consulta Creare un'autorità di certificazione (CA) Windows Server.

Cryptography API: Next Generation (CNG) è un'API di crittografia specifica per il sistema operativo Microsoft Windows. CNG consente agli sviluppatori di utilizzare tecniche di crittografia per proteggere le applicazioni basate su Windows. Ad alto livello, l' AWS CloudHSM implementazione di CNG offre le seguenti funzionalità:

- Primitive crittografiche: consentono di eseguire operazioni crittografiche fondamentali.
- Importazione ed esportazione di chiavi: consente di importare ed esportare chiavi asimmetriche.
- Data Protection API (CNG DPAPI): consente di crittografare e decrittografare facilmente i dati.
- Archiviazione e recupero delle chiavi: consente di archiviare e isolare in modo sicuro la chiave privata di una coppia di chiavi asimmetrica.

Argomenti

- Verifica i fornitori KSP e CNG per AWS CloudHSM
- Prerequisiti per l'utilizzo del client AWS CloudHSM Windows
- Associare una AWS CloudHSM chiave a un certificato
- Esempio di codice per il fornitore di CNG per AWS CloudHSM

Verifica i fornitori KSP e CNG per AWS CloudHSM

I provider KSP e CNG vengono installati quando si installa il client Windows AWS CloudHSM . È possibile installare il client seguendo i passaggi descritti in Installare il client (Windows).

Utilizza le seguenti sezioni per verificare l'installazione dei provider.

Configura ed esegui il AWS CloudHSM client Windows

Per avviare il client Windows CloudHSM, è necessario soddisfare per prima cosa i Prerequisiti. Quindi, aggiorna i file di configurazione utilizzati dai provider e avvia il client completando i passaggi seguenti. È necessario eseguire questi passaggi la prima volta che si utilizzano i provider KSP e CNG e dopo averli aggiunti o rimossi HSMs nel cluster. In questo modo, AWS CloudHSM è in grado di sincronizzare i dati e mantenere la coerenza HSMs in tutto il cluster.

Fase 1: Arrestare il client AWS CloudHSM

Prima di aggiornare i file di configurazione utilizzati dai provider, arrestate il AWS CloudHSM client. Se il client è già stato arrestato, eseguire il comando stop non ha alcun effetto.

- Per client Windows dalla versione 1.1.2 in poi:

```
C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

  Usa Ctrl + C nella finestra di comando in cui hai avviato il AWS CloudHSM client.

Fase 2: Aggiornare i AWS CloudHSM file di configurazione

Questo passaggio utilizza il `-a` parametro dello strumento Configure per aggiungere l'indirizzo IP ENI (elastic network interface) di uno dei HSMs componenti del cluster al file di configurazione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\configure.exe" -a <HSM ENI IP>
```

Per ottenere l'indirizzo IP ENI di un HSM nel cluster, accedi alla AWS CloudHSM console, scegli i cluster e seleziona il cluster desiderato. È inoltre possibile utilizzare l'DescribeClustersoperazione, il comando describe-clusters o il cmdlet. Get-HSM2Cluster PowerShell Digitare solo un indirizzo IP ENI. Non importa l'indirizzo IP ENI utilizzato.

Passaggio 3: avviare il client AWS CloudHSM

Quindi, avvia o riavvia il AWS CloudHSM client. All'avvio, il AWS CloudHSM client utilizza l'indirizzo IP ENI nel suo file di configurazione per interrogare il cluster. Quindi aggiunge gli indirizzi IP ENI di tutti i membri HSMs del cluster al file di informazioni sul cluster.

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:
\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Controllo dei provider KSP e CNG

È possibile utilizzare uno dei seguenti comandi per determinare quali provider sono installati nel sistema. I comandi elencano i provider KSP e CNG registrati. Il client AWS CloudHSM non deve essere in esecuzione.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\ksp_config.exe" -enum
```

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cng_config.exe" -enum
```

Per verificare che i provider KSP e CNG siano installati sulla tua istanza di Windows Server EC2, dovresti vedere le seguenti voci nell'elenco:

```
Cavium CNG Provider
Cavium Key Storage Provider
```

Se il provider CNG manca, esegui il comando seguente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\cng_config.exe" -register
```

Se il provider CNG manca, esegui il comando seguente.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\ksp_config.exe" -register
```

Prerequisiti per l'utilizzo del client AWS CloudHSM Windows

Prima di avviare il AWS CloudHSM client Windows e utilizzare i provider KSP e CNG, è necessario impostare le credenziali di accesso per l'HSM sul sistema. Puoi impostare le credenziali tramite Gestione credenziali di Windows o la variabile di ambiente di sistema. Ti consigliamo di utilizzare Gestione credenziali di Windows per archiviare le credenziali. Questa opzione è disponibile con la versione AWS CloudHSM del client 2.0.4 e successive. L'utilizzo della variabile di ambiente è più semplice da configurare, ma meno sicura rispetto all'utilizzo di Gestione credenziali di Windows.

Gestione credenziali di Windows

Puoi utilizzare l'utility `set_cloudhsm_credentials` o l'interfaccia di Gestione credenziali di Windows.

- Utilizzo dell'utility **`set_cloudhsm_credentials`**:

  L'utility `set_cloudhsm_credentials` è inclusa nel programma di installazione di Windows. Puoi utilizzare questa utility per trasferire le credenziali di accesso HSM a Gestione credenziali di Windows. Se desideri compilare questa utility dall'origine, puoi utilizzare il codice Python incluso nel programma di installazione.

  1. Passare alla cartella `C:\Program Files\Amazon\CloudHSM\tools\`.
  2. Esegui il file `set_cloudhsm_credentials.exe` con il nome utente e i parametri della password del CU.

     ```
     set_cloudhsm_credentials.exe --username <CU USER> --password <CU PASSWORD>
     ```

- Utilizzo dell'interfaccia di gestione delle credenziali:

  Puoi utilizzare l'interfaccia di gestione delle credenziali per gestire manualmente le credenziali.

  1. Per aprire Credential Manager, digita `credential manager` nella casella di ricerca sulla barra delle applicazioni e seleziona Gestione delle credenziali.
  2. Seleziona Credenziali di Windows per gestire le credenziali di Windows.
  3. Seleziona Aggiungi una credenziale generica e compila i dettagli come segue:
     - In Indirizzo Internet o di rete, immetti il nome della destinazione come `cloudhsm_client`.
     - In Nome utente e Password immettere le credenziali CU.
     - Fai clic su OK.

Variabili di ambiente del sistema

Puoi impostare variabili di ambiente di sistema che identificano un HSM e un crypto user (CU) per l'applicazione Windows. Puoi eseguire il comando setx per impostare le variabili di ambiente del sistema temporanee o definitive in modo programmatico oppure nella scheda Avanzate del pannello di controllo Proprietà di sistema di Windows.

> **Warning**
>
> Quando imposti le credenziali tramite variabili di ambiente di sistema, la password è disponibile in testo normale nel sistema di un utente. Per risolvere questo problema, utilizza Gestione credenziali di Windows.

Imposta le seguenti variabili di ambiente del sistema:

`n3fips_password=`*`<CU USERNAME>`*`:`*`<CU PASSWORD>`*

Identifica un crypto user (CU) nell'HSM e fornisce tutte le informazioni di login richieste. La tua applicazione viene autenticata ed eseguita come questo CU. L'applicazione dispone delle autorizzazioni di questo CU e può visualizzare e gestire solo le chiavi di proprietà del CU e quelle con questi condivise. Per creare un nuovo CU, utilizza createUser. Per trovare quelli esistenti CUs, usa ListUsers.

Esempio:

```
setx /m n3fips_password test_user:password123
```

Associare una AWS CloudHSM chiave a un certificato

Prima di poter utilizzare AWS CloudHSM le chiavi con strumenti di terze parti, come Microsoft SignTool, devi importare i metadati della chiave nell'archivio certificati locale e associarli a un certificato. Per importare i metadati della chiave, utilizza l'utilità import_key.exe inclusa in CloudHSM versione 3.0 e successive. La procedura seguente fornisce informazioni aggiuntive e un esempio output.

Passaggio 1: importare il certificato

In Windows, dovresti essere in grado di fare doppio clic sul certificato per importarlo nell'archivio certificati locale.

Tuttavia, se il doppio clic non funziona, utilizza lo strumento Microsoft Certreq per importare il certificato nel gestore certificati. Esempio:

```
certreq -accept <certificatename>
```

Se questa azione non riesce e viene visualizzato l'errore `Key not found`, passa al passaggio 2. Se il certificato viene visualizzato nell'archivio chiavi, l'attività è stata completata con successo e non sono necessarie ulteriori azioni.

Passaggio 2: raccogliere informazioni sull'identificazione dei certificati

Se il passaggio precedente non ha avuto esito positivo, dovrai associare la chiave privata a un certificato. Tuttavia, prima di poter creare l'associazione, devi innanzitutto trovare il nome del container univoco e il numero di serie del certificato. Utilizzate un'utilità, ad esempiocertutil, per visualizzare le informazioni necessarie sul certificato. Il seguente esempio di output di certutil mostra il nome del contenitore e il numero di serie.

```
================ Certificate 1 ================ Serial Number:
   72000000047f7f7a9d41851b4e000000000004Issuer: CN=Enterprise-CANotBefore: 10/8/2019
 11:50
   AM NotAfter: 11/8/2020 12:00 PMSubject: CN=www.example.com, OU=Certificate
 Management,
   O=Information Technology, L=Seattle, S=Washington, C=USNon-root CertificateCert
   Hash(sha1): 7f d8 5c 00 27 bf 37 74 3d 71 5b 54 4e c0 94 20 45 75 bc 65No key
 provider
   information Simple container name: CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
 Unique
   container name: CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
```

Fase 3: Associare la chiave AWS CloudHSM privata al certificato

Per associare la chiave al certificato, assicurati innanzitutto di avviare il demone AWS CloudHSM client. Quindi, utilizza import_key.exe (incluso in CloudHsm versione 3.0 e successive) per associare la chiave privata al certificato. Quando si specifica il certificato, utilizzare il nome del contenitore

semplice. L'esempio seguente mostra il comando e la risposta. Questa azione copia solo i metadati della chiave; la chiave rimane sull'HSM.

```
$> import_key.exe -RSA CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c

Successfully opened Microsoft Software Key Storage Provider : 0NCryptOpenKey failed :
 80090016
```

Passaggio 4: aggiornare l'archivio certificati

Assicurati che il daemon AWS CloudHSM client sia ancora in esecuzione. Quindi, utilizzate il certutil verbo,-repairstore, per aggiornare il numero di serie del certificato. L'esempio seguente mostra il comando e l'output. Per informazioni sul -repairstoreverbo, consulta la documentazione Microsoft.

```
C:\Program Files\Amazon\CloudHSM>certutil -f -csp "Cavium Key Storage Provider"-
repairstore my "72000000047f7f7a9d41851b4e000000000004"
my "Personal"
================ Certificate 1 ================
Serial Number: 72000000047f7f7a9d41851b4e000000000004
Issuer: CN=Enterprise-CA
NotBefore: 10/8/2019 11:50 AM
NotAfter: 11/8/2020 12:00 PM
Subject: CN=www.example.com, OU=Certificate Management, O=Information Technology,
 L=Seattle, S=Washington, C=US
Non-root CertificateCert Hash(sha1): 7f d8 5c 00 27 bf 37 74 3d 71 5b 54 4e c0 94 20 45
 75 bc 65
SDK Version: 3.0
Key Container = CertReq-39c04db0-6aa9-4310-93db-db0d9669f42c
Provider = "Cavium Key Storage Provider"
Private key is NOT exportableEncryption test passedCertUtil: -repairstore command
 completed successfully.
```

Dopo aver aggiornato il numero di serie del certificato, puoi utilizzare questo certificato e la chiave AWS CloudHSM privata corrispondente con qualsiasi strumento di firma di terze parti su Windows.

Esempio di codice per il fornitore di CNG per AWS CloudHSM

⚠ ** Solo codice di esempio, non per uso in produzione**

Questo codice di esempio è solo a scopo illustrativo. Non eseguire questo codice in fase di produzione.

Il seguente esempio mostra come enumerare i provider di crittografia registrati nel sistema per trovare il provider CNG installato con il client CloudHSM per Windows. L'esempio mostra inoltre come creare una coppia di chiavi asimmetriche e come utilizzare la coppia di chiavi per firmare i dati.

Important

Prima di eseguire questo esempio, devi configurare le credenziali HSM come descritto nei prerequisiti. Per informazioni dettagliate, vedi Prerequisiti per l'utilizzo del client AWS CloudHSM Windows.

```
// CloudHsmCngExampleConsole.cpp : Console application that demonstrates CNG
 capabilities.
// This example contains the following functions.
//
//   VerifyProvider()          - Enumerate the registered providers and retrieve Cavium
 KSP and CNG providers.
//   GenerateKeyPair()         - Create an RSA key pair.
//   SignData()                - Sign and verify data.
//

#include "stdafx.h"
#include <Windows.h>

#ifndef NT_SUCCESS
#define NT_SUCCESS(Status) ((NTSTATUS)(Status) >= 0)
#endif

#define CAVIUM_CNG_PROVIDER L"Cavium CNG Provider"
#define CAVIUM_KEYSTORE_PROVIDER L"Cavium Key Storage Provider"

// Enumerate the registered providers and determine whether the Cavium CNG provider
// and the Cavium KSP provider exist.
//
bool VerifyProvider()
```

```
{
  NTSTATUS status;
  ULONG cbBuffer = 0;
  PCRYPT_PROVIDERS pBuffer = NULL;
  bool foundCng = false;
  bool foundKeystore = false;

  // Retrieve information about the registered providers.
  //   cbBuffer - the size, in bytes, of the buffer pointed to by pBuffer.
  //   pBuffer - pointer to a buffer that contains a CRYPT_PROVIDERS structure.
  status = BCryptEnumRegisteredProviders(&cbBuffer, &pBuffer);

  // If registered providers exist, enumerate them and determine whether the
  // Cavium CNG provider and Cavium KSP provider have been registered.
  if (NT_SUCCESS(status))
  {
    if (pBuffer != NULL)
    {
      for (ULONG i = 0; i < pBuffer->cProviders; i++)
      {
        // Determine whether the Cavium CNG provider exists.
        if (wcscmp(CAVIUM_CNG_PROVIDER, pBuffer->rgpszProviders[i]) == 0)
        {
          printf("Found %S\n", CAVIUM_CNG_PROVIDER);
          foundCng = true;
        }

        // Determine whether the Cavium KSP provider exists.
        else if (wcscmp(CAVIUM_KEYSTORE_PROVIDER, pBuffer->rgpszProviders[i]) == 0)
        {
          printf("Found %S\n", CAVIUM_KEYSTORE_PROVIDER);
          foundKeystore = true;
        }
      }
    }
  }
  else
  {
    printf("BCryptEnumRegisteredProviders failed with error code 0x%08x\n", status);
  }

  // Free memory allocated for the CRYPT_PROVIDERS structure.
  if (NULL != pBuffer)
  {
```

```
    BCryptFreeBuffer(pBuffer);
  }

  return foundCng == foundKeystore == true;
}

// Generate an asymmetric key pair. As used here, this example generates an RSA key
 pair
// and returns a handle. The handle is used in subsequent operations that use the key
 pair.
// The key material is not available.
//
// The key pair is used in the SignData function.
//
NTSTATUS GenerateKeyPair(BCRYPT_ALG_HANDLE hAlgorithm, BCRYPT_KEY_HANDLE *hKey)
{
  NTSTATUS status;

  // Generate the key pair.
  status = BCryptGenerateKeyPair(hAlgorithm, hKey, 2048, 0);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptGenerateKeyPair failed with code 0x%08x\n", status);
    return status;
  }

  // Finalize the key pair. The public/private key pair cannot be used until this
  // function is called.
  status = BCryptFinalizeKeyPair(*hKey, 0);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptFinalizeKeyPair failed with code 0x%08x\n", status);
    return status;
  }

  return status;
}

// Sign and verify data using the RSA key pair. The data in this function is hardcoded
// and is for example purposes only.
//
NTSTATUS SignData(BCRYPT_KEY_HANDLE hKey)
{
  NTSTATUS status;
```

```
  PBYTE sig;
  ULONG sigLen;
  ULONG resLen;
  BCRYPT_PKCS1_PADDING_INFO pInfo;

  // Hardcode the data to be signed (for demonstration purposes only).
  PBYTE message = (PBYTE)"d83e7716bed8a20343d8dc6845e57447";
  ULONG messageLen = strlen((char*)message);

  // Retrieve the size of the buffer needed for the signature.
  status = BCryptSignHash(hKey, NULL, message, messageLen, NULL, 0, &sigLen, 0);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptSignHash failed with code 0x%08x\n", status);
    return status;
  }

  // Allocate a buffer for the signature.
  sig = (PBYTE)HeapAlloc(GetProcessHeap(), 0, sigLen);
  if (sig == NULL)
  {
    return -1;
  }

  // Use the SHA256 algorithm to create padding information.
  pInfo.pszAlgId = BCRYPT_SHA256_ALGORITHM;

  // Create a signature.
  status = BCryptSignHash(hKey, &pInfo, message, messageLen, sig, sigLen, &resLen,
BCRYPT_PAD_PKCS1);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptSignHash failed with code 0x%08x\n", status);
    return status;
  }

  // Verify the signature.
  status = BCryptVerifySignature(hKey, &pInfo, message, messageLen, sig, sigLen,
BCRYPT_PAD_PKCS1);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptVerifySignature failed with code 0x%08x\n", status);
    return status;
  }
```

```
  // Free the memory allocated for the signature.
  if (sig != NULL)
  {
    HeapFree(GetProcessHeap(), 0, sig);
    sig = NULL;
  }

  return 0;
}

// Main function.
//
int main()
{
  NTSTATUS status;
  BCRYPT_ALG_HANDLE hRsaAlg;
  BCRYPT_KEY_HANDLE hKey = NULL;

  // Enumerate the registered providers.
  printf("Searching for Cavium providers...\n");
  if (VerifyProvider() == false) {
    printf("Could not find the CNG and Keystore providers\n");
    return 1;
  }

  // Get the RSA algorithm provider from the Cavium CNG provider.
  printf("Opening RSA algorithm\n");
  status = BCryptOpenAlgorithmProvider(&hRsaAlg, BCRYPT_RSA_ALGORITHM,
 CAVIUM_CNG_PROVIDER, 0);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptOpenAlgorithmProvider RSA failed with code 0x%08x\n", status);
    return status;
  }

  // Generate an asymmetric key pair using the RSA algorithm.
  printf("Generating RSA Keypair\n");
  GenerateKeyPair(hRsaAlg, &hKey);
  if (hKey == NULL)
  {
    printf("Invalid key handle returned\n");
    return 0;
  }
```

```
  printf("Done!\n");

  // Sign and verify [hardcoded] data using the RSA key pair.
  printf("Sign/Verify data with key\n");
  SignData(hKey);
  printf("Done!\n");

  // Remove the key handle from memory.
  status = BCryptDestroyKey(hKey);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptDestroyKey failed with code 0x%08x\n", status);
    return status;
  }

  // Close the RSA algorithm provider.
  status = BCryptCloseAlgorithmProvider(hRsaAlg, NULL);
  if (!NT_SUCCESS(status))
  {
    printf("BCryptCloseAlgorithmProvider RSA failed with code 0x%08x\n", status);
    return status;
  }

  return 0;
}
```

Integrazione di applicazioni di terze parti con AWS CloudHSM

Alcuni dei casi d'uso AWS CloudHSM riguardano l'integrazione di applicazioni software di terze parti con l'HSM del cluster. AWS CloudHSM Integrando software di terze parti con AWS CloudHSM, è possibile raggiungere una serie di obiettivi relativi alla sicurezza. Gli argomenti seguenti illustrano come raggiungerne alcuni.

Argomenti

- Migliora la sicurezza del tuo server web con SSL/TLS offload in AWS CloudHSM
- Configurare Windows Server come autorità di certificazione (CA) con AWS CloudHSM
- Crittografia trasparente dei dati (TDE) del database Oracle con AWS CloudHSM
- Usa Microsoft SignTool con AWS CloudHSM per firmare i file
- Integrazione di Java Keytool e Jarsigner con AWS CloudHSM
- Usa Microsoft Manifest Generation and Editing Tool (Mage.exe) con AWS CloudHSM per firmare i file
- Integrazioni con altri fornitori di terze parti con AWS CloudHSM

Migliora la sicurezza del tuo server web con SSL/TLS offload in AWS CloudHSM

I server Web e i relativi client (browser Web) possono utilizzare i protocolli Secure Sockets Layer (SSL) o Transport Layer Security (TLS) per confermare l'identità del server Web e stabilire una connessione sicura che invia e riceve pagine Web o altri dati su Internet. Questo protocollo è comunemente noto come HTTPS. Il server Web utilizza una coppia di chiavi pubblica-privata e un certificato di chiave SSL/TLS pubblica per stabilire una sessione HTTPS con ciascun client. Questo processo comporta molte operazioni di calcolo per i server Web, ma è possibile trasferirle in parte al AWS CloudHSM cluster, operazione denominata accelerazione SSL. L'offloading riduce il carico di calcolo sui server Web e offre una maggiore sicurezza archiviando le chiavi private dei server. HSMs

I seguenti argomenti forniscono una panoramica del AWS CloudHSM funzionamento di SSL/TLS offload with e dei tutorial per la configurazione SSL/TLS di offload with sulle seguenti piattaforme. AWS CloudHSM

Per Linux, utilizza OpenSSL Dynamic Engine sul software dei server Web NGINX o Apache HTTP Server

Per Windows, utilizza il software del server Web Internet Information Services (IIS) for Windows Server

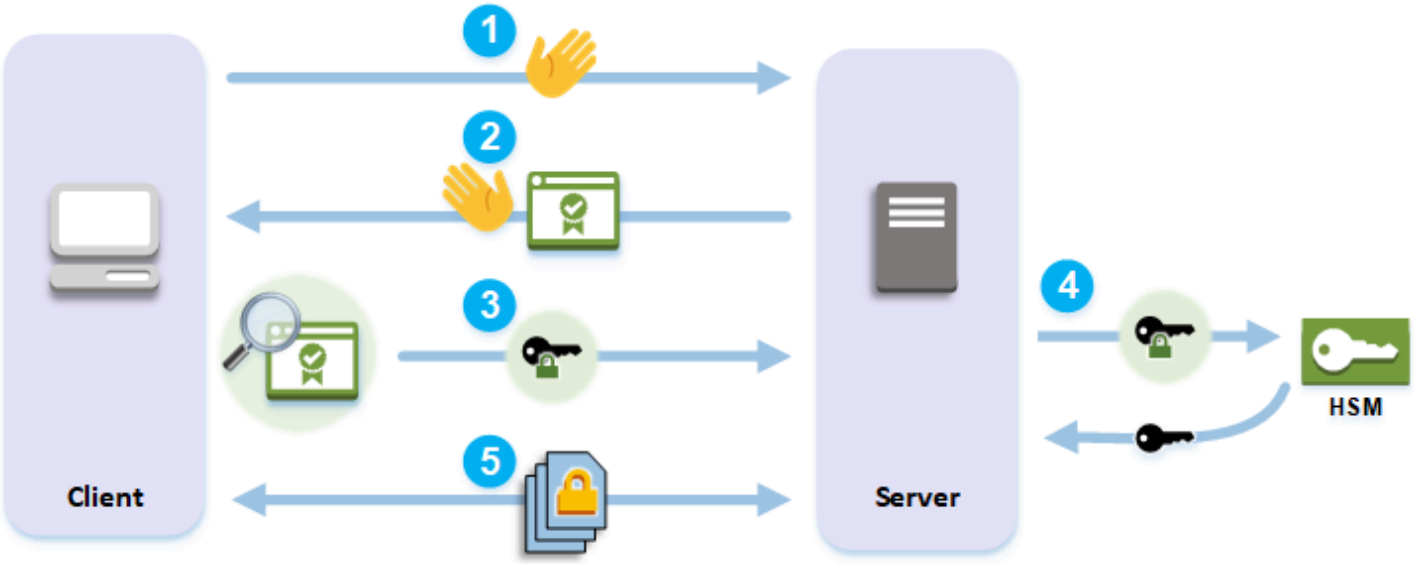
Argomenti

- Come SSL/TLS scaricare con AWS CloudHSM funzionamento
- AWS CloudHSM SSL/TLS offload su Linux usando NGINX o Apache con OpenSSL
- AWS CloudHSM SSL/TLS offload su Linux utilizzando NGINX o HAProxy con OpenSSL Provider
- AWS CloudHSM Offload SSL/TLS su Linux utilizzando Tomcat con JSSE
- AWS CloudHSM Offload SSL/TLS su Windows tramite IIS con KSP
- Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale)

Come SSL/TLS scaricare con AWS CloudHSM funzionamento

Per stabilire una connessione HTTPS, il server Web esegue un processo di handshake con i client. Come parte di questo processo, il server trasferisce parte dell'elaborazione crittografica agli HSM del AWS CloudHSM cluster, come illustrato nella figura seguente. I singoli passaggi del processo sono illustrati sotto la figura.

> **Note**
>
> L'immagine e il processo seguenti presuppongono l'uso dello standard RSA per la verifica del server e lo scambio delle chiavi. Il processo è leggermente diverso quando si usa il protocollo Diffie-Hellman anziché RSA.



1. Il client invia una messaggio di saluto al server.
2. Il server risponde con un messaggio di saluto e invia il certificato del server.
3. Il client effettua le azioni seguenti:
   a. Verifica che il certificato del SSL/TLS server sia firmato da un certificato radice considerato attendibile dal client.
   b. Estrae la chiave pubblica dal certificato del server.
   c. Genera un segreto pre-master e lo crittografa con la chiave pubblica del server.
   d. Invia il segreto pre-master crittografato al server.
4. Per decrittografare il segreto pre-master del client, il server lo invia all'HSM. L'HSM utilizza la chiave privata dell'HSM per decrittografare il segreto pre-master e quindi invia il segreto pre-master al server. Indipendentemente, il client e il server utilizzano ciascuno il segreto premaster e alcune informazioni contenute nei messaggi di benvenuto per calcolare un segreto principale.
5. Il processo di handshake termina. Per il resto della sessione, tutti i messaggi inviati tra il client e il server vengono crittografati con derivati del segreto master.

Per informazioni su come configurare SSL/TLS offload con AWS CloudHSM, consulta uno dei seguenti argomenti:

- AWS CloudHSM SSL/TLS offload su Linux usando NGINX o Apache con OpenSSL
- AWS CloudHSM Offload SSL/TLS su Linux utilizzando Tomcat con JSSE
- AWS CloudHSM Offload SSL/TLS su Windows tramite IIS con KSP

AWS CloudHSM SSL/TLS offload su Linux usando NGINX o Apache con OpenSSL

Questo argomento fornisce istruzioni dettagliate per configurare l' SSL/TLS offload with su un server web Linux. AWS CloudHSM

Argomenti

- Panoramica di
- Fase 1: configurazione dei prerequisiti
- Fase 2: Generare la chiave privata e il certificato SSL/TLS
- Fase 3: configurazione del server Web
- Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Panoramica di

In Linux il software per server Web NGINX e Apache HTTP Server si integra con OpenSSL per supportare HTTPS. Il motore dinamico AWS CloudHSM per OpenSSL offre un'interfaccia che consente al software del server Web di utilizzare i moduli HSM nel cluster per l'offload e lo storage delle chiavi di crittografia. Il motore OpenSSL connette il server Web al cluster AWS CloudHSM .

Per completare questo tutorial, è necessario prima scegliere se utilizzare il software del server Web NGINX o Apache su Linux. Vengono quindi illustrate le seguenti operazioni:

- Installa il software del server Web in un'istanza Amazon EC2.
- Configura il software del server Web in modo tale che supporti HTTPS con una chiave privata archiviata nel cluster AWS CloudHSM .
- (Facoltativo) Utilizza Amazon EC2 per creare una seconda istanza del server Web ed Elastic Load Balancing per creare un sistema di bilanciamento del carico. L'uso di un sistema di bilanciamento del carico può migliorare le prestazioni grazie alla distribuzione del carico in più server. Offre anche ridondanza e una disponibilità più elevata in caso di errore di uno o più server.

Quando sei pronto per iniziare, vai a Fase 1: configurazione dei prerequisiti.

Fase 1: configurazione dei prerequisiti

Piattaforme diverse richiedono prerequisiti diversi. Utilizza la sezione sui prerequisiti riportata di seguito corrispondente alla tua piattaforma.

Prerequisiti per Client SDK 5

Per configurare l' SSL/TLS offload del server Web con Client SDK 5, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno due moduli di sicurezza hardware (HSM)

> **Note**
>
> È possibile utilizzare un singolo cluster HSM, ma bisogna prima disabilitare la durabilità delle chiavi del client. Per ulteriori informazioni, consulta la pagina sulla gestione delle impostazioni di durabilità delle chiavi del client e la pagina sullo strumento di configurazione di Client SDK 5.

- Un'istanza Amazon EC2 che esegue un sistema operativo Linux con il seguente software installato:
  - Un server Web (NGINX o Apache)
  - OpenSSL Dynamic Engine per Client SDK 5
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata del server Web sull'HSM.

Per configurare un'istanza del server Web Linux e creare un CU sull'HSM

1. Installa e configura OpenSSL Dynamic Engine per. AWS CloudHSM Per ulteriori informazioni sull'installazione di OpenSSL Dynamic Engine, consulta la pagina OpenSSL Dynamic Engine per Client SDK 5.
2. Su un'istanza Linux EC2 che ha accesso al tuo cluster, installa il server Web NGINX o Apache:

   Amazon Linux 2

   - Per le informazioni su come scaricare l'ultima versione di NGINX su Amazon Linux 2, consulta il sito Web di NGINX.

     L'ultima versione di NGINX disponibile per Amazon Linux 2 utilizza una versione di OpenSSL più recente rispetto alla versione di sistema di OpenSSL. Dopo aver installato

NGINX, è necessario creare un collegamento simbolico dalla libreria AWS CloudHSM OpenSSL Dynamic Engine alla posizione prevista da questa versione di OpenSSL

```
$ sudo ln -sf /opt/cloudhsm/lib/libcloudhsm_openssl_engine.so /usr/lib64/
engines-1.1/cloudhsm.so
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

La versione più recente di Apache per Amazon Linux 2 si collega a una versione di OpenSSL più recente di quella installata sul tuo sistema. È necessario creare un collegamento simbolico dalla libreria AWS CloudHSM OpenSSL Dynamic Engine al percorso in cui la versione più recente di OpenSSL lo cerca:

```
$ sudo ln -sf /opt/cloudhsm/lib/libcloudhsm_openssl_engine.so /usr/lib64/
engines-1.1/cloudhsm.so
```

Amazon Linux 2023

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Red Hat 8

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Red Hat 9

- NGINX

```
$ sudo yum install nginx
```

- Apache

```
$ sudo yum install httpd mod_ssl
```

Ubuntu 22.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 24.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 26.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

3. Usa CloudHSM CLI per creare un utente crittografico. ??? Per ulteriori informazioni sulla gestione degli utenti HSM, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM.

> **Tip**
>
> Prendere nota del nome utente e della password del CU, perché saranno necessari più avanti per creare o importare il certificato e la chiava privata HTTPS per il server Web.

Dopo aver completato queste operazioni, andare su Fase 2: Generare la chiave privata e il certificato SSL/TLS.

Note

- Per utilizzare Security-Enhanced Linux (SELinux) e i server web, è necessario consentire le connessioni TCP in uscita sulla porta 2223, che è la porta utilizzata da Client SDK 5 per comunicare con l'HSM.
- Per creare e attivare un cluster e consentire a un'istanza EC2 di accedervi, completa la procedura descritta nella pagina Nozioni di base su AWS CloudHSM. La guida offre istruzioni dettagliate per la creazione di un cluster attivo con un HSM e un'istanza client Amazon EC2. È possibile utilizzare questa istanza client come server Web.
- Per evitare di disabilitare la durabilità delle chiavi del client, aggiungi più di un HSM al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
- È possibile utilizzare SSH o PuTTY per connettersi all'istanza del client. Per ulteriori informazioni, consulta le pagine Connessione all'istanza Linux tramite SSH o Connessione all'istanza Linux da Windows tramite PuTTY nella documentazione Amazon EC2.

Fase 2: Generare la chiave privata e il certificato SSL/TLS

Per abilitare HTTPS, l'applicazione del server web (NGINX o Apache) necessita di una chiave privata e di un certificato corrispondente. SSL/TLS Per utilizzare l' SSL/TLS offload del server Web con AWS CloudHSM, è necessario archiviare la chiave privata in un HSM del cluster. AWS CloudHSM Dovrai prima generare una chiave privata e utilizzarla per creare una richiesta di firma del certificato (CSR).

Quindi si esporta una chiave privata PEM falsa dall'HSM, che è un file di chiave privata in formato PEM che contiene un riferimento alla chiave privata memorizzata nell'HSM (non è la chiave privata effettiva). Il server web utilizza il file di chiave privata PEM falso per identificare la chiave privata sull'HSM durante l'offload. SSL/TLS

Generazione di una chiave privata e di un certificato

Generazione di una chiave privata

Questa sezione mostra come generare una coppia di chiavi utilizzando l'interfaccia a riga di comando di CloudHSM. ??? Una volta generata una coppia di chiavi all'interno dell'HSM, è possibile esportarla come file PEM falso e generare il certificato corrispondente.

Installa e configura l'interfaccia a riga di comando di CloudHSM

1. Installa e configura l'interfaccia a riga di comando di CloudHSM.
2. Usa il seguente comando per avviare la CLI di CloudHSM.

   ```
   $ /opt/cloudhsm/bin/cloudhsm-cli interactive
   ```

3. Eseguire il comando seguente per accedere all'HSM. Sostituiscilo *<user name>* con il nome utente del tuo cripto-utente

   ```
   Command: login --username <user name> --role crypto-user
   ```

Generazione di una chiave privata

A seconda del caso d'uso, è possibile generare una coppia di chiavi RSA o EC. Esegui una delle seguenti operazioni:

- Come generare una chiave privata RSA su un HSM

  Utilizza il comando `key generate-asymmetric-pair rsa` per generare una coppia di chiavi RSA. Questo esempio genera una coppia di chiavi RSA con un modulo di 2048, un esponente pubblico di 65537, un'etichetta di chiave pubblica di e un'etichetta di chiave privata di. *`tls_rsa_pub`* *`tls_rsa_private`*

  ```
  aws-cloudhsm > key generate-asymmetric-pair rsa \
  --public-exponent 65537 \
  --modulus-size-bits 2048 \
  ```

```
--public-label tls_rsa_pub \
--private-label tls_rsa_private \
--private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_pub",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
```

```
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
73a80fdb457aa7b20cd61e486c326e2cfd5e124a7f6a996437437812b542e3caf85928aa866f0298580f7967ee6aa
f6e6296d6c116d5744c6d60d14d3bf3cb978fe6b75ac67b7089bafd50d8687213b31abc7dc1bad422780d29c851d5
133022653225bd129f8491101725e9ea33e1ded83fb57af35f847e532eb30cd7e726f23910d2671c6364092e83469
ac3160f0ca9725d38318b7",
        "modulus-size-bits": 2048
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280cc7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_private",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
```

```
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
        "modulus-size-bits": 2048
      }
    }
  }
}
```

- Come generare una chiave privata EC su un HSM

  Utilizza il comando [key generate-asymmetric-pair ec] per generare una coppia di chiavi EC. Questo esempio genera una coppia di chiavi EC con la `prime256v1` curva (corrispondente alla `NID_X9_62_prime256v1` curva), un'etichetta di chiave pubblica di e un'etichetta di *`tls_ec_pub`* chiave privata di. *`tls_ec_private`*

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve prime256v1 \
    --public-label tls_ec_pub \
    --private-label tls_ec_private \
    --private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000012000b",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ec_pub",
        "id": "",
        "check-value": "0xd7c1a7",
```

```
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 57,
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
        "curve": "secp224r1"
      }
    },
"private_key": {
      "key-reference": "0x000000000012000c",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ec_private",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "private-key",
        "encrypt": false,
```

```
        "decrypt": false,
        "token": false,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 122,
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
        "curve": "secp224r1"
      }
    }
  }
}
```


Esportazione di un file di chiave privata PEM falso

Una volta che disponi di una chiave privata nell'HSM, devi esportare un file di chiave privata PEM falso. Questo file non contiene i dati della chiave effettivi, ma consente a OpenSSL Dynamic Engine di identificare la chiave privata nell'HSM, che potrà quindi essere utilizzata per creare una richiesta di firma del certificato (CSR) e firmare la CSR per creare il certificato.

Utilizzate il `key generate-file` comando per esportare la chiave privata in un falso formato PEM e salvarla in un file. Sostituire i valori seguenti con i propri valori.

- *<private_key_label>*— Etichetta della chiave privata generata nel passaggio precedente.
- *<web_server_fake_pem.key>*— Nome del file in cui verrà scritta la tua chiave PEM falsa.

```
aws-cloudhsm > key generate-file --encoding reference-pem --
path <web_server_fake_pem.key> --filter attr.label=<private_key_label>
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Uscire dalla CLI di CloudHSM

Esegui il comando seguente per interrompere la CLI di CloudHSM.

```
aws-cloudhsm > quit
```

Ora dovresti avere un nuovo file sul tuo sistema, situato *<web_server_fake_pem.key>* nel percorso specificato dal comando precedente. Questo file è il file della chiave privata PEM falsa.

Generazione di un certificato auto-firmato

Dopo aver generato una chiave privata PEM falsa, puoi utilizzare questo file per generare una richiesta di firma del certificato (CSR) e un certificato.

In un ambiente di produzione, per creare un certificato da una CSR in genere ci si avvale di un'autorità di certificazione, che non è invece necessaria per un ambiente di test. Se utilizzate una CA, inviate loro il file CSR e utilizzate il SSL/TLS certificato firmato che vi forniscono nel vostro server web per HTTPS.

In alternativa all'utilizzo di una CA, puoi utilizzare l' AWS CloudHSM OpenSSL Dynamic Engine per creare un certificato autofirmato. Self-signed i certificati non sono considerati attendibili dai browser e non devono essere utilizzati negli ambienti di produzione. ma solo negli ambienti di test.

> **Warning**
>
> Self-signed i certificati devono essere utilizzati solo in un ambiente di test. Per un ambiente di produzione, è consigliabile utilizzare un metodo più sicuro, ad esempio un'autorità di certificazione per creare un certificato.

Installa e configura l'OpenSSL Dynamic Engine

1. Effettuare la connessione all'istanza del client.
2. the section called "Installa"

Generazione di un certificato

1. Ottieni una copia del file PEM falso generato in un passaggio precedente.
2. Crea un CSR

   Esegui il comando seguente per utilizzare AWS CloudHSM OpenSSL Dynamic Engine per creare una richiesta di firma del certificato (CSR). Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM. Sostituisci *<web_server.csr>* con il nome del file che contiene la tua CSR.

   Il comando `req` è interattivo. Ogni campo deve essere compilato Le informazioni sul campo vengono copiate nel certificato. SSL/TLS

   ```
   $ openssl req -engine cloudhsm -new -key <web_server_fake_pem.key> -
   out <web_server.csr>
   ```
3. Crea un certificato auto-firmato

   Esegui il comando seguente per utilizzare l' AWS CloudHSM OpenSSL Dynamic Engine per firmare il tuo CSR con la tua chiave privata sul tuo HSM. In questo modo viene creato un certificato autofirmato. Sostituire i valori seguenti nel comando con i propri valori.

   - *<web_server.csr>*— Nome del file che contiene il CSR.
   - *<web_server_fake_pem.key>*— Nome del file che contiene la falsa chiave privata PEM.
   - *<web_server.crt>*— Nome del file che conterrà il certificato del server web.

   ```
   $ openssl x509 -engine cloudhsm -req -days 365 -in <web_server.csr> -
   signkey <web_server_fake_pem.key> -out <web_server.crt>
   ```

Dopo aver completato queste operazioni, andare su Fase 3: configurazione del server Web.

Fase 3: configurazione del server Web

È possibile aggiornare la configurazione del software del server Web per utilizzare il certificato HTTPS e la chiave privata PEM fittizia corrispondente creata nella fase precedente. Ricorda di eseguire il backup dei certificati e delle chiavi esistenti prima di iniziare. Questo completerà la configurazione del software del server web Linux per l' SSL/TLS offload con. AWS CloudHSM

Completa la procedura delineata in una delle seguenti sezioni.

Argomenti

- Configurazione del server Web NGINX
- Configurazione del server Web Apache

Configurazione del server Web NGINX

Fai riferimento a questa sezione per configurare NGINX sulle piattaforme supportate.

Per aggiornare la configurazione del server Web per NGINX

1. Effettuare la connessione all'istanza del client.
2. Eseguire il comando seguente per creare le directory necessarie per il certificato del server Web e la falsa chiave privata PEM.

```
$ sudo mkdir -p /etc/pki/nginx/private
```

3. Eseguire il comando seguente per copiare il certificato del server Web nella posizione richiesta. Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

```
$ sudo cp <web_server.crt> /etc/pki/nginx/server.crt
```

4. Eseguire il comando seguente per copiare la falsa chiave privata PEM nella posizione richiesta. Sostituiscilo *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

```
$ sudo cp <web_server_fake_pem.key> /etc/pki/nginx/private/server.key
```

5. Eseguire il comando seguente per modificare la proprietà dei file in modo che l'utente denominato nginx possa leggerli.

```
$ sudo chown nginx /etc/pki/nginx/server.crt /etc/pki/nginx/private/server.key
```

6. Eseguire il comando seguente per effettuare il backup del file `/etc/nginx/nginx.conf`.

```
$ sudo cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.backup
```

7. Aggiornamento della configurazione per NGINX.

> **Note**
>
> Ciascun cluster può supportare un massimo di 1000 processi di lavoro NGINX su tutti i server web NGINX.

Amazon Linux 2

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2;
```

```
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Amazon Linux 2023

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
```

```
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Red Hat 8

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
```

```
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Red Hat 9

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Ubuntu 22.04 LTS

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
```

```
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Ubuntu 24.04 LTS

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;
```

```
    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Ubuntu 26.04 LTS

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Per farlo sono necessarie le autorizzazioni root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
ssl_engine cloudhsm;
env CLOUDHSM_PIN;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    #ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
```

```
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA";
    ssl_prefer_server_ciphers on;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Salvare il file.

8. Eseguire il backup del file di configurazione `systemd`, quindi impostare il percorso `EnvironmentFile`.

   Amazon Linux 2

   1. Effettuare il backup del file `nginx.service`.

      ```
      $ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
      nginx.service.backup
      ```

   2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

      ```
      EnvironmentFile=/etc/sysconfig/nginx
      ```

Amazon Linux 2023

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Apri il `/lib/systemd/system/nginx.service` file in un editor di testo. Quindi, nella sezione [Servizio], aggiungi il seguente percorso:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Red Hat 8

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Red Hat 9

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 22.04 LTS

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 24.04 LTS

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 26.04 LTS

1. Effettuare il backup del file `nginx.service`.

```
$ sudo cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire il file `/lib/systemd/system/nginx.service` in un editor di testo, quindi nella sezione [Service], aggiungere il percorso seguente:

```
EnvironmentFile=/etc/sysconfig/nginx
```

9. Controllare se il file `/etc/sysconfig/nginx` esiste, quindi eseguire una delle operazioni seguenti:

   - Se il file esiste, effettuare il backup del file eseguendo il seguente comando:

     ```
     $ sudo cp /etc/sysconfig/nginx /etc/sysconfig/nginx.backup
     ```

   - In caso contrario, aprire un editor di testo, quindi creare un file denominato `nginx` nella cartella `/etc/sysconfig/`.

10. Configura l'ambiente NGINX.

    > **Note**
    >
    > Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU.

    Amazon Linux 2

    Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

    ```
    CLOUDHSM_PIN=<CU user name>:<password>
    ```

    Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

    Salvare il file.

    Amazon Linux 2023

    Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

    ```
    CLOUDHSM_PIN=<CU user name>:<password>
    ```

    Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

    Salvare il file.

Red Hat 8

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Salvare il file.

Red Hat 9

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Salvare il file.

Ubuntu 22.04 LTS

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Salvare il file.

Ubuntu 24.04 LTS

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Salvare il file.

Ubuntu 26.04 LTS

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Per farlo sono necessarie le autorizzazioni root di Linux. Aggiungi le credenziali del crypto user (CU):

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Salvare il file.

11. Avviare il server Web NGINX.

Amazon Linux 2

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

Amazon Linux 2023

Arresta tutti i processi NGINX

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia NGINX

```
$ sudo systemctl start nginx
```

Red Hat 8

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

Red Hat 9

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 22.04 LTS

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 24.04 LTS

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

Ubuntu 26.04 LTS

Interrompi qualsiasi processo NGINX in esecuzione

```
$ sudo systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ sudo systemctl daemon-reload
```

Avvia il processo NGINX

```
$ sudo systemctl start nginx
```

12. (Facoltativo) Configura la piattaforma per avviare NGINX all'avvio.

Amazon Linux 2

```
$ sudo systemctl enable nginx
```

Amazon Linux 2023

```
$ sudo systemctl enable nginx
```

Red Hat 8

```
$ sudo systemctl enable nginx
```

Red Hat 9

```
$ sudo systemctl enable nginx
```

Ubuntu 22.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 24.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 26.04 LTS

```
$ sudo systemctl enable nginx
```

Dopo avere aggiornato la configurazione del server Web, vai alla Fase 4: abilitazione del traffico HTTPS e verifica del certificato.

Configurazione del server Web Apache

Fai riferimento a questa sezione per configurare Apache sulle piattaforme supportate.

Per aggiornare la configurazione del server Web per Apache

1. Esegui la connessione all'istanza del client Amazon EC2.

2. Definisci le posizioni predefinite per i certificati e le chiavi private per la piattaforma.

   Amazon Linux 2

   Assicurati che nel file `/etc/httpd/conf.d/ssl.conf` siano presenti questi valori:

   ```
   SSLCertificateFile      /etc/pki/tls/certs/localhost.crt
   SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
   ```

   Amazon Linux 2023

   Nel `/etc/httpd/conf.d/ssl.conf` file, assicurati che esistano questi valori:

   ```
   SSLCertificateFile      /etc/pki/tls/certs/localhost.crt
   SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
   ```

   Red Hat 8

   Assicurati che nel file `/etc/httpd/conf.d/ssl.conf` siano presenti questi valori:

   ```
   SSLCertificateFile      /etc/pki/tls/certs/localhost.crt
   SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
   ```

   Red Hat 9

   Assicurati che nel file `/etc/httpd/conf.d/ssl.conf` siano presenti questi valori:

   ```
   SSLCertificateFile      /etc/pki/tls/certs/localhost.crt
   SSLCertificateKeyFile   /etc/pki/tls/private/localhost.key
   ```

   Ubuntu 22.04 LTS

   Assicurati che nel file `/etc/apache2/sites-available/default-ssl.conf` siano presenti questi valori:

   ```
   SSLCertificateFile      /etc/ssl/certs/localhost.crt
   SSLCertificateKeyFile   /etc/ssl/private/localhost.key
   ```

Ubuntu 24.04 LTS

Assicurati che nel file `/etc/apache2/sites-available/default-ssl.conf` siano presenti questi valori:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

Ubuntu 26.04 LTS

Assicurati che nel file `/etc/apache2/sites-available/default-ssl.conf` siano presenti questi valori:

```
SSLCertificateFile      /etc/ssl/certs/localhost.crt
SSLCertificateKeyFile   /etc/ssl/private/localhost.key
```

3. Copia il certificato del server Web nella posizione richiesta per la piattaforma.

Amazon Linux 2

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Amazon Linux 2023

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Red Hat 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Red Hat 9

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Ubuntu 22.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Ubuntu 24.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

Ubuntu 26.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

4. Copia la tua chiave privata PEM falsa nella posizione richiesta per la piattaforma.

   Amazon Linux 2

   ```
   $ sudo cp <web_server_fake_pem.key> /etc/pki/tls/private/localhost.key
   ```

   Sostituiscilo *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

   Amazon Linux 2023

   ```
   $ sudo cp <web_server_fake_pem.key> /etc/pki/tls/private/localhost.key
   ```

   Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

   Red Hat 8

   ```
   $ sudo cp <web_server_fake_pem.key> /etc/pki/tls/private/localhost.key
   ```

Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

Red Hat 9

```
$ sudo cp <web_server_fake_pem.key> /etc/pki/tls/private/localhost.key
```

Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

Ubuntu 22.04 LTS

```
$ sudo cp <web_server_fake_pem.key> /etc/ssl/private/localhost.key
```

Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

Ubuntu 24.04 LTS

```
$ sudo cp <web_server_fake_pem.key> /etc/ssl/private/localhost.key
```

Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

Ubuntu 26.04 LTS

```
$ sudo cp <web_server_fake_pem.key> /etc/ssl/private/localhost.key
```

Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

5. Cambia la proprietà di questi file se richiesto dalla piattaforma.

   Amazon Linux 2

   ```
   $ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/
   localhost.key
   ```

   Fornisce il permesso di lettura all'utente denominato apache.

Amazon Linux 2023

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/
localhost.key
```

Fornisce il permesso di lettura all'utente denominato apache.

Red Hat 8

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/
localhost.key
```

Fornisce il permesso di lettura all'utente denominato apache.

Red Hat 9

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/
localhost.key
```

Fornisce il permesso di lettura all'utente denominato apache.

Ubuntu 22.04 LTS

Nessuna operazione necessaria.

Ubuntu 24.04 LTS

Nessuna operazione necessaria.

Ubuntu 26.04 LTS

Nessuna operazione necessaria.

6. Configura le direttive Apache per la piattaforma.

Amazon Linux 2

Individua il file SSL per questa piattaforma:

```
/etc/httpd/conf.d/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salvare il file.

Amazon Linux 2023

Individua il file SSL per questa piattaforma:

```
/etc/httpd/conf.d/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

Salvare il file.

Red Hat 8

Individua il file SSL per questa piattaforma:

```
/etc/httpd/conf.d/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLProtocol TLSv1.2 TLSv1.3
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProxyCipherSuite HIGH:!aNULL
```

Salvare il file.

Red Hat 9

Individua il file SSL per questa piattaforma:

```
/etc/httpd/conf.d/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLProtocol TLSv1.2 TLSv1.3
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
```

```
SSLProxyCipherSuite HIGH:!aNULL
```

Salvare il file.

Ubuntu 22.04 LTS

Individua il file SSL per questa piattaforma:

```
/etc/apache2/mods-available/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3
```

Salvare il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 24.04 LTS

Individua il file SSL per questa piattaforma:

```
/etc/apache2/mods-available/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3
```

Salvare il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 26.04 LTS

Individua il file SSL per questa piattaforma:

```
/etc/apache2/mods-available/ssl.conf
```

Questo file contiene le direttive Apache che definiscono la modalità di esecuzione del server. Le direttive vengono visualizzate a sinistra, seguite da un valore. Utilizza un editor di testo per modificare il file. Per farlo sono necessarie le autorizzazioni root di Linux.

Aggiorna o inserisci le seguenti direttive con questi valori:

```
SSLCryptoDevice cloudhsm
SSLCipherSuite ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-
RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES128-SHA:DHE-RSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:ECDHE-ECDSA-AES256-GCM-
```

```
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-SHA
SSLProtocol TLSv1.2 TLSv1.3
```

Salvare il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

7. Configura un file environment-values per la piattaforma.

Amazon Linux 2

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Amazon Linux 2023

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Red Hat 8

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Red Hat 9

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Ubuntu 22.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Ubuntu 24.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Ubuntu 26.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

8. Imposta una variabile di ambiente contenente le credenziali del crypto user (CU) nel file in cui vengono archiviate le variabili di ambiente per la piattaforma:

Amazon Linux 2

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Amazon Linux 2023

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

Red Hat 8

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU.

Red Hat 9

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU.

Ubuntu 22.04 LTS

Utilizza un editor di testo per modificare `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU. In Client SDK 3 le credenziali del CU sono archiviate nella variabile di ambiente `n3fips_password`. Client SDK 5 supporta entrambe le variabili di ambiente, ma si consiglia di utilizzare `CLOUDHSM_PIN`.

Ubuntu 24.04 LTS

Utilizza un editor di testo per modificare `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU. In Client SDK 3 le credenziali del CU sono archiviate nella variabile di ambiente `n3fips_password`. Client SDK 5 supporta entrambe le variabili di ambiente, ma si consiglia di utilizzare `CLOUDHSM_PIN`.

Ubuntu 26.04 LTS

Utilizza un editor di testo per modificare `/etc/apache2/envvars`.

```
export CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU. In Client SDK 3 le credenziali del CU sono archiviate nella

variabile di ambiente `n3fips_password`. Client SDK 5 supporta entrambe le variabili di ambiente, ma si consiglia di utilizzare `CLOUDHSM_PIN`.

9. Avviare il server Web Apache.

   Amazon Linux 2

   ```
   $ sudo systemctl daemon-reload
   $ sudo service httpd start
   ```

   Amazon Linux 2023

   ```
   $ sudo systemctl daemon-reload
   $ sudo service httpd start
   ```

   Red Hat 8

   ```
   $ sudo systemctl daemon-reload
   $ sudo service httpd start
   ```

   Red Hat 9

   ```
   $ sudo systemctl daemon-reload
   $ sudo service httpd start
   ```

   Ubuntu 22.04 LTS

   ```
   $ sudo service apache2 start
   ```

   Ubuntu 24.04 LTS

   ```
   $ sudo service apache2 start
   ```

   Ubuntu 26.04 LTS

   ```
   $ sudo service apache2 start
   ```

10. (Facoltativo) Configura la tua piattaforma per avviare Apache all'avvio.

Amazon Linux 2

```
$ sudo chkconfig httpd on
```

Amazon Linux 2023

```
$ sudo chkconfig httpd on
```

Red Hat 8

```
$ systemctl enable httpd
```

Red Hat 9

```
$ systemctl enable httpd
```

Ubuntu 22.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 24.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 26.04 LTS

```
$ sudo systemctl enable apache2
```

Dopo avere aggiornato la configurazione del server Web, vai alla Fase 4: abilitazione del traffico HTTPS e verifica del certificato.

Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Dopo aver configurato il server Web per l' SSL/TLS offload con AWS CloudHSM, aggiungete l'istanza del server Web a un gruppo di sicurezza che consenta il traffico HTTPS in entrata. Ciò consente ai client, come i browser Web, di stabilire una connessione HTTPS con il server Web. Quindi crea

una connessione HTTPS al tuo server web e verifica che stia utilizzando il certificato con cui hai configurato l' SSL/TLS offload. AWS CloudHSM

Argomenti

- Abilitazione delle connessioni HTTPS in entrata
- Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Abilitazione delle connessioni HTTPS in entrata

Per connetterti al server Web da un client (ad esempio un browser Web), crea un gruppo di sicurezza che consenta le connessioni HTTPS in entrata. Nello specifico, deve consentire le connessioni TCP in entrata sulla porta 443. Assegna questo gruppo di sicurezza al tuo server Web.

Per creare un gruppo di sicurezza per HTTPS e assegnarlo al server Web

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Seleziona Gruppi di sicurezza nel riquadro di navigazione.
3. Scegliere Create Security Group (Crea gruppo di sicurezza).
4. Per Create Security Group (Crea un gruppo di sicurezza), procedere come segue:
    a. Per Security group name (Nome del gruppo di sicurezza), digitare un nome per il gruppo di sicurezza che si sta creando.
    b. (Facoltativo) Digitare una descrizione del gruppo di sicurezza in fase di creazione.
    c. Per VPC, scegli il VPC contenente l'istanza Amazon EC2 del server Web.
    d. Seleziona Aggiungi regola.
    e. Per Tipo, seleziona HTTPS dalla finestra a discesa.
    f. Per Origine, inserisci una posizione di origine.
    g. Scegliere Create Security Group (Crea gruppo di sicurezza).
5. Nel riquadro di navigazione, scegliere Instances (Istanze).
6. Seleziona la casella di controllo accanto all'istanza del server Web.
7. Seleziona il menu a discesa Operazioni nella parte superiore della pagina. Seleziona Sicurezza, quindi Modifica gruppi di sicurezza.
8. Per Gruppi di sicurezza associati, seleziona la casella di ricerca e scegli il gruppo di sicurezza creato per HTTPS. Quindi, scegli Aggiungi i gruppi di sicurezza.
9. Seleziona Salva.

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo aver aggiunto il server web a un gruppo di sicurezza, puoi verificare che SSL/TLS offload stia utilizzando il tuo certificato autofirmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l' SSL/TLS offload con un browser web

1. Utilizza un browser Web per connetterti al server Web utilizzando il nome DNS pubblico o l'indirizzo IP del server. Assicurati che l'URL nella barra degli indirizzi inizi con https://. Ad esempio,**`https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`**.

   > **Tip**
   >
   > Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempiohttps://www.example.com/) al tuo server web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Utilizza il browser Web per visualizzare il certificato del server Web. Per ulteriori informazioni, consulta gli argomenti seguenti:

   - Per Mozilla Firefox, consultare Visualizzare un certificato sul sito Web di supporto di Mozilla.
   - Per Google Chrome, consulta la pagina Understand Security Issues sul sito Web di Google per sviluppatori.

   Altri browser Web potrebbero avere caratteristiche simili da utilizzare per visualizzare il certificato del server Web.

3. Assicurati che il SSL/TLS certificato sia quello per cui hai configurato il tuo server web.

Per verificare l' SSL/TLS offload con OpenSSL s_client

1. Esegui il seguente comando OpenSSL per connetterti al server Web tramite HTTPS. Sostituiscilo *`<server name>`* con il nome DNS pubblico o l'indirizzo IP del tuo server web.

```
openssl s_client -connect <server name>:443
```

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempiohttps://www.example.com/) al tuo server web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Assicurati che il SSL/TLS certificato sia quello per cui hai configurato il tuo server web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata per il server Web è archiviata in un HSM nel AWS CloudHSM cluster.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale).

AWS CloudHSM SSL/TLS offload su Linux utilizzando NGINX o HAProxy con OpenSSL Provider

Questo argomento fornisce istruzioni dettagliate per configurare l'offload dell'identità del server AWS CloudHSM su un SSL/TLS server Web Linux utilizzando NGINX o HAProxy con il provider OpenSSL.

Argomenti

- Panoramica di
- Fase 1: configurazione dei prerequisiti
- Fase 2: Generare o importare una chiave privata e ottenere un certificato
- Fase 3: configurazione del server Web
- Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Panoramica di

Su Linux, il software del server web NGINX e HAProxy si integra con OpenSSL per supportare HTTPS. https://www.openssl.org/ Il provider AWS CloudHSM OpenSSL fornisce un'interfaccia che consente al software del server Web di utilizzare gli HSM del cluster per l'offload crittografico e l'archiviazione delle chiavi. L'OpenSSL Provider è il bridge che collega il server web al cluster. AWS CloudHSM

Per completare questo tutorial, configurerai NGINX o HAProxy per utilizzare il provider OpenSSL. AWS CloudHSM Il tutorial mostra come eseguire le seguenti operazioni:

- Installa il software del server Web in un'istanza Amazon EC2.
- Configura il software del server Web in modo tale che supporti HTTPS con una chiave privata archiviata nel cluster AWS CloudHSM .
- (Facoltativo) Utilizza Amazon EC2 per creare una seconda istanza del server Web ed Elastic Load Balancing per creare un sistema di bilanciamento del carico. L'uso di un sistema di bilanciamento del carico può migliorare le prestazioni grazie alla distribuzione del carico in più server. Offre anche ridondanza e una disponibilità più elevata in caso di errore di uno o più server.

Quando sei pronto per iniziare, vai a Fase 1: configurazione dei prerequisiti.

Fase 1: configurazione dei prerequisiti

Piattaforme diverse richiedono prerequisiti diversi. Utilizza la sezione sui prerequisiti riportata di seguito corrispondente alla tua piattaforma.

Prerequisiti per AWS CloudHSM Provider OpenSSL

Per configurare l'offload dell'identità del SSL/TLS server Web con AWS CloudHSM OpenSSL Provider for Client SDK 5, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno due moduli di sicurezza hardware (HSM)

> **Note**
>
> È possibile utilizzare un singolo cluster HSM, ma bisogna prima disabilitare la durabilità delle chiavi del client. Per ulteriori informazioni, consulta la pagina sulla gestione delle impostazioni di durabilità delle chiavi del client e la pagina sullo strumento di configurazione di Client SDK 5.

- Un'istanza Amazon EC2 che esegue un sistema operativo Linux con il seguente software installato:
  - Un server web (NGINX o HAProxy)
  - Il provider AWS CloudHSM OpenSSL per Client SDK 5
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata del server Web sull'HSM.

Per configurare un'istanza del server Web Linux e creare un CU sull'HSM

> **Note**
>
> Molti dei comandi di questa procedura richiedono privilegi elevati. Potrebbe essere necessario eseguire comandi con `sudo` o come utente root a seconda della configurazione del sistema.

1. Installa e configura AWS CloudHSM OpenSSL Provider for Client SDK 5. Per ulteriori informazioni sull'installazione del provider OpenSSL, vedere AWS CloudHSM OpenSSL Provider for Client SDK 5.
2. Su un'istanza Linux EC2 che ha accesso al tuo cluster, installa il server web NGINX o HAProxy:

   Amazon Linux 2023

   - NGINX

     ```
     $ yum install nginx
     ```

   - HAProxy

     ```
     $ yum install haproxy
     ```

   RHEL 9 (9.2+)

   - NGINX

     ```
     $ yum install nginx
     ```

   - HAProxy

     ```
     $ yum install haproxy
     ```

   RHEL 10 (10.0+)

   - NGINX

```
$ yum install nginx
```

- HAProxy

```
$ yum install haproxy
```

Ubuntu 26.04 LTS

- NGINX

```
$ apt install nginx
```

- HAProxy

```
$ apt install haproxy
```

3. Usa CloudHSM CLI per creare un utente crittografico. ??? Per ulteriori informazioni sulla gestione degli utenti HSM, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM.

> **Tip**
>
> Prendere nota del nome utente e della password del CU, perché saranno necessari più avanti per creare o importare il certificato e la chiava privata HTTPS per il server Web.

Dopo aver completato queste operazioni, andare su Fase 2: Generare o importare una chiave privata e ottenere un certificato.

Note

- Per utilizzare Security-Enhanced Linux (SELinux) e i server web, è necessario consentire le connessioni TCP in uscita sulla porta 2223, che è la porta utilizzata da Client SDK 5 per comunicare con l'HSM.
- Per creare e attivare un cluster e consentire a un'istanza EC2 di accedervi, completa la procedura descritta nella pagina Nozioni di base su AWS CloudHSM. La guida offre istruzioni dettagliate per la creazione di un cluster attivo con un HSM e un'istanza client Amazon EC2. È possibile utilizzare questa istanza client come server Web.

- Per evitare di disabilitare la durabilità delle chiavi del client, aggiungi più di un HSM al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
- È possibile utilizzare SSH o PuTTY per connettersi all'istanza del client. Per ulteriori informazioni, consulta le pagine Connessione all'istanza Linux tramite SSH o Connessione all'istanza Linux da Windows tramite PuTTY nella documentazione Amazon EC2.

Fase 2: Generare o importare una chiave privata e ottenere un certificato

Per abilitare HTTPS, l'applicazione del server web (NGINX o HAProxy) necessita di una chiave privata e di un certificato corrispondente. SSL/TLS Per utilizzare l'offload dell'identità SSL/TLS del server Web con AWS CloudHSM, è necessario archiviare la chiave privata in un HSM nel cluster. AWS CloudHSM Dovrai prima generare una chiave privata e utilizzarla per creare una richiesta di firma del certificato (CSR). Quindi si esporta una chiave privata PEM falsa dall'HSM, che è un file di chiave privata in formato PEM che contiene un riferimento alla chiave privata memorizzata nell'HSM (non è la chiave privata effettiva). Il server web utilizza il file di chiave privata PEM falso per identificare la chiave privata sull'HSM durante l'offload dell'identità del server. SSL/TLS

Generazione di una chiave privata

Questa sezione mostra come generare una coppia di chiavi utilizzando l'interfaccia a riga di comando di CloudHSM. ??? Una volta generata una coppia di chiavi all'interno dell'HSM, puoi esportarla come falso file PEM e generare il certificato corrispondente.

Installa e configura l'interfaccia a riga di comando di CloudHSM

1. Installa e configura l'interfaccia a riga di comando di CloudHSM.
2. Usa il seguente comando per avviare la CLI di CloudHSM.

```
$ /opt/cloudhsm/bin/cloudhsm-cli interactive
```

3. Eseguire il comando seguente per accedere all'HSM. Sostituiscilo *<user name>* con il nome utente del tuo cripto-utente

```
aws-cloudhsm>login --username <user name> --role crypto-user
```

Generazione di una chiave privata

A seconda del caso d'uso, è possibile generare una coppia di chiavi RSA o EC. Puoi anche generare un Ed25519 o una coppia di ML-DSA chiavi da utilizzare con TLS 1.3. Esegui una delle seguenti operazioni:

- Come generare una chiave privata RSA su un HSM

  Utilizza il comando key generate-asymmetric-pair rsa per generare una coppia di chiavi RSA. Questo esempio genera una coppia di chiavi RSA con un modulo di 2048, un esponente pubblico di 65537, un'etichetta di chiave pubblica di e un'etichetta di chiave privata di. *tls_rsa_pub* *tls_rsa_private*

```
aws-cloudhsm > key generate-asymmetric-pair rsa \
--public-exponent 65537 \
--modulus-size-bits 2048 \
--public-label tls_rsa_pub \
--private-label tls_rsa_private \
--private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cc8",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_pub",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "public-key",
        "encrypt": true,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
```

```
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 512,
        "public-exponent": "0x010001",
        "modulus":
"0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
        "modulus-size-bits": 2048
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280cc7",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "tls_rsa_private",
        "id": "",
        "check-value": "0x01fe6e",
        "class": "private-key",
        "encrypt": false,
        "decrypt": true,
        "token": true,
        "always-sensitive": true,
        "derive": false,
```

```
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 1217,
        "public-exponent": "0x010001",
        "modulus":
 "0xb1d27e857a876f4e9fd5de748a763c539b359f937eb4b4260e30d1435485a732c878cdad9c72538e2215351b1
        "modulus-size-bits": 2048
      }
    }
  }
}
```

- Come generare una chiave privata EC su un HSM

  Utilizza il comando [key generate-asymmetric-pair ec](#) per generare una coppia di chiavi EC. Questo esempio genera una coppia di chiavi EC con la `prime256v1` curva (corrispondente alla `NID_X9_62_prime256v1` curva), un'etichetta di chiave pubblica di e un'etichetta di *`tls_ec_pub`* chiave privata di. *`tls_ec_private`*

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve prime256v1 \
    --public-label tls_ec_pub \
    --private-label tls_ec_private \
    --private-attributes sign=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x000000000012000b",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ec_pub",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 57,
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
        "curve": "secp224r1"
      }
    },
"private_key": {
      "key-reference": "0x000000000012000c",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
```

```
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ec_private",
        "id": "",
        "check-value": "0xd7c1a7",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 122,
        "ec-point":
 "0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f3
        "curve": "secp224r1"
      }
    }
  }
}
```

- Per generare una chiave privata Ed25519 su un HSM

> **Disponibilità di Ed25519**
>
> Ed25519 è disponibile solo su cluster non FIPS. Ed25519 supporta solo TLS 1.3.

Usa il key generate-asymmetric-pair ec comando per generare una coppia di chiavi Ed25519. Questo esempio genera una coppia di chiavi Ed25519 con un'etichetta di chiave pubblica *tls_ed25519_pub* e un'etichetta di chiave privata di. *tls_ed25519_private*

```
aws-cloudhsm > key generate-asymmetric-pair ec \
    --curve ed25519 \
    --public-label tls_ed25519_pub \
    --private-label tls_ed25519_private \
    --private-attributes sign=true \
    --public-attributes verify=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000383931",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ed25519_pub",
        "id": "0x",
        "check-value": "0x4cb1b1",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
```

```
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 32,
        "ec-point":
"0x302a300506032b6570032100c31f8a3695f571dda1658bc98e0591ab088c1a657b3199dd50c3956046ebbe75"
        "curve": "ed25519"
      }
    },
    "private_key": {
      "key-reference": "0x0000000000382e60",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ec",
        "label": "tls_ed25519_private",
        "id": "0x",
        "check-value": "0x4cb1b1",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
```

```
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 48,
        "ec-point":
 "0x302a300506032b6570032100c31f8a3695f571dda1658bc98e0591ab088c1a657b3199dd50c3956046ebbe75"
        "curve": "ed25519"
      }
    }
  }
}
```


- Per generare una chiave ML-DSA privata su un HSM

> **ML-DSA disponibilità e requisiti**
>
> ML-DSA richiede OpenSSL 3.5 o successivo per la creazione di CSR, la creazione di certificati e l'offload TLS. ML-DSA supporta solo TLS 1.3.
> Per l'offload ML-DSA TLS sulle piattaforme Amazon Linux 2023 e RHEL, potrebbe essere necessario abilitare la sub-policy crittografica post-quantum (PQ). Per ulteriori informazioni, consulta the section called “Problema: l'handshake ML-DSA TLS non riesce su Amazon Linux 2023 e RHEL senza «algoritmi di firma condivisi»”.

Usa il comando per generare una coppia di `key generate-asymmetric-pair ml-dsa` chiavi. ML-DSA Questo esempio genera una coppia di ML-DSA-44 chiavi con un'etichetta di chiave pubblica *`tls_mldsa44_pub`* e un'etichetta di chiave privata di*`tls_mldsa44_private`*. Puoi anche usare `ML-DSA-65` or `ML-DSA-87` per livelli di sicurezza più elevati.

```
aws-cloudhsm > key generate-asymmetric-pair ml-dsa \
```

```
    --mldsa-algorithm ML-DSA-44 \
    --public-label tls_mldsa44_pub \
    --private-label tls_mldsa44_private \
    --private-attributes sign=true \
    --public-attributes verify=true
{
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x0000000000280cd2",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "tls_mldsa44_pub",
        "id": "",
        "check-value": "0xd4e5f6",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": false,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": false,
        "sign": false,
        "trusted": false,
        "unwrap": false,
        "verify": true,
        "wrap": false,
        "wrap-with-trusted": false,
```

```
        "key-length-bytes": 1312,
        "mldsa-algorithm": "ML-DSA-44"
      }
    },
    "private_key": {
      "key-reference": "0x0000000000280cd3",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "full"
      },
      "attributes": {
        "key-type": "ml-dsa",
        "label": "tls_mldsa44_private",
        "id": "",
        "check-value": "0xd4e5f6",
        "class": "private-key",
        "encrypt": false,
        "decrypt": false,
        "token": true,
        "always-sensitive": true,
        "derive": false,
        "destroyable": true,
        "extractable": true,
        "local": true,
        "modifiable": true,
        "never-extractable": false,
        "private": true,
        "sensitive": true,
        "sign": true,
        "trusted": false,
        "unwrap": false,
        "verify": false,
        "wrap": false,
        "wrap-with-trusted": false,
        "key-length-bytes": 2560,
        "mldsa-algorithm": "ML-DSA-44"
      }
    }
```

```
  }
}
```

Esportazione di un file di chiave privata PEM falso

Una volta che disponi di una chiave privata nell'HSM, devi esportare un file di chiave privata PEM falso. Questo file non contiene i dati chiave effettivi, ma consente al provider OpenSSL di identificare la chiave privata sull'HSM. È quindi possibile utilizzare la chiave privata per creare una richiesta di firma del certificato (CSR) e firmare la CSR per creare il certificato.

Usa il `key generate-file` comando per esportare la chiave privata in un falso formato PEM e salvarla in un file. Sostituire i valori seguenti con i propri valori.

- *<private_key_label>*— Etichetta della chiave privata generata nel passaggio precedente.
- *<web_server_fake_pem.key>*— Nome del file in cui verrà scritta la tua chiave PEM falsa.

Per chiavi RSA ed EC

Usa la `reference-pem` codifica:

```
aws-cloudhsm > key generate-file --encoding reference-pem --
path <web_server_fake_pem.key> --filter attr.label=<private_key_label>
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
}
```

Per Ed25519 e chiavi ML-DSA

Usa la codifica: `cloudhsm-reference-pem`

```
aws-cloudhsm > key generate-file --encoding cloudhsm-reference-pem --
path <web_server_fake_pem.key> --filter attr.label=<private_key_label>
{
  "error_code": 0,
  "data": {
    "message": "Successfully generated key file"
  }
```

```
}
```

Uscire dalla CLI di CloudHSM

Esegui il comando seguente per interrompere la CLI di CloudHSM.

```
aws-cloudhsm > quit
```

Ora dovresti avere un nuovo file sul tuo sistema, situato *<web_server_fake_pem.key>* nel percorso specificato dal comando precedente. Questo file è il file della chiave privata PEM falsa.

Generazione di un certificato auto-firmato

Dopo aver generato una chiave privata PEM falsa, puoi utilizzare questo file per generare una richiesta di firma del certificato (CSR) e un certificato.

In un ambiente di produzione, per creare un certificato da una CSR in genere ci si avvale di un'autorità di certificazione, che non è invece necessaria per un ambiente di test. Se utilizzate una CA, inviate loro il file CSR e utilizzate il SSL/TLS certificato firmato che vi forniscono nel vostro server web per HTTPS.

In alternativa all'utilizzo di una CA, puoi utilizzare il provider AWS CloudHSM OpenSSL per creare un certificato autofirmato. Self-signed i certificati non sono considerati attendibili dai browser e non devono essere utilizzati negli ambienti di produzione. ma solo negli ambienti di test.

> **Warning**
>
> Self-signed i certificati devono essere utilizzati solo in un ambiente di test. Per un ambiente di produzione, è consigliabile utilizzare un metodo più sicuro, ad esempio un'autorità di certificazione per creare un certificato.

Installa e configura il provider OpenSSL

1. Effettuare la connessione all'istanza del client.
2. the section called “Installa”

Generazione di un certificato

1. Ottieni una copia del file PEM falso generato in un passaggio precedente.

2. Crea un CSR

   Esegui il comando seguente per utilizzare il provider AWS CloudHSM OpenSSL per creare una richiesta di firma del certificato (CSR). Sostituisci *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM. Sostituisci *<web_server.csr>* con il nome del file che contiene la tua CSR.

   Il comando `req` è interattivo. Ogni campo deve essere compilato Le informazioni sul campo vengono copiate nel certificato. SSL/TLS

   > **ML-DSA Requisito della versione di OpenSSL**
   >
   > ML-DSA richiede OpenSSL 3.5 o successivo. Sulle piattaforme con un sistema OpenSSL precedente, è necessario utilizzare un binario OpenSSL 3.5+ personalizzato.

   > **Algoritmo Digest richiesto per le chiavi RSA ed EC**
   >
   > Per le chiavi RSA ed EC, è necessario specificare un algoritmo di digest (ad esempio,). `-sha256` Ed25519 e non ML-DSA richiedono un flag digest.

   ```
   $ openssl req -provider cloudhsm -provider default -new -sha256 -
   key <web_server_fake_pem.key> -out <web_server.csr>
   ```

3. Crea un certificato auto-firmato

   Esegui il seguente comando per utilizzare il provider AWS CloudHSM OpenSSL per firmare il tuo CSR con la tua chiave privata sul tuo HSM. In questo modo viene creato un certificato autofirmato. Sostituire i valori seguenti nel comando con i propri valori.

   - *<web_server.csr>*— Nome del file che contiene il CSR.
   - *<web_server_fake_pem.key>*— Nome del file che contiene la falsa chiave privata PEM.
   - *<web_server.crt>*— Nome del file che conterrà il certificato del server web.

   ```
   $ openssl x509 -provider cloudhsm -provider default -req -sha256 -days 365 -
   in <web_server.csr> -signkey <web_server_fake_pem.key> -out <web_server.crt>
   ```

Dopo aver ottenuto la chiave privata e il certificato, vai aFase 3: configurazione del server Web.

Fase 3: configurazione del server Web

È possibile aggiornare la configurazione del software del server Web per utilizzare il certificato HTTPS e la chiave privata PEM fittizia corrispondente creata nella fase precedente. Ricorda di eseguire il backup dei certificati e delle chiavi esistenti prima di iniziare. Questo completerà la configurazione del software del server Web Linux per l'offload dell'identità SSL/TLS del server con. AWS CloudHSM

Completa la procedura delineata in una delle seguenti sezioni.

Argomenti

- Configurazione del server Web NGINX
- Configura il server web HAProxy

Configurazione del server Web NGINX

Usa questa sezione per configurare NGINX con il provider OpenSSL.

Per configurare NGINX per OpenSSL Provider

1. Effettuare la connessione all'istanza del client.
2. Eseguire il comando seguente per creare le directory necessarie per il certificato del server Web e la falsa chiave privata PEM.

```
$ mkdir -p /etc/pki/nginx/private
```

3. Eseguire il comando seguente per copiare il certificato del server Web nella posizione richiesta. Sostituiscilo *<web_server.crt>* con il nome del certificato del tuo server web.

```
$ cp <web_server.crt> /etc/pki/nginx/server.crt
```

4. Eseguire il comando seguente per copiare la falsa chiave privata PEM nella posizione richiesta. Sostituiscilo *<web_server_fake_pem.key>* con il nome del file che contiene la tua falsa chiave privata PEM.

```
$ cp <web_server_fake_pem.key> /etc/pki/nginx/private/server.key
```

5. Eseguire il comando seguente per modificare la proprietà dei file in modo che l'utente denominato nginx possa leggerli.

```
$ chown nginx /etc/pki/nginx/server.crt /etc/pki/nginx/private/server.key
```

6. Configura OpenSSL per utilizzare il provider. AWS CloudHSM Per ulteriori informazioni sulla configurazione del provider OpenSSL, vedere AWS CloudHSM OpenSSL Provider for Client SDK 5.

   1. Individua il tuo file di configurazione OpenSSL:

      ```
      $ openssl version -d
      ```

      Dovresti vedere un risultato simile a:

      ```
      OPENSSLDIR: "/etc/pki/tls"
      ```

      Il file di configurazione si trova `openssl.cnf` in questa directory.

   2.
      > **Note**
      >
      > Non modificate direttamente il file openssl.cnf predefinito del sistema. Ciò impedisce che le operazioni OpenSSL a livello di sistema (SSH, connessioni TLS e altri servizi) vengano instradate involontariamente attraverso il provider CloudHSM.
      > L'utilizzo di un file di configurazione separato consente di limitare l'utilizzo del provider CloudHSM solo ad applicazioni specifiche che richiedono operazioni di crittografia. HSM-backed

      Crea un nuovo file di configurazione OpenSSL con i seguenti contenuti:

      ```
      $ cat > <example-cloudhsm-openssl.cnf> << 'EOF'
      ## NOTE: This should point to the system default openssl config file.
      # Replace /etc/pki/tls with the path to your OpenSSL configuration directory
      .include </etc/pki/tls>/openssl.cnf

      # Override the existing provider_section to include AWS CloudHSM OpenSSL
       Provider as a 3rd party OpenSSL provider
      [provider_sect]
      default = default_sect
      ```

```
# Include AWS CloudHSM CloudHSM OpenSSL provider
cloudhsm = cloudhsm_sect

[default_sect]
activate = 1

[cloudhsm_sect]
activate = 1
EOF
```

3. Assicurati che la variabile di `CLOUDHSM_PIN` ambiente sia impostata con le tue credenziali di utente crittografico (CU):

```
$ export CLOUDHSM_PIN=<username>:<password>
```

4. Imposta la variabile di `OPENSSL_CONF` ambiente in modo che punti al file di configurazione aggiornato e verifica che il provider sia caricato:

```
$ OPENSSL_CONF=/path/to/example-cloudhsm-openssl.cnf openssl list -providers
```

Dovresti vedere elencati sia il provider predefinito che il provider CloudHSM:

```
OPENSSL_CONF=/path/to/example-cloudhsm-openssl.cnf openssl list -providers
Providers:
  default
    name: OpenSSL Default Provider
    version: 3.2.2
    status: active
  cloudhsm
    name: AWS CloudHSM OpenSSL Provider
    version: 5.17.0
    status: active
```

7. Eseguire il comando seguente per effettuare il backup del file `/etc/nginx/nginx.conf`.

```
$ cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.backup
```

8. Aggiornamento della configurazione per NGINX.

> **Note**
>
> Ciascun cluster può supportare un massimo di 1000 processi di lavoro NGINX su tutti i server web NGINX.

Amazon Linux 2023

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Ciò potrebbe richiedere i permessi di root di Linux. All'inizio del file, aggiungi le seguenti righe:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
 ciphers
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    # ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-
RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-
ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-
POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;
```

```
    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

RHEL 9 (9.2+)

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Ciò potrebbe richiedere i permessi di root per Linux. All'inizio del file, aggiungi le seguenti righe:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;

    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
 ciphers
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    # ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
```

```
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-
RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-
ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-
POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

RHEL 10 (10.0+)

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Ciò potrebbe richiedere i permessi di root per Linux. All'inizio del file, aggiungi le seguenti righe:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /usr/share/nginx/html;
```

```
    ssl_certificate "/etc/pki/nginx/server.crt";
    ssl_certificate_key "/etc/pki/nginx/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
 ciphers
    # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048
    # ssl_dhparam "/etc/pki/nginx/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-
RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-
ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-
POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Ubuntu 26.04 LTS

Usare un editor di testo per modificare il file `/etc/nginx/nginx.conf`. Ciò potrebbe richiedere i permessi di root per Linux. All'inizio del file, aggiungi le seguenti righe:

```
env CLOUDHSM_PIN;
env OPENSSL_CONF;
```

Quindi aggiungi quanto segue alla sezione TLS del file:

```
# Settings for a TLS enabled server.
server {
    listen       443 ssl http2 default_server;
    listen       [::]:443 ssl http2 default_server;
    server_name  _;
    root         /var/www/html;

    ssl_certificate "/etc/ssl/certs/server.crt";
    ssl_certificate_key "/etc/ssl/private/server.key";
    # It is *strongly* recommended to generate unique DH parameters for DHE
 ciphers
    # Generate them with: openssl dhparam -out /etc/ssl/certs/dhparams.pem 2048
    # ssl_dhparam "/etc/ssl/certs/dhparams.pem";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-
RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-
RSA-AES256-SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-
POLY1305:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-
ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-CHACHA20-
POLY1305:TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256";
    ssl_prefer_server_ciphers off;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
    }

    error_page 404 /404.html;
    location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }
}
```

Salvare il file.

9. Eseguire il backup del file di configurazione `systemd`, quindi impostare il percorso `EnvironmentFile`.

   Amazon Linux 2023

   1. Effettua il backup del `nginx.service` file:

      ```
      $ cp /lib/systemd/system/nginx.service /lib/systemd/system/
      nginx.service.backup
      ```

   2. Aprire `/lib/systemd/system/nginx.service` in un editor di testo. Nella sezione [Servizio], aggiungi:

      ```
      EnvironmentFile=/etc/sysconfig/nginx
      ```

   RHEL 9 (9.2+)

   1. Esegui il backup del `nginx.service` file:

      ```
      $ cp /lib/systemd/system/nginx.service /lib/systemd/system/
      nginx.service.backup
      ```

   2. Aprire `/lib/systemd/system/nginx.service` in un editor di testo. Nella sezione [Servizio], aggiungi:

      ```
      EnvironmentFile=/etc/sysconfig/nginx
      ```

   RHEL 10 (10.0+)

   1. Esegui il backup del `nginx.service` file:

      ```
      $ cp /lib/systemd/system/nginx.service /lib/systemd/system/
      nginx.service.backup
      ```

   2. Aprire `/lib/systemd/system/nginx.service` in un editor di testo. Nella sezione [Servizio], aggiungi:

```
EnvironmentFile=/etc/sysconfig/nginx
```

Ubuntu 26.04 LTS

1. Esegui il backup del `nginx.service` file:

```
$ cp /lib/systemd/system/nginx.service /lib/systemd/system/
nginx.service.backup
```

2. Aprire `/lib/systemd/system/nginx.service` in un editor di testo. Nella sezione [Servizio], aggiungi:

```
EnvironmentFile=/etc/sysconfig/nginx
```

10. Controllare se il file `/etc/sysconfig/nginx` esiste, quindi eseguire una delle operazioni seguenti:

- Se il file esiste, effettuare il backup del file eseguendo il seguente comando:

```
$ cp /etc/sysconfig/nginx /etc/sysconfig/nginx.backup
```

- In caso contrario, aprire un editor di testo, quindi creare un file denominato `nginx` nella cartella `/etc/sysconfig/`.

11. Configura l'ambiente NGINX.

Amazon Linux 2023

Come utente root di Linux, apri `/etc/sysconfig/nginx` il file in un editor di testo. Ad esempio,

```
vi /etc/sysconfig/nginx
```

Aggiungi le credenziali Cryptography User (CU) e il percorso del tuo file di configurazione OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU. Sostituisci *<path to example-cloudhsm-openssl.cnf>* con il percorso completo del file di configurazione in cui hai creato. Per configurare NGINX per OpenSSL Provider

Salvare il file.

RHEL 9 (9.2+)

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Ciò potrebbe richiedere i permessi di root di Linux. Aggiungi le credenziali Cryptography User (CU) e il percorso del tuo file di configurazione OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU. Sostituisci *<path to example-cloudhsm-openssl.cnf>* con il percorso completo del file di configurazione in cui hai creato. Per configurare NGINX per OpenSSL Provider

Salvare il file.

RHEL 10 (10.0+)

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Ciò potrebbe richiedere i permessi di root di Linux. Aggiungi le credenziali Cryptography User (CU) e il percorso del tuo file di configurazione OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU. Sostituisci *<path to example-cloudhsm-openssl.cnf>* con il percorso completo del file di configurazione in cui hai creato. Per configurare NGINX per OpenSSL Provider

Salvare il file.

Ubuntu 26.04 LTS

Apri il file `/etc/sysconfig/nginx` in un editor di testo. Ciò potrebbe richiedere i permessi di root di Linux. Aggiungi le credenziali Cryptography User (CU) e il percorso del tuo file di configurazione OpenSSL:

```
CLOUDHSM_PIN=<CU user name>:<password>
OPENSSL_CONF=<path to example-cloudhsm-openssl.cnf>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali CU. Sostituisci *<path to example-cloudhsm-openssl.cnf>* con il percorso completo del file di configurazione in cui hai creato. Per configurare NGINX per OpenSSL Provider

Salvare il file.

12. Avviare il server Web NGINX.

Amazon Linux 2023

Arresta tutti i processi NGINX

```
$ systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ systemctl daemon-reload
```

Avvia NGINX

```
$ systemctl start nginx
```

RHEL 9 (9.2+)

Interrompi qualsiasi processo NGINX in esecuzione

```
$ systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ systemctl daemon-reload
```

Avvia il processo NGINX

```
$ systemctl start nginx
```

RHEL 10 (10.0+)

Interrompi qualsiasi processo NGINX in esecuzione

```
$ systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ systemctl daemon-reload
```

Avvia il processo NGINX

```
$ systemctl start nginx
```

Ubuntu 26.04 LTS

Interrompi qualsiasi processo NGINX in esecuzione

```
$ systemctl stop nginx
```

Ricarica la configurazione `systemd` per implementare le modifiche più recenti

```
$ systemctl daemon-reload
```

Avvia il processo NGINX

```
$ systemctl start nginx
```

Dopo aver configurato NGINX, vai a. Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Configura il server web HAProxy

Usa questa sezione per configurare HAProxy con il provider OpenSSL. Gli esempi seguenti mostrano come configurare HAProxy con i certificati e le chiavi CloudHSM.

Per configurare HAProxy per OpenSSL Provider

1. Esegui il backup del file di certificato combinato esistente, se esiste:

```
$ cp server-combined.pem server-combined.pem.backup
```

2. Crea un file di certificato combinato per HAProxy utilizzando il certificato e la chiave PEM falsa di CloudHSM:

```
$ cat server.crt server.key > server-combined.pem
```

3. Esegui il backup della configurazione HAProxy esistente:

```
$ cp /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg.backup
```

4. Crea una nuova configurazione di offload TLS di CloudHSM all'indirizzo: `/etc/haproxy/haproxy.cfg`

```
global
    daemon
    ssl-provider cloudhsm
    # It is *strongly* recommended to generate unique DH parameters
    # Generate them with: openssl dhparam -out /etc/haproxy/dhparams.pem 2048
    # ssl-dh-param-file /etc/haproxy/dhparams.pem
    ssl-default-bind-ciphers ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-
GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-
AES256-SHA384:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-
SHA256:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES256-
GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-CHACHA20-POLY1305
    ssl-default-bind-ciphersuites
 TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
    ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    mode http
    timeout connect 5000ms
    timeout client 50000ms
    timeout server 50000ms

frontend haproxy_frontend
    bind *:443 ssl crt /path/to/server-combined.pem
    default_backend web_servers

backend web_servers
```

```
    server web1 127.0.0.1:8080 check
```

Aggiorna il percorso del certificato in modo che corrisponda alla posizione del file.

5. Configura systemd per utilizzare un file di ambiente per HAProxy. La posizione dipende dalla tua distribuzione Linux.

   Amazon Linux and RHEL

   Eseguire il backup e modificare il file di servizio HAProxy:

   ```
   $ cp /lib/systemd/system/haproxy.service /lib/systemd/system/
   haproxy.service.backup
   ```

   Modifica `/lib/systemd/system/haproxy.service` e aggiungi la riga seguente nella sezione [Servizio]:

   ```
   EnvironmentFile=/etc/sysconfig/haproxy
   ```

   Ubuntu

   Eseguire il backup e modificare il file di servizio HAProxy:

   ```
   $ cp /lib/systemd/system/haproxy.service /lib/systemd/system/
   haproxy.service.backup
   ```

   Modifica `/lib/systemd/system/haproxy.service` e aggiungi la riga seguente nella sezione [Servizio]:

   ```
   EnvironmentFile=/etc/default/haproxy
   ```

6. Crea il file di ambiente nella posizione appropriata per il tuo sistema.

   Amazon Linux and RHEL

   Eseguite il backup del file di ambiente HAProxy, se presente:

   ```
   $ cp /etc/sysconfig/haproxy /etc/sysconfig/haproxy.backup
   ```

   Crea il file di ambiente HAProxy `/etc/sysconfig/haproxy` con i seguenti contenuti:

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Ubuntu

Effettuare il backup del file di ambiente HAProxy, se presente:

```
$ cp /etc/default/haproxy /etc/default/haproxy.backup
```

Crea il file di ambiente HAProxy `/etc/default/haproxy` con i seguenti contenuti:

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le tue credenziali CU.

7. Ricarica la configurazione di systemd:

```
$ systemctl daemon-reload
```

8. Avvia HAProxy con la configurazione di offload TLS di CloudHSM:

```
$ systemctl start haproxy
```

Puoi anche eseguire HAProxy direttamente con un file di configurazione personalizzato:

```
$ haproxy -f /path/to/haproxy-cloudhsm.cfg
```

Dopo aver configurato HAProxy, vai a. Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Dopo aver configurato il server Web per l' SSL/TLS offload con AWS CloudHSM, aggiungete l'istanza del server Web a un gruppo di sicurezza che consenta il traffico HTTPS in entrata. Ciò consente ai client, come i browser Web, di stabilire una connessione HTTPS con il server Web. Quindi crea una connessione HTTPS al tuo server web e verifica che stia utilizzando il certificato con cui hai configurato l' SSL/TLS offload. AWS CloudHSM

Argomenti

- Abilitazione delle connessioni HTTPS in entrata
- Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Abilitazione delle connessioni HTTPS in entrata

Per connetterti al server Web da un client (ad esempio un browser Web), crea un gruppo di sicurezza che consenta le connessioni HTTPS in entrata. Nello specifico, deve consentire le connessioni TCP in entrata sulla porta 443. Assegna questo gruppo di sicurezza al tuo server Web.

Per creare un gruppo di sicurezza per HTTPS e assegnarlo al server Web

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Seleziona Gruppi di sicurezza nel riquadro di navigazione.
3. Scegliere Create Security Group (Crea gruppo di sicurezza).
4. Per Create Security Group (Crea un gruppo di sicurezza), procedere come segue:
    a. Per Security group name (Nome del gruppo di sicurezza), digitare un nome per il gruppo di sicurezza che si sta creando.
    b. (Facoltativo) Digitare una descrizione del gruppo di sicurezza in fase di creazione.
    c. Per VPC, scegli il VPC contenente l'istanza Amazon EC2 del server Web.
    d. Seleziona Aggiungi regola.
    e. Per Tipo, seleziona HTTPS dalla finestra a discesa.
    f. Per Origine, inserisci una posizione di origine.
    g. Scegliere Create Security Group (Crea gruppo di sicurezza).
5. Nel riquadro di navigazione, scegliere Instances (Istanze).
6. Seleziona la casella di controllo accanto all'istanza del server Web.
7. Seleziona il menu a discesa Operazioni nella parte superiore della pagina. Seleziona Sicurezza, quindi Modifica gruppi di sicurezza.
8. Per Gruppi di sicurezza associati, seleziona la casella di ricerca e scegli il gruppo di sicurezza creato per HTTPS. Quindi, scegli Aggiungi i gruppi di sicurezza.
9. Seleziona Salva.

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo aver aggiunto il server web a un gruppo di sicurezza, puoi verificare che SSL/TLS offload stia utilizzando il tuo certificato autofirmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l' SSL/TLS offload con un browser web

1. Utilizza un browser Web per connetterti al server Web utilizzando il nome DNS pubblico o l'indirizzo IP del server. Assicurati che l'URL nella barra degli indirizzi inizi con https://. Ad esempio,**`https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`**.

   > **Tip**
   >
   > Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempiohttps://www.example.com/) al tuo server web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Utilizza il browser Web per visualizzare il certificato del server Web. Per ulteriori informazioni, consulta gli argomenti seguenti:

   - Per Mozilla Firefox, consultare Visualizzare un certificato sul sito Web di supporto di Mozilla.
   - Per Google Chrome, consulta la pagina Understand Security Issues sul sito Web di Google per sviluppatori.

   Altri browser Web potrebbero avere caratteristiche simili da utilizzare per visualizzare il certificato del server Web.

3. Assicurati che il SSL/TLS certificato sia quello per cui hai configurato il tuo server web.

Per verificare l' SSL/TLS offload con OpenSSL s_client

1. Esegui il seguente comando OpenSSL per connetterti al server Web tramite HTTPS. Sostituiscilo *`<server name>`* con il nome DNS pubblico o l'indirizzo IP del tuo server web.

   ```
   openssl s_client -connect <server name>:443
   ```

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempiohttps://www.example.com/) al tuo server web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Assicurati che il SSL/TLS certificato sia quello per cui hai configurato il tuo server web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata per il server Web è archiviata in un HSM nel AWS CloudHSM cluster.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale).

AWS CloudHSM Offload SSL/TLS su Linux utilizzando Tomcat con JSSE

Questo argomento fornisce step-by-step istruzioni per configurare l' SSL/TLS offload utilizzando Java Secure Socket Extension (JSSE) con JCE SDK. AWS CloudHSM

Argomenti

- Panoramica di
- Fase 1: configurazione dei prerequisiti
- Passaggio 2: generare o importare una chiave privata e un certificato SSL/TLS
- Fase 3: configurazione del server Web Tomcat
- Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Panoramica di

In AWS CloudHSM, i server web Tomcat funzionano su Linux per supportare HTTPS. L'SDK AWS CloudHSM JCE fornisce un'interfaccia che può essere utilizzata con JSSE (Java Secure Socket Extension) per consentire l'uso di HSMs tali server Web. AWS CloudHSM JCE è il bridge che collega JSSE al tuo cluster AWS CloudHSM. JSSE è un'API Java per i protocolli Secure Sockets Layer (SSL) e Transport Layer Security (TLS).

Fase 1: configurazione dei prerequisiti

Segui questi prerequisiti per utilizzare un server Web Tomcat con AWS CloudHSM offload su Linux. SSL/TLS Questi prerequisiti devono essere soddisfatti per configurare l' SSL/TLS offload del server Web con Client SDK 5 e un server Web Tomcat.

> **Note**
>
> Piattaforme diverse richiedono prerequisiti diversi. Segui sempre la procedura di installazione corretta per la tua piattaforma.

Prerequisiti

- Un'istanza Amazon EC2 che esegue un sistema operativo Linux su cui è installato un server Web Tomcat.
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata del server Web sull'HSM.
- Un AWS CloudHSM cluster attivo con almeno due moduli di sicurezza hardware (HSMs) su cui è installato e configurato JCE for Client SDK 5.

> **Note**
>
> È possibile utilizzare un singolo cluster HSM, ma bisogna prima disabilitare la durabilità delle chiavi del client. Per ulteriori informazioni, consulta la sezione sulla gestione delle impostazioni di durabilità delle chiavi del client e la pagina sullo strumento di configurazione di Client SDK 5.

Come soddisfare i prerequisiti

1. Installa e configura JCE for AWS CloudHSM su un AWS CloudHSM cluster attivo con almeno due moduli di sicurezza hardware (). HSMs Per ulteriori informazioni sull'installazione, consulta la pagina su JCE per Client SDK 5.
2. Su un'istanza Linux EC2 che ha accesso al tuo AWS CloudHSM cluster, segui le istruzioni di Apache Tomcat per scaricare e installare il server web Tomcat.

3. Utilizza la CLI di CloudHSM per creare un crypto user (CU). Per ulteriori informazioni sulla gestione degli utenti HSM, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM.

   > **Tip**
   >
   > Prendere nota del nome utente e della password del CU, perché saranno necessari più avanti per creare o importare il certificato e la chiava privata HTTPS per il server Web.

4. Per configurare JCE con Java KeyTool, segui le istruzioni riportate nella pagina Usa Client SDK 5 per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner.

Dopo aver completato queste operazioni, andare su Passaggio 2: generare o importare una chiave privata e un certificato SSL/TLS.

Note

- Per utilizzare Security-Enhanced Linux (SELinux) e i server web, devi consentire le connessioni TCP in uscita sulla porta 2223, che è la porta utilizzata da Client SDK 5 per comunicare con l'HSM.
- Per creare e attivare un cluster e consentire a un'istanza EC2 di accedervi, completa la procedura descritta nella pagina Nozioni di base su AWS CloudHSM. Questa sezione offre step-by-step istruzioni per creare un cluster attivo con un HSM e un'istanza client Amazon EC2. È possibile utilizzare questa istanza client come server Web.
- Per evitare di disabilitare la durabilità delle chiavi del client, aggiungi più di un HSM al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
- È possibile utilizzare SSH o PuTTY per connettersi all'istanza del client. Per ulteriori informazioni, consulta le pagine Connessione all'istanza Linux tramite SSH o Connessione all'istanza Linux da Windows tramite PuTTY nella documentazione Amazon EC2.

Passaggio 2: generare o importare una chiave privata e un certificato SSL/TLS

Per abilitare HTTPS, l'applicazione del server Web Tomcat richiede una chiave privata e un certificato corrispondente. SSL/TLS Per utilizzare l' SSL/TLS offload del server Web con AWS CloudHSM, è necessario archiviare la chiave privata in un HSM del cluster. AWS CloudHSM

> **Note**
>
> In mancanza di una chiave privata e di un certificato corrispondente, genera una chiave privata in un HSM, La chiave privata viene utilizzata per creare una richiesta di firma del certificato (CSR), che viene utilizzata per creare il certificato. SSL/TLS

Si crea un AWS CloudHSM KeyStore file locale che contiene un riferimento alla chiave privata sull'HSM e al certificato associato. Il server Web utilizza il AWS CloudHSM KeyStore file per identificare la chiave privata sull'HSM durante SSL/TLS l'offload.

Argomenti

- Generazione di una chiave privata
- Generazione di un certificato auto-firmato

Generazione di una chiave privata

Questa sezione mostra come generare una key pair usando il file KeyTool da JDK. Una volta generata una key pair all'interno dell'HSM, è possibile esportarla come KeyStore file e generare il certificato corrispondente.

A seconda del caso d'uso, è possibile generare una coppia di chiavi RSA o EC. La procedura riportata di seguito illustra come generare una coppia di chiavi RSA.

Usa il **`genkeypair`** comando in KeyTool per generare una coppia di key pair RSA

1. Dopo aver sostituito i *<VARIABLES>* seguenti dati con i tuoi dati specifici, usa il seguente comando per generare un file keystore denominato`jsse_keystore.keystore`, che conterrà un riferimento alla tua chiave privata sull'HSM.

```
$ keytool -genkeypair -alias <UNIQUE ALIAS FOR KEYS> -keyalg <KEY ALGORITHM> -
keysize <KEY SIZE> -sigalg <SIGN ALGORITHM> \
        -keystore <PATH>/<JSSE KEYSTORE NAME>.keystore -storetype CLOUDHSM \
        -dname CERT_DOMAIN_NAME \
        -J-classpath '-J'$JAVA_LIB'/*:/opt/cloudhsm/java/*:./*' \
        -provider "com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider" \
        -providerpath "$CLOUDHSM_JCE_LOCATION" \
        -keypass <KEY PASSWORD> -storepass <KEYSTORE PASSWORD>
```

- ***<PATH>***: Il percorso in cui desiderate generare il file keystore.
- ***<UNIQUE ALIAS FOR KEYS>***: Viene utilizzato per identificare in modo univoco la chiave sull'HSM. Questo alias verrà impostato come attributo ETICHETTA della chiave.
- ***<KEY PASSWORD>***: Memorizziamo il riferimento alla tua chiave nel file keystore locale e questa password protegge quel riferimento locale.
- ***<KEYSTORE PASSWORD>***: Questa è la password per il file keystore locale.
- ***<JSSE KEYSTORE NAME>***: nome del file Keystore.
- ***<CERT DOMAIN NAME>***: X.500 Nome distinto.
- ***<KEY ALGORITHM>***: algoritmo chiave per generare una coppia di chiavi (ad esempio, RSA ed EC).
- ***<KEY SIZE>***: dimensione della chiave per generare una coppia di chiavi (ad esempio, 2048, 3072 e 4096).
- ***<SIGN ALGORITHM>***: dimensione della chiave per generare una coppia di chiavi (ad esempio, SHA1with RSA, SHA224with RSA, SHA256with RSA e SHA384with RSA). SHA512with

2. Per assicurarti che il comando sia stato eseguito correttamente, inserisci il seguente comando e verifica di aver generato correttamente una coppia di chiavi RSA.

```
$ ls <PATH>/<JSSE KEYSTORE NAME>.keystore
```

Generazione di un certificato auto-firmato

Dopo aver generato una chiave privata insieme al file KeyStore, puoi utilizzare questo file per generare una richiesta di firma del certificato (CSR) e un certificato.

In un ambiente di produzione, per creare un certificato da una CSR in genere ci si avvale di un'autorità di certificazione, che non è invece necessaria per un ambiente di test. Se utilizzi una CA, invia loro il file CSR e utilizza il SSL/TLS certificato firmato che ti forniscono nel tuo server web per HTTPS.

In alternativa all'utilizzo di una CA, puoi utilizzare il KeyTool per creare un certificato autofirmato. I certificati autofirmati non sono considerati attendibili dai browser e non devono essere utilizzati negli ambienti di produzione, ma solo negli ambienti di test.

> **Warning**
>
> È consigliabile utilizzare i certificati autofirmati solo in un ambiente di test. Per un ambiente di produzione, è consigliabile utilizzare un metodo più sicuro, ad esempio un'autorità di certificazione, per creare un certificato.

Generazione di un certificato

1. Ottieni una copia del file KeyStore generato in un passaggio precedente.
2. Esegui il comando seguente per utilizzare per KeyTool creare una richiesta di firma del certificato (CSR).

```
$ keytool -certreq -keyalg RSA -alias unique_alias_for_key -file certreq.csr \
        -keystore <JSSE KEYSTORE NAME>.keystore -storetype CLOUDHSM \
        -J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \
        -keypass <KEY PASSWORD> -storepass <KEYSTORE PASSWORD>
```

> **Note**
>
> Il file di output della richiesta di firma del certificato è `certreq.csr`.

Firma di un certificato

- Dopo aver sostituito i *<VARIABLES>* seguenti dati con i tuoi dati specifici, esegui il comando seguente per firmare la CSR con la tua chiave privata sull'HSM. In questo modo viene creato un certificato autofirmato.

```
$ keytool -gencert -infile certreq.csr -outfile certificate.crt \
    -alias <UNIQUE ALIAS FOR KEYS> -keypass <KEY_PASSWORD> -
storepass <KEYSTORE_PASSWORD> -sigalg SIG_ALG \
    -storetype CLOUDHSM -J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \
    -keystore jsse_keystore.keystore
```

> **Note**
>
> `certificate.crt` è il certificato firmato che utilizza la chiave privata dell'alias.

Importazione di un certificato in KeyStore

- Dopo aver sostituito i *<VARIABLES>* seguenti dati con i tuoi dati specifici, esegui il comando seguente per importare un certificato firmato come certificato affidabile. Questo passaggio memorizzerà il certificato nella voce del keystore identificata da alias.

```
$ keytool -import -alias <UNIQUE ALIAS FOR KEYS> -keystore jsse_keystore.keystore \
    -file certificate.crt -storetype CLOUDHSM \
    -v -J-classpath '-J$JAVA_LIB/*:/opt/cloudhsm/java/*:./*' \
    -keypass <KEY PASSWORD> -storepass <KEYSTORE_PASSWORD>
```

Conversione di un certificato in un file PEM

- Esegui il seguente comando per convertire il file del certificato firmato (`.crt`) in un PEM. Il file PEM verrà utilizzato per inviare la richiesta dal client http.

```
$ openssl x509 -inform der -in certificate.crt -out certificate.pem
```

Dopo aver completato questa procedura, vai alla Fase 3: configurazione del server Web.

Fase 3: configurazione del server Web Tomcat

È possibile aggiornare la configurazione del software del server Web per utilizzare il certificato HTTPS e il file PEM corrispondente creato nella fase precedente. Ricorda di eseguire il backup dei certificati e delle chiavi esistenti prima di iniziare. Questo completerà la configurazione del software del server web Linux con cui eseguire l' SSL/TLS offload. AWS CloudHSM Per ulteriori informazioni, consulta la documentazione di riferimento relativa alla configurazione di Apache Tomcat 9.

Arresta il server

- Dopo aver sostituito quanto *<VARIABLES>* segue con i tuoi dati specifici, esegui il seguente comando per arrestare Tomcat Server prima di aggiornare la configurazione

```
$ /<TOMCAT DIRECTORY>/bin/shutdown.sh
```

  - ***<TOMCAT DIRECTORY>***: La tua directory di installazione di Tomcat.

Aggiorna il percorso di classe Tomcat

1. Effettuare la connessione all'istanza del client.
2. Individua la cartella di installazione di Tomcat.
3. Dopo aver sostituito quanto *<VARIABLES>* segue con i tuoi dati specifici, usa il seguente comando per aggiungere la libreria Java e il percorso AWS CloudHSM Java in Tomcatclasspath, che si trova nel file.sh. Tomcat/bin/catalina

```
$ sed -i 's@CLASSPATH="$CLASSPATH""$CATALINA_HOME"\/bin\/
bootstrap.jar@CLASSPATH="$CLASSPATH""$CATALINA_HOME"\/bin\/bootstrap.jar:'"
        <JAVA LIBRARY>"'\/*:\/opt\/cloudhsm\/java\/*:.\/*@' <TOMCAT PATH> /bin/
catalina.sh
```

   - ***<JAVA LIBRARY>***: posizione della libreria Java JRE.
   - ***<TOMCAT PATH>***: cartella di installazione di Tomcat.

Aggiungi un connettore HTTPS nella configurazione del server.

1. Vai alla cartella di installazione di Tomcat.
2. Dopo aver sostituito i *<VARIABLES>* seguenti dati con i tuoi dati specifici, utilizza il seguente comando per aggiungere un connettore HTTPS per utilizzare i certificati generati nei prerequisiti:

```
$ sed -i '/<Connector port="8080"/i <Connector port=\"443\" maxThreads=\"200\"
 scheme=\"https\" secure=\"true\" SSLEnabled=\"true\" keystoreType=\"CLOUDHSM\"
 keystoreFile=\"
        <CUSTOM DIRECTORY>/<JSSE KEYSTORE NAME>.keystore\" keystorePass=\"<KEYSTORE
 PASSWORD>\" keyPass=\"<KEY PASSWORD>
        \" keyAlias=\"<UNIQUE ALIAS FOR KEYS>" clientAuth=\"false\" sslProtocol=
\"TLS\"/>' <TOMCAT PATH>/conf/server.xml
```

   - ***<CUSTOM DIRECTORY>***: cartella in cui si trova il file keystore.
   - ***<JSSE KEYSTORE NAME>***: nome del file Keystore.
   - ***<KEYSTORE PASSWORD>***: Questa è la password per il file keystore locale.
   - ***<KEY PASSWORD>***: Memorizziamo il riferimento alla tua chiave nel file keystore locale e questa password protegge quel riferimento locale.
   - ***<UNIQUE ALIAS FOR KEYS>***: Viene utilizzato per identificare in modo univoco la chiave sull'HSM. Questo alias verrà impostato come attributo ETICHETTA della chiave.

- ***<TOMCAT PATH>***: Il percorso della cartella Tomcat.

Avvio del server

- Dopo aver sostituito i *<VARIABLES>* seguenti dati con i tuoi dati specifici, usa il seguente comando per avviare Tomcat Server:

```
$ /<TOMCAT DIRECTORY>/bin/startup.sh
```

> **Note**
>
> ***<TOMCAT DIRECTORY>***è il nome della directory di installazione di Tomcat.

Dopo avere aggiornato la configurazione del server Web, vai alla Fase 4: abilitazione del traffico HTTPS e verifica del certificato.

Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Dopo aver configurato il server Web per l' SSL/TLS offload con AWS CloudHSM, aggiungete l'istanza del server Web a un gruppo di sicurezza che consente il traffico HTTPS in entrata. Ciò consente ai client, come i browser Web, di stabilire una connessione HTTPS con il server Web. Quindi stabilisci una connessione HTTPS al tuo server web e verifica che stia utilizzando il certificato con cui hai configurato per l' SSL/TLS offload. AWS CloudHSM

Argomenti

- Abilitazione delle connessioni HTTPS in entrata
- Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Abilitazione delle connessioni HTTPS in entrata

Per connetterti al server Web da un client (ad esempio un browser Web), crea un gruppo di sicurezza che consenta le connessioni HTTPS in entrata. Nello specifico, deve consentire le connessioni TCP in entrata sulla porta 443. Assegna questo gruppo di sicurezza al tuo server Web.

Per creare un gruppo di sicurezza per HTTPS e assegnarlo al server Web

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.

2. Seleziona Gruppi di sicurezza nel riquadro di navigazione.
3. Scegliere Create Security Group (Crea gruppo di sicurezza).
4. Per Create Security Group (Crea un gruppo di sicurezza), procedere come segue:
    a. Per Security group name (Nome del gruppo di sicurezza), digitare un nome per il gruppo di sicurezza che si sta creando.
    b. (Facoltativo) Digitare una descrizione del gruppo di sicurezza in fase di creazione.
    c. Per VPC, scegli il VPC contenente l'istanza Amazon EC2 del server Web.
    d. Seleziona Aggiungi regola.
    e. Per Tipo, seleziona HTTPS dalla finestra a discesa.
    f. Per Origine, inserisci una posizione di origine.
    g. Scegliere Create Security Group (Crea gruppo di sicurezza).
5. Nel riquadro di navigazione, scegliere Instances (Istanze).
6. Seleziona la casella di controllo accanto all'istanza del server Web.
7. Seleziona il menu a discesa Operazioni nella parte superiore della pagina. Seleziona Sicurezza, quindi Modifica gruppi di sicurezza.
8. Per Gruppi di sicurezza associati, seleziona la casella di ricerca e scegli il gruppo di sicurezza creato per HTTPS. Quindi, scegli Aggiungi i gruppi di sicurezza.
9. Seleziona Salva.

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo aver aggiunto il server Web a un gruppo di sicurezza, puoi verificare che SSL/TLS offload utilizzi il tuo certificato autofirmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l' SSL/TLS offload con un browser web

1. Utilizza un browser Web per connetterti al server Web utilizzando il nome DNS pubblico o l'indirizzo IP del server. Accertarsi che l'URL nella barra degli indirizzi inizi con https://. Ad esempio, **`https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`**.

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Utilizza il browser Web per visualizzare il certificato del server Web. Per ulteriori informazioni, consulta gli argomenti seguenti:

   - Per Mozilla Firefox, consultare Visualizzare un certificato sul sito Web di supporto di Mozilla.
   - Per Google Chrome, consulta la pagina Understand Security Issues sul sito Web di Google per sviluppatori.

   Altri browser Web potrebbero avere caratteristiche simili da utilizzare per visualizzare il certificato del server Web.

3. Assicurati che il SSL/TLS certificato sia quello che hai configurato per l'uso sul tuo server web.

Per verificare l' SSL/TLS offload con OpenSSL s_client

1. Esegui il seguente comando OpenSSL per connetterti al server Web tramite HTTPS. Sostituisci *<server name>* con il nome DNS pubblico o l'indirizzo IP del tuo server web.

```
openssl s_client -connect <server name>:443
```

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Assicurati che il SSL/TLS certificato sia quello che hai configurato per l'uso sul tuo server web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata per il server Web è archiviata in un HSM nel AWS CloudHSM cluster.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale).

AWS CloudHSM Offload SSL/TLS su Windows tramite IIS con KSP

Questo tutorial fornisce step-by-step istruzioni per configurare l' SSL/TLS offload con AWS CloudHSM su un server Web Windows.

Argomenti

- Panoramica di
- Fase 1: configurazione dei prerequisiti
- Fase 2: creazione di una richiesta di firma del certificato (CSR) e di un certificato
- Fase 3: configurazione del server Web
- Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Panoramica di

In Windows Internet Information Services (IIS) per l'applicazione del server Web Windows Server supporta HTTPS in modo nativo. Il provider di archiviazione delle AWS CloudHSM chiavi (KSP) per l'API di crittografia di Microsoft: Next Generation (CNG) fornisce l'interfaccia che consente a IIS di utilizzare IIS HSMs nel cluster per l'offload crittografico e l'archiviazione delle chiavi. AWS CloudHSM KSP è il bridge che collega IIS al cluster. AWS CloudHSM

In questo tutorial vengono illustrate le seguenti operazioni:

- Installa il software del server Web in un'istanza Amazon EC2.
- Configura il software del server Web in modo tale che supporti HTTPS con una chiave privata archiviata nel cluster AWS CloudHSM .
- (Facoltativo) Utilizza Amazon EC2 per creare una seconda istanza del server Web ed Elastic Load Balancing per creare un sistema di bilanciamento del carico. L'uso di un sistema di bilanciamento del carico può migliorare le prestazioni grazie alla distribuzione del carico in più server. Offre anche ridondanza e una disponibilità più elevata in caso di errore di uno o più server.

Quando sei pronto per iniziare, vai a Fase 1: configurazione dei prerequisiti.

Fase 1: configurazione dei prerequisiti

Piattaforme diverse richiedono prerequisiti diversi. Utilizza la sezione sui prerequisiti riportata di seguito corrispondente alla tua piattaforma.

Argomenti

- Prerequisiti per Client SDK 5
- Prerequisiti per Client SDK 3

Prerequisiti per Client SDK 5

Per configurare l' SSL/TLS offload del server Web con, è necessario quanto segue: AWS CloudHSM

- Un AWS CloudHSM cluster attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Windows su cui sia installato il seguente software:
  - Il software AWS CloudHSM client per Windows.
  - Internet Information Services (IIS) per Windows Server.
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata del server Web sull'HSM.

> **Note**
>
> Questo tutorial utilizza Microsoft Windows Server 2019. Sono supportati anche Microsoft Windows Server 2016 e 2022.

Per configurare un'istanza di Windows Server e creare un CU sull'HSM

1. Completa le fasi descritte in Nozioni di base. Quando avvii il client Amazon EC2, scegli un'AMI Windows Server 2019. Dopo aver completato queste fasi, sarà presente un cluster attivo con almeno un HSM. Hai anche un'istanza client Amazon EC2 che esegue Windows Server con il software AWS CloudHSM client per Windows installato.
2. (Facoltativo) Aggiungine altri HSMs al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.

3. Connettersi al server Windows. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
4. Utilizza la CLI di CloudHSM per creare un crypto user (CU). Prendere nota del nome utente e della password del CU, Saranno necessari per completare la fase successiva.

   > **Note**
   >
   > Per le informazioni sulla creazione di un utente, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM.

5. Impostare le credenziali di accesso per l'HSM, utilizzando il nome utente e la password CU creati nella fase precedente.
6. Nel passaggio 5, se hai utilizzato Windows Credentials Manager per impostare le credenziali HSM, scarica `psexec.exe`da SysInternals per eseguire il comando seguente come NT Authority \ SYSTEM:

   ```
   psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"
    --username <USERNAME> --password <PASSWORD>
   ```

   Sostituisci *<USERNAME>* e *<PASSWORD>* con le credenziali HSM.

Per installare IIS in Windows Server

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager scegliere Add roles and features (Aggiungi ruoli e funzionalità).
4. Leggere le informazioni Before you begin (Prima di iniziare), quindi scegliere Next (Successivo).
5. Per Installation Type (Tipo di installazione) scegliere Role-based or feature-based installation (Installazione basata su ruoli o su funzionalità). Quindi scegli Successivo.
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Successivo.
7. Per Server Roles (Ruoli server) utilizzare la seguente procedura:

a. Selezionare Web Server (IIS).

b. Per Add features that are required for Web Server (IIS) (Aggiungi le funzionalità richieste per Web Server (IIS)) scegliere Add Features (Aggiungi caratteristiche).

c. Selezionare Next (Successivo) per terminare la selezione dei ruoli server.

8. Per Caratteristiche, accettare le impostazioni predefinite. Quindi scegli Successivo.
9. Leggere le informazioni su Web Server Role (IIS) (Ruolo Web Server). Quindi scegli Successivo.
10. Per Select server roles (Seleziona ruoli server), accettare le impostazioni predefinite o modificarle in base alle esigenze. Quindi scegli Successivo.
11. Per Confirmation (Conferma), leggere le informazioni di conferma. Quindi scegliere Install (Istalla).
12. Al termine dell'installazione, scegliere Close (Chiudi).

Dopo aver completato queste operazioni, andare su Fase 2: creazione di una richiesta di firma del certificato (CSR) e di un certificato.

Prerequisiti per Client SDK 3

Per configurare l' SSL/TLS offload del server Web con AWS CloudHSM, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Windows su cui sia installato il seguente software:
  - Il software AWS CloudHSM client per Windows.
  - Internet Information Services (IIS) per Windows Server.
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata del server Web sull'HSM.

> **Note**
>
> In questo tutorial viene utilizzato Microsoft Windows Server 2016. È supportato anche Microsoft Windows Server 2012, ma non è supportato Microsoft Windows Server 2012 R2.

Per configurare un'istanza di Windows Server e creare un CU sull'HSM

1. Completa le fasi descritte in Nozioni di base. All'avvio del client Amazon EC2, scegliere un'AMI Windows Server 2016 o Windows Server 2012. Dopo aver completato queste fasi, sarà presente un cluster attivo con almeno un HSM. Hai anche un'istanza client Amazon EC2 che esegue Windows Server con il software AWS CloudHSM client per Windows installato.
2. (Facoltativo) Aggiungine altri HSMs al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
3. Connettersi al server Windows. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
4. Utilizza la CLI di CloudHSM per creare un crypto user (CU). Prendere nota del nome utente e della password del CU, Saranno necessari per completare la fase successiva.

   > **Note**
   >
   > Per le informazioni sulla creazione di un utente, consulta la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM.

5. Impostare le credenziali di accesso per l'HSM, utilizzando il nome utente e la password CU creati nella fase precedente.
6. Nel passaggio 5, se hai utilizzato Windows Credentials Manager per impostare le credenziali HSM, scarica `psexec.exe`da SysInternals per eseguire il comando seguente come NT Authority \ SYSTEM:

   ```
   psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"
    --username <USERNAME> --password <PASSWORD>
   ```

   Sostituisci *<USERNAME>* e *<PASSWORD>* con le credenziali HSM.

Per installare IIS in Windows Server

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Su Windows Server avviare Server Manager.

3. Nel pannello di controllo Server Manager scegliere Add roles and features (Aggiungi ruoli e funzionalità).
4. Leggere le informazioni Before you begin (Prima di iniziare), quindi scegliere Next (Successivo).
5. Per Installation Type (Tipo di installazione) scegliere Role-based or feature-based installation (Installazione basata su ruoli o su funzionalità). Quindi scegli Successivo.
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Successivo.
7. Per Server Roles (Ruoli server) utilizzare la seguente procedura:
    a. Selezionare Web Server (IIS).
    b. Per Add features that are required for Web Server (IIS) (Aggiungi le funzionalità richieste per Web Server (IIS)) scegliere Add Features (Aggiungi caratteristiche).
    c. Selezionare Next (Successivo) per terminare la selezione dei ruoli server.
8. Per Caratteristiche, accettare le impostazioni predefinite. Quindi scegli Successivo.
9. Leggere le informazioni su Web Server Role (IIS) (Ruolo Web Server). Quindi scegli Successivo.
10. Per Select server roles (Seleziona ruoli server), accettare le impostazioni predefinite o modificarle in base alle esigenze. Quindi scegli Successivo.
11. Per Confirmation (Conferma), leggere le informazioni di conferma. Quindi scegliere Install (Istalla).
12. Al termine dell'installazione, scegliere Close (Chiudi).

Dopo aver completato queste operazioni, andare su Fase 2: creazione di una richiesta di firma del certificato (CSR) e di un certificato.

Fase 2: creazione di una richiesta di firma del certificato (CSR) e di un certificato

Per abilitare HTTPS, il server Web necessita di un SSL/TLS certificato e della chiave privata corrispondente. Per utilizzare SSL/TLS offload con AWS CloudHSM, memorizzi la chiave privata nell'HSM del cluster. AWS CloudHSM Per eseguire questa operazione, è possibile utilizzare il provider di archiviazione delle chiavi (KSP)AWS CloudHSM per Cryptography API: Next Generation (CNG) di Microsoft per creare una richiesta di firma del certificato (CSR). La CSR viene quindi inviata a un'autorità di certificazione (CA), che la firma per ottenere un certificato.

Argomenti

- Crea una CSR con Client SDK 5

- Crea una CSR con Client SDK 3
- Come ottenere un certificato firmato e importarlo

Crea una CSR con Client SDK 5

1. Nel Windows Server utilizzare un editor di testo per creare un file di richiesta di certificato denominato `IISCertRequest.inf`. Di seguito viene mostrato il contenuto di un file `IISCertRequest.inf` di esempio. Per ulteriori informazioni sulle sezioni, le chiavi e i valori che è possibile specificare nel file, vedi la documentazione Microsoft. Non modificare il valore `ProviderName`.

   ```
   [Version]
   Signature = "$Windows NT$"
   [NewRequest]
   Subject = "CN=example.com,C=US,ST=Washington,L=Seattle,O=ExampleOrg,OU=WebServer"
   HashAlgorithm = SHA256
   KeyAlgorithm = RSA
   KeyLength = 2048
   ProviderName = "CloudHSM Key Storage Provider"
   KeyUsage = 0xf0
   MachineKeySet = True
   [EnhancedKeyUsageExtension]
   OID=1.3.6.1.5.5.7.3.1
   ```

2. Utilizzate il certreqcomando Windows per creare una CSR dal `IISCertRequest.inf` file creato nel passaggio precedente. L'esempio seguente salva la CSR in un file denominato `IISCertRequest.csr`. Se hai utilizzato un nome di file diverso per il file di richiesta del certificato, sostituiscilo *`IISCertRequest.inf`* con il nome file appropriato. Facoltativamente, puoi sostituirlo *`IISCertRequest.csr`* con un nome di file diverso per il tuo file CSR.

   ```
   C:\>certreq -new IISCertRequest.inf IISCertRequest.csr

   CertReq: Request Created
   ```

   Il file `IISCertRequest.csr` contene la CSR. Questa CSR è necessaria per ottenere un certificato firmato.

Crea una CSR con Client SDK 3

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Usa il seguente comando per avviare il demone AWS CloudHSM client.

   Amazon Linux

   ```
   $ sudo start cloudhsm-client
   ```

   Amazon Linux 2

   ```
   $ sudo service cloudhsm-client start
   ```

   CentOS 7

   ```
   $ sudo service cloudhsm-client start
   ```

   CentOS 8

   ```
   $ sudo service cloudhsm-client start
   ```

   RHEL 7

   ```
   $ sudo service cloudhsm-client start
   ```

   RHEL 8

   ```
   $ sudo service cloudhsm-client start
   ```

   Ubuntu 16.04 LTS

   ```
   $ sudo service cloudhsm-client start
   ```

   Ubuntu 18.04 LTS

   ```
   $ sudo service cloudhsm-client start
   ```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe
 C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

3. Nel Windows Server utilizzare un editor di testo per creare un file di richiesta di certificato denominato `IISCertRequest.inf`. Di seguito viene mostrato il contenuto di un file `IISCertRequest.inf` di esempio. Per ulteriori informazioni sulle sezioni, le chiavi e i valori che è possibile specificare nel file, vedi la documentazione Microsoft. Non modificare il valore `ProviderName`.

```
[Version]
Signature = "$Windows NT$"
[NewRequest]
Subject = "CN=example.com,C=US,ST=Washington,L=Seattle,O=ExampleOrg,OU=WebServer"
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "Cavium Key Storage Provider"
KeyUsage = 0xf0
MachineKeySet = True
[EnhancedKeyUsageExtension]
OID=1.3.6.1.5.5.7.3.1
```

4. Utilizzate il certreqcomando Windows per creare una CSR dal `IISCertRequest.inf` file creato nel passaggio precedente. L'esempio seguente salva la CSR in un file denominato `IISCertRequest.csr`. Se hai utilizzato un nome di file diverso per il file di richiesta del certificato, sostituiscilo *`IISCertRequest.inf`* con il nome file appropriato. Facoltativamente, puoi sostituirlo *`IISCertRequest.csr`* con un nome di file diverso per il tuo file CSR.

```
C:\>certreq -new IISCertRequest.inf IISCertRequest.csr
        SDK Version: 2.03

CertReq: Request Created
```

Il file `IISCertRequest.csr` contene la CSR. Questa CSR è necessaria per ottenere un certificato firmato.

Come ottenere un certificato firmato e importarlo

In un ambiente di produzione, per creare un certificato da una CSR in genere ci si avvale di un'autorità di certificazione, che non è invece necessaria per un ambiente di test. Se utilizzi una CA, inviagli il file CSR (`IISCertRequest.csr`) e usa la CA per creare un certificato firmato. SSL/TLS

In alternativa all'utilizzo di una CA, è possibile utilizzare uno strumento come OpenSSL per creare un certificato autofirmato.

> **Warning**
>
> I certificati autofirmati non sono considerati attendibili dai browser e non devono essere utilizzati negli ambienti di produzione, ma solo negli ambienti di test.

Le procedure seguenti illustrano come creare un certificato autofirmato e utilizzarlo per firmare la CSR del proprio server Web.

Per creare un certificato autofirmato

1. Eseguire il comando OpenSSL seguente per creare una chiave privata. Facoltativamente, puoi sostituirlo *`SelfSignedCA.key`* con il nome del file per contenere la tua chiave privata.

   ```
   openssl genrsa -aes256 -out SelfSignedCA.key 2048
   Generating RSA private key, 2048 bit long modulus
   ..............................................................+++
   .....................................+++
   e is 65537 (0x10001)
   Enter pass phrase for SelfSignedCA.key:
   Verifying - Enter pass phrase for SelfSignedCA.key:
   ```

2. Eseguire il comando OpenSSL seguente per creare un certificato autofirmato mediante la chiave privata creata nella fase precedente. Si tratta di un comando interattivo. Leggi le istruzioni a video e segui i prompt. Sostituiscilo *`SelfSignedCA.key`* con il nome del file che contiene la tua chiave privata (se diverso). Facoltativamente, puoi sostituirlo *`SelfSignedCA.crt`* con il nome del file per contenere il tuo certificato autofirmato.

```
openssl req -new -x509 -days 365 -key SelfSignedCA.key -out SelfSignedCA.crt
Enter pass phrase for SelfSignedCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Per utilizzare il certificato autofirmato per firmare la CSR del server Web

- Eseguire il seguente comando OpenSSL per utilizzare la chiave privata e il certificato autofirmato per firmare il CSR. Sostituire gli elementi seguenti con i nomi dei file che contengono i dati corrispondenti (se diversi).
  - *IISCertRequest.csr*— Il nome del file che contiene la CSR del server web
  - *SelfSignedCA.crt*— Il nome del file che contiene il certificato autofirmato
  - *SelfSignedCA.key*— Il nome del file che contiene la chiave privata
  - *IISCert.crt*— Il nome del file che contiene il certificato firmato del server web

```
openssl x509 -req -days 365 -in IISCertRequest.csr \
                            -CA SelfSignedCA.crt \
                            -CAkey SelfSignedCA.key \
                            -CAcreateserial \
                            -out IISCert.crt
Signature ok
subject=/ST=IIS-HSM/L=IIS-HSM/OU=IIS-HSM/O=IIS-HSM/CN=IIS-HSM/C=IIS-HSM
Getting CA Private Key
Enter pass phrase for SelfSignedCA.key:
```

Una volta completato il passaggio precedente, avrai un certificato firmato per il server Web (`IISCert.crt`) e un certificato autofirmato (`SelfSignedCA.crt`). A questo punto, vai alla sezione Fase 3: configurazione del server Web.

Fase 3: configurazione del server Web

È possibile aggiornare la configurazione del sito Web IIS per utilizzare il certificato HTTPS creato alla fine della fase precedente. Verrà completata la configurazione del software del server Web Windows (IIS) per l' SSL/TLS offload con. AWS CloudHSM

Se hai utilizzato un certificato autofirmato per firmare la CSR, devi innanzitutto importare il certificato autofirmato nelle Autorità di certificazione fonti attendibili Windows.

Per importare il certificato autofirmato nelle Autorità di certificazione fonti attendibili Windows

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Copiare il certificato autofirmato nel server Windows.
3. Nel server Windows aprire il Pannello di controllo.
4. In Cerca nel Pannello di controllo digitare **`certificates`**. Quindi scegliere Gestisci i certificati computer.
5. Nella finestra Certificati - Computer locale, fai doppio clic su Trusted Root Certification Authorities.
6. Fai clic con il pulsante destro del mouse su Certificati e quindi scegliere Tutte le attività, Importa.
7. In Importazione guidata certificati scegliere Avanti.
8. Scegliere Sfoglia, quindi individuare e selezionare il certificato autofirmato. Se il certificato autofirmato è stato creato seguendo le istruzioni nella fase precedente di questo tutorial, il nome del certificato sarà `SelfSignedCA.crt`. Seleziona Apri.
9. Scegli Next (Successivo).
10. Per Archivio certificati scegliere Mettere tutti i certificati nel seguente archivio. Quindi accertarsi che sia selezionata l'opzione Autorità di certificazione radice attendibili per Archivio certificati.
11. Scegliere Avanti, quindi scegliere Fine.

Pe aggiornare la configurazione del sito Web IIS

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Avvia il daemon AWS CloudHSM del client.
3. Copia il certificato firmato del server Web (quello creato al termine della fase precedente di questo tutorial) nel server Windows.
4. Sul tuo Windows Server, usa il certreqcomando Windows per accettare il certificato firmato, come nell'esempio seguente. Sostituiscilo *IISCert.crt* con il nome del file che contiene il certificato firmato del tuo server Web.

   ```
   C:\>certreq -accept IISCert.crt
           SDK Version: 2.03
   ```

5. Su Windows Server avviare Server Manager.
6. Nel pannello di controllo di Server Manager, nell'angolo in alto a destra, scegliere Strumenti, Gestione Internet Information Services (IIS).
7. Nella finestra Gestione Internet Information Services (IIS) fare doppio clic sul nome del server. Quindi fare doppio clic su Siti. Selezionare il sito Web.
8. Selezionare Impostazioni SSL. Quindi, sulla parte destra della finestra, scegliere Binding.
9. Nella finestra Binding sito Web scegliere Aggiungi.
10. Per Tipo, scegliere https. Per Certificato SSL, scegliere il certificato HTTPS creato alla fine della fase precedente di questo tutorial.

    > **Note**
    >
    > Se si verifica un errore durante l'associazione del certificato, riavviare il server e riprovare l'operazione.

11. Scegli OK.

Dopo avere aggiornato la configurazione del sito Web, vai a Fase 4: abilitazione del traffico HTTPS e verifica del certificato.

Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Dopo aver configurato il server Web per l' SSL/TLS offload con AWS CloudHSM, aggiungi l'istanza del server Web a un gruppo di sicurezza che consente il traffico HTTPS in entrata. Ciò consente ai client, come i browser Web, di stabilire una connessione HTTPS con il server Web. Quindi stabilisci una connessione HTTPS al tuo server web e verifica che stia utilizzando il certificato con cui hai configurato per l' SSL/TLS offload. AWS CloudHSM

Argomenti

- Abilitazione delle connessioni HTTPS in entrata
- Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Abilitazione delle connessioni HTTPS in entrata

Per connetterti al server Web da un client (ad esempio un browser Web), crea un gruppo di sicurezza che consenta le connessioni HTTPS in entrata. Nello specifico, deve consentire le connessioni TCP in entrata sulla porta 443. Assegna questo gruppo di sicurezza al tuo server Web.

Per creare un gruppo di sicurezza per HTTPS e assegnarlo al server Web

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Seleziona Gruppi di sicurezza nel riquadro di navigazione.
3. Scegliere Create Security Group (Crea gruppo di sicurezza).
4. Per Create Security Group (Crea un gruppo di sicurezza), procedere come segue:
    a. Per Security group name (Nome del gruppo di sicurezza), digitare un nome per il gruppo di sicurezza che si sta creando.
    b. (Facoltativo) Digitare una descrizione del gruppo di sicurezza in fase di creazione.
    c. Per VPC, scegli il VPC contenente l'istanza Amazon EC2 del server Web.
    d. Seleziona Aggiungi regola.
    e. Per Tipo, seleziona HTTPS dalla finestra a discesa.
    f. Per Origine, inserisci una posizione di origine.
    g. Scegliere Create Security Group (Crea gruppo di sicurezza).
5. Nel riquadro di navigazione, scegliere Instances (Istanze).
6. Seleziona la casella di controllo accanto all'istanza del server Web.

7. Seleziona il menu a discesa Operazioni nella parte superiore della pagina. Seleziona Sicurezza, quindi Modifica gruppi di sicurezza.
8. Per Gruppi di sicurezza associati, seleziona la casella di ricerca e scegli il gruppo di sicurezza creato per HTTPS. Quindi, scegli Aggiungi i gruppi di sicurezza.
9. Seleziona Salva.

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo aver aggiunto il server Web a un gruppo di sicurezza, puoi verificare che SSL/TLS offload utilizzi il tuo certificato autofirmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l' SSL/TLS offload con un browser web

1. Utilizza un browser Web per connetterti al server Web utilizzando il nome DNS pubblico o l'indirizzo IP del server. Accertarsi che l'URL nella barra degli indirizzi inizi con https://. Ad esempio, **`https://ec2-52-14-212-67.us-east-2.compute.amazonaws.com/`**.

   > **Tip**
   >
   > Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Utilizza il browser Web per visualizzare il certificato del server Web. Per ulteriori informazioni, consulta gli argomenti seguenti:

   - Per Mozilla Firefox, consultare Visualizzare un certificato sul sito Web di supporto di Mozilla.
   - Per Google Chrome, consulta la pagina Understand Security Issues sul sito Web di Google per sviluppatori.

   Altri browser Web potrebbero avere caratteristiche simili da utilizzare per visualizzare il certificato del server Web.
3. Assicurati che il SSL/TLS certificato sia quello che hai configurato per l'uso sul tuo server web.

Per verificare l' SSL/TLS offload con OpenSSL s_client

1. Esegui il seguente comando OpenSSL per connetterti al server Web tramite HTTPS. Sostituisci *<server name>* con il nome DNS pubblico o l'indirizzo IP del tuo server web.

   ```
   openssl s_client -connect <server name>:443
   ```

   > **Tip**
   >
   > Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Assicurati che il SSL/TLS certificato sia quello che hai configurato per l'uso sul tuo server web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata per il server Web è archiviata in un HSM nel AWS CloudHSM cluster.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale).

Aggiungi un sistema di bilanciamento del carico con Elastic Load Balancing AWS CloudHSM per (opzionale)

Dopo aver SSL/TLS configurato l'offload con un server Web, puoi creare più server Web e un sistema di bilanciamento del carico Elastic Load Balancing che indirizza il traffico HTTPS verso i server Web. Un sistema di bilanciamento del carico è in grado di ridurre il carico sui singoli server Web bilanciando il traffico tra due o più server. È inoltre in grado di aumentare la disponibilità del sito Web, poiché il sistema di bilanciamento del carico monitora lo stato dei server Web e instrada solo il traffico verso i server integri. Se un server Web presenta dei problemi, il sistema di bilanciamento del carico interrompe automaticamente l'instradamento del traffico verso quel server.

Argomenti

- Passaggio 1. Creazione di una sottorete per un secondo server Web
- Passaggio 2. Creazione del secondo server Web

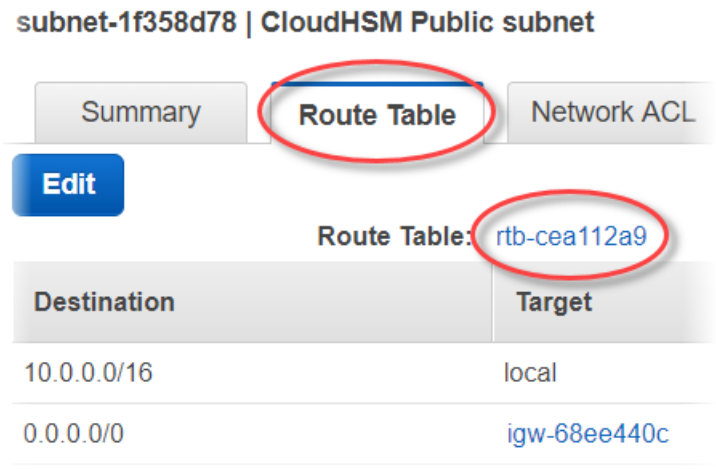
- Fase 3. Creazione del sistema di bilanciamento del carico

Passaggio 1. Creazione di una sottorete per un secondo server Web

Prima di poter creare un altro server Web, è necessario creare una nuova sottorete nello stesso VPC che contiene il server AWS CloudHSM Web e il cluster esistenti.

Per creare una nuova sottorete

1. Apri la sezione Sottoreti della console Amazon VPC.
2. Seleziona Create Subnet (Crea sottorete).
3. Nella finestra di dialogo Create Subnet (Crea sottorete), seguire questi passaggi:
   a. In Name tag (Assegna un nome al tag), digitare un nome per la sottorete.
   b. Per VPC, scegli il AWS CloudHSM VPC che contiene il server Web e il cluster esistenti. AWS CloudHSM
   c. Per Availability Zone (Zona di disponibilità), scegliere una zona di disponibilità diversa da quella che contiene il server Web esistente.
   d. In IPv4 CIDR block (Blocco CIDR), digitare il blocco CIDR da usare per la sottorete. Ad esempio, digita **`10.0.10.0/24`**.
   e. Selezionare Yes, Create (Sì, crea).
4. Seleziona la casella di controllo accanto alla sottorete pubblica che contiene il server Web esistente. Si tratta di una sottorete pubblica diversa da quella creata nella fase precedente.
5. Nel riquadro dei contenuti, scegli la scheda Tabella di routing. Quindi scegliere il link per la tabella di routing.



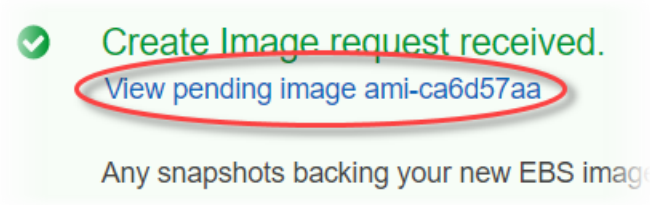
6. Selezionare la casella di controllo accanto alla tabella di routing.
7. Scegli la scheda Associazioni sottoreti. Quindi scegliere Edit (Modifica).
8. Seleziona la casella di controllo accanto alla sottorete pubblica creata precedentemente in questa procedura. Quindi scegli Save (Salva).

Passaggio 2. Creazione del secondo server Web

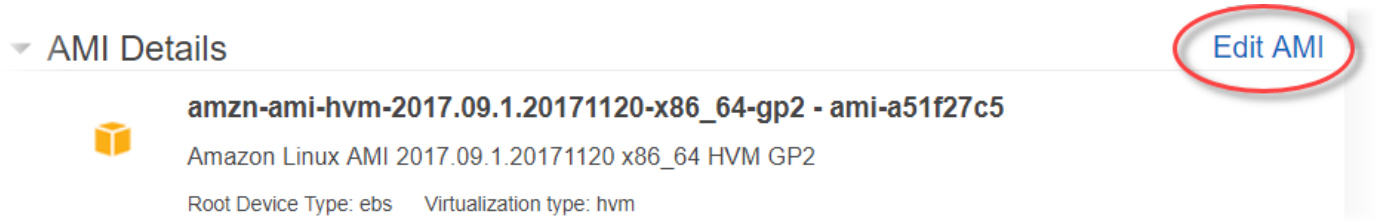
Completa le fasi seguenti per creare un secondo server Web con la stessa configurazione del tuo server Web esistente.

Per creare un secondo server Web

1. Apri la sezione Istanze nella console Amazon EC2.
2. Selezionare la casella di controllo accanto all'istanza del server Web esistente.
3. Scegliere Actions (Operazioni), Image (Immagine), quindi Create Image (Crea immagine).
4. Nella finestra di dialogo Crea Image (Crea immagine), seguire questi passaggi:
   a. Per Image name (Nome immagine), digitare un nome per l'immagine.
   b. Per Image description (Descrizione immagine), digitare una descrizione per l'immagine.
   c. Scegliere Create Image (Crea immagine). Questa operazione riavvia il server Web esistente.
   d. Scegli l'ami- link Visualizza immagine in sospeso. *<AMI ID>*



   Nella colonna Stato, prendi nota dello stato dell'immagine. Se lo stato dell'immagine è available (disponibile) (potrebbe essere necessario qualche minuto), passare alla fase successiva.
5. Nel riquadro di navigazione, scegliere Instances (Istanze).
6. Selezionare la casella di controllo accanto al server Web esistente.
7. Scegliere Actions (Operazioni), quindi Launch More Like This (Avvia altre come questa).
8. Selezionare Edit AMI (Modifica AMI).



9. Nel riquadro di navigazione a sinistra, scegli Mio AMIs. Quindi cancellare il testo nella casella di ricerca.
10. Accanto all'immagine del server Web, scegliere Select (Seleziona).
11. Scegli Sì, voglio continuare con questo AMI (***<image name>***- ami-***<AMI ID>***).
12. Scegli Next (Successivo).
13. Seleziona un tipo di istanza, quindi scegli Next: Configure Instance Details (Successivo: configura dettagli dell'istanza).
14. Per Step 3: Configure Instance Details (Fase 3: Configurare i dettagli dell'istanza), procedere come segue:
    a. Per Network (Rete), scegliere il VPC che contiene il server Web esistente.
    b. Per Subnet (Sottorete), scegliere la sottorete pubblica creata per il secondo server Web.
    c. Per Auto-assign Public IP (Assegna automaticamente IP pubblico), sceglieреEnable (Abilita).
    d. Modifica i dettagli rimanenti dell'istanza in base alle preferenze. Quindi, scegliere Next: Add Storage (Fase successiva: aggiungi storage).
15. Modifica le impostazioni di storage come desiderato. Quindi selezionare Next: Add Tags (Fase successiva: aggiungere tag).
16. Aggiungi o modifica i tag come preferito, quindi scegliere Next: Configure Security Group (Fase successiva: configurare il gruppo di sicurezza) .
17. Per Step 6: Configure Security Group (Fase 6: configurare il gruppo di sicurezza), procedere come segue:
    a. Per Assign a security group (Assegna un gruppo di sicurezza), scegliere Select an existing security group (Seleziona un gruppo di sicurezza esistente).
    b. Seleziona la casella di controllo accanto al gruppo di sicurezza denominato cloudhsm-***<cluster ID>*** -sg. AWS CloudHSM ha creato questo gruppo di sicurezza per tuo conto quando hai creato il cluster. È necessario scegliere questo gruppo di sicurezza per consentire all'istanza del server Web di connettersi al gruppo presente HSMs nel cluster.

c. Seleziona la casella di controllo accanto al gruppo di sicurezza che consente il traffico HTTPS in entrata. Il gruppo di sicurezza è stato creato in precedenza.

d. (Facoltativo) Seleziona la casella di controllo accanto a un gruppo di sicurezza che consente il traffico SSH (per Linux) o RDP (per Windows) in entrata dalla rete. Ovvero, il gruppo di sicurezza deve consentire il traffico TCP in entrata sulla porta 22 (per SSH su Linux) o sulla porta 3389 (per RDP su Windows). In caso contrario, non è possibile connettersi all'istanza del client. Se non disponi di un gruppo di sicurezza come questo, è necessario crearne uno e assegnarlo all'istanza del client in un secondo momento.

Scegli Analizza e avvia.

18. Consultare i dettagli dell'istanza e scegliere Launch (Avvia).
19. Scegliere se avviare l'istanza con una coppia di chiavi esistente, creare una nuova coppia di chiavi o avviare l'istanza senza alcuna coppia di chiavi.

- Per utilizzare una coppia di chiavi esistente, eseguire quanto descritto di seguito.
  1. Scegli Choose an existing key pair (Scegli una coppia di chiavi esistente).
  2. Per Select a key pair (Selezionare una coppia di chiavi), scegliere la coppia di chiavi da utilizzare.
  3. Seleziona la casella di controllo accanto a Riconosco di avere accesso al file di chiave privata selezionato (***<private key file name>***.pem) e che senza questo file non sarò in grado di accedere alla mia istanza.
- Per creare una nuova coppia di chiavi, eseguire quanto descritto di seguito:
  1. Scegliere Create a new key pair (Crea nuova coppia di chiavi).
  2. Per Key pair name (Nome coppia di chiavi), digitare un nome per la coppia di chiavi.
  3. Scegliere Download Key Pair (Scarica la coppia di chiavi) e salvare il file della chiave privata in una posizione protetta e accessibile.

  > **Warning**
  >
  > Non è possibile scaricare nuovamente il file della chiave privata dopo questo punto. Se il file della chiave privata non viene scaricato a questo punto, non sarà possibile accedere all'istanza del client.

- Per avviare l'istanza senza una coppia di chiavi, procedere nel seguente modo:
  1. Scegliere Proceed without a key pair (Procedi senza una coppia di chiavi).

2. Selezionare la casella di controllo accanto a I acknowledge that I will not be able to connect to this instance unless I already know the password built into this AMI. (Prendo atto che non potrò collegarmi a questa istanza, a meno che io non conosca già la password integrata in questa AMI.)

Scegliere Launch Instances (Avvia istanze).

Fase 3. Creazione del sistema di bilanciamento del carico

Completa la procedura seguente per creare un sistema di bilanciamento del carico Elastic Load Balancing che indirizzi il traffico HTTPS ai tuoi server Web.

Come creare un bilanciatore del carico

1. Apri la sezione Sistemi di bilanciamento del carico nella console Amazon EC2.
2. Seleziona Crea sistema di bilanciamento del carico.
3. Nella sezione Network Load Balancer (Sistema di bilanciamento del carico della rete), selezionare Create (Crea).
4. Per Step 1: Configure Load Balancer (Fase 1: configurare il sistema di bilanciamento del carico), procedere come segue:
   a. Per Name (Nome), digitare un nome per il sistema di bilanciamento del carico in fase di creazione.
   b. Nella sezione Ascoltatori, per Porta del sistema di bilanciamento del carico, modifica il valore impostandolo su **443**.
   c. Nella sezione Availability Zones (Zone di disponibilità), per VPC scegliere il VPC che contiene i server Web.
   d. Nella sezione Availability Zones (Zone di disponibilità), scegliere le sottoreti che contengono i server Web.
   e. Seleziona Successivo: Configurazione del routing.
5. Per Step 2: Configure Routing (Fase 2: configurare l'instradamento), procedere come segue:
   a. Per Name (Nome), digitare un nome per il gruppo target in fase di creazione.
   b. Per Porta, modifica il valore impostandolo su **443**.
   c. Seleziona Next: Register Targets (Fase successiva: registrazione delle destinazioni).

6. Per Step 3: Register Targets (Fase 3: registrare i target), procedere come segue:

   a. Nella sezione Istanze, seleziona le caselle di controllo accanto alle istanze del server Web. Quindi scegliere Add to registered (Aggiungi a elementi registrati).

   b. Scegli Prossimo: Rivedi.

7. Esaminare i dettagli del sistema di bilanciamento del carico, quindi scegliere Create (Crea).

8. Se il sistema di bilanciamento del carico è stato creato correttamente, scegliere Close (Chiudi).

Dopo aver completato i passaggi precedenti, la console Amazon EC2 mostra il sistema di bilanciamento del carico Elastic Load Balancing.

Quando lo stato del sistema di bilanciamento del carico è attivo, puoi verificare se il sistema è in funzione. Cioè, puoi verificare che stia inviando traffico HTTPS ai tuoi server web con SSL/TLS offload with. AWS CloudHSM Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare se il tuo sistema di bilanciamento del carico funziona con un browser Web

1. Nella console Amazon EC2, individua il nome DNS del sistema di bilanciamento del carico appena creato. quindi selezionare il nome DNS e copiarlo.

2. Utilizza un browser Web, ad esempio Mozilla Firefox o Google Chrome, per connetterti al sistema di bilanciamento del carico utilizzando il relativo nome DNS. Accertarsi che l'URL nella barra degli indirizzi inizi con https://.

   > **Tip**
   >
   > Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

3. Utilizza il browser Web per visualizzare il certificato del server Web. Per ulteriori informazioni, consulta gli argomenti seguenti:

   - Per Mozilla Firefox, consultare Visualizzare un certificato sul sito Web di supporto di Mozilla.

- Per Google Chrome, consulta la pagina Understand Security Issues sul sito Web di Google per sviluppatori.

Altri browser Web potrebbero avere caratteristiche simili da utilizzare per visualizzare il certificato del server Web.

4. Assicurati che il certificato corrisponda a quello configurato per l'uso da parte del server Web.

Per verificare se il sistema di bilanciamento del carico funziona con OpenSSL s_client

1. Utilizza il seguente comando OpenSSL per connetterti al sistema di bilanciamento del carico tramite HTTPS. *<DNS name>*Sostituiscilo con il nome DNS del tuo sistema di bilanciamento del carico.

```
openssl s_client -connect <DNS name>:443
```

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per indirizzare il nome di dominio del tuo sito Web (ad esempio, https://www.example.com/) al tuo server Web. Per ulteriori informazioni, consulta la sezione Routing del traffico a un'istanza Amazon EC2 nella Guida per gli sviluppatori di Amazon Route 53 oppure nella documentazione del servizio DNS in uso.

2. Assicurati che il certificato corrisponda a quello configurato per l'uso da parte del server Web.

Ora hai un sito Web protetto con HTTPS, con la chiave privata del server Web archiviata in un HSM del tuo cluster. AWS CloudHSM Il tuo sito Web ha due server Web e un sistema di bilanciamento del carico per aiutare a migliorare l'efficienza e la disponibilità.

Configurare Windows Server come autorità di certificazione (CA) con AWS CloudHSM

AWS CloudHSM offre supporto per configurare Windows Server come autorità di certificazione (CA) tramite Client SDK 3 e Client SDK 5. I passaggi per utilizzare questi strumenti variano a seconda

della versione del client SDK in cui è stato attualmente effettuato il download. Le seguenti sezioni forniscono informazioni su ciascun SDK.

Argomenti

- Configurare Windows Server come autorità di certificazione (CA) con Client SDK 5
- Configurare Windows Server come autorità di certificazione (CA) con Client SDK 3

Configurare Windows Server come autorità di certificazione (CA) con Client SDK 5

In un'infrastruttura a chiave pubblica (PKI), un'autorità di certificazione (CA) è un'entità attendibile che emette certificati digitali. Tali certificati digitali associano una chiave pubblica a un'identità (una persona o un'organizzazione) mediante crittografia a chiave pubblica e firme digitali. Per gestire una CA, è necessario mantenere l'attendibilità proteggendo la chiave privata che firma i certificati emessi dalla CA. È possibile archiviare la chiave privata nell'HSM del cluster AWS CloudHSM e utilizzare l'HSM per eseguire le operazioni di firma crittografica.

In questo tutorial, si utilizza Windows Server e si AWS CloudHSM configura una CA. Puoi installare il client software AWS CloudHSM per Windows sul server Windows e quindi aggiungere il ruolo Active Directory Certificate Services (AD CS) a Windows Server. Quando si configura questo ruolo, si utilizza un provider di archiviazione delle AWS CloudHSM chiavi (KSP) per creare e archiviare la chiave privata della CA nel AWS CloudHSM cluster. Il KSP è il bridge che collega il server Windows al AWS CloudHSM cluster. Nell'ultima fase, firmerai una richiesta di firma del certificato (CSR) con la tua CA Windows Server.

Per ulteriori informazioni, consulta i seguenti argomenti:

Argomenti

- Fase 1: configurazione dei prerequisiti
- Passaggio 2: creare una CA Windows Server con AWS CloudHSM
- Passaggio 3: Firma una richiesta di firma del certificato (CSR) con la CA di Windows Server con AWS CloudHSM

Fase 1: configurazione dei prerequisiti

Per configurare Windows Server come autorità di certificazione (CA) con AWS CloudHSM, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Windows Server con il software AWS CloudHSM client per Windows installato. In questo tutorial viene utilizzato Microsoft Windows Server 2016.
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata della CA sull'HSM.

Per configurare i prerequisiti per una CA Windows Server con AWS CloudHSM

1. Completa le fasi descritte in Nozioni di base. All'avvio del client Amazon EC2, scegli un'AMI Windows Server. In questo tutorial viene utilizzato Microsoft Windows Server 2016. Dopo aver completato queste fasi, sarà presente un cluster attivo con almeno un HSM. Hai anche un'istanza client Amazon EC2 che esegue Windows Server con il software AWS CloudHSM client per Windows installato.
2. (Facoltativo) Aggiungine altri HSMs al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
3. Effettuare la connessione all'istanza del client. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
4. Crea un utente crittografico (CU) utilizzando Managing HSM users with CloudHSM CLI o Managing HSM users with CloudHSM Management Utility (CMU). Prendere nota del nome utente e della password del CU, Saranno necessari per completare la fase successiva.
5. Impostare le credenziali di accesso per l'HSM, utilizzando il nome utente e la password CU creati nella fase precedente.
6. Nel passaggio 5, se hai utilizzato Windows Credentials Manager per impostare le credenziali HSM, scarica *psexec.exe*da per eseguire il comando seguente come NT Authority\ SYSTEM: SysInternals

   ```
   psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools\set_cloudhsm_credentials.exe"
    --username <USERNAME> --password <PASSWORD>
   ```

   Sostituisci *<USERNAME>* e *<PASSWORD>* con le credenziali HSM.

Per creare una CA Windows Server con AWS CloudHSM, vai a. Creare un'autorità di certificazione (CA) Windows Server

Passaggio 2: creare una CA Windows Server con AWS CloudHSM

Per creare un'autorità di certificazione (CA) Windows Server, aggiungi il ruolo Active Directory Certificate Services (AD CS) al tuo Windows Server. Quando aggiungi questo ruolo, utilizzi un provider di archiviazione delle AWS CloudHSM chiavi (KSP) per creare e archiviare la chiave privata della CA nel tuo AWS CloudHSM cluster.

> **Note**
>
> Quando crei la tua CA Windows Server, puoi scegliere di creare una CA radice o una CA subordinata. La scelta dipende in genere da come la tua infrastruttura a chiave pubblica è stata progettata e dalle policy di sicurezza della tua organizzazione. Per semplicità, questo tutorial spiega come creare una CA radice.

Per aggiungere un ruolo AD CS al tuo Windows Server e creare una chiave privata CA

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager scegliere Add roles and features (Aggiungi ruoli e funzionalità).
4. Leggere le informazioni Before you begin (Prima di iniziare), quindi scegliere Next (Successivo).
5. Per Installation Type (Tipo di installazione) scegliere Role-based or feature-based installation (Installazione basata su ruoli o su funzionalità). Quindi scegli Successivo.
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Successivo.
7. Per Server Roles (Ruoli server) utilizzare la seguente procedura:
   a. Selezionare Active Directory Certificate Services.
   b. Per Add features that are required for Active Directory Certificate Services (Aggiungi le funzionalità richieste per Active Directory Certificate Services) scegliere Add Features (Aggiungi funzionalità).
   c. Selezionare Next (Successivo) per terminare la selezione dei ruoli server.
8. Per Features (Funzionalità), accettare i valori predefiniti, quindi scegliere Next (Successivo).
9. Per AD CS procedere nel seguente modo:

a. Scegli Next (Successivo).

b. Selezionare Certification Authority (Autorità di certificazione), quindi Next (Successivo).

10. Per Confirmation (Conferma), leggere le informazioni di conferma, quindi scegliere Install (Installa). Non chiudere la finestra.
11. Scegliere il link evidenziato Configure Active Directory Certificate Services on the destination server (Configura Active Directory Certificate Services sul server di destinazione).
12. Per Credentials (Credenziali), verificare o cambiare le credenziali visualizzate. Quindi scegli Successivo.
13. Per Role Services (Servizi ruolo), selezionare Certification Authority (Autorità di certificazione). Quindi scegli Successivo.
14. Per Setup Type (Tipo di installazione) selezionare Standalone CA (CA indipendente). Quindi scegli Successivo.
15. Per CA Type (Tipo CA) selezionare Root CA (CA radice). Quindi scegli Successivo.

> **Note**
>
> La scelta tra creare una CA radice o una CA subordinata dipende da come l'infrastruttura a chiave pubblica è stata progettata e dalle policy di sicurezza dell'organizzazione. Per semplicità, questo tutorial spiega come creare una CA radice.

16. Per Private Key (Chiave privata) selezionare Create a new private key (Crea una nuova chiave privata). Quindi scegli Successivo.
17. Per Cryptography (Crittografia) procedere nel seguente modo:

    a. Per Seleziona un provider di crittografia, scegli una delle opzioni di CloudHSM Key Storage Provider dal menu. Questi sono i provider di archiviazione chiavi AWS CloudHSM . Ad esempio, puoi scegliere RSA #CloudHSM Key Storage Provider.

    b. Per Key length (Lunghezza chiave) scegliere una delle opzioni disponibili.

    c. Per Select the hash algorithm for signing certificates issued by this CA (Seleziona l'algoritmo hash per firmare i certificati rilasciati da questa CA) scegliere una delle opzioni di algoritmo hash disponibili.

    Scegli Next (Successivo).

18. Per CA Name (Nome CA) procedere nel seguente modo:

a. (Opzionale) Modificare il nome comune.

b. (Opzionale) Digitare un suffisso del nome distinto.

Scegli Next (Successivo).

19. Per Validity Period (Periodo di validità) specificare un periodo di tempo in anni, mesi, settimane o giorni. Quindi scegli Successivo.
20. Per Certificate Database (Database certificati), è possibile accettare i valori predefiniti oppure modificare la posizione per il database e il log del database. Quindi scegli Successivo.
21. Per Confirmation (Conferma) rivedere le informazioni relative alla CA, quindi scegliere Configure (Configura).
22. Selezionare Close (Chiudi) e poi ancora Close (Chiudi).

Ora disponi di una CA Windows Server con AWS CloudHSM. Per informazioni su come firmare una richiesta di firma del certificato con la tua CA, consulta Firma di una CSR.

Passaggio 3: Firma una richiesta di firma del certificato (CSR) con la CA di Windows Server con AWS CloudHSM

Puoi utilizzare la tua CA di Windows Server con AWS CloudHSM per firmare una richiesta di firma del certificato (CSR). Per completare queste fasi, è necessaria una CSR valida. Puoi creare una CSR in diversi modi, compreso quanto segue:

- Utilizzo di OpenSSL
- Utilizzo di Windows Server Internet Information Services (IIS) Manager
- Utilizzo di snap-in di certificati in Microsoft Management Console
- Utilizzo dell'utility a riga di comando certreq su Windows

Le fasi per la creazione di una CSR non rientrano nell'ambito di questo tutorial. Quando si dispone di una CSR, è possibile firmarla con la CA Windows Server.

Per firmare una CSR con la CA Windows Server

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.

2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager, nell'angolo in alto a destra, scegliere Tools (Strumenti), Certification Authority (Autorità di certificazione).
4. Nella finestra Certification Authority (Autorità di certificazione), scegliere il nome del computer.
5. Dal menu Action (Azione), scegliere All Tasks (Tutte le attività), Submit new request (Invia nuova richiesta).
6. Selezionare il file CSR, quindi scegliere Open (Apri).
7. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic su Pending Requests (Richieste in sospeso).
8. Selezionare la richiesta in sospeso. Quindi, dal menu Action (Azione), scegliere All Tasks (Tutte le attività), Issue (Invia).
9. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic su Issued Requests (Richieste emesse) per visualizzare il certificato firmato.
10. (Opzionale) Per esportare il certificato firmato in un file, completare i seguenti passaggi:

    a. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic sul certificato.
    b. Scegliere la scheda Details (Dettagli), quindi scegliere Copy to File (Copia su file).
    c. Seguire le istruzioni in Certificate Export Wizard (Esportazione guidata certificati).

Ora disponi di una CA di Windows Server con AWS CloudHSM e di un certificato valido firmato dalla CA di Windows Server.

Configurare Windows Server come autorità di certificazione (CA) con Client SDK 3

In un'infrastruttura a chiave pubblica (PKI), un'autorità di certificazione (CA) è un'entità attendibile che emette certificati digitali. Tali certificati digitali associano una chiave pubblica a un'identità (una persona o un'organizzazione) mediante crittografia a chiave pubblica e firme digitali. Per gestire una CA, è necessario mantenere l'attendibilità proteggendo la chiave privata che firma i certificati emessi dalla CA. È possibile archiviare la chiave privata nell'HSM del cluster AWS CloudHSM e utilizzare l'HSM per eseguire le operazioni di firma crittografica.

In questo tutorial, si utilizza Windows Server e si AWS CloudHSM configura una CA. Puoi installare il client software AWS CloudHSM per Windows sul server Windows e quindi aggiungere il ruolo Active Directory Certificate Services (AD CS) a Windows Server. Quando si configura questo ruolo,

si utilizza un provider di archiviazione delle AWS CloudHSM chiavi (KSP) per creare e archiviare la chiave privata della CA nel AWS CloudHSM cluster. Il KSP è il bridge che collega il server Windows al AWS CloudHSM cluster. Nell'ultima fase, firmerai una richiesta di firma del certificato (CSR) con la tua CA Windows Server.

Per ulteriori informazioni, consulta i seguenti argomenti:

Argomenti

- Fase 1: configurazione dei prerequisiti
- Passaggio 2: creare una CA Windows Server con AWS CloudHSM
- Passaggio 3: Firma una richiesta di firma del certificato (CSR) con la CA di Windows Server con AWS CloudHSM

Fase 1: configurazione dei prerequisiti

Per configurare Windows Server come autorità di certificazione (CA) con AWS CloudHSM, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Windows Server con il software AWS CloudHSM client per Windows installato. In questo tutorial viene utilizzato Microsoft Windows Server 2016.
- Un utente di crittografia (CU) che sia proprietario e che gestisca la chiave privata della CA sull'HSM.

Per configurare i prerequisiti per una CA Windows Server con AWS CloudHSM

1. Completa le fasi descritte in Nozioni di base. All'avvio del client Amazon EC2, scegli un'AMI Windows Server. In questo tutorial viene utilizzato Microsoft Windows Server 2016. Dopo aver completato queste fasi, sarà presente un cluster attivo con almeno un HSM. Hai anche un'istanza client Amazon EC2 che esegue Windows Server con il software AWS CloudHSM client per Windows installato.
2. (Facoltativo) Aggiungine altri HSMs al cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
3. Effettuare la connessione all'istanza del client. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.

4. Crea un utente crittografico (CU) utilizzando Managing HSM users with CloudHSM CLI o Managing HSM users with CloudHSM Management Utility (CMU). Prendere nota del nome utente e della password del CU, Saranno necessari per completare la fase successiva.
5. Impostare le credenziali di accesso per l'HSM, utilizzando il nome utente e la password CU creati nella fase precedente.
6. Nel passaggio 5, se hai utilizzato Windows Credentials Manager per impostare le credenziali HSM, scarica *psexec.exe*da per eseguire il comando seguente come NT Authority\ SYSTEM: SysInternals

   ```
   psexec.exe -s "C:\Program Files\Amazon\CloudHsm\tools
\set_cloudhsm_credentials.exe" --username <USERNAME> --password <PASSWORD>
   ```

   Sostituisci *<USERNAME>* e *<PASSWORD>* con le credenziali HSM.

Per creare una CA Windows Server con AWS CloudHSM, vai a. Creare un'autorità di certificazione (CA) Windows Server

Passaggio 2: creare una CA Windows Server con AWS CloudHSM

Per creare un'autorità di certificazione (CA) Windows Server, aggiungi il ruolo Active Directory Certificate Services (AD CS) al tuo Windows Server. Quando aggiungi questo ruolo, utilizzi un provider di archiviazione delle AWS CloudHSM chiavi (KSP) per creare e archiviare la chiave privata della CA nel tuo AWS CloudHSM cluster.

> **Note**
>
> Quando crei la tua CA Windows Server, puoi scegliere di creare una CA radice o una CA subordinata. La scelta dipende in genere da come la tua infrastruttura a chiave pubblica è stata progettata e dalle policy di sicurezza della tua organizzazione. Per semplicità, questo tutorial spiega come creare una CA radice.

Per aggiungere un ruolo AD CS al tuo Windows Server e creare una chiave privata CA

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.

2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager scegliere Add roles and features (Aggiungi ruoli e funzionalità).
4. Leggere le informazioni Before you begin (Prima di iniziare), quindi scegliere Next (Successivo).
5. Per Installation Type (Tipo di installazione) scegliere Role-based or feature-based installation (Installazione basata su ruoli o su funzionalità). Quindi scegli Successivo.
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Successivo.
7. Per Server Roles (Ruoli server) utilizzare la seguente procedura:
    a. Selezionare Active Directory Certificate Services.
    b. Per Add features that are required for Active Directory Certificate Services (Aggiungi le funzionalità richieste per Active Directory Certificate Services) scegliere Add Features (Aggiungi funzionalità).
    c. Selezionare Next (Successivo) per terminare la selezione dei ruoli server.
8. Per Features (Funzionalità), accettare i valori predefiniti, quindi scegliere Next (Successivo).
9. Per AD CS procedere nel seguente modo:
    a. Scegli Next (Successivo).
    b. Selezionare Certification Authority (Autorità di certificazione), quindi Next (Successivo).
10. Per Confirmation (Conferma), leggere le informazioni di conferma, quindi scegliere Install (Installa). Non chiudere la finestra.
11. Scegliere il link evidenziato Configure Active Directory Certificate Services on the destination server (Configura Active Directory Certificate Services sul server di destinazione).
12. Per Credentials (Credenziali), verificare o cambiare le credenziali visualizzate. Quindi scegli Successivo.
13. Per Role Services (Servizi ruolo), selezionare Certification Authority (Autorità di certificazione). Quindi scegli Successivo.
14. Per Setup Type (Tipo di installazione) selezionare Standalone CA (CA indipendente). Quindi scegli Successivo.
15. Per CA Type (Tipo CA) selezionare Root CA (CA radice). Quindi scegli Successivo.

> **Note**
>
> La scelta tra creare una CA radice o una CA subordinata dipende da come l'infrastruttura a chiave pubblica è stata progettata e dalle policy di sicurezza dell'organizzazione. Per semplicità, questo tutorial spiega come creare una CA radice.

16. Per Private Key (Chiave privata) selezionare Create a new private key (Crea una nuova chiave privata). Quindi scegli Successivo.
17. Per Cryptography (Crittografia) procedere nel seguente modo:

    a. Per Select a cryptographic provider (Seleziona un provider di crittografia) scegliere una delle opzioni Cavium Key Storage Provider (Provider di archiviazione chiavi Cavium) nel menu. Questi sono i provider di archiviazione chiavi AWS CloudHSM . Ad esempio, è possibile scegliere RSA#Cavium Key Storage Provider (Provider di archiviazione chiavi RSA#Cavium).

    b. Per Key length (Lunghezza chiave) scegliere una delle opzioni disponibili.

    c. Per Select the hash algorithm for signing certificates issued by this CA (Seleziona l'algoritmo hash per firmare i certificati rilasciati da questa CA) scegliere una delle opzioni di algoritmo hash disponibili.

    Scegli Next (Successivo).
18. Per CA Name (Nome CA) procedere nel seguente modo:

    a. (Opzionale) Modificare il nome comune.

    b. (Opzionale) Digitare un suffisso del nome distinto.

    Scegli Next (Successivo).
19. Per Validity Period (Periodo di validità) specificare un periodo di tempo in anni, mesi, settimane o giorni. Quindi scegli Successivo.
20. Per Certificate Database (Database certificati), è possibile accettare i valori predefiniti oppure modificare la posizione per il database e il log del database. Quindi scegli Successivo.
21. Per Confirmation (Conferma) rivedere le informazioni relative alla CA, quindi scegliere Configure (Configura).
22. Selezionare Close (Chiudi) e poi ancora Close (Chiudi).

Ora disponi di una CA Windows Server con AWS CloudHSM. Per informazioni su come firmare una richiesta di firma del certificato con la tua CA, consulta Firma di una CSR.

Passaggio 3: Firma una richiesta di firma del certificato (CSR) con la CA di Windows Server con AWS CloudHSM

Puoi utilizzare la tua CA di Windows Server con AWS CloudHSM per firmare una richiesta di firma del certificato (CSR). Per completare queste fasi, è necessaria una CSR valida. Puoi creare una CSR in diversi modi, compreso quanto segue:

- Utilizzo di OpenSSL
- Utilizzo di Windows Server Internet Information Services (IIS) Manager
- Utilizzo di snap-in di certificati in Microsoft Management Console
- Utilizzo dell'utility a riga di comando certreq su Windows

Le fasi per la creazione di una CSR non rientrano nell'ambito di questo tutorial. Quando si dispone di una CSR, è possibile firmarla con la CA Windows Server.

Per firmare una CSR con la CA Windows Server

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager, nell'angolo in alto a destra, scegliere Tools (Strumenti), Certification Authority (Autorità di certificazione).
4. Nella finestra Certification Authority (Autorità di certificazione), scegliere il nome del computer.
5. Dal menu Action (Azione), scegliere All Tasks (Tutte le attività), Submit new request (Invia nuova richiesta).
6. Selezionare il file CSR, quindi scegliere Open (Apri).
7. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic su Pending Requests (Richieste in sospeso).
8. Selezionare la richiesta in sospeso. Quindi, dal menu Action (Azione), scegliere All Tasks (Tutte le attività), Issue (Invia).
9. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic su Issued Requests (Richieste emesse) per visualizzare il certificato firmato.

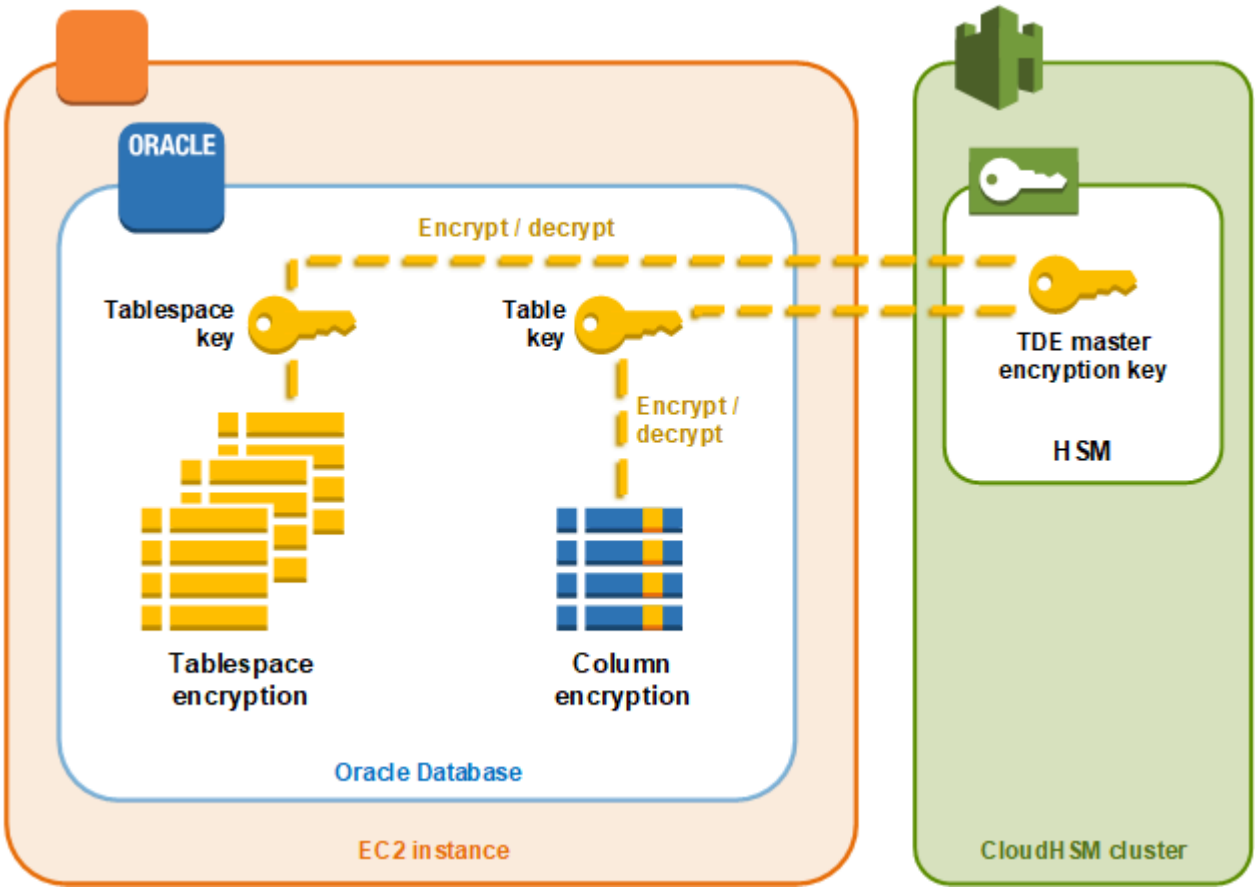
10. (Opzionale) Per esportare il certificato firmato in un file, completare i seguenti passaggi:

    a. Nella finestra Certification Authority (Autorità di certificazione), fare doppio clic sul certificato.
    b. Scegliere la scheda Details (Dettagli), quindi scegliere Copy to File (Copia su file).
    c. Seguire le istruzioni in Certificate Export Wizard (Esportazione guidata certificati).

Ora disponi di una CA di Windows Server con AWS CloudHSM e di un certificato valido firmato dalla CA di Windows Server.

Crittografia trasparente dei dati (TDE) del database Oracle con AWS CloudHSM

Transparent Data Encryption (TDE) viene utilizzato per crittografare i file di database. Utilizzando TDE, il software del database crittografa i dati prima di archiviarli su disco. I dati nelle colonne o negli spazi della tabella del database vengono crittografati con una chiave di tabella o una chiave di spazio di tabella. Alcune versioni del software del database di Oracle offrono TDE. In Oracle TDE, queste chiavi sono crittografate con una chiave di crittografia principale TDE. È possibile ottenere una maggiore sicurezza archiviando la chiave di crittografia principale TDE negli HSM del cluster. AWS CloudHSM



In questa soluzione, si utilizza Oracle Database installato su un'istanza Amazon EC2. Oracle Database si integra con la libreria software AWS CloudHSM per PKCS #11 per archiviare la chiave principale TDE negli HSM del cluster.

> **Important**
>
> - Consigliamo di installare Oracle Database su un'istanza Amazon EC2.

Completare la procedura seguente per effettuare l'integrazione di Oracle TDE con AWS CloudHSM.

Argomenti

- Passaggio 1. Imposta i prerequisiti
- Fase 2: Aggiornare la configurazione del database Oracle
- Fase 3: Generazione della chiave di crittografia principale Oracle TDE

Passaggio 1. Imposta i prerequisiti

Per realizzare l'integrazione con Oracle TDE con AWS CloudHSM, è necessario quanto segue:

- Un AWS CloudHSM cluster attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue il sistema operativo Amazon Linux con il seguente software installato:
  - Il AWS CloudHSM client e gli strumenti da riga di comando.
  - La libreria AWS CloudHSM software per PKCS #11.
  - Database Oracle. AWS CloudHSM supporta l'integrazione con Oracle TDE. Client SDK 5.6 e le versioni successive supportano Oracle TDE per Oracle Database 19c. Client SDK 3 supporta Oracle TDE per Oracle Database versione 11g e 12c.
- Un utente di crittografia (CU) che possieda e gestisca la chiave di cifratura master TDE sugli HSM nel cluster.

Completa la procedura seguente per impostare tutti i prerequisiti.

Per configurare i prerequisiti per l'integrazione di Oracle TDE con AWS CloudHSM

1. Completa le fasi descritte in Nozioni di base. Dopo aver completato questa fase, avrai un cluster attivo con un HSM. Avrai inoltre un'istanza Amazon EC2 che esegue il sistema operativo Amazon Linux. Verranno inoltre installati e configurati gli strumenti AWS CloudHSM client e da riga di comando.
2. (Facoltativo) Aggiungi altri HSM sul cluster. Per ulteriori informazioni, consulta Aggiungere un HSM a un AWS CloudHSM cluster.
3. Connettiti alla tua istanza del client Amazon EC2 ed esegui le operazioni descritte di seguito:
   a. Installa la libreria AWS CloudHSM software per PKCS #11.
   b. Installa Oracle Database. Per ulteriori informazioni, consulta la documentazione relativa a Oracle Database. Client SDK 5.6 e le versioni successive supportano Oracle TDE per Oracle Database 19c. Client SDK 3 supporta Oracle TDE per Oracle Database versione 11g e 12c.
   c. Utilizza lo strumento a riga di comando cloudhsm_mgmt_util per creare un utente di crittografia (CU) sul tuo cluster. Per ulteriori informazioni sulla creazione di un CU, consulta la pagina Come gestire gli utenti HSM con CMU e Utenti HSM.

Fase 2: Aggiornare la configurazione del database Oracle

Per aggiornare la configurazione di Oracle Database per utilizzare un HSM nel cluster come modulo di sicurezza esterno, attieniti alla seguente procedura. Per informazioni sui moduli di sicurezza esterni, vedi la sezione Introduction to Transparent Data Encryption nella Oracle Database Advanced Security Guide.

Per aggiornare la configurazione di Oracle

1. Esegui la connessione all'istanza del client Amazon EC2. Si tratta dell'istanza in cui è installato Oracle Database.
2. Crea una copia di backup del file `sqlnet.ora`. Per informazioni sul percorso del file, consulta la documentazione Oracle.
3. Usare un editor di testo per modificare il fine denominato `sqlnet.ora`. Aggiungi la seguente riga. Se una linea esistente nel file inizia con `encryption_wallet_location`, sostituiscila con quella riportata di seguito.

   ```
   encryption_wallet_location=(source=(method=hsm))
   ```

   Salvare il file.
4. Eseguire il comando seguente per creare la directory in cui Oracle Database prevede di trovare il file di libreria per la libreria software AWS CloudHSM PKCS #11.

   ```
   sudo mkdir -p /opt/oracle/extapi/64/hsm
   ```
5. Esegui il comando seguente per copiare la libreria AWS CloudHSM software per il file PKCS #11 nella directory creata nel passaggio precedente.

   ```
   sudo cp /opt/cloudhsm/lib/libcloudhsm_pkcs11.so /opt/oracle/extapi/64/hsm/
   ```

   > **Note**
   >
   > La directory `/opt/oracle/extapi/64/hsm` deve contenere solo un file della libreria. Rimuovi tutti gli altri file presenti in tale directory.
6. Esegui il seguente comando per modificare le proprietà della directory `/opt/oracle` e del relativo contenuto.

```
sudo chown -R oracle:dba /opt/oracle
```

7. Avvia Oracle Database.

Fase 3: Generazione della chiave di crittografia principale Oracle TDE

Per creare la chiave principale di Oracle TDE negli HSM del cluster, completa le fasi della seguente procedura.

Per generare la chiave principale

1. Utilizza il comando seguente per aprire Oracle SQL*Plus. Quando richiesto, immetti la password di sistema impostata al momento dell'installazione di Oracle Database.

```
sqlplus / as sysdba
```

> **Note**
>
> Per Client SDK 3 è necessario impostare la variabile di ambiente `CLOUDHSM_IGNORE_CKA_MODIFIABLE_FALSE` ogni volta che si genera una chiave principale. Questa variabile è necessaria solo per la generazione di chiavi principali. Per ulteriori informazioni, vedere «Problema: Oracle imposta l'attributo PCKS #11 `CKA_MODIFIABLE` durante la generazione della chiave master, ma l'HSM non lo supporta» in Problemi noti per l'integrazione delle applicazioni. Third-Party

2. Esegui l'istruzione SQL che crea la chiave di crittografia principale, come mostrato negli esempi di seguito. Utilizza l'istruzione corrispondente alla versione in uso di Oracle Database. Sostituire *<CU user name>* con il nome utente dell'utente crittografico (CU). Sostituire *<password>* con la password CU.

> **Important**
>
> Esegui il seguente comando solo una volta. Ogni volta che lo esegui, il comando crea una nuova chiave di crittografia principale.

- In Oracle Database versione 11, esegui la seguente istruzione SQL.

```
SQL> alter system set encryption key identified by "<CU user name>:<password>";
```

- In Oracle Database versione 12 e 19c, esegui la seguente istruzione SQL.

```
SQL> administer key management set key identified by "<CU user
 name>:<password>";
```

Se la risposta è `System altered` oppure `keystore altered`, la creazione della chiave principale di Oracle TDE è stata completata senza errori.

3. (Opzionale) Esegui il seguente comando per verificare lo stato di Oracle Wallet.

```
SQL> select * from v$encryption_wallet;
```

Se il wallet non è aperto, utilizza uno dei seguenti comandi per aprirlo. Sostituire *<CU user name>* con il nome dell'utente crittografico (CU). Sostituire *<password>* con la password CU.

- In Oracle 11, esegui il seguente comando per aprire il wallet.

```
SQL> alter system set encryption wallet open identified by "<CU user
 name>:<password>";
```

Per chiudere manualmente il wallet, esegui il seguente comando.

```
SQL> alter system set encryption wallet close identified by "<CU user
 name>:<password>";
```

- In Oracle 12 e Oracle 19c, esegui il seguente comando per aprire il wallet.

```
SQL> administer key management set keystore open identified by "<CU user
 name>:<password>";
```

Per chiudere manualmente il wallet, esegui il seguente comando.

```
SQL> administer key management set keystore close identified by "<CU user
 name>:<password>";
```

Usa Microsoft SignTool con AWS CloudHSM per firmare i file

AWS CloudHSM offre supporto per l'utilizzo di Microsoft Signtool per firmare file tramite Client SDK 3 e Client SDK 5. I passaggi per utilizzare questi strumenti variano a seconda della versione del client SDK in cui è stato attualmente effettuato il download. Le seguenti sezioni forniscono informazioni su ciascun SDK.

Argomenti

- Usa Microsoft SignTool con Client SDK 5 per firmare i file
- Usa Microsoft SignTool con Client SDK 3 per firmare i file

Usa Microsoft SignTool con Client SDK 5 per firmare i file

In crittografia e infrastruttura a chiave pubblica (PKI), le firme digitali vengono utilizzate per confermare che i dati sono stati inviati da un'entità attendibile. Le firme, inoltre, indicano che i dati non sono stati danneggiati durante il transito. Una firma è un hash crittografato generato con la chiave privata del mittente. Il destinatario può verificare l'integrità dei dati decifrando la firma hash con la chiave pubblica del mittente. In cambio, è responsabilità del mittente mantenere un certificato digitale. Il certificato digitale mostra la titolarità del mittente della chiave privata e fornisce al destinatario la chiave pubblica necessaria per la decrittografia. Finché la chiave privata è di proprietà del mittente, la firma può essere considerata attendibile. AWS CloudHSM fornisce hardware sicuro con convalida FIPS 140-2 di livello 3 per proteggere queste chiavi con un accesso esclusivo single-tenant.

Molte organizzazioni utilizzano Microsoft SignTool, uno strumento da riga di comando che firma, verifica e marca temporale i file per semplificare il processo di firma del codice. È possibile utilizzarlo per AWS CloudHSM archiviare in modo sicuro le coppie di chiavi fino al momento in cui sono necessarie SignTool, creando così un flusso di lavoro facilmente automatizzabile per la firma dei dati.

I seguenti argomenti forniscono una panoramica sull'utilizzo SignTool di. AWS CloudHSM

Argomenti

- Fase 1: configurazione dei prerequisiti
- Fase 2: Creare un certificato di firma
- Fase 3: Firmare un file

Fase 1: configurazione dei prerequisiti

Per utilizzare Microsoft SignTool con AWS CloudHSM, è necessario quanto segue:

- Un'istanza del client Amazon EC2 in esecuzione su un sistema operativo Windows.
- Un'autorità di certificazione (CA), o mantenuta in modo autonomo o istituita da un fornitore esterno.
- Un AWS CloudHSM cluster attivo nello stesso cloud pubblico virtuale (VPC) dell'istanza EC2. Il cluster deve contenere almeno un HSM.
- Un utente crittografico (CU) che possiede e gestisce le chiavi nel cluster. AWS CloudHSM
- Un file non firmato o eseguibile.
- Microsoft Windows Software Development Kit (SDK).

Per impostare i prerequisiti per l'utilizzo AWS CloudHSM con Windows SignTool

1. Segui le istruzioni nella sezione Nozioni di base di questa guida per avviare un'istanza di EC2 Windows e un cluster AWS CloudHSM .
2. Se desideri ospitare la tua CA di Windows Server, segui i passaggi 1 e 2 in Configurazione di Windows Server come autorità di certificazione con AWS CloudHSM. Altrimenti, continua a utilizzare la tua CA di terze parti pubblicamente attendibile.
3. Scaricare e installare una delle seguenti versioni di Microsoft Windows SDK sull'istanza Windows EC2:

   - Microsoft Windows SDK 10
   - Microsoft Windows SDK 8.1
   - Microsoft Windows SDK 7

   L' eseguibile `SignTool` fa parte di Windows SDK Signing Tools per la funzione di installazione delle app desktop. È possibile omettere l'installazione di altre funzionalità, se non è necessario. Il percorso di installazione predefinito è:

   ```
   C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\<CPU
    architecture>\signtool.exe
   ```

Ora puoi usare Microsoft Windows SDK, il tuo AWS CloudHSM cluster e la tua CA per creare un certificato di firma.

Fase 2: Creare un certificato di firma

Ora che hai scaricato l'SDK Windows SDK nell'istanza di EC2, puoi utilizzarlo per generare una richiesta di firma del certificato (CSR). CSR è un certificato non firmato che viene alla fine passato al CA per la procedura di firma. In questo esempio, si utilizza l'eseguibile `certreq` incluso con l'SDK Windows SDK per generare il CSR.

Per generare una CSR utilizzando l'eseguibile certreq

1. Effettua la connessione all'istanza EC2 Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Creare un file denominato `request.inf` che contiene le righe qui di seguito. Sostituisci le informazioni `Subject` con quelle della tua organizzazione. Per una spiegazione di ciascun parametro, consulta la documentazione di Microsoft.

   ```
   [Version]
   Signature= $Windows NT$
   [NewRequest]
   Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-Unit>,L=<City>,S=<State>"
   RequestType=PKCS10
   HashAlgorithm = SHA256
   KeyAlgorithm = RSA
   KeyLength = 2048
   ProviderName = "CloudHSM Key Storage Provider"
   KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
   MachineKeySet = True
   Exportable = False
   ```

3. Esegui `certreq.exe`. Per questo esempio, abbiamo salvato il CSR come `request.csr`.

   ```
   certreq.exe -new request.inf request.csr
   ```

   Internamente, viene generata una nuova coppia di chiavi sul AWS CloudHSM cluster e la chiave privata della coppia viene utilizzata per creare la CSR.
4. Invia la CSR alla CA. Se stai usando un Windows Server CA, procedi nel seguente modo:

   a. Inserire il comando seguente per aprire lo strumento CA:

      ```
      certsrv.msc
      ```

b. Nella nuova finestra, fare clic con il pulsante destro del mouse sul nome del server CA. Scegliere All tasks (Tutte le attività), quindi scegliere Submit new request (Sottometti nuova richiesta).

c. Accedere all'ubicazione d `request.csr` e scegliere Open (Apri).

d. Passa alla cartella Richieste in sospeso espandendo il menu Server CA. Right-click sulla richiesta appena creata e in Tutte le attività scegli Problema.

e. Ora passare alla cartella Issued Certificates (Certificati emessi) (sopra la cartella Pending Requests (Richieste in sospeso)).

f. Scegliere Apri per visualizzare il certificato, quindi scegliere la schedaDetails (Dettagli).

g. Scegliere Copy to File (Copia su File) per avviare la procedura guidata di esportazione dei certificati. Salva il DER-encoded X.509 file in una posizione sicura come`signedCertificate.cer`.

h. Esci dallo strumento CA e utilizza il comando seguente, che consente di spostare il file del certificato al Personal Certificate Store in Windows. A questo punto, può essere utilizzato da altre applicazioni.

```
certreq.exe -accept signedCertificate.cer
```

Ora puoi utilizzare il tuo certificato importato per Firmare un file .

Fase 3: Firmare un file

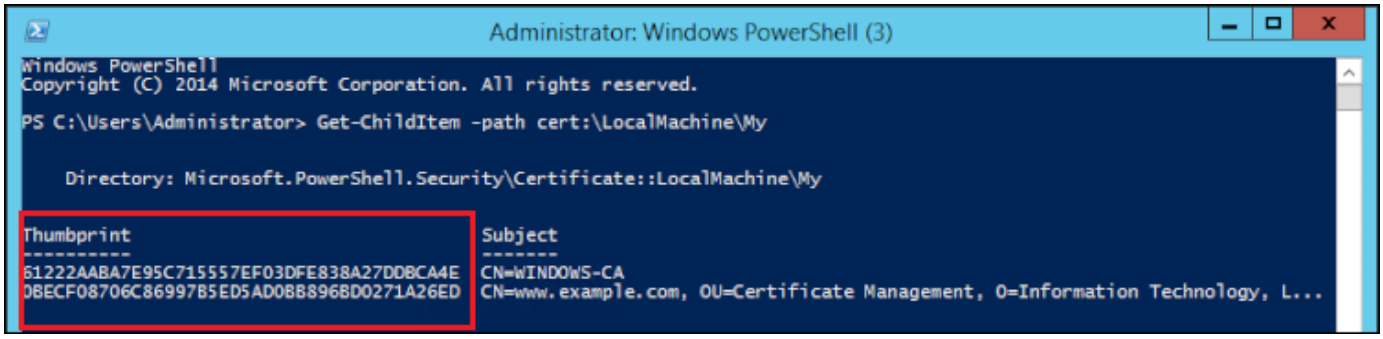
Ora sei pronto per l'uso SignTool e il certificato importato per firmare il tuo file di esempio. Per fare ciò, devi conoscere l' SHA-1 hash o l'impronta digitale del certificato. L'impronta digitale viene utilizzata per garantire che vengano utilizzati SignTool solo certificati verificati da. AWS CloudHSM In questo esempio, lo usiamo PowerShell per ottenere l'hash del certificato. È inoltre possibile utilizzare l'interfaccia utente grafica del CA o l'SDK di Windows `certutil` eseguibile.

Per ottenere l'identificazione personale del certificato e utilizzarlo per firmare un file

1. Apri PowerShell come amministratore ed esegui il seguente comando:

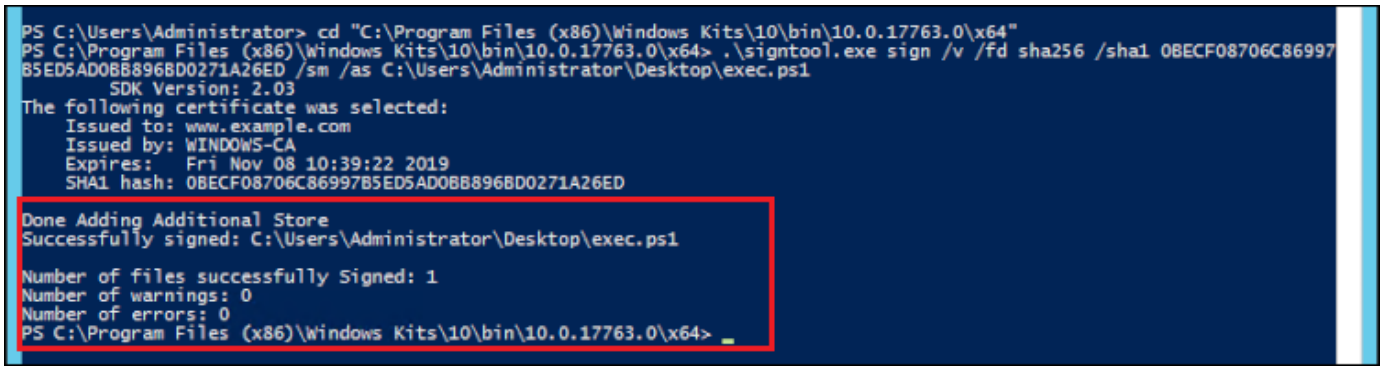
```
Get-ChildItem -path cert:\LocalMachine\My
```

Copia `Thumbprint` che viene restituito.



2. Passa alla directory all'interno della PowerShell quale è contenutoSignTool.exe. Il percorso predefinito è C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64.
3. Infine, firmare il file eseguendo il seguente comando. Se il comando ha esito positivo, PowerShell restituisce un messaggio di successo.

```
signtool.exe sign /v /fd sha256 /sha1 <thumbprint> /sm C:\Users\Administrator
\Desktop\<test>.ps1
```



4. (facoltativo) Per verificare la firma sul file, utilizzare il comando seguente:

```
signtool.exe verify /v /pa C:\Users\Administrator\Desktop\<test>.ps1
```

Usa Microsoft SignTool con Client SDK 3 per firmare i file

In crittografia e infrastruttura a chiave pubblica (PKI), le firme digitali vengono utilizzate per confermare che i dati sono stati inviati da un'entità attendibile. Le firme, inoltre, indicano che i dati non sono stati danneggiati durante il transito. Una firma è un hash crittografato generato con la chiave privata del mittente. Il destinatario può verificare l'integrità dei dati decifrando la firma hash con la chiave pubblica del mittente. In cambio, è responsabilità del mittente mantenere un certificato digitale. Il certificato digitale mostra la titolarità del mittente della chiave privata e fornisce al destinatario la

chiave pubblica necessaria per la decrittografia. Finché la chiave privata è di proprietà del mittente, la firma può essere considerata attendibile. AWS CloudHSM fornisce hardware sicuro con convalida FIPS 140-2 di livello 3 per proteggere queste chiavi con un accesso esclusivo single-tenant.

Molte organizzazioni utilizzano Microsoft SignTool, uno strumento da riga di comando che firma, verifica e marca temporale i file per semplificare il processo di firma del codice. È possibile utilizzarlo per AWS CloudHSM archiviare in modo sicuro le coppie di chiavi fino al momento in cui sono necessarie SignTool, creando così un flusso di lavoro facilmente automatizzabile per la firma dei dati.

I seguenti argomenti forniscono una panoramica sull'utilizzo SignTool di. AWS CloudHSM

Argomenti

- Fase 1: configurazione dei prerequisiti
- Fase 2: Creare un certificato di firma
- Fase 3: Firmare un file

Fase 1: configurazione dei prerequisiti

Per utilizzare Microsoft SignTool con AWS CloudHSM, è necessario quanto segue:

- Un'istanza del client Amazon EC2 in esecuzione su un sistema operativo Windows.
- Un'autorità di certificazione (CA), o mantenuta in modo autonomo o istituita da un fornitore esterno.
- Un AWS CloudHSM cluster attivo nello stesso cloud pubblico virtuale (VPC) dell'istanza EC2. Il cluster deve contenere almeno un HSM.
- Un utente crittografico (CU) che possiede e gestisce le chiavi nel cluster. AWS CloudHSM
- Un file non firmato o eseguibile.
- Microsoft Windows Software Development Kit (SDK).

Per impostare i prerequisiti per l'utilizzo AWS CloudHSM con Windows SignTool

1. Segui le istruzioni nella sezione Nozioni di base di questa guida per avviare un'istanza di EC2 Windows e un cluster AWS CloudHSM .
2. Se desideri ospitare la tua CA di Windows Server, segui i passaggi 1 e 2 in Configurazione di Windows Server come autorità di certificazione con AWS CloudHSM. Altrimenti, continua a utilizzare la tua CA di terze parti pubblicamente attendibile.

3. Scaricare e installare una delle seguenti versioni di Microsoft Windows SDK sull'istanza Windows EC2:

   - Microsoft Windows SDK 10
   - Microsoft Windows SDK 8.1
   - Microsoft Windows SDK 7

   L' eseguibile `SignTool` fa parte di Windows SDK Signing Tools per la funzione di installazione delle app desktop. È possibile omettere l'installazione di altre funzionalità, se non è necessario. Il percorso di installazione predefinito è:

   ```
   C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\<CPU
    architecture>\signtool.exe
   ```

Ora puoi usare Microsoft Windows SDK, il tuo AWS CloudHSM cluster e la tua CA per creare un certificato di firma.

Fase 2: Creare un certificato di firma

Ora che hai scaricato l'SDK Windows SDK nell'istanza di EC2, puoi utilizzarlo per generare una richiesta di firma del certificato (CSR). CSR è un certificato non firmato che viene alla fine passato al CA per la procedura di firma. In questo esempio, si utilizza l'eseguibile `certreq` incluso con l'SDK Windows SDK per generare il CSR.

Per generare una CSR utilizzando l'eseguibile certreq

1. Effettua la connessione all'istanza EC2 Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Creare un file denominato `request.inf` che contiene le righe qui di seguito. Sostituisci le informazioni `Subject` con quelle della tua organizzazione. Per una spiegazione di ciascun parametro, consulta la documentazione di Microsoft.

   ```
   [Version]
   Signature= $Windows NT$
   [NewRequest]
   Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-
   Unit>,L=<City>,S=<State>"
   RequestType=PKCS10
   ```

```
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "Cavium Key Storage Provider"
KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
MachineKeySet = True
Exportable = False
```

3. Esegui `certreq.exe`. Per questo esempio, abbiamo salvato il CSR come `request.csr`.

```
certreq.exe -new request.inf request.csr
```

Internamente, viene generata una nuova coppia di chiavi sul AWS CloudHSM cluster e la chiave privata della coppia viene utilizzata per creare la CSR.

4. Invia la CSR alla CA. Se stai usando un Windows Server CA, procedi nel seguente modo:

   a. Inserire il comando seguente per aprire lo strumento CA:

   ```
   certsrv.msc
   ```

   b. Nella nuova finestra, fare clic con il pulsante destro del mouse sul nome del server CA. Scegliere All tasks (Tutte le attività), quindi scegliere Submit new request (Sottometti nuova richiesta).

   c. Accedere all'ubicazione d `request.csr` e scegliere Open (Apri).

   d. Passa alla cartella Richieste in sospeso espandendo il menu Server CA. Right-click sulla richiesta appena creata e in Tutte le attività scegli Problema.

   e. Ora passare alla cartella Issued Certificates (Certificati emessi) (sopra la cartella Pending Requests (Richieste in sospeso)).

   f. Scegliere Apri per visualizzare il certificato, quindi scegliere la schedaDetails (Dettagli).

   g. Scegliere Copy to File (Copia su File) per avviare la procedura guidata di esportazione dei certificati. Salva il DER-encoded X.509 file in una posizione sicura come`signedCertificate.cer`.

   h. Esci dallo strumento CA e utilizza il comando seguente, che consente di spostare il file del certificato al Personal Certificate Store in Windows. A questo punto, può essere utilizzato da altre applicazioni.

   ```
   certreq.exe -accept signedCertificate.cer
   ```

Ora puoi utilizzare il tuo certificato importato per Firmare un file .

Fase 3: Firmare un file

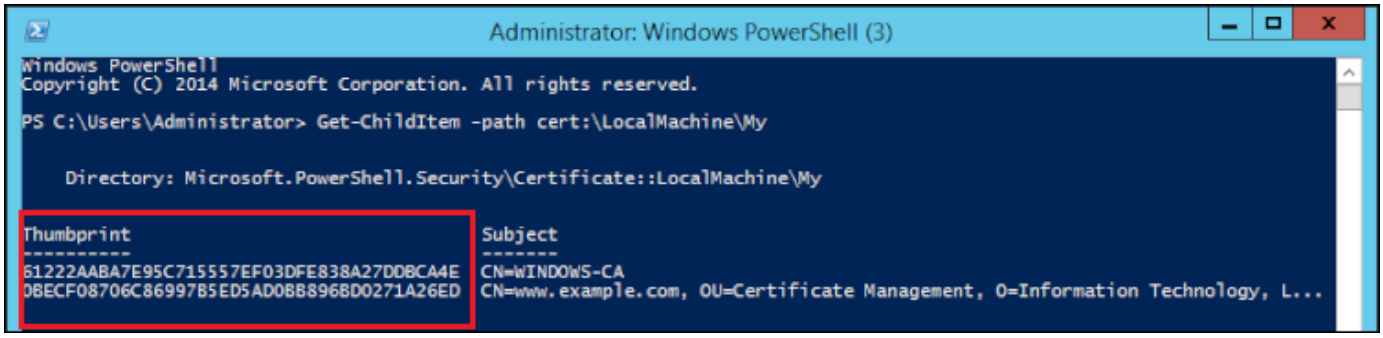
Ora sei pronto per l'uso SignTool e il certificato importato per firmare il tuo file di esempio. Per fare ciò, devi conoscere l' SHA-1 hash o l'impronta digitale del certificato. L'impronta digitale viene utilizzata per garantire che vengano utilizzati SignTool solo certificati verificati da. AWS CloudHSM In questo esempio, lo usiamo PowerShell per ottenere l'hash del certificato. È inoltre possibile utilizzare l'interfaccia utente grafica del CA o l'SDK di Windows `certutil` eseguibile.

Per ottenere l'identificazione personale del certificato e utilizzarlo per firmare un file

1. Apri PowerShell come amministratore ed esegui il seguente comando:

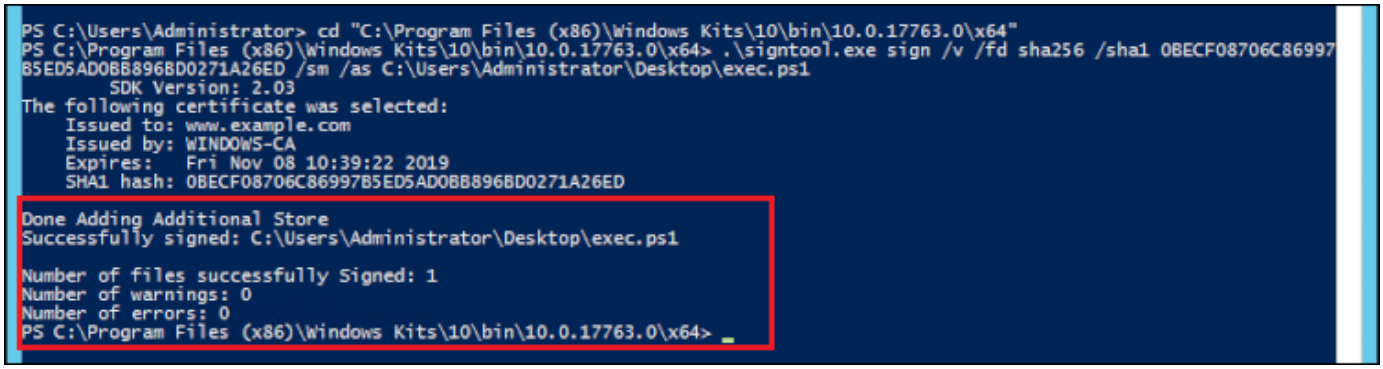
   ```
   Get-ChildItem -path cert:\LocalMachine\My
   ```

   Copia `Thumbprint` che viene restituito.

2. Passa alla directory all'interno della PowerShell quale è contenuto`SignTool.exe`. Il percorso predefinito è `C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x64`.
3. Infine, firmare il file eseguendo il seguente comando. Se il comando ha esito positivo, PowerShell restituisce un messaggio di successo.

   ```
   signtool.exe sign /v /fd sha256 /sha1 <thumbprint> /sm C:\Users\Administrator
   \Desktop\<test>.ps1
   ```



4. (facoltativo) Per verificare la firma sul file, utilizzare il comando seguente:

```
signtool.exe verify /v /pa C:\Users\Administrator\Desktop\<test>.ps1
```

Integrazione di Java Keytool e Jarsigner con AWS CloudHSM

AWS CloudHSM offre l'integrazione con le utilità Java Keytool e Jarsigner tramite Client SDK 3 e Client SDK 5. I passaggi per utilizzare questi strumenti variano a seconda della versione del client SDK in cui è attualmente stato scaricato. Le seguenti sezioni forniscono informazioni su ciascun SDK.

Argomenti

- Usa Client SDK 5 per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner
- Usa Client SDK 3 per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner

Usa Client SDK 5 per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner

AWS CloudHSM key store è un archivio di chiavi JCE per scopi speciali che utilizza i certificati associati alle chiavi del modulo di sicurezza hardware (HSM) tramite strumenti di terze parti come e. `keytool` `jarsigner` AWS CloudHSM non memorizza i certificati sull'HSM, in quanto i certificati sono dati pubblici e non riservati. Il AWS CloudHSM key store memorizza i certificati in un file locale e li associa alle chiavi corrispondenti sull'HSM.

Quando si utilizza l'archivio AWS CloudHSM chiavi per generare nuove chiavi, non viene generata alcuna voce nel file di archivio delle chiavi locale: le chiavi vengono create sull'HSM. Allo stesso modo, quando si utilizza l'archivio chiavi AWS CloudHSM per cercare le chiavi, la ricerca viene passata all'HSM. Quando si archiviano i certificati nel AWS CloudHSM key store, il provider verifica

che nell'HSM esista una coppia di chiavi con l'alias corrispondente e quindi associa il certificato fornito alla coppia di chiavi corrispondente.

Argomenti

- Prerequisiti per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner utilizzando Client SDK 5
- Utilizzo AWS CloudHSM archivio di chiavi con keytool utilizzando Client SDK 5
- Utilizzo AWS CloudHSM archivio chiavi con Jarsigner utilizzando Client SDK 5
- Problemi noti per AWS CloudHSM integrazione Java Keytool e Jarsigner tramite Client SDK 5

Prerequisiti per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner utilizzando Client SDK 5

Per utilizzare il AWS CloudHSM key store, è necessario prima inizializzare e configurare JCE SDK. AWS CloudHSM Utilizza le fasi seguenti per farlo.

Fase 1: Installare JCE

Per installare JCE, inclusi i prerequisiti del AWS CloudHSM client, segui i passaggi per l'installazione della libreria Java.

Passaggio 2: aggiungere le credenziali di accesso HSM alle variabili di ambiente

Imposta le variabili di ambiente per contenere le credenziali di accesso all'HSM.

Linux

```
$ export HSM_USER=<HSM user name>
```

```
$ export HSM_PASSWORD=<HSM password>
```

Windows

```
PS C:\> $Env:HSM_USER=<HSM user name>
```

```
PS C:\> $Env:HSM_PASSWORD=<HSM password>
```

> **Note**
>
> Il AWS CloudHSM JCE offre diverse opzioni di accesso. Per utilizzare il AWS CloudHSM key store con applicazioni di terze parti, è necessario utilizzare l'accesso implicito con variabili di ambiente. Se desideri utilizzare l'accesso esplicito tramite il codice dell'applicazione, devi creare la tua applicazione utilizzando il AWS CloudHSM key store. Per ulteriori informazioni, consulta l'articolo sull'uso di AWS CloudHSM Key Store.

Passaggio 3: Registrazione del provider JCE

Per registrare il provider JCE nella CloudProvider configurazione Java, seguite questi passaggi:

1. Apri il file di configurazione `java.security` nell'installazione Java, per la modifica.
2. Nel file di configurazione `java.security`, aggiungi `com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider` come ultimo provider. Ad esempio, se sono presenti nove provider nel file `java.security`, aggiungi il seguente provider come ultimo provider nella sezione.

   ```
   security.provider.10=com.amazonaws.cloudhsm.jce.provider.CloudHsmProvider
   ```

> **Note**
>
> L'aggiunta del AWS CloudHSM provider come priorità più alta può influire negativamente sulle prestazioni del sistema, in quanto a quest'ultimo verrà assegnata la AWS CloudHSM priorità per le operazioni che potrebbero essere trasferite in tutta sicurezza al software. Come procedura consigliata, specificate sempre il provider che desiderate utilizzare per un'operazione, sia esso un provider basato su software AWS CloudHSM .

> **Note**
>
> La specificazione e `-providerName` le `-providerclass` opzioni della riga di `-providerpath` comando durante la generazione delle chiavi utilizzando keytool il key store di AWS CloudHSM può causare errori.

Utilizzo AWS CloudHSM archivio di chiavi con keytool utilizzando Client SDK 5

Keytool è una utility a riga di comando molto usata per le attività comuni di chiavi e certificati sui sistemi Linux. Nella documentazione AWS CloudHSM non è incluso un tutorial completo su keytool. Questo articolo spiega i parametri specifici da utilizzare con le varie funzioni di keytool quando si utilizza il key store AWS CloudHSM come root of trust. AWS CloudHSM

Quando usi keytool con il AWS CloudHSM key store, specifica i seguenti argomenti per qualsiasi comando keytool:

> **ML-DSA e keytool**
>
> Keytoolsupporta ML-DSA in JDK 24 o versioni successive.
> I seguenti comandi keytool sono supportati con i ML-DSA tasti:
>
> - -certreq— Genera una richiesta di firma del certificato per una ML-DSA chiave.
> - -importcert— Importazione di un certificato per una ML-DSA chiave.
> - -exportcert— Esporta un certificato per una ML-DSA chiave.
>
> Il comando seguente non è supportato con ML-DSA le chiavi:
>
> - -genkeypair— AWS CloudHSM non supporta l'importazione di ML-DSA chiavi e keytool genera le chiavi nel software prima dell'importazione. Usa `KeyPairGenerator` la nostra CLI CloudHSM per generare coppie di chiavi. ML-DSA

Linux

```
-storetype CLOUDHSM -J-classpath< '-J/opt/cloudhsm/java/*'>
```

Windows

```
-storetype CLOUDHSM -J-classpath<'-J"C:\Program Files\Amazon\CloudHSM\java\*"'>
```

Se desideri creare un nuovo file di key store utilizzando AWS CloudHSM key store, consulta. Usa for Client SDK 3 AWS CloudHSM KeyStore AWS CloudHSM Per utilizzare un archivio di chiavi esistente, specificane il nome (incluso il percorso) utilizzando l'argomento -keystore per keytool. Se

si specifica un file key store inesistente in un comando keytool, il AWS CloudHSM key store crea un nuovo file key store.

Crea un nuovo AWS CloudHSM chiavi con keytool

È possibile utilizzare keytool per generare i tipi di chiave RSA, EC (Ed25519), AES e DeSede supportati da JCE SDK. AWS CloudHSM

> **Important**
>
> Una chiave generata tramite keytool viene generata nel software e quindi importata come chiave estraibile e persistente. AWS CloudHSM

> **ML-DSA generazione di chiavi**
>
> AWS CloudHSM non supporta la generazione di ML-DSA chiavi tramite. keytool -genkeypair Usa `KeyPairGenerator` o l'interfaccia a riga di comando CloudHSM per generare coppie di ML-DSA chiavi e archiviarle in CloudHSM utilizzando. KeyStore `KeyStore.setKeyEntry()` Per un esempio, consulta il sito Web. MldsaKeyStoreExampleRunner GitHub

Ti consigliamo vivamente di generare chiavi non esportabili al di fuori del keytool e quindi di importare i certificati corrispondenti nell'archivio chiavi. Se si utilizzano chiavi RSA o EC estraibili tramite keytool e Jarsigner, i provider esportano le chiavi da AWS CloudHSM e quindi utilizzano la chiave localmente per le operazioni di firma.

Se hai più istanze client connesse al tuo AWS CloudHSM cluster, tieni presente che l'importazione di un certificato nel key store di un'istanza client non renderà automaticamente disponibili i certificati su altre istanze client. Per registrare la chiave e i certificati associati su ogni istanza del client è necessario eseguire un'applicazione Java come descritto in the section called "Generare una CSR usando Keytool". In alternativa, è possibile apportare le modifiche necessarie su un client e copiare il file dell'archivio delle chiavi risultante in ogni altra istanza del client.

Esempio 1: generare una chiave simmetrica e salvarla in un file di AES-256 key store denominato «example_keystore.store», nella directory di lavoro. Sostituisci con un'etichetta univoca. *<secret label>*

Linux

```
$ keytool -genseckey -alias <secret label> -keyalg aes \
 -keysize 256 -keystore example_keystore.store \
 -storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
```

Windows

```
PS C:\> keytool -genseckey -alias <secret label> -keyalg aes `
 -keysize 256 -keystore example_keystore.store `
 -storetype CloudHSM -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Esempio 2: generare una coppia di chiavi RSA 2048 e salvarla in un file di archivio di chiavi denominato «example_keystore.store» nella directory di lavoro. Sostituisci con un'etichetta univoca. *<RSA key pair label>*

Linux

```
$ keytool -genkeypair -alias <RSA key pair label> \
 -keyalg rsa -keysize 2048 \
 -sigalg sha512withrsa \
 -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -genkeypair -alias <RSA key pair label> `
 -keyalg rsa -keysize 2048 `
 -sigalg sha512withrsa `
 -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

È possibile trovare un elenco di algoritmi di firma supportati nella libreria Java.

Eliminare un AWS CloudHSM chiave usando keytool

Il AWS CloudHSM key store non supporta l'eliminazione delle chiavi. È possibile eliminare le chiavi utilizzando il metodo destrot dell'interfaccia Distruggibile.

```
((Destroyable) key).destroy();
```

Genera un AWS CloudHSM CSR utilizzando keytool

Ottieni la massima flessibilità nella generazione di una richiesta di firma del certificato (CSR) se utilizzi OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5. Il comando seguente utilizza keytool per generare un CSR per una coppia di chiavi con l'alias, `example-key-pair`.

Linux

```
$ keytool -certreq -alias <key pair label> \
 -file my_csr.csr \
 -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -certreq -alias <key pair label> `
 -file my_csr.csr `
 -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

> **Note**
>
> Per utilizzare una coppia di chiavi da keytool, tale coppia di chiavi deve avere una voce nel file dell'archivio chiavi specificato. Se si desidera utilizzare una coppia di chiavi generata al di fuori di keytool, è necessario importare i metadati della chiave e del certificato nell'archivio chiavi. Per istruzioni sull'importazione dei dati dell'archivio chiavi, vedi the section called "Usa keytool per importare i certificati nel key store ".

Usa keytool per importare certificati intermedi e principali in AWS CloudHSM archivio chiavi

Per importare un certificato CA in AWS CloudHSM, è necessario abilitare la verifica di una catena completa di certificati su un certificato appena importato. Il comando seguente mostra un esempio.

Linux

```
$ keytool -import -trustcacerts -alias rootCAcert \
 -file rootCAcert.cert -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -import -trustcacerts -alias rootCAcert `
 -file rootCAcert.cert -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se connetti più istanze client al AWS CloudHSM cluster, l'importazione di un certificato nel key store di un'istanza client non renderà automaticamente disponibile il certificato su altre istanze client. È necessario importare il certificato in ogni istanza del client.

Usa keytool per eliminare i certificati da AWS CloudHSM archivio chiavi

Il comando seguente mostra un esempio di come eliminare un AWS CloudHSM certificato da un key store Java keytool.

Linux

```
$ keytool -delete -alias mydomain \
 -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -delete -alias mydomain `
 -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se si connettono più istanze client al AWS CloudHSM cluster, l'eliminazione di un certificato nell'archivio chiavi di un'istanza client non rimuoverà automaticamente il certificato dalle altre istanze client. È necessario eliminare il certificato su ogni istanza del client.

Importa un certificato funzionante in AWS CloudHSM archivio di chiavi utilizzando keytool

Una volta firmata una richiesta di firma del certificato (CSR), è possibile importarla nell'archivio chiavi AWS CloudHSM e associarla alla coppia di chiavi appropriata. Il seguente comando fornisce un esempio.

Linux

```
$ keytool -importcert -noprompt -alias <key pair label> \
 -file my_certificate.crt \
 -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -importcert -noprompt -alias <key pair label> `
 -file my_certificate.crt `
 -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

L'alias deve essere una coppia di chiavi con un certificato associato nell'archivio chiavi. Se la chiave viene generata al di fuori di keytool o viene generata in un'istanza del client diversa, è necessario prima importare i metadati della chiave e del certificato nell'archivio chiavi.

La catena di certificati deve essere verificabile. Se non è possibile verificare il certificato, potrebbe essere necessario importare il certificato di firma (autorità di certificazione) nell'archivio chiavi in modo che la catena possa essere verificata.

Esporta un certificato da AWS CloudHSM usando keytool

L'esempio seguente genera un certificato in X.509 formato binario. Per esportare un certificato leggibile dall'uomo da AWS CloudHSM, aggiungilo `-rfc` al `-exportcert` comando.

Linux

```
$ keytool -exportcert -alias <key pair label> \
 -file my_exported_certificate.crt \
 -keystore example_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -exportcert -alias <key pair label> `
 -file my_exported_certificate.crt `
 -keystore example_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Utilizzo AWS CloudHSM archivio chiavi con Jarsigner utilizzando Client SDK 5

Jarsigner è una popolare utilità a riga di comando per firmare file JAR utilizzando una chiave archiviata in modo sicuro su un modulo di sicurezza hardware (HSM). Un tutorial completo su Jarsigner non rientra nell'ambito della documentazione. AWS CloudHSM Questa sezione spiega i parametri di Jarsigner da utilizzare per firmare e verificare le firme con AWS CloudHSM come root of trust tramite il key store. AWS CloudHSM

> **ML-DSA requisiti jarsigner**
>
> ML-DSA gli algoritmi di firma (ad esempio`-sigalg ML-DSA-44`) richiedono JDK 26 o versioni successive se utilizzati con. jarsigner

Configurazione AWS CloudHSM chiavi e certificati con Jarsigner

Prima di poter firmare i file AWS CloudHSM JAR con Jarsigner, assicurati di aver impostato o completato i seguenti passaggi:

1. Segui le indicazioni contenute nei prerequisiti dell'archivio chiavi AWS CloudHSM.
2. Configura le chiavi di firma, i certificati e la catena di certificati associati che devono essere archiviati nell'archivio AWS CloudHSM chiavi dell'istanza corrente del server o del client. Crea le chiavi su AWS CloudHSM e poi importa i metadati associati nel tuo AWS CloudHSM key store. Se

desideri utilizzare keytool per impostare le chiavi e i certificati, vedi the section called "Crea nuove chiavi con Keytool". Se utilizzi più istanze del client per firmare i JAR, crearla chiave e importa la catena di certificati. Quindi copia il file dell'archivio delle chiavi risultante in ogni istanza del client. Se generi frequentemente nuove chiavi, è possibile che sia più semplice importare singolarmente i certificati in ogni istanza client.

3. L'intera catena di certificati dovrebbe essere verificabile. Affinché la catena di certificati sia verificabile, potrebbe essere necessario aggiungere il certificato CA e i certificati intermedi al key store. AWS CloudHSM Vedere lo snippet di codice in the section called "Firma un file JAR" per istruzione su come utilizzare il codice Java per verificare la catena di certificati. Se preferisci, puoi utilizzare keytool per importare i certificati. Per istruzioni su come usare keytool, vedi the section called "Usa keytool per importare i certificati nel key store ".

Firma un file JAR utilizzando AWS CloudHSM e Jarsigner

Firma e verifica con chiavi RSA o ECDSA

Usa il seguente comando per firmare un file JAR usando una chiave RSA o ECDSA con e: AWS CloudHSM jarsigner

Linux;

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -keystore example_keystore.store \
 -signedjar signthisclass_signed.jar \
 -sigalg sha512withrsa \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass.jar <key pair label>
```

Windows

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -keystore example_keystore.store `
 -signedjar signthisclass_signed.jar `
 -sigalg sha512withrsa `
 -storetype CloudHSM `
```

```
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'` 
  "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" ` 
 signthisclass.jar <key pair label>
```

Utilizza il seguente comando per verificare un file JAR firmato:

Linux

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -verify \
 -keystore example_keystore.store \
 -sigalg sha512withrsa \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass_signed.jar <key pair label>
```

Windows

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -verify `
 -keystore example_keystore.store `
 -sigalg sha512withrsa `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'` 
 "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" ` 
 signthisclass_signed.jar <key pair label>
```

Firma e ML-DSA verifica con le chiavi

Usa il comando seguente per firmare un file JAR usando una ML-DSA chiave con AWS CloudHSM ejarsigner:

Linux

Per OpenJDK 26 o versioni successive

```
jarsigner -keystore example_keystore.store \
 -signedjar signthisclass_signed.jar \
 -sigalg ML-DSA-44 \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass.jar <key pair label>
```

Windows

Per OpenJDK 26 o versioni successive

```
jarsigner -keystore example_keystore.store `
 -signedjar signthisclass_signed.jar `
 -sigalg ML-DSA-44 `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'`
  "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass.jar <key pair label>
```

Usa il comando seguente per verificare un JAR firmato con una ML-DSA chiave:

Linux

Per OpenJDK 26 o versioni successive

```
jarsigner -verify \
 -keystore example_keystore.store \
 -sigalg ML-DSA-44 \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass_signed.jar <key pair label>
```

Windows

Per OpenJDK 26 o versioni successive

```
jarsigner -verify `
```

```
 -keystore example_keystore.store `
 -sigalg ML-DSA-44 `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'`
 "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass_signed.jar <key pair label>
```

Firma e verifica con le chiavi Ed25519

Usa il seguente comando per firmare un file JAR usando una chiave Ed25519 con e: AWS CloudHSM jarsigner

Linux

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -keystore example_keystore.store \
 -signedjar signthisclass_signed.jar \
 -sigalg Ed25519 \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass.jar <key pair label>
```

Windows

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -keystore example_keystore.store `
 -signedjar signthisclass_signed.jar `
 -sigalg Ed25519 `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'`
  "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass.jar <key pair label>
```

Usa il seguente comando per verificare un JAR firmato con una chiave Ed25519:

Linux

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -verify \
 -keystore example_keystore.store \
 -sigalg Ed25519 \
 -storetype CloudHSM \
 -J-classpath '-J/opt/cloudhsm/java/*' \
 -J-Djava.library.path=/opt/cloudhsm/lib \
 signthisclass_signed.jar <key pair label>
```

Windows

Per OpenJDK 17, OpenJDK 21 e OpenJDK 25

```
jarsigner -verify `
 -keystore example_keystore.store `
 -sigalg Ed25519 `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*'`
 "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass_signed.jar <key pair label>
```

Problemi noti per AWS CloudHSM integrazione Java Keytool e Jarsigner tramite Client SDK 5

L'elenco seguente fornisce l'elenco attuale dei problemi noti relativi alle integrazioni con Java Keytool AWS CloudHSM e Jarsigner utilizzando Client SDK 5.

1. Non supportiamo chiavi EC non ED25519 con Keytool e Jarsigner.
2. Non supportiamo la generazione di chiavi tramite keytool. ML-DSA Utilizza la `KeyPairGenerator` nostra CLI CloudHSM per generare coppie di chiavi. ML-DSA
3. ML-DSA algoritmi di firma con jarsigner require JDK 26 o successivo.

Usa Client SDK 3 per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner

AWS CloudHSM key store è un archivio di chiavi JCE per scopi speciali che utilizza certificati associati alle chiavi del modulo di sicurezza hardware (HSM) tramite strumenti di terze parti come e. `keytool` `jarsigner` AWS CloudHSM non archivia i certificati sull'HSM, poiché i certificati sono dati pubblici e non riservati. L'archivio delle AWS CloudHSM chiavi archivia i certificati in un file locale e li associa alle chiavi corrispondenti sull'HSM.

Quando si utilizza l'archivio AWS CloudHSM chiavi per generare nuove chiavi, non viene generata alcuna voce nel file dell'archivio chiavi locale: le chiavi vengono create sull'HSM. Allo stesso modo, quando si utilizza l'archivio chiavi AWS CloudHSM per cercare le chiavi, la ricerca viene passata all'HSM. Quando si archiviano i certificati nell'archivio delle AWS CloudHSM chiavi, il provider verifica che esista una coppia di chiavi con l'alias corrispondente sull'HSM, quindi associa il certificato fornito alla coppia di chiavi corrispondente.

Argomenti

- Prerequisiti per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner utilizzando Client SDK 3
- Usa l'archivio AWS CloudHSM delle chiavi con keytool utilizzando Client SDK 3
- Usa l'archivio di AWS CloudHSM chiavi con Jarsigner utilizzando Client SDK 3
- Problemi noti per AWS CloudHSM l'integrazione di Java Keytool e Jarsigner utilizzando Client SDK 3
- AWS CloudHSM Registra chiavi preesistenti con key store

Prerequisiti per l'integrazione AWS CloudHSM con Java Keytool e Jarsigner utilizzando Client SDK 3

Per utilizzare l'archivio AWS CloudHSM chiavi, è necessario prima inizializzare e configurare l'SDK JCE. AWS CloudHSM Utilizza le fasi seguenti per farlo.

Fase 1: Installare JCE

Per installare JCE, inclusi i prerequisiti del AWS CloudHSM client, seguite i passaggi per l'installazione della libreria Java.

Passaggio 2: aggiungere le credenziali di accesso HSM alle variabili di ambiente

Imposta le variabili di ambiente per contenere le credenziali di accesso all'HSM.

```
export HSM_PARTITION=PARTITION_1
export HSM_USER=<HSM user name>
export HSM_PASSWORD=<HSM password>
```

> **Note**
>
> Il CloudHSM JCE offre varie opzioni di accesso. Per utilizzare l'archivio AWS CloudHSM chiavi con applicazioni di terze parti, è necessario utilizzare l'accesso implicito con variabili di ambiente. Se desideri utilizzare l'accesso esplicito tramite il codice dell'applicazione, devi creare la tua applicazione utilizzando l'archivio delle AWS CloudHSM chiavi. Per ulteriori informazioni, consulta l'articolo sull'utilizzo di AWS CloudHSM Key Store.

Fase 3: Registrare il provider JCE

Per registrare il provider JCE, nella configurazione Java CloudProvider .

1. Aprire il file di configurazione java.security nell'installazione Java, per la modifica.
2. Nel file di configurazione java.security, aggiungere `com.cavium.provider.CaviumProvider` come ultimo provider. Ad esempio, se sono presenti nove provider nel file java.security, aggiungere il seguente provider come ultimo provider nella sezione. L'aggiunta del provider Cavium come priorità più elevata può influire negativamente sulle prestazioni del sistema.

   ```
   security.provider.10=com.cavium.provider.CaviumProvider
   ```

   > **Note**
   >
   > Gli utenti esperti possono essere abituati a specificare `-providerName`, `-providerclass` e `-providerpath` come opzioni della riga di comando quando usano il keytool, invece di aggiornare il file di configurazione di sicurezza. Se si tenta di specificare le opzioni della riga di comando durante la generazione di chiavi con AWS CloudHSM key store, si verificheranno degli errori.

Usa l'archivio AWS CloudHSM delle chiavi con keytool utilizzando Client SDK 3

Keytool è una utility a riga di comando molto usata per le attività comuni di chiavi e certificati sui sistemi Linux. Nella documentazione AWS CloudHSM non è incluso un tutorial completo su keytool. Questo articolo spiega i parametri specifici da utilizzare con varie funzioni di keytool quando si utilizzano AWS CloudHSM come root of trust tramite il AWS CloudHSM key store.

Quando usi keytool con il AWS CloudHSM key store, specifica i seguenti argomenti per qualsiasi comando keytool:

```
-storetype CLOUDHSM \
  -J-classpath '-J/opt/cloudhsm/java/*' \
  -J-Djava.library.path=/opt/cloudhsm/lib
```

Se desideri creare un nuovo file di archivio chiavi utilizzando l'archivio AWS CloudHSM chiavi, consulta. Usa for Client SDK 3 AWS CloudHSM KeyStore AWS CloudHSM Per utilizzare un archivio di chiavi esistente, specificane il nome (incluso il percorso) utilizzando l'argomento -keystore per keytool. Se specificate un file di archivio chiavi inesistente in un comando keytool, l'archivio AWS CloudHSM chiavi crea un nuovo file di archivio chiavi.

Crea nuove AWS CloudHSM chiavi con keytool

Puoi usare keytool per generare qualsiasi tipo di chiave supportata da AWS CloudHSM JCE SDK. Vedere un elenco completo di chiavi e lunghezze nell'articolo Chiavi supportate nella libreria Java.

> **Important**
>
> Una chiave generata tramite keytool viene generata nel software e quindi importata AWS CloudHSM come chiave persistente estraibile.

Le istruzioni per creare chiavi non estraibili direttamente sul modulo di sicurezza hardware (HSM) e quindi utilizzarle con keytool o Jarsigner sono mostrate nell'esempio di codice in Registrazione di chiavi preesistenti con Key Store. AWS CloudHSM Ti consigliamo vivamente di generare chiavi non esportabili al di fuori del keytool e quindi di importare i certificati corrispondenti nell'archivio chiavi. Se si utilizzano chiavi RSA o EC estraibili tramite keytool e jarsigner, i provider esportano le chiavi da e quindi utilizzano la chiave localmente per le operazioni di firma. AWS CloudHSM

Se si dispone di più istanze client connesse al cluster CloudHSM, tenere presente che l'importazione di un certificato nell'archivio chiavi di un'istanza del client non renderà automaticamente disponibili i

certificati in altre istanze del client. Per registrare la chiave e i certificati associati su ogni istanza del client è necessario eseguire un'applicazione Java come descritto in Generare una CSR utilizzando Keytool. In alternativa, è possibile apportare le modifiche necessarie su un client e copiare il file dell'archivio delle chiavi risultante in ogni altra istanza del client.

Esempio 1: generare una chiave AES-256 simmetrica e salvarla in un file di archivio chiavi denominato «example_keystore.store», nella directory di lavoro. Sostituisci con un'etichetta univoca. *<secret label>*

```
keytool -genseckey -alias <secret label> -keyalg aes \
  -keysize 256 -keystore example_keystore.store \
  -storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
  -J-Djava.library.path=/opt/cloudhsm/lib/
```

Esempio 2: generare una coppia di chiavi RSA 2048 e salvarla in un file di archivio chiavi denominato «example_keystore.store» nella directory di lavoro. Sostituisci con un'etichetta univoca. *<RSA key pair label>*

```
keytool -genkeypair -alias <RSA key pair label> \
        -keyalg rsa -keysize 2048 \
        -sigalg sha512withrsa \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Esempio 3: generare una chiave ED p256 e salvarla in un file di archivio chiavi denominato «example_keystore.store» nella directory di lavoro. Sostituisci con un'etichetta univoca. *<ec key pair label>*

```
keytool -genkeypair -alias <ec key pair label> \
        -keyalg ec -keysize 256 \
        -sigalg SHA512withECDSA \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

È possibile trovare un elenco di algoritmi di firma supportati nella libreria Java.

Elimina una AWS CloudHSM chiave usando keytool

L'archivio AWS CloudHSM chiavi non supporta l'eliminazione delle chiavi. Per eliminare la chiave, è necessario utilizzare la `deleteKey` funzione AWS CloudHSM dello strumento da riga di comando,Eliminare una AWS CloudHSM chiave usando KMU.

Genera una AWS CloudHSM CSR usando keytool

Ottieni la massima flessibilità nella generazione di una richiesta di firma del certificato (CSR) se utilizzi OpenSSL Dynamic Engine per AWS CloudHSM Client SDK 5. Il comando seguente utilizza keytool per generare un CSR per una coppia di chiavi con l'alias, `example-key-pair`.

```
keytool -certreq -alias <key pair label> \
        -file example_csr.csr \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

> **Note**
>
> Per utilizzare una coppia di chiavi da keytool, tale coppia di chiavi deve avere una voce nel file dell'archivio chiavi specificato. Se si desidera utilizzare una coppia di chiavi generata al di fuori di keytool, è necessario importare i metadati della chiave e del certificato nell'archivio chiavi. Per istruzioni sull'importazione dei dati del keystore, consulta Importazione di certificati intermedi e root in Key Store utilizzando Keytool. AWS CloudHSM

Usa keytool per importare certificati intermedi e root nel key store AWS CloudHSM

Per importare un certificato CA AWS CloudHSM, è necessario abilitare la verifica di una catena completa di certificati su un certificato appena importato. Il comando seguente mostra un esempio.

```
keytool -import -trustcacerts -alias rootCAcert \
        -file rootCAcert.cert -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Se connetti più istanze client al AWS CloudHSM cluster, l'importazione di un certificato nell'archivio delle chiavi di un'istanza client non renderà automaticamente il certificato disponibile su altre istanze client. È necessario importare il certificato in ogni istanza del client.

Usa keytool per eliminare i certificati dall'archivio delle chiavi AWS CloudHSM

Il comando seguente mostra un esempio di come eliminare un AWS CloudHSM certificato da un archivio di chiavi Java keytool.

```
keytool -delete -alias mydomain -keystore \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Se si connettono più istanze client al AWS CloudHSM cluster, l'eliminazione di un certificato nell'archivio chiavi di un'istanza client non rimuoverà automaticamente il certificato dalle altre istanze client. È necessario eliminare il certificato su ogni istanza del client.

Importa un certificato funzionante nell'archivio delle AWS CloudHSM chiavi utilizzando keytool

Una volta firmata una richiesta di firma del certificato (CSR), è possibile importarla nell'archivio chiavi AWS CloudHSM e associarla alla coppia di chiavi appropriata. Il seguente comando fornisce un esempio.

```
keytool -importcert -noprompt -alias <key pair label> \
        -file example_certificate.crt \
        -keystore example_keystore.store
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

L'alias deve essere una coppia di chiavi con un certificato associato nell'archivio chiavi. Se la chiave viene generata al di fuori di keytool o viene generata in un'istanza del client diversa, è necessario prima importare i metadati della chiave e del certificato nell'archivio chiavi. Per istruzioni sull'importazione dei metadati del certificato, consulta l'esempio di codice in Registrazione di chiavi preesistenti con Key Store. AWS CloudHSM

La catena di certificati deve essere verificabile. Se non è possibile verificare il certificato, potrebbe essere necessario importare il certificato di firma (autorità di certificazione) nell'archivio chiavi in modo che la catena possa essere verificata.

Esporta un certificato utilizzando keytool AWS CloudHSM

Nell'esempio seguente viene generato un certificato in formato binario X.509. Per esportare un certificato leggibile dall'uomo AWS CloudHSM, aggiungilo `-rfc` al `-exportcert` comando.

```
keytool -exportcert -alias <key pair label> \
        -file example_exported_certificate.crt \
        -keystore example_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Usa l'archivio di AWS CloudHSM chiavi con Jarsigner utilizzando Client SDK 3

Jarsigner è una popolare utilità a riga di comando per firmare file JAR utilizzando una chiave archiviata in modo sicuro su un modulo di sicurezza hardware (HSM). Un tutorial completo su Jarsigner non rientra nell'ambito della documentazione. AWS CloudHSM Questa sezione spiega i parametri Jarsigner da utilizzare per firmare e verificare le firme AWS CloudHSM come radice di fiducia tramite il key store. AWS CloudHSM

Configura AWS CloudHSM chiavi e certificati con Jarsigner

Prima di poter firmare i file AWS CloudHSM JAR con Jarsigner, assicurati di aver configurato o completato i seguenti passaggi:

1. Seguire le indicazioni contenute nei prerequisiti dell'archivio chiavi AWS CloudHSM.
2. Configura le chiavi di firma e i certificati e la catena di certificati associati, che devono essere archiviati nell'archivio AWS CloudHSM chiavi dell'istanza corrente del server o del client. Crea le chiavi su AWS CloudHSM e poi importa i metadati associati nel tuo AWS CloudHSM key store. Usa l'esempio di codice in Registrazione di chiavi preesistenti con AWS CloudHSM Key Store per importare i metadati nel key store. Se desideri utilizzare keytool per impostare le chiavi e i certificati, vedi Crea nuove AWS CloudHSM chiavi con keytool. Se utilizzi più istanze client per firmare il tuo JARs, crea la chiave e importa la catena di certificati. Quindi copia il file dell'archivio delle chiavi risultante in ogni istanza del client. Se generi frequentemente nuove chiavi, è possibile che sia più semplice importare singolarmente i certificati in ogni istanza client.
3. L'intera catena di certificati dovrebbe essere verificabile. Affinché la catena di certificati sia verificabile, potrebbe essere necessario aggiungere il certificato CA e i certificati intermedi all'archivio delle chiavi. AWS CloudHSM Consulta lo snippet di codice in Sign a JAR file using

AWS CloudHSM e Jarsigner per istruzioni su come utilizzare il codice Java per verificare la catena di certificati. Se preferisci, puoi utilizzare keytool per importare i certificati. Per istruzioni sull'uso di keytool, consulta Uso di Keytool per importare certificati intermedi e root in Key Store. AWS CloudHSM

Firma un file JAR utilizzando e Jarsigner AWS CloudHSM

Usa il seguente comando per firmare un file JAR usando AWS CloudHSM e jarsigner:

```
jarsigner -keystore example_keystore.store \
        -signedjar signthisclass_signed.jar \
        -sigalg sha512withrsa \
        -storetype CloudHSM \
        -J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \
        -J-Djava.library.path=/opt/cloudhsm/lib \
        signthisclass.jar <key pair label>
```

Utilizza il seguente comando per verificare un file JAR firmato:

```
jarsigner -verify \
        -keystore example_keystore.store \
        -sigalg sha512withrsa \
        -storetype CloudHSM \
        -J-classpath '-J/opt/cloudhsm/java/*:/usr/lib/jvm/java-1.8.0/lib/tools.jar' \
        -J-Djava.library.path=/opt/cloudhsm/lib \
        signthisclass_signed.jar <key pair label>
```

Problemi noti per AWS CloudHSM l'integrazione di Java Keytool e Jarsigner utilizzando Client SDK 3

L'elenco seguente fornisce l'elenco corrente dei problemi noti relativi alle integrazioni con AWS CloudHSM e Java Keytool e Jarsigner utilizzando Client SDK 3.

- Quando si generano chiavi utilizzando keytool, il primo provider nella configurazione del provider non può esserlo. CaviumProvider
- Quando si generano chiavi utilizzando keytool, viene utilizzato il primo provider (supportato) nel file di configurazione di sicurezza per generare la chiave. Questo è generalmente un provider di software. Alla chiave generata viene quindi assegnato un alias e importata nell' AWS CloudHSM HSM come chiave persistente (token) durante il processo di aggiunta della chiave.

- Quando si utilizza keytool con AWS CloudHSM key store, non specificare `-providerName` né `-providerpath` opzioni sulla riga di comando. `-providerclass` Specificare queste opzioni nel file del provider di protezione come descritto nei prerequisiti dell'archivio chiavi.
- Quando si utilizzano chiavi EC non estraibili tramite keytool e Jarsigner, il provider SunEC deve appartenere all'elenco dei provider nel file removed/disabled java.security. Se si utilizzano chiavi EC estraibili tramite keytool e Jarsigner, i provider esportano i bit chiave dall'HSM e utilizzano la chiave localmente per le operazioni di firma. AWS CloudHSM Si sconsiglia di utilizzare chiavi esportabili con keytool o Jarsigner.

AWS CloudHSM Registra chiavi preesistenti con key store

Per la massima sicurezza e flessibilità negli attributi e nell'etichettatura, ti consigliamo di generare le chiavi di AWS CloudHSM firma utilizzando key_mgmt_util. È inoltre possibile utilizzare un'applicazione Java per generare la chiave in AWS CloudHSM.

La sezione seguente fornisce un esempio di codice che dimostra come generare una nuova coppia di chiavi sull'HSM e registrarla utilizzando le chiavi esistenti importate nel AWS CloudHSM key store. Le chiavi importate sono disponibili per l'uso con strumenti di terze parti come keytool e Jarsigner.

Per utilizzare una chiave preesistente, modificare il codice di esempio per cercare una chiave per etichetta invece di generare una nuova chiave. Il codice di esempio per la ricerca di una chiave per etichetta è disponibile nell'KeyUtilitiesRunneresempio.java su. GitHub

> **Important**
>
> La registrazione di una chiave AWS CloudHSM memorizzata in un key store locale non comporta l'esportazione della chiave. Quando la chiave è registrata, l'archivio chiavi registra l'alias (o l'etichetta) della chiave e crea una correlazione tra gli oggetti certificati archiviati localmente e una coppia di chiavi sul AWS CloudHSM. Finché la coppia di chiavi viene creata come non esportabile, i bit chiave non lasceranno l'HSM.

```
                    //
// Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.
//
```

```
 // Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
 // software and associated documentation files (the "Software"), to deal in the
 Software
 // without restriction, including without limitation the rights to use, copy, modify,
 // merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
 // permit persons to whom the Software is furnished to do so.
 //
 // THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
 // INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
 // PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
 // HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
 // OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
 // SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 //

package com.amazonaws.cloudhsm.examples;

import com.cavium.key.CaviumKey;
import com.cavium.key.parameter.CaviumAESKeyGenParameterSpec;
import com.cavium.key.parameter.CaviumRSAKeyGenParameterSpec;
import com.cavium.asn1.Encoder;
import com.cavium.cfm2.Util;

import javax.crypto.KeyGenerator;

import java.io.ByteArrayInputStream;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.FileNotFoundException;

import java.math.BigInteger;

import java.security.*;
import java.security.cert.Certificate;
import java.security.cert.CertificateException;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
import java.security.interfaces.RSAPrivateKey;
import java.security.interfaces.RSAPublicKey;
import java.security.KeyStore.PasswordProtection;
import java.security.KeyStore.PrivateKeyEntry;
import java.security.KeyStore.Entry;
```

```
import java.util.Calendar;
import java.util.Date;
import java.util.Enumeration;

//
// KeyStoreExampleRunner demonstrates how to load a keystore, and associate a
 certificate with a
// key in that keystore.
//
// This example relies on implicit credentials, so you must setup your environment
 correctly.
//
// https://docs.aws.amazon.com/cloudhsm/latest/userguide/java-library-
install.html#java-library-credentials
//

public class KeyStoreExampleRunner {

    private static byte[] COMMON_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
 (byte) 0x03 };
    private static byte[] COUNTRY_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
 (byte) 0x06 };
    private static byte[] LOCALITY_NAME_OID = new byte[] { (byte) 0x55, (byte) 0x04,
 (byte) 0x07 };
    private static byte[] STATE_OR_PROVINCE_NAME_OID = new byte[] { (byte) 0x55,
 (byte) 0x04, (byte) 0x08 };
    private static byte[] ORGANIZATION_NAME_OID = new byte[] { (byte) 0x55, (byte)
 0x04, (byte) 0x0A };
    private static byte[] ORGANIZATION_UNIT_OID = new byte[] { (byte) 0x55, (byte)
 0x04, (byte) 0x0B };

    private static String helpString = "KeyStoreExampleRunner%n" +
           "This sample demonstrates how to load and store keys using a keystore.%n%n"
 +
           "Options%n" +
           "\t--help\t\t\tDisplay this message.%n" +
           "\t--store <filename>\t\tPath of the keystore.%n" +
           "\t--password <password>\t\tPassword for the keystore (not your CU
 password).%n" +
           "\t--label <label>\t\t\tLabel to store the key and certificate under.%n" +
           "\t--list\t\t\tList all the keys in the keystore.%n%n";

   public static void main(String[] args) throws Exception {
       Security.addProvider(new com.cavium.provider.CaviumProvider());
```

```
        KeyStore keyStore = KeyStore.getInstance("CloudHSM");

        String keystoreFile = null;
        String password = null;
        String label = null;
        boolean list = false;
        for (int i = 0; i < args.length; i++) {
            String arg = args[i];
            switch (args[i]) {
                case "--store":
                    keystoreFile = args[++i];
                    break;
                case "--password":
                    password = args[++i];
                    break;
                case "--label":
                    label = args[++i];
                    break;
                case "--list":
                    list = true;
                    break;
                case "--help":
                    help();
                    return;
            }
        }

        if (null == keystoreFile || null == password) {
            help();
            return;
        }

        if (list) {
            listKeys(keystoreFile, password);
            return;
        }

        if (null == label) {
            label = "Keystore Example Keypair";
        }

        //
        // This call to keyStore.load() will open the pkcs12 keystore with the supplied
        // password and connect to the HSM. The CU credentials must be specified using
```

```
        // standard CloudHSM login methods.
        //
        try {
            FileInputStream instream = new FileInputStream(keystoreFile);
            keyStore.load(instream, password.toCharArray());
        } catch (FileNotFoundException ex) {
            System.err.println("Keystore not found, loading an empty store");
            keyStore.load(null, null);
        }

        PasswordProtection passwd = new PasswordProtection(password.toCharArray());
        System.out.println("Searching for example key and certificate...");

        PrivateKeyEntry keyEntry = (PrivateKeyEntry) keyStore.getEntry(label, passwd);
        if (null == keyEntry) {
            //
            // No entry was found, so we need to create a key pair and associate a
certificate.
            // The private key will get the label passed on the command line. The
keystore alias
            // needs to be the same as the private key label. The public key will have
":public"
            // appended to it. The alias used in the keystore will We associate the
certificate
            // with the private key.
            //
            System.out.println("No entry found, creating...");
            KeyPair kp = generateRSAKeyPair(2048, label + ":public", label);
            System.out.printf("Created a key pair with the handles %d/%d%n",
((CaviumKey) kp.getPrivate()).getHandle(), ((CaviumKey) kp.getPublic()).getHandle());

            //
            // Generate a certificate and associate the chain with the private key.
            //
            Certificate self_signed_cert = generateCert(kp);
            Certificate[] chain = new Certificate[1];
            chain[0] = self_signed_cert;
            PrivateKeyEntry entry = new PrivateKeyEntry(kp.getPrivate(), chain);

            //
            // Set the entry using the label as the alias and save the store.
            // The alias must match the private key label.
            //
            keyStore.setEntry(label, entry, passwd);
```

```
            FileOutputStream outstream = new FileOutputStream(keystoreFile);
            keyStore.store(outstream, password.toCharArray());
            outstream.close();

            keyEntry = (PrivateKeyEntry) keyStore.getEntry(label, passwd);
        }

        long handle = ((CaviumKey) keyEntry.getPrivateKey()).getHandle();
        String name = keyEntry.getCertificate().toString();
        System.out.printf("Found private key %d with certificate %s%n", handle, name);
    }

    private static void help() {
        System.out.println(helpString);
    }

    //
    // Generate a non-extractable / non-persistent RSA keypair.
    // This method allows us to specify the public and private labels, which
    // will make KeyStore aliases easier to understand.
    //
    public static KeyPair generateRSAKeyPair(int keySizeInBits, String publicLabel,
String privateLabel)
            throws InvalidAlgorithmParameterException, NoSuchAlgorithmException,
NoSuchProviderException {

        boolean isExtractable = false;
        boolean isPersistent = false;
        KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("rsa", "Cavium");
        CaviumRSAKeyGenParameterSpec spec = new
CaviumRSAKeyGenParameterSpec(keySizeInBits, new BigInteger("65537"), publicLabel,
privateLabel, isExtractable, isPersistent);

        keyPairGen.initialize(spec);

        return keyPairGen.generateKeyPair();
    }

    //
    // Generate a certificate signed by a given keypair.
    //
    private static Certificate generateCert(KeyPair kp) throws CertificateException {
        CertificateFactory cf = CertificateFactory.getInstance("X509");
```

```
        PublicKey publicKey = kp.getPublic();
        PrivateKey privateKey = kp.getPrivate();
        byte[] version = Encoder.encodeConstructed((byte) 0,
Encoder.encodePositiveBigInteger(new BigInteger("2"))); // version 1
        byte[] serialNo = Encoder.encodePositiveBigInteger(new BigInteger(1,
Util.computeKCV(publicKey.getEncoded())));

        // Use the SHA512 OID and algorithm.
        byte[] signatureOid = new byte[] {
            (byte) 0x2A, (byte) 0x86, (byte) 0x48, (byte) 0x86, (byte) 0xF7, (byte)
0x0D, (byte) 0x01, (byte) 0x01, (byte) 0x0D };
        String sigAlgoName = "SHA512WithRSA";

         byte[] signatureId = Encoder.encodeSequence(
                                        Encoder.encodeOid(signatureOid),
                                        Encoder.encodeNull());

         byte[] issuer = Encoder.encodeSequence(
                                    encodeName(COUNTRY_NAME_OID, "<Country>"),
                                    encodeName(STATE_OR_PROVINCE_NAME_OID, "<State>"),
                                    encodeName(LOCALITY_NAME_OID, "<City>"),
                                    encodeName(ORGANIZATION_NAME_OID,
"<Organization>"),
                                    encodeName(ORGANIZATION_UNIT_OID, "<Unit>"),
                                    encodeName(COMMON_NAME_OID, "<CN>")
                                );

         Calendar c = Calendar.getInstance();
         c.add(Calendar.DAY_OF_YEAR, -1);
         Date notBefore = c.getTime();
         c.add(Calendar.YEAR, 1);
         Date notAfter = c.getTime();
         byte[] validity = Encoder.encodeSequence(
                                        Encoder.encodeUTCTime(notBefore),
                                        Encoder.encodeUTCTime(notAfter)
                                    );
         byte[] key = publicKey.getEncoded();

         byte[] certificate = Encoder.encodeSequence(
                                        version,
                                        serialNo,
                                        signatureId,
                                        issuer,
                                        validity,
```

```
                                        issuer,
                                        key);
        Signature sig;
        byte[] signature = null;
        try {
            sig = Signature.getInstance(sigAlgoName, "Cavium");
            sig.initSign(privateKey);
            sig.update(certificate);
            signature = Encoder.encodeBitstring(sig.sign());

        } catch (Exception e) {
            System.err.println(e.getMessage());
            return null;
        }

        byte [] x509 = Encoder.encodeSequence(
                        certificate,
                        signatureId,
                        signature
                        );
        return cf.generateCertificate(new ByteArrayInputStream(x509));
    }

     //
     // Simple OID encoder.
     // Encode a value with OID in ASN.1 format
     //
     private static byte[] encodeName(byte[] nameOid, String value) {
         byte[] name = null;
         name = Encoder.encodeSet(
                     Encoder.encodeSequence(
                             Encoder.encodeOid(nameOid),
                             Encoder.encodePrintableString(value)
                     )
                 );
         return name;
     }

    //
    // List all the keys in the keystore.
    //
    private static void listKeys(String keystoreFile, String password) throws Exception
{
        KeyStore keyStore = KeyStore.getInstance("CloudHSM");
```

```
        try {
            FileInputStream instream = new FileInputStream(keystoreFile);
            keyStore.load(instream, password.toCharArray());
        } catch (FileNotFoundException ex) {
            System.err.println("Keystore not found, loading an empty store");
            keyStore.load(null, null);
        }

        for(Enumeration<String> entry = keyStore.aliases(); entry.hasMoreElements();) {
            System.out.println(entry.nextElement());
        }
    }

}
```

Usa Microsoft Manifest Generation and Editing Tool (Mage.exe) con AWS CloudHSM per firmare i file

> **Note**
>
> AWS CloudHSM supporta solo lo strumento Mage a 64 bit incluso in Windows SDK for .NET Framework 4.8.1 e versioni successive.

I seguenti argomenti forniscono una panoramica sull'utilizzo di. Mage.exe AWS CloudHSM

Argomenti

- Fase 1: configurazione dei prerequisiti
- Fase 2: Creare un certificato di firma
- Fase 3: Firma un file

Fase 1: configurazione dei prerequisiti

Per utilizzare Microsoft Mage.exe con AWS CloudHSM, è necessario quanto segue:

- Un'istanza Amazon EC2 che esegue un sistema operativo Windows
- Un'autorità di certificazione (CA), gestita autonomamente o fornita da un fornitore terzo
- Un AWS CloudHSM cluster attivo nello stesso cloud privato virtuale (VPC) dell'istanza EC2, con almeno un HSM
- Un utente crittografico (CU) per possedere e gestire le chiavi nel cluster AWS CloudHSM
- Un file o eseguibile non firmato
- Il kit di sviluppo software (SDK) per Microsoft Windows

Per configurare i prerequisiti per l'utilizzo AWS CloudHSM con Mage.exe

1. Avvia un'istanza Windows EC2 e un AWS CloudHSM cluster seguendo le istruzioni nella sezione Getting Started di questa guida.
2. Se desideri ospitare la tua CA di Windows Server, completa i passaggi 1 e 2 in Configurazione di Windows Server come autorità di certificazione con. AWS CloudHSM Altrimenti, utilizza la tua CA di terze parti pubblicamente attendibile.
3. Scarica e installa Microsoft Windows SDK for .NET Framework 4.8.1 o versione successiva sulla tua istanza Windows EC2:

   - Microsoft Windows SDK 10

   L'`mage.exe`eseguibile fa parte degli strumenti di Windows SDK. Il percorso di installazione predefinito è:

   ```
   C:\Program Files (x86)\Windows Kits\<SDK version>\bin\<version number>\x64\Mage.exe
   ```

Dopo aver completato questi passaggi, puoi utilizzare Microsoft Windows SDK, il tuo AWS CloudHSM cluster e la tua CA per creare un certificato di firma.

Fase 2: Creare un certificato di firma

Ora che hai installato Windows SDK sulla tua istanza EC2, puoi utilizzarlo per generare una richiesta di firma del certificato (CSR). Il CSR è un certificato non firmato che invii alla tua CA per la firma. In questo esempio, utilizziamo l'`certreq`eseguibile incluso in Windows SDK per generare la CSR.

Per generare una CSR utilizzando l'eseguibile certreq

1. Connect alla tua istanza Windows EC2. Per ulteriori informazioni, consulta Connect to Your Instance nella Amazon EC2 User Guide.
2. Crea un archivio denominato `request.inf` con i seguenti contenuti. Sostituisci le `Subject` informazioni con i dettagli della tua organizzazione:

```
[Version]
Signature= $Windows NT$
[NewRequest]
Subject = "C=<Country>,CN=<www.website.com>,O=<Organization>,OU=<Organizational-Unit>,L=<City>,S=<State>"
RequestType=PKCS10
HashAlgorithm = SHA256
KeyAlgorithm = RSA
KeyLength = 2048
ProviderName = "CloudHSM Key Storage Provider"
KeyUsage = "CERT_DIGITAL_SIGNATURE_KEY_USAGE"
MachineKeySet = True
Exportable = False
```

   Per una spiegazione di ciascun parametro, consulta la documentazione di Microsoft.
3. Esegui `certreq.exe` per generare la CSR:

```
certreq.exe -new request.inf request.csr
```

   Questo comando genera una nuova coppia di chiavi sul AWS CloudHSM cluster e utilizza la chiave privata per creare la CSR.
4. Invia la CSR alla CA. Se utilizzi una CA di Windows Server, procedi nel seguente modo:

   a. Apri lo strumento CA:

```
certsrv.msc
```

   b. Nella nuova finestra, fare clic con il pulsante destro del mouse sul nome del server CA. Scegliere All tasks (Tutte le attività), quindi scegliere Submit new request (Sottometti nuova richiesta).

   c. Vai alla posizione di `request.csr` e scegli Apri.

d. Espandi il menu Server CA e vai alla cartella Richieste in sospeso. Right-click nella richiesta appena creata, scegli Tutte le attività, quindi scegli Emetti.

e. Vai alla cartella Certificati emessi.

f. Scegliere Apri per visualizzare il certificato, quindi scegliere la schedaDetails (Dettagli).

g. Scegliere Copy to File (Copia su File) per avviare la procedura guidata di esportazione dei certificati. Salva il DER-encoded X.509 file in una posizione sicura come`signedCertificate.cer`.

h. Uscire dallo strumento CA ed eseguire il comando seguente per spostare il file del certificato nel Personal Certificate Store in Windows:

```
certreq.exe -accept signedCertificate.cer
```

È ora possibile utilizzare il certificato importato per firmare un file.

Fase 3: Firma un file

Ora che hai Mage.exe importato il certificato, puoi firmare un file. Devi conoscere l' SHA-1 hash o l'impronta digitale del certificato. L'impronta digitale garantisce che vengano utilizzati Mage.exe solo certificati verificati da. AWS CloudHSM In questo esempio, lo usiamo PowerShell per ottenere l'hash del certificato.

Per ottenere l'identificazione personale del certificato e utilizzarlo per firmare un file

1. Passa alla directory contenente`mage.exe`. La posizione predefinita è:

```
C:\Program Files (x86)\Microsoft SDKs\Windows\v10.0A\bin\NETFX 4.8.1 Tools\x64
```

2. Per creare un file di applicazione di esempio utilizzando Mage.exe, esegui il seguente comando:

```
mage.exe -New Application -ToFile C:\Users\Administrator\Desktop\sample.application
```

3. Apri PowerShell come amministratore ed esegui il seguente comando:

```
Get-ChildItem -path cert:\LocalMachine\My
```

Copia i `Provider` valori `ThumbprintKey Container`, e dall'output.

```
================ Certificate 1 ================
Serial Number: 7ad8df4f5bfaf5a3405d32aa4a01b85a
Issuer: CN=EC2AMAZ-0BVI3HH-CA
 NotBefore: 2/4/2025 7:55 PM
 NotAfter: 2/4/2030 8:05 PM
Subject: CN=EC2AMAZ-0BVI3HH-CA
CA Version: V0.0
Signature matches Public Key
Root Certificate: Subject matches Issuer
Cert Hash(sha1): 08f295934325ee2c37ea01ab605ceeb81610d3be   thumbprint
  Key Container = EC2AMAZ-0BVI3HH-CA   key container
  Provider = CloudHSM Key Storage Provider     provider
Signature test passed
CertUtil: -store command completed successfully.
```

4. Firma il file eseguendo il seguente comando:

```
mage.exe -Sign -CertHash <thumbprint> -KeyContainer <keycontainer> -
CryptoProvider <CloudHSM Key Storage Provider/Cavium Key Storage Provider> C:\Users
\Administrator\Desktop\<sample.application>
```

   Se il comando ha esito positivo, PowerShell restituisce un messaggio di successo.

5. Per verificare la firma sul file, utilizzate il seguente comando:

```
mage.exe -Verify -CryptoProvider <CloudHSM Key Storage Provider/Cavium Key Storage
 Provider> C:\Users\Administrator\Desktop\<sample.application>
```

Integrazioni con altri fornitori di terze parti con AWS CloudHSM

Diversi fornitori di terze parti supportano AWS CloudHSM come base di fiducia. Questo significa che puoi utilizzare una soluzione software di tua scelta durante la creazione e l'archiviazione delle chiavi sottostanti nel cluster CloudHSM. Di conseguenza, il carico di lavoro in ingresso AWS può contare sui vantaggi di latenza, disponibilità, affidabilità ed elasticità di CloudHSM. L'elenco seguente include i fornitori di terze parti che supportano CloudHSM.

> **Note**
>
> AWS non approva né garantisce alcun fornitore terzo.

- Hashicorp Vault è uno strumento di gestione dei segreti progettato per consentire la collaborazione e la governance tra le organizzazioni. Supporta AWS Key Management Service e AWS CloudHSM fonda la fiducia per una protezione aggiuntiva.
- Thycotic Secrets Server consente ai clienti di gestire credenziali sensibili su account privilegiati. Supporta AWS CloudHSM come base di fiducia.
- L'adattatore KMIP di P6R consente di utilizzare le AWS CloudHSM istanze tramite un'interfaccia KMIP standard.
- PrimeKey EJBCA è una popolare soluzione open source per PKI. Consente di creare e archiviare coppie di chiavi in modo sicuro con. AWS CloudHSM
- Box KeySafe fornisce la gestione delle chiavi di crittografia per i contenuti cloud a molte organizzazioni con rigorosi requisiti di sicurezza, privacy e conformità normativa. I clienti possono proteggere ulteriormente KeySafe le chiavi direttamente AWS Key Management Service o indirettamente AWS CloudHSM tramite AWS KMS Custom Key Store.
- Insyde Software supporta la AWS CloudHSM firma del firmware come root of trust.
- F5 BIG-IP LTM funge AWS CloudHSM da radice di fiducia.
- Cloudera Navigator Key HSM consente di utilizzare il cluster CloudHSM per creare e archiviare chiavi per Cloudera Navigator Key Trustee Server.
- Venafi Trust Protection Platform offre una gestione completa dell'identità delle macchine per TLS, SSH e per la firma del codice con la generazione di chiavi AWS CloudHSM e la relativa protezione.

Monitoraggio AWS CloudHSM

Oltre alle funzionalità di registrazione integrate nel Client SDK, puoi anche utilizzare AWS CloudTrail Amazon CloudWatch Logs e Amazon CloudWatch per il monitoraggio. AWS CloudHSM

Log del Client SDK

Utilizza i log di Client SDK per monitorare le informazioni di diagnostica e la risoluzione dei problemi delle applicazioni che crei.

CloudTrail

CloudTrail Utilizzalo per monitorare tutte le chiamate API nel tuo AWS account, incluse le chiamate effettuate per creare ed eliminare cluster, moduli di sicurezza hardware (HSM) e tag di risorse.

CloudWatch Registri

Usa CloudWatch Logs per monitorare i log delle tue istanze HSM, che includono eventi per la creazione e l'eliminazione di utenti HSM, la modifica delle password degli utenti, la creazione e l'eliminazione di chiavi e altro ancora.

CloudWatch

Utilizzalo CloudWatch per monitorare lo stato del cluster in tempo reale.

Argomenti

- Utilizzo dei log SDK AWS CloudHSM del client
- Lavorare conAWS CloudTrailandAWS CloudHSM
- Utilizzo di Amazon CloudWatch Logs e AWS CloudHSM Audit Logs
- Ottenere CloudWatch metriche per AWS CloudHSM

Utilizzo dei log SDK AWS CloudHSM del client

È possibile recuperare i log generati dal Client SDK. AWS CloudHSM offre un'implementazione della registrazione con Client SDK 3 e Client SDK 5.

Argomenti

- Logging con Client SDK 5
- Logging con Client SDK 3

Logging con Client SDK 5

I log di Client SDK 5 contengono le informazioni di ciascun componente all'interno di un file denominato a seconda del componente. È possibile utilizzare lo strumento di configurazione per Client SDK 5 per configurare il logging per ciascun componente.

Se non si specifica una posizione per il file, il sistema scrive i log nella posizione predefinita:

AWS CloudHSM CLI

- Linux

  ```
  /opt/cloudhsm/run/cloudhsm-cli.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-cli.log
  ```

PKCS #11 library

- Linux

  ```
  /opt/cloudhsm/run/cloudhsm-pkcs11.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-pkcs11.log
  ```

OpenSSL Dynamic Engine

- Linux

  ```
  stderr
  ```

JCE provider

- Linux

  ```
  /opt/cloudhsm/run/cloudhsm-jce.log
  ```

  Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-jce.log
  ```

Key Storage Provider

- Windows

  ```
  C:\Program Files\Amazon\CloudHSM\cloudhsm-ksp.log
  ```

OpenSSL Dynamic Engine Provider

- Linux

  ```
  stderr
  ```

Per le informazioni su come configurare il logging per Client SDK 5, consulta la pagina sullo strumento di configurazione di Client SDK 5

Logging con Client SDK 3

I log di Client SDK 3 contengono informazioni dettagliate dal demone client. AWS CloudHSM La posizione dei log dipende dal sistema operativo dell'istanza del EC2 client Amazon in cui esegui il daemon del client.

Amazon Linux

In Amazon Linux, i log dei AWS CloudHSM client vengono scritti nel file denominato`/opt/cloudhsm/run/cloudhsm_client.log`. È possibile utilizzare logrotate o uno strumento simile per ruotare e gestire tali log.

Amazon Linux 2

In Amazon Linux 2, i log AWS CloudHSM del client vengono raccolti e archiviati nel journal. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, usa il seguente comando per visualizzare i log del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

CentOS 7

In CentOS 7, i log del AWS CloudHSM client vengono raccolti e archiviati nel diario. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizzare il seguente comando per visualizzare i registri del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

CentOS 8

In CentOS 8, i log del AWS CloudHSM client vengono raccolti e archiviati nel diario. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizzare il seguente comando per visualizzare i registri del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

RHEL 7

In Red Hat Enterprise Linux 7, i log del AWS CloudHSM Client vengono raccolti e archiviati nel journal. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizzate il seguente comando per visualizzare i log del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

RHEL 8

In Red Hat Enterprise Linux 8, i log del AWS CloudHSM client vengono raccolti e archiviati nel journal. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizzate il seguente comando per visualizzare i log del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 16.04

In Ubuntu 16.04, i log del AWS CloudHSM client vengono raccolti e archiviati nel diario. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizzare il seguente comando per visualizzare i log del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 18.04

In Ubuntu 18.04, i log del AWS CloudHSM client vengono raccolti e archiviati nel diario. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, usa il seguente comando per visualizzare i log del AWS CloudHSM client.

```
journalctl -f -u cloudhsm-client
```

Windows

- Per client Windows dalla versione 1.1.2+:

  AWS CloudHSM i log dei client vengono scritti in un `cloudhsm-kps.log` file nella cartella dei file di AWS CloudHSM programma ()`C:\Program Files\Amazon\CloudHSM\`. Ogni nome di file di registro ha un suffisso con un timestamp che indica quando il client è stato avviato. AWS CloudHSM
- Per Windows client 1.1.1 e versioni precedenti:

  Il log del client non vengono scritti su un file. I log vengono visualizzati al prompt dei comandi o nella PowerShell finestra in cui è stato avviato il client. AWS CloudHSM

Lavorare conAWS CloudTrailandAWS CloudHSM

AWS CloudHSMè integrato conAWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, un ruolo o un AWS servizio inAWS CloudHSM. CloudTrail acquisisce tutte le chiamate API AWS CloudHSM come eventi. Le chiamate acquisite includono chiamate dalla AWS CloudHSM console e chiamate di codice alle operazioni AWS CloudHSM API. Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per. AWS CloudHSM Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte

da CloudTrail, puoi determinare a quale richiesta è stata inviataAWS CloudHSM, 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 AWS CloudTrail per l'utente. Per un elenco completo delle operazioni AWS CloudHSM API, consulta Azioni nel riferimento AWS CloudHSM API.

AWS CloudHSMinformazioni in CloudTrail

CloudTrail è abilitato sul tuo AWS account al momento della creazione dell'account. Quando si verifica un'attività inAWS CloudHSM, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi AWS di servizio nella cronologia degli eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS. Per ulteriori informazioni, vedere Visualizzazione degli eventi con la cronologia degli CloudTrail eventi.

Per una registrazione continua degli eventi nel tuo AWS account, inclusi gli eventi diAWS CloudHSM, 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 trail
- CloudTrail Servizi e integrazioni supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni e ricezione di file di CloudTrail registro da più account

CloudTrail registra tutte le AWS CloudHSM operazioni, incluse le operazioni di sola lettura, come `DescribeClusters` e`ListTags`, e le operazioni di gestione, ad esempio, e. `InitializeCluster` `CreatHsm` `DeleteBackup`

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.

ComprensioneAWS CloudHSMvoci del file di log

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'AWS CloudHSM`CreateHsm`azione.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAJZVM5NEGZSTCITAMM:ExampleSession",
        "arn": "arn:aws:sts::111122223333:assumed-role/AdminRole/ExampleSession",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIY22AX6VRYNBGJSA",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2017-07-11T03:48:44Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAJZVM5NEGZSTCITAMM",
                "arn": "arn:aws:iam::111122223333:role/AdminRole",
                "accountId": "111122223333",
                "userName": "AdminRole"
            }
        }
    },
    "eventTime": "2017-07-11T03:50:45Z",
```

```
    "eventSource": "cloudhsm.amazonaws.com",
    "eventName": "CreateHsm",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.179",
    "userAgent": "aws-internal/3",
    "requestParameters": {
        "availabilityZone": "us-west-2b",
        "clusterId": "cluster-fw7mh6mayb5"
    },
    "responseElements": {
        "hsm": {
            "eniId": "eni-65338b5a",
            "clusterId": "cluster-fw7mh6mayb5",
            "state": "CREATE_IN_PROGRESS",
            "eniIp": "10.0.2.7",
            "hsmId": "hsm-6lz2hfmnzbx",
            "subnetId": "subnet-02c28c4b",
            "availabilityZone": "us-west-2b"
        }
    },
    "requestID": "1dae0370-65ec-11e7-a770-6578d63de907",
    "eventID": "b73a5617-8508-4c3d-900d-aa8ac9b31d08",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```


AWS CloudHSMeventi di gestione in CloudTrail

Gli eventi di gestione forniscono informazioni sulle operazioni di gestione eseguite sulle risorse dell'aziendaAccount AWS. Queste operazioni sono definite anche operazioni del piano di controllo (control-plane). Per impostazione predefinita, CloudTrail registra gli eventi di gestione.

AWS CloudHSMregistra tutte le operazioni del piano di controllo come eventi di gestione. Per un elenco delle operazioni del piano di AWS CloudHSM controllo a cui si AWS CloudHSM effettua l'accesso CloudTrail, vedere Azioni nell'AWS CloudHSMAPI Reference.

AWS CloudHSMeventi di servizio in CloudTrail

AWS CloudHSMregistra gli eventi di servizio CloudTrail quando il servizio esegue automaticamente azioni sulle risorse. Questi eventi non possono essere attivati direttamente da una chiamata a un'API pubblicaAWS CloudHSM. Per questi eventi, il campo `eventType` è `AwsServiceEvent`.

Di seguito sono riportati gli eventi di servizio a cui viene effettuato l'AWS CloudHSMaccesso. CloudTrail

- `DeleteBackup`— Registrato quando elimina AWS CloudHSM automaticamente un backup in base alla politica di conservazione dei backup.
- `BackupDeletedPermanently`— Registrato quando un backup PENDING_DELETION viene eliminato definitivamente e non può più essere ripristinato.
- `MigrationStart`— Registrato quando inizia una migrazione di tipo HSM per un cluster.
- `MigrationEnd`— Registrato quando viene completata una migrazione di tipo HSM per un cluster.
- `RollbackStart`— Registrato quando inizia un rollback di migrazione di tipo HSM per un cluster.
- `RollbackEnd`— Registrato quando viene completato un rollback di migrazione di tipo HSM per un cluster.

È possibile identificare gli eventi AWS CloudHSM di servizio nei CloudTrail registri in base ai seguenti valori di attributo:

- `eventSource`: `cloudhsm.amazonaws.com`
- `eventType`: `AwsServiceEvent`

Utilizzo di Amazon CloudWatch Logs e AWS CloudHSM Audit Logs

Quando un HSM del tuo account riceve un comando dagli strumenti a riga di AWS CloudHSM comando o dalle librerie software, registra l'esecuzione del comando in un registro di controllo. I log di audit dell'HSM includono tutti i comandi di gestione iniziati dal client, inclusi quelli che creano ed eliminano HSM, eseguono l'accesso e la disconnessione dall'HSM e gestiscono utenti e chiavi. Questi log forniscono un record affidabile di azioni che hanno modificato lo stato dell'HSM.

AWS CloudHSM raccoglie i log di controllo HSM e li invia ad Amazon CloudWatch Logs per tuo conto. Puoi utilizzare le funzionalità di CloudWatch Logs per gestire i log di AWS CloudHSM controllo, tra cui la ricerca e il filtraggio dei log e l'esportazione dei dati di log in Amazon S3. Puoi lavorare con i log di controllo HSM nella _CloudWatch console Amazon o utilizzare i comandi CloudWatch Logs in and Logs. AWS CLICloudWatch SDKs

Argomenti

- Funzionamento del log degli audit dell'HSM
- Visualizzazione AWS CloudHSM dei registri di controllo nei CloudWatch registri

- Interpretazione dei registri di AWS CloudHSM controllo
- AWS CloudHSM riferimento al registro di controllo

Funzionamento del log degli audit dell'HSM

La registrazione di controllo viene abilitata automaticamente in tutti i AWS CloudHSM cluster. Non può essere disattivata o disattivata e nessuna impostazione può AWS CloudHSM impedire l'esportazione dei log in Logs. CloudWatch Ogni evento di log dispone di un timestamp e di un numero di sequenza che indicano l'ordine degli eventi e consentono di rilevare qualsiasi manomissione dei log.

Ogni istanza HSM genera il proprio log. È probabile che i registri di controllo di diversi registri HSMs, anche quelli dello stesso cluster, siano diversi. Ad esempio, solo il primo HSM in ogni cluster registra l'inizializzazione dell'HSM. Gli eventi di inizializzazione non compaiono nei log HSMs che vengono clonati dai backup. Analogamente, quando si crea una chiave, l'HSM che genera la chiave registra un evento di generazione della chiave. Gli altri HSMs nel cluster registrano un evento quando ricevono la chiave tramite sincronizzazione.

AWS CloudHSM raccoglie i log e li pubblica in Logs in your CloudWatch account. Per comunicare con il servizio CloudWatch Logs per conto dell'utente, AWS CloudHSM utilizza un ruolo collegato al servizio. La policy IAM associata al ruolo consente di eseguire solo le attività necessarie AWS CloudHSM per inviare i log di controllo a Logs. CloudWatch

> **Important**
>
> Se hai creato un cluster prima del 20 gennaio 2018 ma non un ruolo collegato al servizio, devi crearne uno manualmente. Ciò è necessario per CloudWatch ricevere i log di controllo dal cluster. AWS CloudHSM Per ulteriori informazioni sulla creazione di ruoli collegati al servizio, vedi Capire i ruoli collegati al servizio e Creare un ruolo collegato a un servizio nella Guida utente IAM.

Visualizzazione AWS CloudHSM dei registri di controllo nei CloudWatch registri

Amazon CloudWatch Logs organizza i log di controllo in gruppi di log e, all'interno di un gruppo di log, in flussi di log. Ogni immissione di registro è un evento. AWS CloudHSM crea un gruppo di log per

ogni cluster e un flusso di log per ogni HSM del cluster. Non è necessario creare alcun componente CloudWatch Logs o modificare alcuna impostazione.

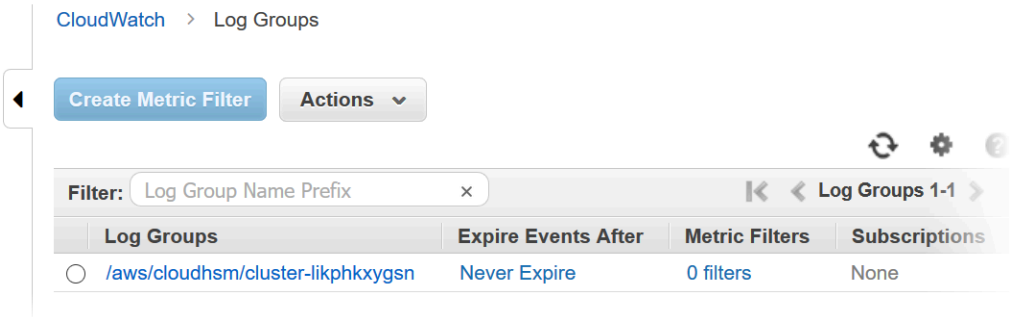
- Il nome del gruppo di log è `/aws/cloudhsm/<cluster ID>`, ad esempio `/aws/cloudhsm/cluster-likphkxygsn`. Quando utilizzate il nome del gruppo di log in un PowerShell comando AWS CLI or, assicuratevi di racchiuderlo tra virgolette doppie.
- Il nome del flusso di log è l'ID HSM; ad esempio, `hsm-nwbbiqbj4jk`.

  In generale, c'è un flusso di log per ogni HSM. Tuttavia, qualsiasi azione che modifica l'ID dell'HSM, ad esempio quando un HSM ha esito negativo e viene sostituito, crea un nuovo flusso di log.

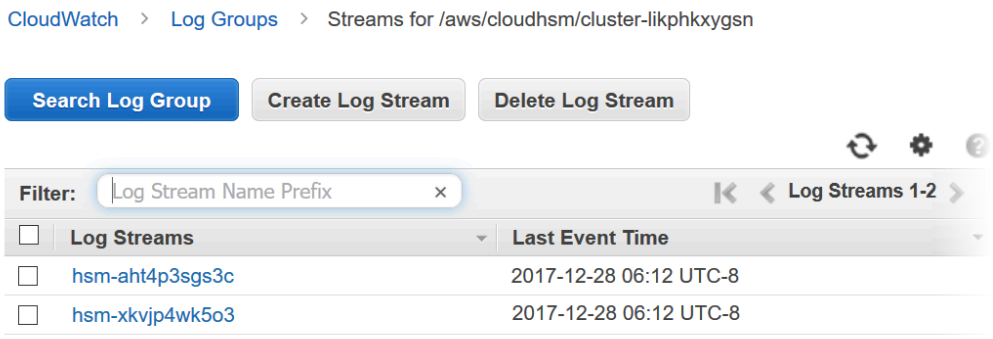
Per ulteriori informazioni sui concetti di CloudWatch Logs, consulta Concepts nella Amazon CloudWatch Logs User Guide.

È possibile visualizzare i log di controllo per un HSM dalla pagina CloudWatch Logs in, dai comandi Logs in Console di gestione AWS, dai CloudWatch cmdlet Logs o dai CloudWatch PowerShellLogs. AWS CLI CloudWatch SDKs Per istruzioni, consulta View Log Data nella Amazon CloudWatch Logs User Guide.

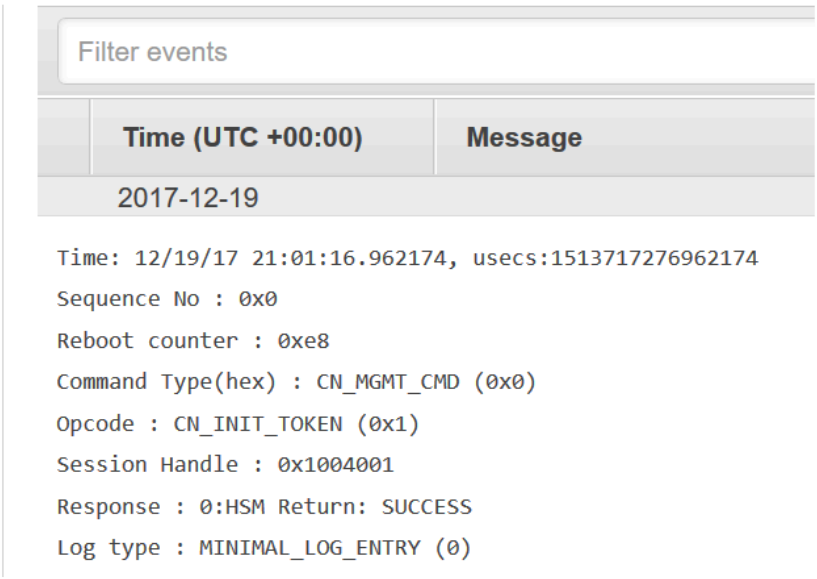
Ad esempio, di seguito è riportata l'immagine del gruppo di log per il cluster `cluster-likphkxygsn` nel Console di gestione AWS.



Quando si sceglie il nome del gruppo di log del cluster, è possibile visualizzare il flusso di log per ciascuno dei membri del HSMs cluster. L'immagine seguente mostra i flussi di log relativi HSMs al `cluster-likphkxygsn` cluster.

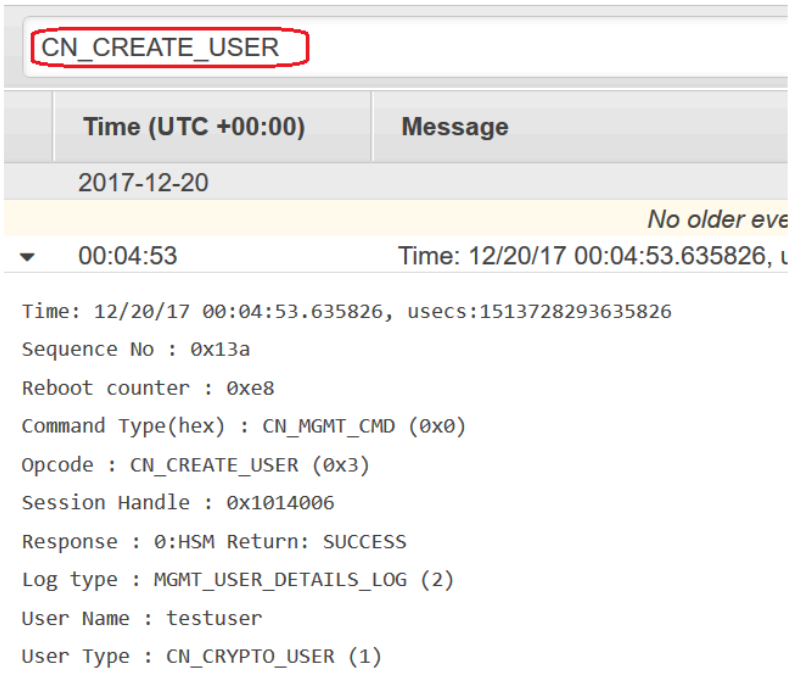


Quando si sceglie un nome per il flusso di log HSM, è possibile visualizzare gli eventi nel log di audit. Ad esempio, questo evento, che ha un numero di sequenza 0x0 e un `Opcode` di `CN_INIT_TOKEN`, è in genere il primo evento per il primo HSM in ogni cluster. Registra l'inizializzazione dell'HSM nel cluster.



È possibile utilizzare tutte le numerose funzionalità di CloudWatch Logs per gestire i registri di controllo. Ad esempio, puoi utilizzare la funzionalità Filtra eventi per trovare un testo particolare in un evento, come `CN_CREATE_USER Opcode`.

Per cercare tutti gli eventi che non includono il testo specificato, aggiungere un segno meno (-) prima del testo. Ad esempio, per trovare gli eventi che non includono `CN_CREATE_USER`, immettere -CN_CREATE_USER.



Interpretazione dei registri di AWS CloudHSM controllo

Gli eventi nei log di audit HSM hanno campi standard. Alcuni tipi di eventi hanno campi aggiuntivi che permettono di acquisire informazioni utili sull'evento. Ad esempio, gli eventi di accesso e di gestione degli utenti includono il nome utente e il tipo di utente. I comandi di gestione delle chiavi includono l'handle della chiave.

Diversi campi forniscono informazioni particolarmente importanti. `Opcode` identifica il comando della gestione in fase di registrazione. `Sequence No` identifica un evento nel flusso di log e indica l'ordine in cui è stato registrato.

Ad esempio, il seguente evento di esempio è il secondo evento (`Sequence No: 0x1`) nel flusso di log per un HSM. Mostra l'HSM che genera una chiave di crittografia della password, parte della routine della sua startup.

```
Time: 12/19/17 21:01:17.140812, usecs:1513717277140812
Sequence No : 0x1
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GEN_PSWD_ENC_KEY (0x1d)
```

```
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

I seguenti campi sono comuni a tutti gli AWS CloudHSM eventi del registro di controllo.

Orario

La data e l'ora in cui si è verificato l'evento nel fuso orario UTC. Il tempo è visualizzato in formato leggibile e in formato Unix in microsecondi.

Riavvia contatore

Un contatore ordinale persistente a 32 bit viene incrementato quando l'hardware HSM viene riavviato.

Tutti gli eventi in un flusso di log hanno lo stesso valore del contatore di riavvio. Tuttavia, il contatore di riavvio potrebbe non essere unico per un flusso di log, in quanto può variare in diverse istanze HSM nello stesso cluster.

Numero sequenza

Un contatore ordinale a 64 bit incrementato per ogni evento di log. Il primo evento in ciascun flusso di log ha un numero di sequenza 0x0. Non ci dovrebbero essere lacune nei valori `Sequence No`. Il numero di sequenza è univoco solo all'interno di un flusso di log.

Tipo di comando

Un valore esadecimale che rappresenta la categoria del comando. I comandi nei flussi di log AWS CloudHSM hanno un tipo di comando `CN_MGMT_CMD` (0x0) o `CN_CERT_AUTH_CMD` (0x9).

Codice operativo

Identifica il comando di gestione che è stato eseguito. Per un elenco dei `Opcode` valori nei log AWS CloudHSM di controllo, vedereAWS CloudHSM riferimento al registro di controllo.

Handle di sessione

Identifica la sessione in cui il comando è stato eseguito e l'evento registrato.

Risposta

Registra la risposta al comando di gestione. È possibile cercare il campo `Response` per i valori `SUCCESS` e `ERROR`.

Tipo di log

Indica il tipo di registro del AWS CloudHSM registro che ha registrato il comando.

- `MINIMAL_LOG_ENTRY (0)`
- `MGMT_KEY_DETAILS_LOG (1)`
- `MGMT_USER_DETAILS_LOG (2)`
- `GENERIC_LOG`

Esempi di eventi di log di audit

Gli eventi in un flusso di log registrano la storia dell'HSM, dalla sua creazione alla sua cancellazione. È possibile utilizzare il registro per esaminare il ciclo di vita del file HSMs e ottenere informazioni dettagliate sul suo funzionamento. Quando si interpretano gli eventi, nota `Opcode`, che indica il comando o l'operazione di gestione e `Sequence No`, che indica l'ordine degli eventi.

Argomenti

- Esempio: inizializza il primo HSM in un cluster
- Eventi di login e logout
- Esempio: creare ed eliminare utenti
- Esempio: creare ed eliminare una coppia di chiavi
- Esempio: generare e sincronizzare una chiave
- Esempio: Esportare una chiave
- Esempio: Importare una chiave
- Esempio: Condividi e Annulla la condivisione di una chiave

Esempio: inizializza il primo HSM in un cluster

Il flusso di log di controllo per il primo HSM di ogni cluster si differenzia notevolmente dai flussi di log degli altri membri del cluster. HSMs Il log di audit per il primo HSM in ogni cluster registra la propria creazione e inizializzazione. I log aggiuntivi HSMs presenti nel cluster, generati dai backup, iniziano con un evento di accesso.

> **Important**
>
> Le seguenti voci di inizializzazione non verranno visualizzate nei CloudWatch log dei cluster inizializzati prima del rilascio della funzionalità di registrazione di audit CloudHSM (30 agosto 2018). Per ulteriori informazioni vedi Cronologia dei documenti.

Gli eventi di esempio seguenti appaiono nel flusso di log per il primo HSM in un cluster. Il primo evento nel log, quello con `Sequence No 0x0`, rappresenta il comando per inizializzare l'HSM (`CN_INIT_TOKEN`). La risposta indica che il comando è stato eseguito con successo (`Response : 0: HSM Return: SUCCESS`).

```
Time: 12/19/17 21:01:16.962174, usecs:1513717276962174
Sequence No : 0x0
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INIT_TOKEN (0x1)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

Il secondo evento in questo flusso di log di esempio (`Sequence No 0x1`) registra il comando per creare la chiave di crittografia della password utilizzata da HSM (`CN_GEN_PSWD_ENC_KEY`).

Si tratta di una tipica sequenza di avvio per il primo HSM in ogni cluster. Poiché nello stesso cluster sono HSMs presenti cloni successivi del primo, utilizzano la stessa chiave di crittografia delle password.

```
Time: 12/19/17 21:01:17.140812, usecs:1513717277140812
Sequence No : 0x1
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GEN_PSWD_ENC_KEY (0x1d)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

Il terzo evento in questo flusso di log di esempio (`Sequence No 0x2`) è la creazione dell' utente dell'applicazione (AU), cioè il servizio AWS CloudHSM . Gli eventi che coinvolgono gli utenti HSM includono campi aggiuntivi per il nome utente e il tipo di utente.

```
Time: 12/19/17 21:01:17.174902, usecs:1513717277174902
Sequence No : 0x2
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_APPLIANCE_USER (0xfc)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : app_user
User Type : CN_APPLIANCE_USER (5)
```

Il quarto evento in questo flusso di log di esempio (`Sequence No 0x3`) registra l'evento `CN_INIT_DONE` che completa l'inizializzazione dell'HSM.

```
Time: 12/19/17 21:01:17.298914, usecs:1513717277298914
Sequence No : 0x3
Reboot counter : 0xe8
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INIT_DONE (0x95)
Session Handle : 0x1004001
Response : 0:HSM Return: SUCCESS
Log type : MINIMAL_LOG_ENTRY (0)
```

È possibile seguire i restanti eventi nella sequenza di avvio. Questi eventi possono includere diversi eventi di accesso e disconnessione e la generazione della chiave di crittografia (KEK). Il seguente evento registra il comando che modifica la password del precrypto officer (PRECO). Questo comando attiva il cluster.

```
Time: 12/13/17 23:04:33.846554, usecs:1513206273846554
Sequence No: 0x1d
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_CHANGE_PSWD (0x9)
Session Handle: 0x2010003
Response: 0:HSM Return: SUCCESS
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: admin
User Type: CN_CRYPTO_PRE_OFFICER (6)
```

Eventi di login e logout

Durante l'interpretazione del log di audit, sono da notare gli eventi che registrano l'accesso e la disconnessione degli utenti dall'HSM. Questi eventi aiutano a capire quale utente è responsabile per i comandi di gestione che appaiono in sequenza tra i comandi di login e di logout.

Ad esempio, questa voce di log registra un login da parte di un crypto officer denominato `admin`. Il numero di sequenza, `0x0`, indica che questo è il primo evento in questo flusso di log.

Quando un utente accede a un HSM, anche l'altro HSMs utente del cluster registra un evento di accesso per l'utente. È possibile trovare gli eventi di accesso corrispondenti nei flussi di registro di altri HSMs membri del cluster poco dopo l'evento di accesso iniziale.

```
Time: 01/16/18 01:48:49.824999, usecs:1516067329824999
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
User Type : CN_CRYPTO_OFFICER (2)
```

Il seguente evento di esempio registra la disconnessione del crypto officer `admin`. Il numero di sequenza, `0x2`, indica che questo è il terzo evento in questo flusso di log.

Se l'utente che ha effettuato l'accesso chiude la sessione senza uscire, il flusso di log include un `CN_APP_FINALIZE` o un evento di chiusura di sessione (`CN_SESSION_CLOSE`) invece di un evento `CN_LOGOUT`. A differenza degli eventi di login, questo evento di disconnessione in genere viene registrato solo dall'HSM che esegue il comando.

```
Time: 01/16/18 01:49:55.993404, usecs:1516067395993404
Sequence No : 0x2
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGOUT (0xe)
Session Handle : 0x7014000
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
```

```
User Type : CN_CRYPTO_OFFICER (2)
```

Se un tentativo di accesso ha esito negativo perché il nome utente non è valido, l'HSM registra un evento `CN_LOGIN` con il nome utente e il tipo forniti nel comando di accesso. La risposta visualizza il messaggio di errore 157 che spiega che il nome utente non esiste.

```
Time: 01/24/18 17:41:39.037255, usecs:1516815699037255
Sequence No : 0x4
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 157:HSM Error: user isn't initialized or user with this name doesn't exist
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : ExampleUser
User Type : CN_CRYPTO_USER (1)
```

Se un tentativo di accesso ha esito negativo perché la password non è valida, l'HSM registra un evento `CN_LOGIN` con il nome utente e il tipo forniti nel comando di accesso. La risposta mostra il messaggio di errore con il codice `RET_USER_LOGIN_FAILURE`.

```
Time: 01/24/18 17:44:25.013218, usecs:1516815865013218
Sequence No : 0x5
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 163:HSM Error: RET_USER_LOGIN_FAILURE
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPTO_USER (1)
```

Esempio: creare ed eliminare utenti

Questo esempio mostra gli eventi di log registrati quando un crypto officer (CO) crea ed elimina utenti.

Il primo evento registra un CO, `admin`, che accede all'HSM. Il numero di sequenza, `0x0`, indica che questo è il primo evento nel flusso di log. Il nome e il tipo di utente che ha effettuato l'accesso sono inclusi nell'evento.

```
Time: 01/16/18 01:48:49.824999, usecs:1516067329824999
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : admin
User Type : CN_CRYPTO_OFFICER (2)
```

L'evento successivo nel flusso di log (sequenza `0x1`) registra il CO che crea un nuovo crypto user (CU). Il nome e il tipo del nuovo utente sono inclusi nell'evento.

```
Time: 01/16/18 01:49:39.437708, usecs:1516067379437708
Sequence No : 0x1
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_USER (0x3)
Session Handle : 0x7014006
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : bob
User Type : CN_CRYPTO_USER (1)
```

Quindi, il CO crea un altro crypto officer, `alice`. Il numero di sequenza indica che questa azione segue quella precedente, senza altre operazioni intermedie.

```
Time: 01/16/18 01:49:55.993404, usecs:1516067395993404
Sequence No : 0x2
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_CREATE_CO (0x4)
Session Handle : 0x7014007
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : alice
User Type : CN_CRYPTO_OFFICER (2)
```

Successivamente, il CO denominato `admin` effettua l'accesso ed elimina il crypto officer denominato `alice`. L'HSM registra un evento `CN_DELETE_USER`. Il nome e il tipo dell'utente eliminato sono inclusi nell'evento.

```
Time: 01/23/18 19:58:23.451420, usecs:1516737503451420
Sequence No : 0xb
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_DELETE_USER (0xa1)
Session Handle : 0x7014007
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : alice
User Type : CN_CRYPTO_OFFICER (2)
```

Esempio: creare ed eliminare una coppia di chiavi

Questo esempio illustra gli eventi che vengono registrati in un log di audit HSM al momento della creazione ed eliminazione di una coppia di chiavi.

Il seguente evento registra il crypto user (CU) denominato `crypto_user` che si collega all'HSM.

```
Time: 12/13/17 23:09:04.648952, usecs:1513206544648952
Sequence No: 0x28
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_LOGIN (0xd)
Session Handle: 0x2014005
Response: 0:HSM Return: SUCCESS
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: crypto_user
User Type: CN_CRYPTO_USER (1)
```

Poi, il CU genera una coppia di chiavi (`CN_GENERATE_KEY_PAIR`). La chiave privata presenta l'handle `131079`. La chiave pubblica ha l'handle `131078`.

```
Time: 12/13/17 23:09:04.761594, usecs:1513206544761594
Sequence No: 0x29
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_GENERATE_KEY_PAIR (0x19)
Session Handle: 0x2014004
```

```
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131079
Public Key Handle: 131078
```

Il CU immediatamente elimina la coppia di chiavi. Un evento CN_DISTRUGGI_OGGETTO registra l'eliminazione della chiave pubblica (131078).

```
Time: 12/13/17 23:09:04.813977, usecs:1513206544813977
Sequence No: 0x2a
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_DESTROY_OBJECT (0x11)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131078
Public Key Handle: 0
```

Quindi, un secondo evento CN_DESTROY_OBJECT registra l'eliminazione della chiave privata (131079).

```
Time: 12/13/17 23:09:04.815530, usecs:1513206544815530
Sequence No: 0x2b
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_DESTROY_OBJECT (0x11)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
Log type: MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle: 131079
Public Key Handle: 0
```

Infine, il CU di disconnette.

```
Time: 12/13/17 23:09:04.817222, usecs:1513206544817222
Sequence No: 0x2c
Reboot counter: 0xe8
Command Type(hex): CN_MGMT_CMD (0x0)
Opcode: CN_LOGOUT (0xe)
Session Handle: 0x2014004
Response: 0:HSM Return: SUCCESS
```

```
Log type: MGMT_USER_DETAILS_LOG (2)
User Name: crypto_user
User Type: CN_CRYPTO_USER (1)
```

Esempio: generare e sincronizzare una chiave

Questo esempio mostra l'effetto della creazione di una chiave in un cluster con più HSMs chiavi. La chiave viene generata su un HSM, estratta dall'HSM come oggetto mascherato e inserita nell'altro HSMs come oggetto mascherato.

> **Note**
>
> Gli strumenti del client potrebbero non riuscire a sincronizzare la chiave. Oppure il comando potrebbe includere il min_srv parametro, che sincronizza la chiave solo con il numero specificato di. HSMs In entrambi i casi, il AWS CloudHSM servizio sincronizza la chiave con l'altra del HSMs cluster. Poiché i HSMs registri registrano solo i comandi di gestione lato client, la sincronizzazione lato server non viene registrata nel registro HSM.

Prima di tutto considera il flusso di log dell'HSM che riceve ed esegue i comandi. Il flusso di log viene denominato per l'ID dell'HSM, `hsm-abcde123456`, ma l'ID dell'HSM non appare negli eventi del log.

In primo luogo, il crypto user (CU)`testuser` esegue l'accesso all'HSM `hsm-abcde123456`.

```
Time: 01/24/18 00:39:23.172777, usecs:1516754363172777
Sequence No : 0x0
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0xc008002
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPTO_USER (1)
```

La CU esegue un comando per generare una chiave simmetrica. exSymKey L'HSM `hsm-abcde123456` genera una chiave simmetrica con un handle di `262152`. L'HSM registra un evento `CN_GENERATE_KEY` nel proprio log.

```
Time: 01/24/18 00:39:30.328334, usecs:1516754370328334
Sequence No : 0x1
```

```
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_GENERATE_KEY (0x17)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

L'evento successivo nel flusso di log per `hsm-abcde123456` registra il primo passo nel processo di sincronizzazione delle chiavi. La nuova chiave (handle `262152`) viene estratta dall'HSM come oggetto mascherato.

```
Time: 01/24/18 00:39:30.330956, usecs:1516754370330956
Sequence No : 0x2
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_EXTRACT_MASKED_OBJECT_USER (0xf0)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

Ora considera il flusso di log per l'HSM `hsm-zyxwv987654`, un altro HSM nello stesso cluster. Questo flusso di log include anche un evento di accesso per il CU `testuser`. Il valore temporale mostra che si verifica subito dopo il login dell'utente all'HSM `hsm-abcde123456`.

```
Time: 01/24/18 00:39:23.199740, usecs:1516754363199740
Sequence No : 0xd
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7004004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPTO_USER (1)
```

Questo flusso di log per questo HSM non dispone di un evento `CN_GENERATE_KEY`. Tuttavia, dispone di un evento che registra la sincronizzazione della chiave per questo HSM. L'evento

`CN_INSERT_MASKED_OBJECT_USER` registra la ricezione della chiave `262152` come oggetto mascherato. Ora la chiave `262152` esiste su entrambi HSMs nel cluster.

```
Time: 01/24/18 00:39:30.408950, usecs:1516754370408950
Sequence No : 0xe
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INSERT_MASKED_OBJECT_USER (0xf1)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 262152
Public Key Handle : 0
```

Quando l'utente CU si disconnette, questo evento `CN_LOGOUT` appare solo nel flusso di log dell'HSM che ha ricevuto i comandi.

Esempio: Esportare una chiave

Questo esempio mostra gli eventi del registro di controllo che vengono registrati quando un utente crittografico (CU) esporta le chiavi da un cluster con più HSMs chiavi.

L'evento seguente registra il CU (`testuser`) che esegue l'accesso a key_mgmt_util.

```
Time: 01/24/18 19:42:22.695884, usecs:1516822942695884
Sequence No : 0x26
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_LOGIN (0xd)
Session Handle : 0x7004004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_USER_DETAILS_LOG (2)
User Name : testuser
User Type : CN_CRYPTO_USER (1)
```

La CU esegue un exSymKeycomando per esportare la chiave7, una chiave AES a 256 bit. Il comando utilizza la chiave6, una chiave AES a 256 bit su HSMs, come chiave di avvolgimento.

L'HSM che riceve il comando registra un evento `CN_WRAP_KEY` per la chiave 7 in fase di esportazione.

```
Time: 01/24/18 19:51:12.860123, usecs:1516823472860123
```

```
Sequence No : 0x27
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_WRAP_KEY (0x1a)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 7
Public Key Handle : 0
```

Quindi, l'HSM registra un evento CN_NIST_AES_WRAP per la chiave di wrapping, la chiave 6. Viene eseguito il wrapping della chiave e subito dopo annullato, ma l'HSM registra solo un evento.

```
Time: 01/24/18 19:51:12.905257, usecs:1516823472905257
Sequence No : 0x28
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_NIST_AES_WRAP (0x1e)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 6
Public Key Handle : 0
```

Il comando exSymKey scrive la chiave esportata su un file, ma non cambia la chiave sull'HSM. Di conseguenza, non ci sono eventi corrispondenti nei log di altri HSMs membri del cluster.

Esempio: Importare una chiave

Questo esempio mostra gli eventi del registro di controllo che vengono registrati quando si importano le chiavi HSMs in un cluster. In questo esempio, l'utente crittografico (CU) utilizza il imSymKeycomando per importare una chiave AES in. HSMs Il comando utilizza la chiave 6 come chiave di wrapping.

L'HSM che riceve i comandi registra per prima cosa un evento CN_NIST_AES_WRAP per la chiave 6, la chiave di wrapping

```
Time: 01/24/18 19:58:23.170518, usecs:1516823903170518
Sequence No : 0x29
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_NIST_AES_WRAP (0x1e)
```

```
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 6
Public Key Handle : 0
```

Quindi, l'HSM registra un evento CN_UNWRAP_KEY che rappresenta l'operazione di importazione. Alla chiave importata viene assegnato un handle di 11.

```
Time: 01/24/18 19:58:23.200711, usecs:1516823903200711
Sequence No : 0x2a
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_UNWRAP_KEY (0x1b)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

Quando viene generata o importata una nuova chiave, gli strumenti client tentano automaticamente di sincronizzare la nuova chiave con altre chiavi del HSMs cluster. In questo caso, l'HSM registra un evento CN_EXTRACT_MASKED_OBJECT_USER quando la chiave 11 viene estratta dall'HSM come oggetto mascherato.

```
Time: 01/24/18 19:58:23.203350, usecs:1516823903203350
Sequence No : 0x2b
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_EXTRACT_MASKED_OBJECT_USER (0xf0)
Session Handle : 0x7004003
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

I flussi di log degli altri membri HSMs del cluster riflettono l'arrivo della chiave appena importata.

Ad esempio, questo evento è stato registrato nel flusso di log di un HSM differente nello stesso cluster. Questo evento CN_INSERT_MASKED_OBJECT_USER registra l'arrivo di un oggetto mascherato che rappresenta la chiave 11.

```
Time: 01/24/18 19:58:23.286793, usecs:1516823903286793
Sequence No : 0xb
Reboot counter : 0x107
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_INSERT_MASKED_OBJECT_USER (0xf1)
Session Handle : 0xc008004
Response : 0:HSM Return: SUCCESS
Log type : MGMT_KEY_DETAILS_LOG (1)
Priv/Secret Key Handle : 11
Public Key Handle : 0
```

Esempio: Condividi e Annulla la condivisione di una chiave

Questo esempio illustra l'evento di log di controllo che viene registrato quando un crypto user (CU) condivide o annulla la condivisione della chiave privata ECC con altri utenti di crittografia. Il CU usa il comando shareKey e offre la chiave di gestione, l'ID utente e il valore 1 per condividere la chiave o il valore 0 per annullare la condivisione della chiave.

In questo esempio, l'HSM che riceve il comando, registra un evento CM_SHARE_OBJECT che rappresenta l'operazione di condivisione.

```
Time: 02/08/19 19:35:39.480168, usecs:1549654539480168
Sequence No : 0x3f
Reboot counter : 0x38
Command Type(hex) : CN_MGMT_CMD (0x0)
Opcode : CN_SHARE_OBJECT (0x12)
Session Handle : 0x3014007
Response : 0:HSM Return: SUCCESS
Log type : UNKNOWN_LOG_TYPE (5)
```

AWS CloudHSM riferimento al registro di controllo

AWS CloudHSM registra i comandi di gestione HSM negli eventi del registro di controllo. Ogni evento ha un valore di codice operativo (Opcode) che identifica l'operazione avvenuta e la relativa risposta. È possibile utilizzare i valori Opcode per cercare, ordinare e filtrare i log.

La tabella seguente definisce i Opcode valori in un registro di AWS CloudHSM controllo.

Codice operazione (Opcode)	Description
Login utente: questi eventi includono il nome utente e il tipo di utente	
`CN_LOGIN (0xd)`	Login utente
`CN_LOGOUT (0xe)`	Logout utente
`CN_APP_FINALIZE`	La connessione con l'HSM è stata chiusa. Tutte le chiavi di sessione o i token di quorum da questa connessione sono stati eliminati.
`CN_CLOSE_SESSION`	La sessione con l'HSM è stata chiusa. Tutte le chiavi di sessione o i token di quorum di questa sessione sono stati eliminati.
Gestione utenti: questi eventi includono il nome utente e il tipo di utente	
`CN_CREATE_USER (0x3)`	Creazione di un utente di crittografia (CU)
`CN_CREATE_CO`	Creazione di un responsabile della crittografia (CO)
`CN_DELETE_USER`	Eliminazione di un utente
`CN_CHANGE_PSWD`	Modifica della password di un utente
`CN_SET_M_VALUE`	Imposta l'autenticazione del quorum (M of N) per un'azione dell'utente
`CN_APPROVE_TOKEN`	Approva un token di autenticazione quorum per un'azione dell'utente
`CN_DELETE_TOKEN`	Eliminare uno o più token di quorum
`CN_GET_TOKEN`	Richiedi un token di firma per avviare un'operazione di quorum
Gestione chiavi: questi eventi includono l'handle della chiave.	
`CN_GENERATE_KEY`	Generare una chiave simmetrica

Codice operazione (Opcode)	Description
`CN_GENERATE_KEY_PAIR (0x19)`	Genera una coppia di key pair asimmetrica
`CN_CREATE_OBJECT`	Importare una chiave pubblica (senza il wrapping)
`CN_MODIFY_OBJECT`	Imposta un attributo chiave
`CN_DESTROY_OBJECT (0x11)`	Eliminazione di una chiave di sessione
`CN_TOMBSTONE_OBJECT`	Eliminazione di una chiave token
`CN_SHARE_OBJECT`	Condivisione e annullamento della condivisione di una chiave
`CN_WRAP_KEY`	Esportare una copia crittografata di una chiave (wrapKey)
`CN_UNWRAP_KEY`	Importare una copia crittografata di una chiave (unwrapKey)
`CN_DERIVE_KEY`	Deriva una chiave simmetrica da una chiave esistente
`CN_NIST_AES_WRAP`	Crittografa o decrittografa una chiave con una chiave AES
`CN_INSERT_MASKED_OBJECT_USER`	Inserisci una chiave crittografata con gli attributi di un altro HSM nel cluster.
`CN_EXTRACT_MASKED_OBJECT_USER`	Racchiude e crittografa una chiave con gli attributi dell'HSM per inviarla a un altro HSM del cluster.
Session Management	
`CN_ENCRYPT_SESSION_V2 (0x107)`	Stabilisce una sessione crittografata autentica ta. end-to-end

Codice operazione (Opcode)	Description
`END_MARKER_OPCODE (0xffff)`	Inserisce un indicatore finale nel buffer dei registri di controllo che indica che non sono consentiti altri comandi registrabili sull'HSM
Back up HSMs	
`CN_BACKUP_BEGIN`	Inizia il processo di backup
`CN_BACKUP_END`	Completato il processo di backup
`CN_RESTORE_BEGIN`	Inizia il ripristino da un backup
`CN_RESTORE_END`	È stato completato il processo di ripristino da un backup
Certificate-Based Authentication	
`CN_CERT_AUTH_STORE_CERT`	Memorizza il certificato del cluster
HSM Instance Commands	
`CN_INIT_TOKEN (0x1)`	Avvia il processo di inizializzazione HSM
`CN_INIT_DONE`	Il processo di inizializzazione HSM è terminato
`CN_GEN_KEY_ENC_KEY`	Generare una chiave di cifratura a chiave (KEK)
`CN_GEN_PSWD_ENC_KEY (0x1d)`	Generare una chiave di cifratura della password (PEK)
HSM crypto commands	
`CN_FIPS_RAND`	Genera un numero casuale conforme a FIPS [1]

[1] Viene registrato solo per i cluster hsm1.medium.

Ottenere CloudWatch metriche per AWS CloudHSM

CloudWatch Usalo per monitorare il AWS CloudHSM cluster in tempo reale. I parametri possono essere raggruppati per regione, ID del cluster oppure ID del cluster e ID dell'HSM.

Lo spazio dei nomi `AWS/CloudHSM` include i parametri descritti di seguito:

Metrica	Description
HsmUnhealthy	L'istanza HSM non funziona correttamente. AWS CloudHSM sostituisce automaticamente le istanze non integre per te. Puoi scegliere di espandere la dimensione del cluster in modo proattivo per ridurre l'impatto delle prestazioni durante la sostituzione dell'HSM.
HsmTemperature [1]	Temperatura di collegamento del processore hardware. Se la temperatura raggiunge 110 gradi centigradi, il sistema si arresta.
HsmKeysSessionOccupied	Numero di chiavi di sessione utilizzate dall'istanza HSM.
HsmKeysTokenOccupied	Numero di chiavi token utilizzate dall'istanza HSM e dal cluster.
HsmSslCtxsOccupied [1]	Il numero di canali end-to-end crittografati attualmente stabiliti per l'istanza HSM. Sono ammessi fino a 2048 canali.
HsmSessionCount	Numero di connessioni aperte all'istanza HSM. Sono ammesse fino a 2048 connessioni. Per impostazione predefinita, il daemon client è configurato per aprire due sessioni con ciascuna istanza HSM in un canale crittografato. end-to-end AWS CloudHSM può anche avere fino a 2 connessioni aperte con l'HSM per monitorare lo stato di. HSMs
HsmUsersAvailable	Numero di utenti aggiuntivi che è possibile creare. Ciò equivale al numero massimo di utenti (elencati in HsmUsersMax) meno gli utenti creati fino ad oggi.
HsmUsersMax [1]	Numero massimo di utenti che è possibile creare nell'istanza HSM.

Metrica	Description
Interface Eth2OctetsInput [1]	La somma cumulativa del traffico in entrata verso l'HSM finora.
Interface Eth2Octet sOutput [1]	La somma cumulativa del traffico in uscita verso l'HSM finora.

- [1] Questa metrica non è disponibile per hsm2m.medium.

AWS CloudHSM informazioni sulle prestazioni

Per AWS CloudHSM i cluster di produzione, è necessario disporre di almeno due istanze HSM (Hardware Security Module) distribuite in diverse zone di disponibilità in una regione. Ti consigliamo di effettuare test di carico sul cluster per determinare il carico massimo da prevedere, quindi di aggiungere un altro modulo HSM per garantire un'elevata disponibilità. Per le applicazioni che richiedono la durabilità delle chiavi appena generate, ti consigliamo almeno tre istanze HSM distribuite su diverse zone di disponibilità in una regione.

Dati di prestazioni

Le prestazioni dei AWS CloudHSM cluster variano in base al carico di lavoro specifico. Per aumentare le prestazioni, puoi aggiungere ulteriori istanze HSM ai tuoi cluster. Le prestazioni possono variare in base alla configurazione, alla dimensione dei dati e al carico aggiuntivo delle applicazioni sulle istanze EC2. Consigliamo di testare il carico dell'applicazione per determinare le esigenze di scalabilità.

La tabella seguente mostra le prestazioni approssimative degli algoritmi crittografici comuni in esecuzione su un'istanza EC2 con istanze hsm1.medium.

Dati sulle prestazioni per hsm1.medium

Operation	Two-HSM cluster[1]	Three-HSM cluster[2]	Six-HSM cluster[3]
Segno RSA a 2048 bit	2.000 ops/sec	3.000 ops/sec	5.000 ops/sec
Segno EC P256	500 ops/sec	750 ops/sec	1.500 ops/sec

La tabella seguente mostra le prestazioni approssimative dei comuni algoritmi crittografici in esecuzione su un'istanza EC2 con hsm2m.medium.

Dati sulle prestazioni per hsm2m.medium

Operation	Two-HSM cluster[1]	Three-HSM cluster[2]	Six-HSM cluster[3]
Segno RSA a 2048 bit	2.000 ops/sec	3.000 ops/sec	5.000 ops/sec
Segno EC P256	3.000 ops/sec	4.500 ops/sec	7.000 ops/sec

Operation	Two-HSM cluster[1]	Three-HSM cluster[2]	Six-HSM cluster[3]
ML-DSA-44 segno	80 ops/sec	140 ops/sec	190 ops/sec

- [1] Un cluster a due HSM con l'applicazione multithread di Java in esecuzione su un'istanza EC2 c4.large con un HSM nella stessa zona di disponibilità dell'istanza EC2.
- [2] Un cluster a tre HSM con l'applicazione multithread di Java in esecuzione su un'istanza EC2 c4.large con un HSM nella stessa zona di disponibilità dell'istanza EC2.
- [3] Un cluster a sei HSM con l'applicazione multithread di Java in esecuzione su un'istanza EC2 c4.large con due HSM nella stessa zona di disponibilità dell'istanza EC2.

Limitazione HSM

Quando il carico di lavoro supera la capacità HSM del cluster, riceverai messaggi di errore che indicano che gli HSM sono occupati o con limitazioni. Per informazioni dettagliate su cosa fare quando ciò accade, consulta Limitazione HSM

Sicurezza in AWS CloudHSM

La sicurezza del cloud AWS è la massima priorità. Come AWS cliente, puoi beneficiare di data center e architetture di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra voi. AWS Il modello di responsabilità condivisa descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi in Cloud AWS. 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 informazioni sui programmi di conformità a cui si applicano AWS CloudHSM, consulta la sezione AWS Servizi nell'ambito del programma di conformità di conformità.
- 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.

Questa documentazione ti aiuta a capire come applicare il modello di responsabilità condivisa durante l'utilizzo AWS CloudHSM. Gli argomenti seguenti mostrano come effettuare la configurazione per AWS CloudHSM soddisfare i tuoi obiettivi di sicurezza e conformità. Imparerai anche come utilizzare altri servizi AWS che ti aiutano a monitorare e proteggere AWS CloudHSM le tue risorse.

Indice

- Controlla l'accesso alle API con le policy IAM
- Protezione dei dati in AWS CloudHSM
- Gestione delle identità e degli accessi per AWS CloudHSM
- Convalida della conformità per AWS CloudHSM
- Resilienza in AWS CloudHSM
- Sicurezza dell'infrastruttura in AWS CloudHSM
- AWS CloudHSM ed endpoint VPC
- Gestione degli aggiornamenti in AWS CloudHSM

Controlla l'accesso alle API con le policy IAM

Aggiornamento delle policy IAM a IPv6

AWS CloudHSM i clienti utilizzano le policy IAM per controllare l'accesso alle AWS CloudHSM API e impedire a qualsiasi indirizzo IP al di fuori dell'intervallo configurato di accedere alle API. AWS CloudHSM

Il cloudhsmv2. *<region>*L'endpoint dual-stack .api.aws in cui AWS CloudHSM sono ospitate le API supporta IPv6 oltre a IPv4.

I clienti che devono supportare sia IPv4 che IPv6 devono aggiornare le proprie politiche di filtro degli indirizzi IP per gestire gli indirizzi IPv6, altrimenti ciò influirà sulla loro capacità di connettersi tramite IPv6. AWS CloudHSM

Chi dovrebbe effettuare l'aggiornamento?

I clienti che utilizzano il doppio indirizzamento con policy contenenti AWS:SourceIP sono interessati da questo aggiornamento. Il doppio indirizzamento significa che la rete supporta sia IPv4 che IPv6.

Se si utilizza il doppio indirizzamento, è necessario aggiornare le politiche IAM attualmente configurate con indirizzi in formato IPv4 per includere gli indirizzi in formato IPv6.

Per ricevere assistenza in caso di problemi di accesso, contattare Supporto.

> **Note**
>
> I seguenti clienti non sono interessati da questo aggiornamento:
>
> - Clienti che utilizzano solo reti IPv4.

Cos'è IPv6?

IPv6 è lo standard IP di nuova generazione destinato a sostituire eventualmente l'IPv4. La versione precedente, IPv4, utilizza uno schema di indirizzamento a 32 bit per supportare 4,3 miliardi di dispositivi. L'IPv6 utilizza invece l'indirizzamento a 128 bit per supportare circa 340 trilioni di trilioni di trilioni di dispositivi (ovvero da 2 alla 128a potenza).

Per maggiori dettagli, consulta la pagina web VPC IPv6.

```
2001:cdba:0000:0000:0000:0000:3257:9652
2001:cdba:0:0:0:0:3257:9652
2001:cdba::3257:965
```

Aggiornamento di una policy IAM per IPv6

Le policy IAM vengono attualmente utilizzate per impostare un intervallo consentito di indirizzi IP utilizzando il `aws:SourceIp` filtro.

Il doppio indirizzamento supporta sia il traffico IPv4 che IPv6. Se la rete utilizza il doppio indirizzamento, è necessario aggiornare tutte le policy IAM utilizzate per il filtraggio degli indirizzi IP in modo da includere gli intervalli di indirizzi IPv6.

Ad esempio, la politica riportata di seguito identifica gli intervalli di indirizzi IPv4 consentiti e contenuti nell'elemento. `192.0.2.0.*` `203.0.113.0.*` `Condition`

```
# https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_policies_examples_aws_deny-ip.html
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "*aws:SourceIp*": [
                    "*192.0.2.0/24*",
                    "*203.0.113.0/24*"
                ]
            },
            "Bool": {
                "aws:ViaAWSService": "false"
            }
        }
    }
}
```

Per aggiornare questa politica, modifica l'`Condition`elemento per includere gli intervalli di indirizzi IPv6 e. `2001:DB8:1234:5678::/64` `2001:cdba:3257:8593::/64`

> **Note**
>
> NON RIMUOVERE gli indirizzi IPv4 esistenti perché sono necessari per la compatibilità con le versioni precedenti.

```
"Condition": {
                "NotIpAddress": {
                    "*aws:SourceIp*": [
                        "*192.0.2.0/24*", <<DO NOT REMOVE existing IPv4 address>>
                        "*203.0.113.0/24*", <<DO NOT REMOVE existing IPv4 address>>
                        "*2001:DB8:1234:5678::/64*", <<New IPv6 IP address>>
                        "*2001:cdba:3257:8593::/64*" <<New IPv6 IP address>>
                    ]
                },
                "Bool": {
                    "aws:ViaAWSService": "false"
                }
            }
```

Verifica che il tuo client supporti IPv6

Clienti che utilizzano cloudhsmv2. Si consiglia agli endpoint {region} .api.aws di verificare se sono in grado di connettersi ad esso. I passaggi seguenti descrivono come eseguire la verifica.

Questo esempio utilizza Linux e curl versione 8.6.0 e utilizza gli endpoint di AWS CloudHSM servizio con endpoint abilitati per IPv6 situati all'endpoint api.aws.

> **Note**
>
> Passa alla stessa regione in cui si trova il client. Regione AWS In questo esempio, utilizziamo l'endpoint Stati Uniti orientali (Virginia settentrionale) - `us-east-1`.

1. Determina se l'endpoint si risolve con un indirizzo IPv6 utilizzando il seguente comando. `dig`

```
dig +short AAAA cloudhsmv2.us-east-1.api.aws
2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
```

2. Determina se la rete client può stabilire una connessione IPv6 utilizzando il seguente comando. `curl` Un codice di risposta 404 indica che la connessione è riuscita, mentre un codice di risposta 0 indica che la connessione non è riuscita.

```
curl --ipv6 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 2600:1f18:e2f:4e05:1a8a:948e:7c08:c1c3
response code: 404
```

Se è stato identificato un IP remoto e il codice di risposta no`0`, è stata stabilita correttamente una connessione di rete all'endpoint utilizzando IPv6. L'IP remoto deve essere un indirizzo IPv6 perché il sistema operativo deve selezionare il protocollo valido per il client. Se l'IP remoto non è un indirizzo IPv6, usa il comando seguente per `curl` forzare l'uso di IPv4.

```
curl --ipv4 -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 3.123.154.250
response code: 404
```

Se l'IP remoto è vuoto o il codice di risposta lo è`0`, lo è la rete del client o il percorso di rete verso l'endpoint. IPv4-only È possibile verificare questa configurazione con il seguente `curl` comando.

```
curl -o /dev/null --silent -w "\nremote ip: %{remote_ip}\nresponse code:
 %{response_code}\n" https://cloudhsmv2.us-east-1.api.aws

remote ip: 3.123.154.250
response code: 404
```

Protezione dei dati in AWS CloudHSM

Ulteriori informazioni su come il modello di responsabilità condivisa AWS si applica alla protezione dei dati in AWS CloudHSM. 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 CloudHSM 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.

Crittografia dei dati a riposo

Quando AWS CloudHSM esegue un backup da un HSM, l'HSM crittografa i dati prima di inviarli a. AWS CloudHSM I dati vengono crittografati utilizzando una chiave di crittografia unica e temporanea. Per ulteriori informazioni, consulta AWS CloudHSM backup dei cluster.

Crittografia dei dati in transito

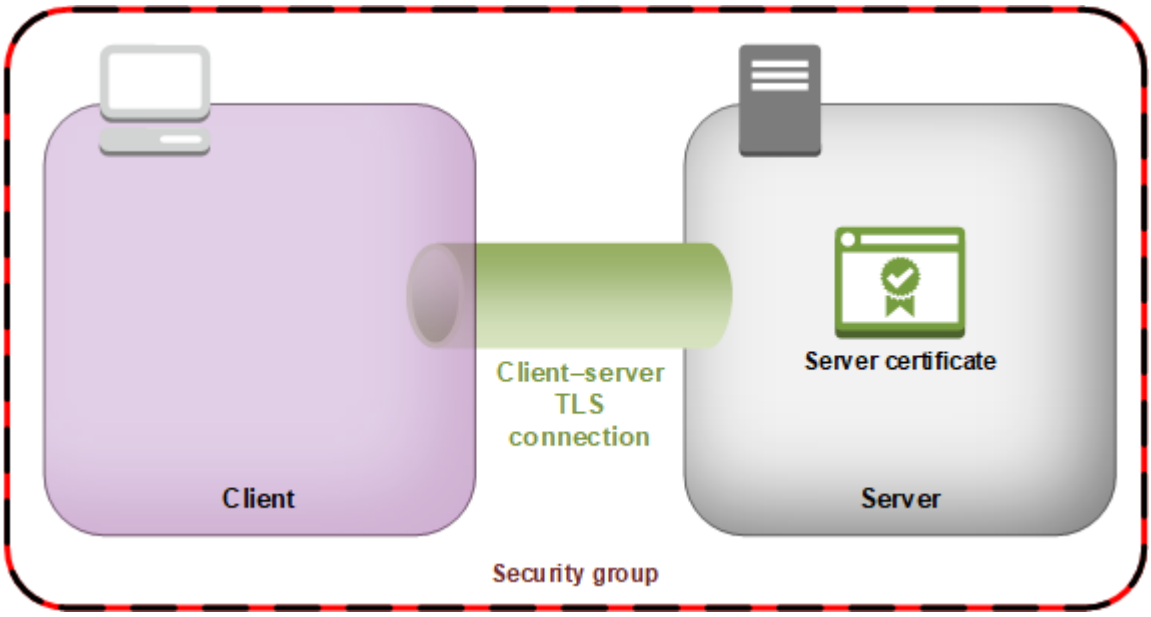
La comunicazione tra il AWS CloudHSM client e l'HSM nel cluster è crittografata da un capo all'altro. Questa comunicazione può essere decifrata solo dal client e dagli HSM. Per ulteriori informazioni, consulta End-to-end crittografia.

AWS CloudHSM crittografia end-to-end del client

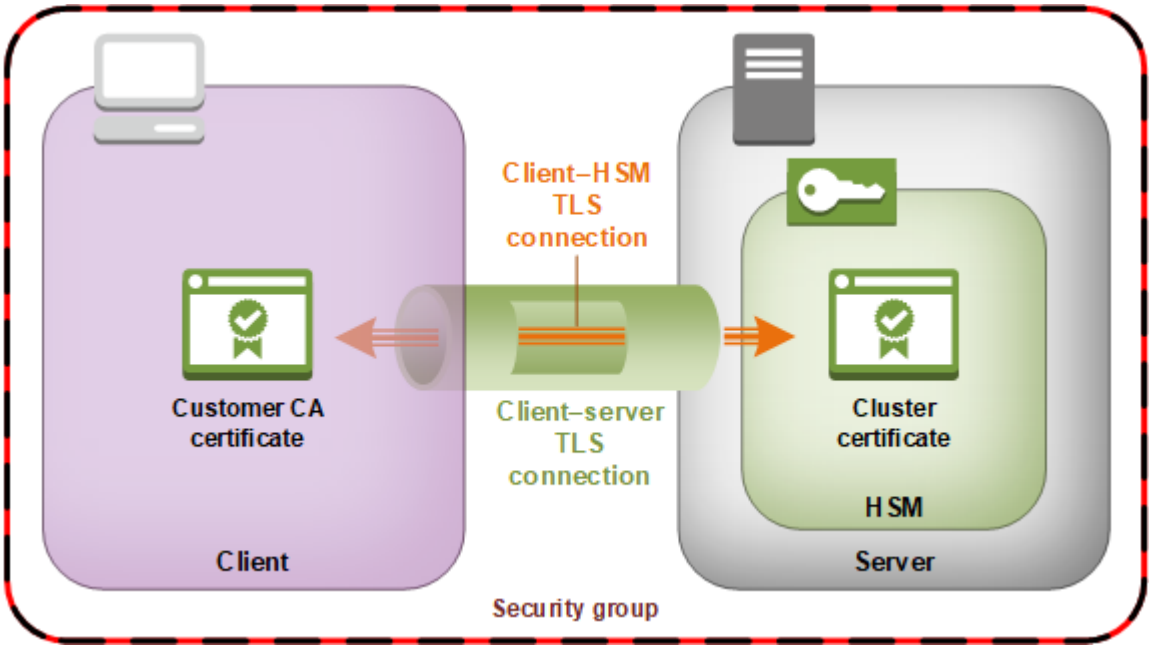
La comunicazione tra l'istanza del client e l'HSM nel tuo cluster è crittografata end-to-end. Solo il tuo client e il tuo HSM possono decifrarla.

Il seguente processo spiega come il client stabilisce una comunicazione crittografata end-to-end con un HSM.

1. Il client stabilisce una connessione Transport Layer Security (TLS) con il server che ospita l'hardware dell'HSM. Il gruppo di sicurezza del cluster consente il traffico in entrata al server solo da istanze client nel gruppo di sicurezza. Il client controlla anche il certificato del server per assicurare che sia un server affidabile.



2. Il client stabilisce poi una connessione crittografata con l'hardware dell'HSM. L'HSM dispone del certificato del cluster firmato con la tua autorità di certificazione (CA) e il client dispone del certificato di origine della CA. Prima di stabilire la connessione crittografata tra client e HSM, il client verifica il certificato del cluster dell'HSM con il relativo certificato di origine. La connessione viene stabilita solo una volta che il client ha verificato correttamente l'affidabilità dell'HSM.



Sicurezza dei backup dei cluster

Quando AWS CloudHSM esegue un backup dall'HSM, quest'ultimo crittografa tutti i dati prima di inviarli a. AWS CloudHSM I dati non lasciano mai l'HSM sotto forma di testo normale. Inoltre, i backup non possono essere decrittografati AWS perché AWS non ha accesso alla chiave utilizzata per decrittografare i backup.

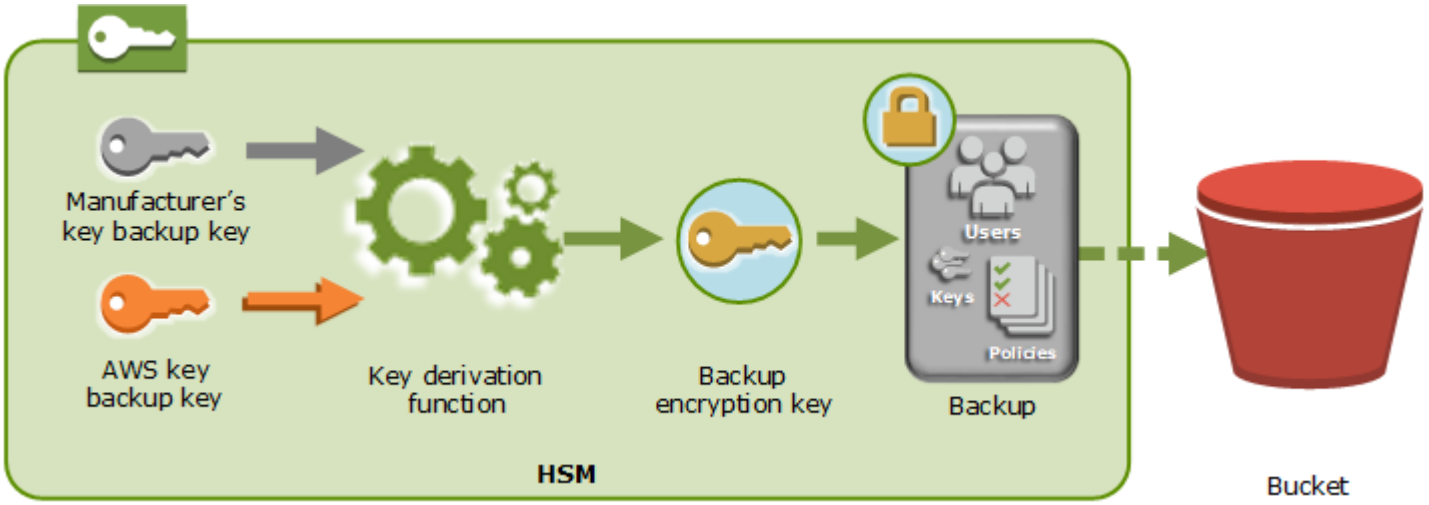
Per crittografare i dati, l'HSM utilizza una chiave di crittografia univoca e temporanea (EBK). L'EBK è una chiave di crittografia AES a 256 bit generata all'interno dell'HSM quando si effettua un backup. AWS CloudHSM L'HSM genera l'EBK, quindi lo utilizza per crittografare i dati dell'HSM con un metodo di key wrapping FIPS-approved AES conforme alla pubblicazione speciale del NIST 800-38F. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf Quindi l'HSM fornisce i dati crittografati a. AWS CloudHSM I dati crittografati includono una copia crittografata dell'EBK.

Per crittografare l'EBK, l'HSM utilizza un'altra chiave di crittografia nota come chiave di backup permanente (PBK). Anche la PBK è una chiave di crittografia AES a 256 bit. Per generare il PBK, l'HSM utilizza una funzione di derivazione FIPS-approved chiave (KDF) in modalità contatore conforme alla pubblicazione speciale 800-108 del NIST. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-108.pdf Gli input per questa KDF includono i seguenti elementi:

- Una chiave di backup del produttore (MKBK, Manufacturer Key Backup Key) integrata in modo permanente nell'hardware dell'HSM dal produttore dell'hardware.

- Una AWS chiave di backup delle chiavi (AKBK), installata in modo sicuro nell'HSM quando è inizialmente configurato da. AWS CloudHSM

La figura che segue riepiloga i processi della crittografia. La chiave di crittografia di backup rappresenta la chiave di backup permanente (PBK) e la chiave di backup temporanea (EBK).



AWS CloudHSM è in grado di ripristinare i backup solo su HSM AWS di proprietà e realizzati dallo stesso produttore. Poiché ciascun backup contiene tutti gli utenti, le chiavi e la configurazione dell'HSM originale, l'HSM ripristinato conterrà le stesse protezioni e gli stessi controlli d'accesso dell'originale. I dati ripristinati sovrascrivono tutti gli altri dati eventualmente presenti sull'HSM prima del ripristino.

Un backup contiene solo dati crittografati. Prima che il servizio memorizzi un backup in Amazon S3, crittografa nuovamente il backup utilizzando (). AWS Key Management Service AWS KMS

Gestione delle identità e degli accessi per AWS CloudHSM

AWS utilizza le credenziali di sicurezza per identificarti e per concederti l'accesso alle risorse AWS. Puoi utilizzare le funzionalità di AWS Identity and Access Management (IAM) per consentire ad altri utenti, servizi e applicazioni di utilizzare le tue risorse AWS completamente o in modo limitato. Il tutto senza condividere le credenziali di sicurezza.

Per impostazione predefinita, gli utenti IAM non dispongono dell'autorizzazione per creare, visualizzare o modificare le risorse AWS. Per consentire a un utente IAM di accedere a risorse come un sistema di bilanciamento del carico ed eseguire attività, è necessario:

1. Crea una policy IAM che conceda all'utente IAM l'autorizzazione per utilizzare le risorse specifiche e le operazioni API di cui ha bisogno.
2. Collegare la policy all'utente IAM o al gruppo a cui l'utente IAM appartiene.

Quando si collega una policy a un utente o a un gruppo di utenti viene concessa o rifiutata agli utenti l'autorizzazione per l'esecuzione delle attività specificate sulle risorse specificate.

Ad esempio è possibile utilizzare l'IAM per creare utenti e gruppi nell'account AWS. Un utente IAM può essere una persona, un sistema o un'applicazione. Quindi puoi concedere le autorizzazioni a utenti e gruppi affinché eseguano operazioni specifiche sulle risorse specificate utilizzando una policy IAM.

Concessione di autorizzazioni tramite le policy IAM

Quando si collega una policy a un utente o a un gruppo di utenti, viene concessa o rifiutata agli utenti l'autorizzazione per l'esecuzione delle attività specificate sulle risorse specificate.

Una policy IAM è un documento JSON costituito da una o più istruzioni.

Il seguente esempio di policy IAM consente agli utenti di descrivere i AWS CloudHSM backup negli Stati Uniti orientali (Virginia settentrionale). Sono consentite solo le richieste di descrizione provenienti dagli Stati Uniti orientali (Virginia settentrionale).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloudhsm:DescribeBackups",
      "Resource": "arn:aws:cloudhsm:us-east-1:123456789012:backup/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "us-east-1"
         }
      }
    }
  ]
```

```
}
```

- Effetto: l'elemento effetto può essere `Allow` o `Deny`. Per impostazione predefinita, gli utenti IAM non dispongono dell'autorizzazione per l'utilizzo delle risorse e per operazioni API, pertanto tutte le richieste vengono rifiutate. Un permesso esplicito sostituisce l'impostazione predefinita. Un rifiuto esplicito sovrascrive tutti i consensi.
- Operazione: l'elemento operazione corrisponde all'operazione API specifica per la quale si concede o si nega l'autorizzazione. Per ulteriori informazioni su come specificare l'operazione, consulta Azioni API per AWS CloudHSM.
- Risorsa: la risorsa interessata dall'azione. AWS CloudHSM non supporta le autorizzazioni a livello di risorsa. È necessario utilizzare il carattere jolly * per specificare tutte le risorse. AWS CloudHSM
- Condizione— Opzionalmente, puoi utilizzare le condizioni per controllare quando la tua policy è in vigore. Per ulteriori informazioni, consulta Chiavi di condizione per AWS CloudHSM.

Per ulteriori informazioni, consultare la Guida per l'utente IAM.

Azioni API per AWS CloudHSM

Nell'elemento Action della tua dichiarazione di policy IAM, puoi specificare qualsiasi azione API AWS CloudHSM offerta. Occorre applicare un prefisso al nome dell'operazione con la stringa minuscola `cloudhsm:`, come nell'esempio seguente.

```
"Action": "cloudhsm:DescribeClusters"
```

Per specificare più operazioni in una sola istruzione, racchiudile tra parentesi quadre e separale con una virgola, come illustrato nell'esempio seguente.

```
"Action": [
    "cloudhsm:DescribeClusters",
    "cloudhsm:DescribeHsm"
]
```

Puoi anche specificare più operazioni tramite il simbolo *. L'esempio seguente specifica tutti i nomi delle azioni API AWS CloudHSM che iniziano con`List`.

```
"Action": "cloudhsm:List*"
```

Per specificare tutte le azioni API per AWS CloudHSM, usa il carattere jolly *, come mostrato nell'esempio seguente.

```
"Action": "cloudhsm:*"
```

Per l'elenco delle azioni API per AWS CloudHSM, consulta AWS CloudHSM Azioni.

Chiavi di condizione per AWS CloudHSM

Quando si crea una policy, puoi specificare le condizioni che controllano quando la policy è in vigore. Ogni condizione contiene una o più coppie chiave/valore. Sono disponibili chiavi di condizione globali e chiavi di condizione specifiche per il servizio.

AWS CloudHSM non dispone di chiavi di contesto specifiche del servizio.

Per ulteriori informazioni sulle chiavi di condizione globali, consulta Chiavi di contesto delle condizioni globali AWS nella Guida dell'utente IAM.

Policy gestite da AWS predefinite per AWS CloudHSM

Le policy gestite create da AWS concedono le autorizzazioni necessarie per i casi d'utilizzo più comuni. Puoi collegare queste policy agli utenti IAM in base all'accesso all' AWS CloudHSM da loro richiesto:

- AWSCloudHSMFullAccess— Garantisce l'accesso completo necessario per utilizzare AWS CloudHSM le funzionalità.
- AWSCloudHSMReadOnlyAccess— Consente l'accesso in sola lettura alle funzionalità. AWS CloudHSM

Policy gestite dal cliente per AWS CloudHSM

Ti consigliamo di creare un gruppo di amministratori IAM AWS CloudHSM che contenga solo le autorizzazioni necessarie per l'esecuzione. AWS CloudHSM Collegare la policy con le autorizzazioni appropriate a questo gruppo. Aggiungere utenti IAM al gruppo, se necessario. Ogni utente aggiunto eredita la policy dal gruppo degli amministratori.

Inoltre, ti consigliamo di creare gruppi di utenti aggiuntivi in base alle autorizzazioni necessarie agli utenti. Ciò garantisce che solo gli utenti attendibili abbiano accesso alle operazioni API critiche. Ad

esempio, è possibile creare un gruppo di utenti che puoi usare per concedere l'accesso in modalità di sola lettura per cluster e HSM. Poiché questo gruppo non consente a un utente di eliminare cluster o HSM, un utente non attendibile non può influire sulla disponibilità di un carico di lavoro di produzione.

Man mano che nel tempo vengono aggiunte nuove funzionalità di AWS CloudHSM gestione, puoi assicurarti che solo gli utenti fidati abbiano accesso immediato. Assegnando autorizzazioni limitate alle policy in fase di creazione, è possibile assegnare manualmente le autorizzazioni per le nuove funzionalità n un secondo momento.

Di seguito sono riportati alcuni esempi di politiche per AWS CloudHSM. Per informazioni su come creare una policy e collegarla a un gruppo di utenti IAM, consulta Creazione di policy nella scheda JSON nella Guida per l'utente IAM.

Esempi

- Autorizzazioni di sola lettura
- Autorizzazioni utente avanzato
- Autorizzazioni amministratore

Example Esempio: Read-only autorizzazioni

Questa policy consente l'accesso al `DescribeClusters` e `DescribeBackups` alle operazioni API. Include anche autorizzazioni aggiuntive per operazioni API Amazon EC2 specifiche. Tuttavia, non consente all'utente di eliminare i cluster o gli HSM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": [
         "cloudhsm:DescribeClusters",
         "cloudhsm:DescribeBackups",
         "cloudhsm:ListTags"
      ],
      "Resource": "*"
   }
}
```

Example Esempio: Autorizzazioni utente avanzato

Questa policy consente l'accesso a un sottoinsieme delle azioni AWS CloudHSM API. Include anche autorizzazioni aggiuntive per operazioni specifiche Amazon EC2. Tuttavia, non consente all'utente di eliminare i cluster o gli HSM. Devi includere l'`iam:CreateServiceLinkedRole`azione per consentire AWS CloudHSM la creazione automatica del ruolo AWSServiceRoleForCloudHSM collegato al servizio nel tuo account. Questo ruolo consente di registrare AWS CloudHSM gli eventi. Per ulteriori informazioni, consulta Service-linked ruoli per AWS CloudHSM.

> **Note**
>
> Per un elenco dei permessi specifici di ogni API, consulta Operazioni, risorse e chiavi di condizione per l' AWS CloudHSM nella Guida all'autorizzazione del servizio.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": [
         "cloudhsm:DescribeClusters",
         "cloudhsm:DescribeBackups",
         "cloudhsm:CreateCluster",
         "cloudhsm:CreateHsm",
         "cloudhsm:RestoreBackup",
         "cloudhsm:CopyBackupToRegion",
         "cloudhsm:InitializeCluster",
         "cloudhsm:ListTags",
         "cloudhsm:TagResource",
         "cloudhsm:UntagResource",
         "ec2:CreateNetworkInterface",
         "ec2:DescribeNetworkInterfaces",
         "ec2:DescribeNetworkInterfaceAttribute",
         "ec2:DetachNetworkInterface",
         "ec2:DeleteNetworkInterface",
         "ec2:CreateSecurityGroup",
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
```

```
            "ec2:DescribeSecurityGroups",
            "ec2:DeleteSecurityGroup",
            "ec2:CreateTags",
            "ec2:DescribeVpcs",
            "ec2:DescribeSubnets",
            "iam:CreateServiceLinkedRole"
        ],
        "Resource": "*"
    }
}
```

Example Esempio: Autorizzazioni admin

Questa policy consente l'accesso a tutte le azioni AWS CloudHSM API, incluse le azioni per eliminare HSM e cluster. Include anche autorizzazioni aggiuntive per operazioni specifiche Amazon EC2. Devi includere l'`iam:CreateServiceLinkedRole`azione per consentire AWS CloudHSM la creazione automatica del ruolo AWSServiceRoleForCloudHSM collegato al servizio nel tuo account. Questo ruolo consente di registrare AWS CloudHSM gli eventi. Per ulteriori informazioni, consulta Service-linked ruoli per AWS CloudHSM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":{
      "Effect":"Allow",
      "Action":[
         "cloudhsm:*",
         "ec2:CreateNetworkInterface",
         "ec2:DescribeNetworkInterfaces",
         "ec2:DescribeNetworkInterfaceAttribute",
         "ec2:DetachNetworkInterface",
         "ec2:DeleteNetworkInterface",
         "ec2:CreateSecurityGroup",
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:DescribeSecurityGroups",
         "ec2:DeleteSecurityGroup",
         "ec2:CreateTags",
         "ec2:DescribeVpcs",
```

```
            "ec2:DescribeSubnets",
            "iam:CreateServiceLinkedRole"
        ],
        "Resource":"*"
    }
}
```

Service-linked ruoli per AWS CloudHSM

La policy IAM che hai creato in precedenza Policy gestite dal cliente per AWS CloudHSM include l'`iam:CreateServiceLinkedRole`azione. AWS CloudHSM definisce un ruolo https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_terms-and-concepts.html#iam-term-service-linked-role collegato al servizio denominato. AWSServiceRoleForCloudHSM Il ruolo è predefinito AWS CloudHSM e include le autorizzazioni AWS CloudHSM necessarie per chiamare altri AWS servizi per conto dell'utente. Il ruolo rende più semplice l'impostazione del servizio perché non devi aggiungere manualmente la policy del ruolo e le autorizzazioni della policy di attendibilità.

La policy sui ruoli consente di AWS CloudHSM creare gruppi di CloudWatch log e stream di log di Amazon Logs e scrivere eventi di log per tuo conto. È possibile visualizzarli qui sotto e nella console IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
```

```
}
```

La politica di fiducia per il AWSServiceRoleForCloudHSM ruolo consente di AWS CloudHSM assumere il ruolo.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudhsm.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Creazione di un ruolo collegato ai servizi (Automatico)

AWS CloudHSM crea il AWSServiceRoleForCloudHSM ruolo quando si crea un cluster se si include l'`iam:CreateServiceLinkedRole`azione nelle autorizzazioni definite durante la creazione del gruppo di AWS CloudHSM amministratori. Consulta Policy gestite dal cliente per AWS CloudHSM.

Se disponi già di uno o più cluster e desideri semplicemente aggiungere il AWSServiceRoleForCloudHSM ruolo, puoi utilizzare la console, il comando create-cluster o l'operazione CreateCluster API per creare un cluster. Quindi usa la console, il comando delete-cluster o l'operazione API per eliminarlo. DeleteCluster La creazione del nuovo cluster genera il ruolo collegato al servizio e lo applica a tutti i cluster nel tuo account. In alternativa, puoi creare manualmente il ruolo. Per ulteriori informazioni, consulta la sezione seguente.

> **Note**
>
> Non è necessario eseguire tutti i passaggi descritti in Guida introduttiva con AWS CloudHSM per creare un cluster se lo si crea solo per aggiungere il ruolo. AWSServiceRoleForCloudHSM

Creazione di un ruolo collegato ai servizi (Manuale)

Puoi utilizzare la console IAM o AWS CLI l'API per creare il AWSServiceRoleForCloudHSM ruolo. Per ulteriori informazioni, consulta Creating a Service-Linked Role nella IAM User Guide.

Modifica del ruolo collegato ai servizi

AWS CloudHSM non consente di modificare il AWSServiceRoleForCloudHSM ruolo. Dopo aver creato il ruolo, ad esempio, non è possibile modificarne il nome perché varie entità possono referenziare il ruolo sulla base del nome. Inoltre, non è possibile modificare la policy del ruolo. Puoi tuttavia utilizzare IAM; per modificare la descrizione del ruolo. Per ulteriori informazioni, consulta la sezione Modifica di un ruolo collegato ai servizi nella Guida per l'utente IAM.

Eliminazione del ruolo collegato ai servizi

Non è possibile eliminare un ruolo collegato ai servizi poiché il cluster a cui è stato applicato è ancora disponibile. Per eliminare il ruolo, è necessario eliminare prima tutti i moduli HSM nel cluster e quindi eliminare il cluster. Ogni cluster nel tuo account deve essere eliminato. Puoi quindi utilizzare la console IAM o AWS CLI l'API per eliminare il ruolo. Per ulteriori informazioni sull'eliminazione di un cluster, consulta Eliminazione di un cluster AWS CloudHSM. Per ulteriori informazioni, consulta Eliminazione di un Service-Linked ruolo nella IAM User Guide.

Convalida della conformità per AWS CloudHSM

Per i cluster in modalità FIPS, AWS CloudHSM fornisce FIPS-approved HSM che soddisfano PCI-PIN i requisiti di conformità SOC2. PCI-3DS AWS CloudHSM offre inoltre ai clienti la possibilità di scegliere cluster in modalità non FIPS. Per informazioni dettagliate sui requisiti di certificazione e conformità applicabili a ciascuno di essi, consulta. AWS CloudHSM modalità cluster

Affidarsi a un FIPS-validated HSM può aiutarti a soddisfare i requisiti di conformità aziendale, contrattuale e normativa per la sicurezza dei dati nel AWS cloud.

Conformità a FIPS 140-2

Il Federal Information Processing Standard (FIPS) Publication 140-2 è uno standard di sicurezza del governo degli Stati Uniti che specifica i requisiti di sicurezza previsti per i moduli crittografici a protezione delle informazioni sensibili. Il tipo di istanza AWS CloudHSM hsm1.medium è certificato FIPS 140-2 di livello 3 (Certificato #4218). https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program/Certificate/4218 Il 4 gennaio 2026 il certificato per hsm1.medium passa all'elenco storico. Consigliamo ai clienti di passare a hsm2m.medium, che è certificato FIPS 140-3 (Certificato #4703). https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4703 Per ulteriori informazioni, consulta la sezione Convalida FIPS per l'hardware. https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program

Conformità allo standard FIPS 140-3

La pubblicazione 140-3 del Federal Information Processing Standard (FIPS) è uno standard di sicurezza del governo degli Stati Uniti che specifica i requisiti di sicurezza per i moduli crittografici che proteggono le informazioni sensibili. Gli HSM di tipo hsm2m.medium forniti da AWS CloudHSM sono certificati FIPS 140-3 di livello 3 (Certificate #4703). https://csrc.nist.gov/projects/cryptographic-module-validation-program/certificate/4703 Per ulteriori informazioni, fare riferimento alla convalida FIPS per l'hardware. https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program

Conformità PCI DSS

Il Payment Card Industry Data Security Standard (PCI DSS) è uno standard proprietario di tutela delle informazioni gestito dal PCI Security Standards Council. Gli HSM forniti da sono AWS CloudHSM conformi allo standard PCI DSS.

Conformità PCI DSS

Il PCI PIN fornisce requisiti di sicurezza e standard di valutazione per la trasmissione, l'elaborazione e la gestione dei dati del numero di identificazione personale (PIN), informazioni utilizzate per le transazioni presso gli sportelli automatici e i terminali POS. Gli HSM hsm1.medium e hsm2m.medium forniti da sono entrambi conformi al PCI PIN. AWS CloudHSM Per ulteriori informazioni, consulta l'articolo AWS CloudHSM è ora certificato PCI PIN.

Conformità PCI-3DS

PCI 3DS (o Three Domain Secure, 3-D Secure) fornisce la sicurezza dei dati per pagamenti e-commerce sicuri EMV 3D. PCI 3DS offre un altro livello di sicurezza per gli acquisti online.

Gli HSM hsm1.medium e hsm2m.medium forniti da sono entrambi conformi. AWS CloudHSM PCI-3DS

SOC2

SOC2 è un framework che aiuta le organizzazioni di servizi a dimostrare i propri controlli di sicurezza nel cloud e nei data center. AWS CloudHSM ha implementato i controlli SOC2 in aree critiche per aderire ai principi di servizio affidabili. Per ulteriori informazioni, consulta la pagina delle domande frequenti su AWS SOC.

AWS CloudHSM PCI-PIN Domande frequenti sulla conformità

Il PCI PIN fornisce requisiti di sicurezza e standard di valutazione per la trasmissione, l'elaborazione e la gestione dei dati relativi al numero di identificazione personale (PIN), informazioni utilizzate per le transazioni presso gli sportelli automatici e i terminali POS.

L' PCI-PIN attestazione di conformità (AOC) e il riepilogo della responsabilità sono disponibili per i clienti tramite AWS Artifact, un portale self-service per l'accesso su richiesta ai report di conformità AWS. Per ulteriori informazioni, accedi ad AWS Artifact dalla Console di gestione AWS o scopri di più su Inizia ad usare AWS Artifact.

FAQs

D: Cos'è l'attestato di conformità e il riepilogo della responsabilità?

L'attestazione di conformità (AOC) è prodotta da un Qualified PIN Assessor (QPA) che attesta la conformità ai controlli applicabili previsti dallo AWS CloudHSM standard. PCI-PIN La matrice riassuntiva delle responsabilità descrive i controlli che rientrano nelle rispettive responsabilità e nei relativi clienti. AWS CloudHSM

D: Come posso ottenere l' AWS CloudHSM attestato di conformità?

L' PCI-PIN Attestation of Compliance (AOC) è disponibile per i clienti tramite AWS Artifact, un portale self-service per l'accesso su richiesta ai report di conformità AWS. Per ulteriori informazioni, accedi ad AWS Artifact dalla Console di gestione AWS o scopri di più su Inizia ad usare AWS Artifact.

D: Come posso sapere di quali controlli PCI PIN sono responsabile?

Per informazioni dettagliate, consulta la sezione «Riepilogo della responsabilitàAWS CloudHSM PCI PIN» del pacchetto di conformità PCI PIN di AWS, disponibile per i clienti tramite AWS Artifact, un

portale self-service per l'accesso su richiesta ai report di conformità AWS. Per ulteriori informazioni, accedi ad AWS Artifact dalla Console di gestione AWS o scopri di più su Inizia ad usare AWS Artifact.

D: In qualità di AWS CloudHSM cliente, posso fare affidamento sull' PCI-PIN Attestation of Compliance (AOC)?

I clienti devono gestire la propria PCI-PIN conformità. È necessario eseguire un processo di PCI-PIN attestazione formale tramite un Qualified PIN Assessor (QPA) per verificare che il carico di lavoro dei pagamenti soddisfi tutti i requisiti. PCI-PIN controls/requirements Tuttavia, per i controlli di cui è responsabile AWS, il tuo QPA può fare affidamento sull' AWS CloudHSM Attestation of Compliance (AOC) senza ulteriori test.

D: È AWS CloudHSM responsabile dei PCI-PIN requisiti relativi al ciclo di vita della gestione delle chiavi?

AWS CloudHSM è responsabile del ciclo di vita dei dispositivi fisici degli HSM. I clienti sono responsabili dei principali requisiti del ciclo di vita della gestione previsti dallo standard. PCI-PIN

D: Quali AWS CloudHSM controlli sono PCI-PIN conformi?

L'AOC riassume i AWS CloudHSM controlli valutati dal QPA. Il Riepilogo delle PCI-PIN responsabilità è disponibile per i clienti tramite AWS Artifact, un portale self-service per l'accesso su richiesta ai report di conformità AWS.

D: AWS CloudHSM Supporta funzioni di pagamento come la conversione del PIN e il DUKPT?

No, AWS CloudHSM fornisce HSM generici. Nel tempo potremmo fornire funzioni di pagamento. Sebbene il servizio non esegua direttamente le funzioni di pagamento, l'attestazione di conformità AWS CloudHSM PCI PIN consente ai clienti di ottenere la propria conformità PCI per i propri servizi in esecuzione. AWS CloudHSM Se sei interessato a utilizzare i servizi di crittografia di AWS Payment per il tuo carico di lavoro, consulta il blog "Move Payment Processing to the Cloud with AWS Payment Cryptography".

Notifiche di deprecazione

Di tanto in tanto, AWS CloudHSM può rendere obsoleta la funzionalità per rimanere conforme ai requisiti di FIPS 140,,, PCI-DSS, SOC2 o a causa della fine del supporto hardware. PCI-PIN PCI-3DS Questa pagina elenca le modifiche attualmente in vigore.

Obsolescenza HSM1

Il tipo di istanza AWS CloudHSM hsm1.medium terminerà il supporto il 31 marzo 2026. Per garantire la continuità del servizio, stiamo introducendo le seguenti modifiche:

- A partire da aprile 2025, non sarai in grado di creare nuovi cluster hsm1.medium.
- A partire da gennaio 2026, inizieremo a migrare automaticamente i cluster hsm1.medium esistenti al nuovo tipo di istanza hsm2m.medium.

Il tipo di istanza hsm2m.medium è compatibile con il tipo di istanza corrente e offre prestazioni migliorate. AWS CloudHSM Per evitare interruzioni delle applicazioni, è necessario eseguire l'aggiornamento alla versione più recente dell'SDK client. Per istruzioni sull'aggiornamento, consulta. ???

Sono disponibili due opzioni per la migrazione:

1. Attiva una CloudHSM-managed migrazione quando sei pronto. Per ulteriori informazioni, consulta ???.
2. Crea un nuovo cluster hsm2m.medium da un backup del tuo cluster hsm1 e reindirizza l'applicazione al nuovo cluster. Consigliamo di utilizzare una strategia di distribuzione per questo approccio. blue/green Per ulteriori informazioni, consulta ???.

Conformità FIPS 140: meccanismo di deprecazione 2024

Il National Institute of Standards and Technology (NIST) [1] segnala che il supporto per la crittografia Triple DES (DESede, 3DES, DES3) e la funzione RSA di wrapping e unwrapping delle chiavi con PKCS #1 v1.5 non sarà consentito dopo il 31 dicembre 2023. Pertanto, il loro supporto terminerà il 1° gennaio 2024 nei nostri cluster in modalità Federal Information Processing Standard (FIPS). Il supporto per questi rimane valido per i cluster in modalità non FIPS.

Questa guida si applica alle seguenti operazioni crittografiche:

- Generazione di chiavi Triple DES
  - `CKM_DES3_KEY_GEN` per la libreria PKCS #11
  - `DESede` generazione di chiavi per il provider JCE
  - `genSymKey` con `-t=21` per la KMU
- Crittografia con chiavi Triple DES (nota: sono consentite operazioni di decifrazione)

  - Per la libreria PKCS #11: crittografare `CKM_DES3_CBC`, crittografare `CKM_DES3_CBC_PAD` e crittografare `CKM_DES3_ECB`
  - Per il provider JCE: crittografare `DESede/CBC/PKCS5Padding`, crittografare `DESede/CBC/NoPadding`, crittografare `DESede/ECB/Padding` e crittografare `DESede/ECB/NoPadding`
- Wrap, unwrap, crittografa e decifrazione RSA delle chiavi con il padding PKCS #1 v1.5
  - `CKM_RSA_PKCS` wrap, unwrap, crittografa e decifrazione per l'SDK PKCS #11
  - `RSA/ECB/PKCS1Padding` wrap, unwrap, crittografa e decrittografa per JCE SDK
  - `wrapKey` e `unWrapKey` con `-m 12` per la KMU (nota: `12` è il valore del meccanismo `RSA_PKCS`)

[1] Per i dettagli su questa modifica, fai riferimento alla Tabella 1 e alla Tabella 5 in Transizione nell'uso degli algoritmi crittografici e della lunghezza delle chiavi.

Resilienza in AWS CloudHSM

L'infrastruttura AWS globale è costruita attorno a AWS regioni e zone di disponibilità. AWS Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, collegate a reti a bassa latenza, alta produttività e altamente ridondanti. Con le zone di disponibilità è possibile progettare e gestire applicazioni e database che eseguono automaticamente il failover 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 Global Infrastructure. AWS Per ulteriori informazioni sulle caratteristiche per supportare la resilienza dell' AWS CloudHSM , consulta AWS CloudHSM elevata disponibilità e bilanciamento del carico del cluster.

Sicurezza dell'infrastruttura in AWS CloudHSM

In quanto servizio gestito, AWS CloudHSM 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 CloudHSM tramite la rete. Inoltre, le richieste devono essere firmate utilizzando un chiave di accesso ID e una chiave di accesso segreta associata a un account principale IAM. O puoi utilizzare Servizio di token di sicurezza AWS (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Isolamento della rete

Un cloud privato virtuale (VPC) è una rete virtuale nella propria area logicamente isolata in Cloud AWS. Puoi creare un cluster in una sottorete privata nel VPC. Puoi creare sottoreti private quando crei un VPC. Per ulteriori informazioni, consulta Crea un cloud privato virtuale (VPC) per AWS CloudHSM.

Quando crei un HSM, AWS CloudHSM inserisci un'interfaccia di rete elastica (ENI) nella tua sottorete in modo da poter interagire con i tuoi HSM. Per ulteriori informazioni, consulta AWS CloudHSM architettura dei cluster.

AWS CloudHSM crea un gruppo di sicurezza che consente la comunicazione in entrata e in uscita tra gli HSM del cluster. Puoi utilizzare questo gruppo di sicurezza per consentire alle istanze EC2 di comunicare con i moduli HSM nel cluster. Per ulteriori informazioni, consulta Configurazione dei gruppi di sicurezza delle istanze Client Amazon EC2 per AWS CloudHSM.

Autorizzazione degli utenti

Con AWS CloudHSM, le operazioni eseguite sull'HSM richiedono le credenziali di un utente HSM autenticato. Per ulteriori informazioni, consulta the section called “Tipi di utente”.

AWS CloudHSM ed endpoint VPC

Puoi stabilire una connessione privata tra il tuo VPC e creando un endpoint VPC AWS CloudHSM di interfaccia. Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere privatamente alle AWS CloudHSM API 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 le API. AWS CloudHSM Il traffico tra il tuo VPC e l' AWS CloudHSM non esce dalla rete Amazon.

Ogni endpoint dell'interfaccia è rappresentato da una o più interfacce di rete elastiche nelle sottoreti.

Per ulteriori informazioni, consulta Interface VPC endpoints (AWS PrivateLink) nella Amazon VPC User Guide.

Considerazioni per AWS CloudHSM Endpoint VPC

Prima di configurare un endpoint VPC di interfaccia per AWS CloudHSM, assicurati di aver esaminato le proprietà e le limitazioni degli endpoint di interfaccia nella Amazon VPC User Guide.

- AWS CloudHSM supporta l'esecuzione di chiamate a tutte le sue azioni API dal tuo VPC.

Creazione di un endpoint VPC di interfaccia per AWS CloudHSM

Puoi creare un endpoint VPC per il AWS CloudHSM servizio utilizzando la console Amazon VPC o (). AWS Command Line Interface AWS CLI Per ulteriori informazioni, consultare Creazione di un endpoint di interfaccia nella Guida per l'utente di Amazon VPC.

Per creare un endpoint VPC per AWS CloudHSM, utilizza il seguente nome di servizio:

```
com.amazonaws.<region>.cloudhsmv2
```

Ad esempio, nella regione Stati Uniti occidentali (Oregon) (`us-west-2`), il nome del servizio sarebbe:

```
com.amazonaws.us-west-2.cloudhsmv2
```

Per semplificare l'utilizzo dell'endpoint VPC, è possibile abilitare un nome host DNS privato per l'endpoint VPC. Se selezioni l'opzione Abilita nome DNS privato, i nomi host AWS CloudHSM DNS standard (`https://cloudhsmv2.<region>.amazonaws.comehttps://cloudhsmv2.<region>.api.aws`) si risolvono nell'endpoint VPC.

Questa opzione rende più semplice utilizzare l'endpoint VPC. Gli AWS SDK AWS CLI utilizzano il nome host AWS CloudHSM DNS standard per impostazione predefinita, quindi non è necessario specificare l'URL dell'endpoint VPC nelle applicazioni e nei comandi.

Per ulteriori informazioni, consultare Accesso a un servizio tramite un endpoint di interfaccia in Guida per l'utente di Amazon VPC.

Creazione di una policy sugli endpoint VPC per AWS CloudHSM

È possibile allegare un criterio all'endpoint VPC che controlla l'accesso all' AWS CloudHSM. La policy specifica le informazioni riportate di seguito:

- Il principale che può eseguire operazioni.
- Le azioni che possono essere eseguite.
- Le risorse sui cui si possono eseguire operazioni.

Per ulteriori informazioni, consulta Controllo degli accessi ai servizi con endpoint VPC in Guida per l'utente di Amazon VPC.

Esempio: politica sugli endpoint VPC per AWS CloudHSM actions

Di seguito è riportato un esempio di policy sugli endpoint per. AWS CloudHSM Se collegata a un endpoint, questa politica consente l'accesso alle AWS CloudHSM azioni elencate a tutti i principali su tutte le risorse. Consulta Gestione delle identità e degli accessi per AWS CloudHSM le altre AWS CloudHSM azioni e le relative autorizzazioni IAM.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "<cloudhsm>:<DescribeBackups>",
            "<cloudhsm>:<DescribeClusters>",
            "<cloudhsm>:<ListTags>",
         ],
         "Resource":"*"
      }
   ]
}
```

Gestione degli aggiornamenti in AWS CloudHSM

AWS gestisce il firmware. Il firmware è gestito da una terza parte e deve essere valutato dal NIST per la conformità allo standard FIPS 140-2 Level 3 o FIPS 140-3 Level 3 a seconda del tipo di hsm. È possibile installare solo firmware con crittografia firmata secondo lo standard FIPS a cui AWS non ha accesso.

Risoluzione dei problemi AWS CloudHSM

Se riscontri problemi con AWS CloudHSM, i seguenti argomenti possono aiutarti a risolverli.

Argomenti

- AWS CloudHSM problemi noti
- AWS CloudHSM Errori di sincronizzazione delle chiavi Client SDK 3
- AWS CloudHSM Client SDK 3 verifica le prestazioni HSM con lo strumento pkpspeed
- AWS CloudHSM L'utente o la policy di Client SDK 5 contiene valori non coerenti
- AWS CloudHSM Errori di replica degli utenti di Client SDK 5
- AWS CloudHSM Errori di replica chiave di Client SDK 5
- AWS CloudHSM errore rilevato durante il controllo della disponibilità delle chiavi
- AWS CloudHSM estrazione di chiavi usando JCE
- Limitazione HSM
- Connessione persa al AWS CloudHSM cluster
- Log AWS CloudHSM di controllo mancanti CloudWatch
- IV personalizzati con lunghezza non conforme per l'inserimento delle chiavi AES AWS CloudHSM
- Risoluzione degli errori di creazione dei AWS CloudHSM cluster
- Recupero dei log di configurazione AWS CloudHSM del client

AWS CloudHSM problemi noti

AWS CloudHSM presenta i seguenti problemi noti. Scegli un argomento per saperne di più.

Argomenti

- Problemi noti per tutte le istanze HSM
- Problemi noti per AWS CloudHSM istanze hsm1.medium
- Problemi noti per AWS CloudHSM istanze hsm2m.medium
- Problemi noti della libreria PKCS #11 per AWS CloudHSM
- Problemi noti dell'SDK JCE per AWS CloudHSM
- Problemi noti di OpenSSL Dynamic Engine per AWS CloudHSM
- Problemi noti del Key Storage Provider (KSP) per AWS CloudHSM

- Problemi noti del provider OpenSSL per AWS CloudHSM
- Problemi noti dell'interfaccia a riga di comando di CloudHSM per AWS CloudHSM
- Problemi noti per le istanze Amazon EC2 che eseguono Amazon Linux 2 con AWS CloudHSM
- Problemi noti relativi all'integrazione di applicazioni di terze parti con AWS CloudHSM
- Problemi noti per AWS CloudHSM modifica cluster
- Problemi noti relativi al mancato funzionamento utilizzando AWS CloudHSM versione client 5.12.0 su hsm2.medium

Problemi noti per tutte le istanze HSM

I seguenti problemi riguardano tutti gli AWS CloudHSM utenti indipendentemente dal fatto che utilizzino lo strumento da riga di comando key_mgmt_util, l'SDK PKCS #11, l'SDK JCE o l'SDK OpenSSL.

Argomenti

- Problema: il wrapping della chiave AES utilizza il riempimento PKCS #5 invece di fornire un'implementazione conforme agli standard del wrapping della chiave con riempimento a zeri
- Problema: il daemon del client richiede almeno un indirizzo IP valido nel suo file di configurazione per la corretta connessione al cluster
- Problema: c'era un limite massimo di 16 KB per i dati che potevano essere sottoposti a hashing e firmati da AWS CloudHSM utilizzando Client SDK 3
- Problema: le chiavi importate non possono essere specificate come non esportabili
- Problema: il meccanismo predefinito per i WrapKey comandi wrapKey e un in key_mgmt_util è stato rimosso
- Problema: se si dispone di un singolo HSM nel cluster, il failover non funziona correttamente
- Problema: se si supera la capacità della chiave per gli HSM nel cluster all'interno di un breve periodo di tempo, il client immette un errore non gestito
- Problema: le operazioni digest con le chiavi HMAC di dimensioni superiori a 800 byte non sono supportate
- Problema: lo strumento client_info distribuito con Client SDK 3 elimina il contenuto del percorso specificato dall'argomento di output opzionale
- Problema: viene visualizzato un errore quando si esegue lo strumento di configurazione SDK 5 utilizzando l'argomento --cluster-id in ambienti containerizzati

- Problema: viene visualizzato l'errore «Creazione cert/key dal file pfx fornito non riuscita. Errore: NotPkcs 8"
- Problema: la firma ECDSA non riesce con l'errore «meccanismo non valido» che inizia con SDK 5.16
- Problema: le operazioni di firma con dati prehashati non cancellano correttamente i token di sessione in modalità interattiva
- Problema: il certificato client predefinito della libreria client CloudHSM scade il 31 gennaio 2026
- Problema: la connessione Client SDK 5 non riesce a causa di una chiave di certificato CA debole
- Problema: le operazioni crittografiche falliscono con errori «Il numero di sequenza è al di fuori della finestra» in caso di concorrenza elevata
- Problema: non ML-DSA è supportato nei FIPS-mode cluster

Problema: il wrapping della chiave AES utilizza il riempimento PKCS #5 invece di fornire un'implementazione conforme agli standard del wrapping della chiave con riempimento a zeri

Inoltre, il wrapping della chiave senza riempimento e riempimento a zeri non è supportato.

- Impatto: non c'è alcun impatto se esegui operazioni di avvolgimento e disimballaggio utilizzando questo algoritmo interno. AWS CloudHSM Tuttavia, le chiavi impacchettate AWS CloudHSM non possono essere scartate all'interno di altri HSM o software che prevedono la conformità alla specifica no-padding. Questo perché otto byte di dati di riempimento potrebbero essere aggiunti alla fine dei dati della chiave durante un wrapping conforme agli standard. Le chiavi racchiuse esternamente non possono essere estratte correttamente in un'istanza. AWS CloudHSM
- Soluzione: per annullare esternamente il wrapping di una chiave eseguito con wrapping della chiave AES con riempimento PKCS #5 su un'istanza AWS CloudHSM, eliminare il riempimento supplementare prima di tentare di utilizzare la chiave. Per farlo, è possibile tagliare i byte supplementari in un editor di file o copiare solo i byte della chiave in un nuovo buffer nel codice.
- Stato della risoluzione: con il client 3.1.0 e la versione software, AWS CloudHSM fornisce opzioni conformi agli standard per il wrapping delle chiavi AES. Per ulteriori informazioni, vedi wrapping delle chiavi AES.

Problema: il daemon del client richiede almeno un indirizzo IP valido nel suo file di configurazione per la corretta connessione al cluster

- Impact: (Impatto) se si elimina ogni HSM nel cluster e si aggiunge poi un altro HSM, che ottiene un nuovo indirizzo IP, il daemon del client continua a cercare gli HSM ai loro indirizzi IP originali.
- Soluzione alternativa: se si esegue un carico di lavoro intermittente, si consiglia di utilizzare l'`IpAddress`argomento nella CreateHsm funzione per impostare l'interfaccia di rete elastica (ENI) sul valore originale. Si noti che l'ENI è specifica per una zona di disponibilità (AZ). In alternativa, è possibile eliminare il file `/opt/cloudhsm/daemon/1/cluster.info` e quindi reimpostare la configurazione del client sull'indirizzo IP del nuovo HSM. È possibile utilizzare il comando `client -a <IP address>`. Per ulteriori informazioni, vedere Installare e configurare il AWS CloudHSM client (Linux) o Installare e configurare il AWS CloudHSM client (Windows).

Problema: c'era un limite massimo di 16 KB per i dati che potevano essere sottoposti a hashing e firmati da AWS CloudHSM utilizzando Client SDK 3

- Resolution status (Stato di risoluzione): i dati di dimensioni inferiori ai 16 KB continuano a essere inviati all'HSM per l'hashing. Abbiamo aggiunto la capacità che consente di eseguire l'hashing in locale, nel software, per i dati di dimensioni comprese tra 16 KB e 64 KB. Client SDK 5 fallirà esplicitamente se il buffer di dati è maggiore di 64 KB. È necessario aggiornare il client e gli SDK a una versione successiva alla 5.0.0 o superiore per beneficiare della correzione.

Problema: le chiavi importate non possono essere specificate come non esportabili

- Resolution Status (Stato di risoluzione): questo problema è stato risolto. Per trarre vantaggio dalla correzione non è richiesta alcuna operazione.

Problema: il meccanismo predefinito per i WrapKey comandi wrapKey e un in key_mgmt_util è stato rimosso

- Risoluzione: quando si utilizzano i WrapKey comandi wrapKey o un, è necessario utilizzare l'opzione per specificare il meccanismo. `-m` Per ulteriori informazioni, consultate gli esempi negli WrapKey articoli wrapKey o un.

Problema: se si dispone di un singolo HSM nel cluster, il failover non funziona correttamente

- Impatto: se la singola istanza HSM nel cluster perde la connettività, il client si non riconnette con essa anche se l'istanza HSM viene successivamente ripristinata.
- Soluzione. consigliamo almeno due istanze HSM in tutti i cluster di produzione. Se si utilizza questa configurazione, non verrà riscontrato questo problema. Per i cluster HSM singoli, non recapitare il daemon del client per ripristinare la connettività.
- Stato della risoluzione: questo problema è stato risolto nella versione 1.1.2 del client AWS CloudHSM . È necessario effettuare l'upgrade a questo client per sfruttare i vantaggi offerti dalla soluzione.

Problema: se si supera la capacità della chiave per gli HSM nel cluster all'interno di un breve periodo di tempo, il client immette un errore non gestito

- Impatto: quando il client incontra l'errore non gestito, si blocca e deve essere riavviato.
- Soluzione. prova il throughput per garantire che non vengono cerate chiavi di sessione a una velocità che il client non è in grado di gestire. È possibile ridurre la velocità aggiungendo un HSM al cluster o rallentando la creazione di chiavi di sessione.
- Stato della risoluzione: questo problema è stato risolto nella versione 1.1.2 del client AWS CloudHSM . È necessario effettuare l'upgrade a questo client per sfruttare i vantaggi offerti dalla soluzione.

Problema: le operazioni digest con le chiavi HMAC di dimensioni superiori a 800 byte non sono supportate

- Impatto: le chiavi HMAC di dimensioni superiori a 800 byte possono essere generate o importate in HSM. Tuttavia, se si utilizza questa chiave di dimensioni maggiori in un'operazione digest tramite JCE o key_mgmt_util, l'operazione avrà esito negativo. Si noti che se si sta utilizzando PKCS11, le chiavi HMAC possono raggiungere al massimo una dimensione di 64 byte.
- Soluzione. se si utilizzano le chiavi HMAC per le operazioni digest su HSM, assicurarsi che la dimensione sia inferiore a 800 byte.
- Stato risoluzione: nessuno in questo momento.

Problema: lo strumento client_info distribuito con Client SDK 3 elimina il contenuto del percorso specificato dall'argomento di output opzionale

- Impatto: tutti i file e le sottodirectory esistenti nel percorso di output specificato potrebbero andare persi definitivamente.
- Soluzione alternativa: non utilizzare l'argomento opzionale `-output` *`path`* quando si utilizza lo strumento `client_info`.
- Stato della risoluzione: questo problema è stato risolto nella versione 3.3.2 dell'SDK del client. È necessario effettuare l'upgrade a questo client per sfruttare i vantaggi offerti dalla soluzione.

Problema: viene visualizzato un errore quando si esegue lo strumento di configurazione SDK 5 utilizzando l'argomento --cluster-id in ambienti containerizzati

Quando utilizzi l'argomento --cluster-id con lo strumento di configurazione, viene visualizzato l'errore seguente:

```
No credentials in the property bag
```

Questo errore è causato da un aggiornamento alla versione 2 di Instance Metadata Service (IMDSv2). Per ulteriori informazioni, consulta la documentazione di IMDSv2.

- Impatto: questo problema si ripercuoterà sugli utenti che eseguono lo strumento di configurazione nelle versioni 5.5.0 e successive dell'SDK in ambienti containerizzati e che utilizzano i metadati dell'istanza EC2 per fornire le credenziali.
- Soluzione alternativa: imposta il limite di hop di risposta PUT su almeno due. Per indicazioni su come eseguire questa operazione, consulta Configurare le opzioni dei metadati dell'istanza.

Problema: viene visualizzato l'errore «Creazione cert/key dal file pfx fornito non riuscita. Errore: NotPkcs 8"

- Soluzione alternativa: puoi convertire la chiave privata SSL personalizzata in formato PKCS8 con il comando openssl: `openssl pkcs8 -topk8 -inform PEM -outform PEM -in` *`ssl_private_key`* `-out` *`ssl_private_key_pkcs8`*
- Stato della risoluzione: questo problema è stato risolto nella versione client SDK 5.12.0. ??? È necessario eseguire l'aggiornamento a questa versione del client o successiva per beneficiare della correzione.

Problema: la firma ECDSA non riesce con l'errore «meccanismo non valido» che inizia con SDK 5.16

- Impatto: le operazioni di firma ECDSA falliscono quando si utilizzano funzioni hash più deboli della forza della chiave. Questo errore si verifica perché FIPS 186-5 richiede che la funzione hash sia forte almeno quanto la forza della chiave.

  Potresti vedere un errore simile a questo nei log dei tuoi clienti:

  ```
  [cloudhsm_provider::hsm1::session::ecdsa::sign::common][][] Digest security strength
   (80) is weaker than the key security strength (128)
  ```

- Soluzione alternativa: se non riesci ad aggiornare le tue funzioni hash, puoi migrare verso cluster non FIPS, che non applicano i requisiti di resistenza dell'hash. Tuttavia, consigliamo di aggiornare le funzioni hash per mantenere la conformità FIPS.

  Come soluzione alternativa, abbiamo aggiunto un'opzione di configurazione per aggirare questo requisito. Tieni presente che questa opzione non è consigliata, poiché l'uso di ECDSA con funzioni hash più deboli non segue le migliori pratiche di sicurezza. Per utilizzare questa opzione, esegui il comando seguente (sostituendolo `configure-cli` con lo strumento di configurazione per l'SDK in uso:): the section called "Sintassi"

  ```
  sudo /opt/cloudhsm/bin/configure-cli --enable-ecdsa-with-weak-hash-function
  ```

- Risoluzione: usate una funzione hash che sia forte almeno quanto la vostra chiave ECDSA. Per informazioni sulla funzione hash e sui punti di forza delle chiavi ECDSA, vedere le tabelle 2 e 3 del NIST SP 800-57 Part 1 Rev 5.

Problema: le operazioni di firma con dati prehashati non cancellano correttamente i token di sessione in modalità interattiva

- Impatto: quando si utilizza la CLI CloudHSM in modalità interattiva con la versione SDK 5.16.1, le operazioni di firma con dati prehash non riescono a cancellare correttamente i token di sessione.
- Soluzione alternativa: utilizza la modalità a comando singolo anziché la modalità interattiva quando esegui operazioni di firma con dati prehashati. Ciò garantisce una corretta pulizia dei token dopo ogni operazione.
- Stato della risoluzione: questo problema è stato risolto in CloudHSM SDK 5.16.2. Esegui l'aggiornamento alla versione 5.16.2 o successiva per beneficiare della correzione.

Problema: il certificato client predefinito della libreria client CloudHSM scade il 31 gennaio 2026

- Impatto: non vi è alcun impatto sui clienti oltre il 31 gennaio 2026. Tutte le comunicazioni tra il client e HSM sono protette da Client-HSM TLS come descritto qui. ??? Il TLS Client-HSM utilizza i owned/managed certificati del cliente che vengono configurati dal cliente durante l'inizializzazione del cluster.
- Stato della risoluzione: risolto.

Problema: la connessione Client SDK 5 non riesce a causa di una chiave di certificato CA debole

Dopo l'aggiornamento a Client SDK 5 versione 5.17.0 o 5.17.1, l'applicazione non riesce a connettersi al cluster con il seguente errore: AWS CloudHSM

```
Certificate verification error with error code 67 and depth 1: CA certificate key too
 weak
```

- Impatto: i cluster inizializzati con un certificato CA utilizzando una chiave RSA più debole di 2048 bit non riusciranno a connettersi alle versioni 5.17.0 e 5.17.1 di Client SDK 5.
- Soluzione alternativa: eseguire il downgrade a una versione Client SDK 5 precedente alla 5.17.0 per ripristinare la connettività al cluster.
- Stato della risoluzione: questo problema è stato risolto in Client SDK 5.17.2. ??? Esegui l'aggiornamento alla versione 5.17.2 o successiva per beneficiare della correzione.

Problema: le operazioni crittografiche falliscono con errori «Il numero di sequenza è al di fuori della finestra» in caso di concorrenza elevata

- Impatto: in caso di carichi di lavoro ad alta concorrenza, Client SDK 5 può occasionalmente restituire errori operativi transitori. Client SDK 5 aumenta il volume delle operazioni parallele all'HSM come parte del suo miglioramento del livello di sicurezza. In caso di carico elevato, i ritardi nella pianificazione dei thread possono causare errori transitori che possono essere ripristinati al nuovo tentativo.
- Soluzione alternativa: implementa la logica di ripetizione a livello di applicazione per gli errori transitori. Consulta la nostra guida alle best practice. ??? Puoi anche ridurre il carico sul client

limitando il numero di operazioni crittografiche parallele o passando a un tipo di istanza con più core CPU.

- Stato della risoluzione: abbiamo aggiunto nuovi tentativi in Client SDK 5.17.2. ??? È necessario eseguire l'aggiornamento a questa versione del client o a una versione successiva per beneficiare degli aggiornamenti.

Problema: non ML-DSA è supportato nei FIPS-mode cluster

- Impatto: la generazione, la firma e la verifica delle chiavi ML-DSA (Module-Lattice-Based Digital Signature Algorithm) non sono attualmente supportate nei FIPS-mode cluster.
- Risoluzione: questo problema è stato risolto. I clienti possono ora utilizzare ML-DSA in modalità FIPS e non FIPS in AWS CloudHSM Client SDK 5.18.

Problemi noti per AWS CloudHSM istanze hsm1.medium

I seguenti problemi riguardano tutte le istanze di hsm1.medium AWS CloudHSM .

Argomenti

- Problema: l'HSM non può creare più di 250 utenti

Problema: l'HSM non può creare più di 250 utenti

- Soluzione alternativa: questo problema è stato risolto sui tipi di istanza AWS CloudHSM hsm2m.medium.
- Stato risoluzione: nessuno in questo momento.

Problemi noti per AWS CloudHSM istanze hsm2m.medium

I seguenti problemi riguardano tutte le istanze di hsm2m.medium. AWS CloudHSM

Argomenti

- Problema: aumento della latenza di accesso su hsm2m.medium
- Problema: aumento della latenza della chiave di ricerca su hsm2m.medium
- Problema: un CO che tenta di impostare l'attributo trusted di una chiave avrà esito negativo con Client SDK 5.12.0 e versioni precedenti

- Problema: la verifica ECDSA avrà esito negativo con Client SDK 5.12.0 e versioni precedenti per i cluster in modalità FIPS
- Problema: solo i PEM-formatted certificati possono essere registrati come mtls trust anchor con CloudHSM CLI
- Problema: le applicazioni dei clienti interromperanno l'elaborazione di tutte le richieste quando utilizzano MTL con una passphrase protetta chiave privata del client.
- Problema: la replica utente non riesce quando si utilizza l'interfaccia a riga di comando di CloudHSM
- Problema: le operazioni possono fallire durante la creazione del backup
- Problema: Client SDK 5.8 e versioni successive non eseguono tentativi automatici per le operazioni HSM con limitazioni in alcuni scenari su hsm2m.medium
- Problema: le operazioni di AES/CBC unwrap con zero IV non riescono su hsm2m.medium
- Problema: mancata inizializzazione della connessione HSM durante l'avvio a freddo dell'applicazione su hsm2m.medium

Problema: aumento della latenza di accesso su hsm2m.medium

- Impatto: l'accesso a hsm2m.medium segue un'interpretazione troppo rigida dei requisiti di conformità, con conseguente aumento della latenza.
- Risoluzione: se hai creato una nuova istanza hsm2m.medium o sei migrato a hsm2m.medium da hsm1.medium prima del 20 dicembre 2025, dovrai reimpostare la password per sfruttare i miglioramenti delle prestazioni che abbiamo implementato per le operazioni di accesso. ???Fai riferimento alla modifica della password per le istruzioni.

Problema: aumento della latenza della chiave di ricerca su hsm2m.medium

- Impatto: l'istanza HSM hsm2m.medium ha migliorato l'architettura fair share, il che si traduce in prestazioni prevedibili più coerenti rispetto a hsm1.medium. Con hsm1.medium, i clienti potrebbero riscontrare prestazioni chiave di ricerca più elevate a causa dell'uso irregolare delle risorse HSM. Tuttavia, le prestazioni delle chiavi di ricerca di hsm1.medium diminuiranno quando l'istanza HSM viene patchata o aggiornata con un nuovo firmware. Questo problema riguarda operazioni come in JCE. `KeyStore.getKey()`
- Risoluzione: questo problema è stato risolto. Come procedura consigliata, memorizza nella cache i risultati delle operazioni di ricerca chiave. La memorizzazione nella cache ridurrà il numero totale

di operazioni di ricerca delle chiavi in quanto è un'operazione che richiede molte risorse in HSM. Inoltre, implementate i tentativi sul lato client con backoff e jitter esponenziali per ridurre gli errori di limitazione dell'HSM.

Problema: un CO che tenta di impostare l'attributo trusted di una chiave avrà esito negativo con Client SDK 5.12.0 e versioni precedenti

- Impatto: qualsiasi utente CO che tenti di impostare l'attributo trusted di una chiave riceverà un errore che lo indica. `User type should be CO or CU`
- Risoluzione: le versioni future di Client SDK risolveranno questo problema. Gli aggiornamenti verranno annunciati nella nostra guida per l'utente. Cronologia dei documenti

Problema: la verifica ECDSA avrà esito negativo con Client SDK 5.12.0 e versioni precedenti per i cluster in modalità FIPS

- Impatto: l'operazione di verifica ECDSA eseguita per gli HSM in modalità FIPS avrà esito negativo.
- Stato della risoluzione: questo problema è stato risolto nella versione client SDK 5.13.0. È necessario eseguire l'aggiornamento a questa versione del client o a una versione successiva per beneficiare della correzione.

Problema: solo i PEM-formatted certificati possono essere registrati come mtls trust anchor con CloudHSM CLI

- Impatto: i certificati in formato DER non possono essere registrati come trust anchor MTL con CloudHSM CLI.
- Soluzione alternativa: puoi convertire un certificato in formato DER in formato PEM con il comando openssl: `openssl x509 -inform DER -outform PEM -in certificate.der -out certificate.pem`

Problema: le applicazioni dei clienti interromperanno l'elaborazione di tutte le richieste quando utilizzano MTL con una passphrase protetta chiave privata del client.

- Impatto: tutte le operazioni eseguite dall'applicazione verranno interrotte e all'utente verrà richiesta la passphrase sullo standard input più volte per tutta la durata dell'applicazione. Le operazioni scadranno e falliranno se la passphrase non viene fornita prima della durata del timeout dell'operazione.
- Soluzione alternativa: le chiavi private crittografate con passphrase non sono supportate per gli MTL. Rimuovi la crittografia con passphrase dalla chiave privata del client

Problema: la replica utente non riesce quando si utilizza l'interfaccia a riga di comando di CloudHSM

- Impatto: la replica degli utenti non riesce nelle istanze hsm2m.medium quando si utilizza la CLI CloudHSM. Il comando funziona come previsto sulle istanze hsm1.medium. `user replicate`
- Risoluzione: questo problema è stato risolto.

Problema: le operazioni possono fallire durante la creazione del backup

- Impatto: operazioni come la generazione di numeri casuali possono fallire sulle istanze hsm2m.medium mentre AWS CloudHSM crea un backup.
- Risoluzione: per ridurre al minimo le interruzioni del servizio, implementa queste best practice:
  - Crea un cluster multi-HSM
  - Configura le tue applicazioni per riprovare le operazioni del cluster

  Per ulteriori informazioni sulle best practice, vedereLe migliori pratiche per AWS CloudHSM.

Problema: Client SDK 5.8 e versioni successive non eseguono tentativi automatici per le operazioni HSM con limitazioni in alcuni scenari su hsm2m.medium

- Impatto: Client SDK 5.8 e versioni successive non riproveranno alcune operazioni HSM con limitazioni
- Soluzione alternativa: segui le best practice per progettare il cluster in modo da gestire il caricamento e implementare i tentativi a livello di applicazione. Attualmente stiamo lavorando a una soluzione. Gli aggiornamenti saranno annunciati nella nostra guida per l'utenteCronologia dei documenti.

- Stato della risoluzione: questo problema è stato risolto nel AWS CloudHSM Client SDK 5.16.2. È necessario eseguire l'aggiornamento a questa versione del client o successiva per beneficiare della correzione.

Problema: le operazioni di AES/CBC unwrap con zero IV non riescono su hsm2m.medium

- Impatto: quando si utilizza un AES/CBC meccanismo per decomprimere le chiavi utilizzando il provider AWS CloudHSM JCE, le operazioni con un IV a riempimento zero a 16 byte non riescono sulle istanze hsm2m.medium, a causa di un ulteriore controllo di convalida che non era presente nelle istanze hsm1.medium.
- Stato della risoluzione: stiamo lavorando a una soluzione che consentirà l'accettazione di IV a zero byte durante le operazioni di unwrapping. AES/CBC

Problema: mancata inizializzazione della connessione HSM durante l'avvio a freddo dell'applicazione su hsm2m.medium

- Impatto: questo problema riguarda gli avviamenti a freddo, ad esempio le distribuzioni o i riavvii delle applicazioni client. L'istanza HSM hsm2m.medium ha migliorato l'architettura fair share che garantisce prestazioni, throughput e latenza più coerenti per tutti i clienti. Attualmente su hsm1.medium, è possibile osservare prestazioni superiori a quelle previste per l'inizializzazione simultanea della connessione HSM. Tuttavia, le prestazioni di inizializzazione della connessione hsm1.medium varieranno in base agli aggiornamenti di sistema sottostanti.
- Risoluzione: segui le best practice e scagliona le distribuzioni e i riavvii delle applicazioni client per limitare il numero di applicazioni client che inizializzano le connessioni HSM contemporaneamente. Si consiglia inoltre di implementare i tentativi a livello di applicazione per l'inizializzazione delle applicazioni client. Inoltre, esegui il bootstrap utilizzando configure tool with `--cluster-id <cluster ID>` per aggiungere tutti gli IP HSM al file di configurazione del client. Questo comportamento è stato migliorato nella versione 5.17.1 e successive di AWS CloudHSM Client SDK. Ti consigliamo di eseguire l'aggiornamento alla versione SDK più recente per beneficiare di questo miglioramento.

Problemi noti della libreria PKCS #11 per AWS CloudHSM

I seguenti problemi riguardano la libreria PKCS #11 per. AWS CloudHSM

Argomenti

- Problema: il wrapping della chiave AES nella versione 3.0.0 della libreria PKCS #11 non convalida gli IV prima dell'uso
- Problema: PKCS #11 SDK 2.0.4 e versioni precedenti utilizzavano sempre l'IV predefinito di 0xA6A6A6A6A6A6A6A6 per AES key wrap and unwrap
- Problema: l'attributo CKA_DERIVE non era supportato e non è stato gestito
- Problema: l'attributo CKA_SENSITIVE non era supportato e non è stato gestito
- Problema: l'hashing e la firma in più parti non sono supportati
- Problema: C_ GenerateKeyPair non gestisce CKA_MODULUS_BITS o CKA_PUBLIC_EXPONENT nel modello privato in modo conforme agli standard
- Problema: i buffer per le operazioni delle API C_Encrypt e C_Decrypt non possono superare i 16 KB quando si utilizza il meccanismo CKM_AES_GCM
- Problema: la derivazione della chiave Elliptic-curve Diffie-Hellman (ECDH) viene eseguita parzialmente all'interno dell'HSM
- Problema: la verifica delle firme secp256k1 non riesce su piattaforme EL6 come CentOS6 e RHEL 6
- Problema: una sequenza errata di chiamate di funzione fornisce risultati indefiniti anziché dare errore
- Problema: la sessione di sola lettura non è supportata in SDK 5
- Problema: il file di intestazione cryptoki.h è Windows-only
- Problema: C_ GetAttributeValue restituisce CKR_DEVICE_ERROR quando è limitato
- Problema: la libreria PKCS #11 non riprova le operazioni di archiviazione dei certificati con limitazioni

Problema: il wrapping della chiave AES nella versione 3.0.0 della libreria PKCS #11 non convalida gli IV prima dell'uso

Se specifichi un IV di lunghezza inferiore a 8 byte, viene riempito con byte imprevedibili prima dell'uso.

> **Note**
>
> Questo ha impatto su `C_WrapKey` solo con il meccanismo `CKM_AES_KEY_WRAP`.

- Impatto: se fornisci un IV inferiore a 8 byte nella versione 3.0.0 della libreria PKCS #11, potrebbe non essere possibile annullare il wrapping della chiave.
- Soluzioni alternative:
  - Si consiglia vivamente di eseguire l'aggiornamento alla versione 3.0.1 o successiva della libreria PKCS #11, che applica correttamente la lunghezza degli IV durante il wrapping delle chiavi AES. Modifica il codice di wrapping per passare un IV NULL o specifica l'IV predefinito `0xA6A6A6A6A6A6A6A6`. Per ulteriori informazioni, vedere IV personalizzati con Non-Compliant lunghezza per AES Key Wrap.
  - Se hai eseguito il wrapping delle chiavi con la versione 3.0.0 della libreria PKCS #11 utilizzando un IV inferiore a 8 byte, contattaci per ricevere supporto.
- Stato della risoluzione: questo problema è stato risolto nella versione 3.0.1 della libreria PKCS #11. Per eseguire il wrapping delle chiavi utilizzando il wrapper delle chiavi AES, specificare un IV di lunghezza NULL o 8 byte.

Problema: PKCS #11 SDK 2.0.4 e versioni precedenti utilizzavano sempre l'IV predefinito di **`0xA6A6A6A6A6A6A6A6` per** AES key wrap and unwrap

User-provided Gli IV sono stati ignorati silenziosamente.

> **Note**
>
> Questo ha impatto su `C_WrapKey` solo con il meccanismo `CKM_AES_KEY_WRAP`.

- Impatto:
  - Se utilizzi PKCS #11 SDK 2.0.4 o una versione precedente e un IV fornito dall'utente, le chiavi vengono sottoposte al wrapping con l'IV predefinito `0xA6A6A6A6A6A6A6A6`.
  - Se hai utilizzato PKCS #11 SDK 3.0.0 o versione successiva e un IV fornito dall'utente, le chiavi vengono sottoposte al wrapping con l'IV fornito dall'utente.
- Soluzioni alternative:
  - Per annullare il wrapping delle chiavi sottoposte al wrapping con PKCS #11 SDK 2.0.4 o versioni precedenti utilizza l'IV predefinito `0xA6A6A6A6A6A6A6A6`.
  - Per annullare il wrapping delle chiavi sottoposte al wrapping con PKCS #11 SDK 3.0.0 o versioni successive, utilizza l'IV fornito dall'utente.

- Stato della risoluzione: si consiglia vivamente di modificare il codice di wrapping e annullamento del wrapping per passare un IV NULL o specificare l'IV predefinito `0xA6A6A6A6A6A6A6A6`.

Problema: l'**attributo** CKA_DERIVE non era supportato e non è stato gestito

- Stato risoluzione: abbiamo implementato le correzioni per accettare `CKA_DERIVE` se impostato su `FALSE`. `CKA_DERIVE` impostato su `TRUE` non sarà supportato fino a quando non si aggiungerà il supporto per la funzione di derivazione della chiave su AWS CloudHSM. Per trarre vantaggio dalla correzione, è necessario aggiornare il client e gli SDK alla versione 1.1.1 o successiva.

Problema: l'**attributo** CKA_SENSITIVE non era supportato e non è stato gestito

- Resolution status (Stato di risoluzione): abbiamo implementato correzioni per accettare e rispettare correttamente l'attributo `CKA_SENSITIVE`. Per trarre vantaggio dalla correzione, è necessario aggiornare il client e gli SDK alla versione 1.1.1 o successiva.

Problema: l'hashing e la firma in più parti non sono supportati

- Impact (Impatto): `C_DigestUpdate` e `C_DigestFinal` non sono implementati. `C_SignFinal` anche non è implementato e avrà esito negativo con `CKR_ARGUMENTS_BAD` per un buffer non-`NULL`.
- Soluzione alternativa: esegui l'hashing dei dati all'interno dell'applicazione e utilizzalo AWS CloudHSM solo per firmare l'hash.
- Resolution status: (Stato di risoluzione) stiamo correggendo il client e gli SDK per implementare correttamente l'hashing in più parti. Gli aggiornamenti saranno annunciati nel forum AWS CloudHSM e nella pagina della cronologia delle versioni.

Problema: **`C_ GenerateKeyPair`** non gestisce **`CKA_MODULUS_BITS`** o **`CKA_PUBLIC_EXPONENT nel modello privato in modo conforme`** agli standard

- Impact: (Impatto) `C_GenerateKeyPair` dovrebbe restituire `CKA_TEMPLATE_INCONSISTENT` quando il modello privato contiene `CKA_MODULUS_BITS` o `CKA_PUBLIC_EXPONENT`. Genera invece una chiave privata per la quale tutti i campi di utilizzo sono impostati su `FALSE`. La chiave non può essere utilizzata.

- Workaround: (Soluzione) consigliamo che l'applicazione verifichi i valori dei campi di utilizzo oltre al codice di errore.
- Resolution status: (Stato di risoluzione) stiamo implementando correzioni per restituire il corretto messaggio di errore quando viene utilizzato un modello di chiavi private non corretto. L'aggiornamento della libreria PKCS #11 sarà annunciato nella pagina della cronologia delle versioni.

Problema: i buffer per le operazioni delle API C_Encrypt e C_Decrypt non possono superare i 16 KB quando si utilizza il meccanismo CKM_AES_GCM

AWS CloudHSM non supporta AES-GCM la crittografia multiparte.

- Impact: (Impatto) non è possibile utilizzare il meccanismo `CKM_AES_GCM` per crittografare dati di dimensioni superiori a 16 KB.
- Soluzione alternativa: puoi utilizzare un meccanismo alternativo come`CKM_AES_CBC`,`CKM_AES_CBC_PAD`, oppure puoi dividere i tuoi dati in parti e crittografare ogni parte singolarmente. `AES_GCM` Se si utilizza`AES_GCM`, è necessario gestire la divisione dei dati e la successiva crittografia. AWS CloudHSM non esegue la AES-GCM crittografia in più parti per te. Nota che FIPS richiede che il vettore di inizializzazione (IV) `AES-GCM` sia generato sull'HSM. Pertanto, l'IV per ogni parte dei dati AES-GCM crittografati sarà diverso.
- Resolution status: (Stato di risoluzione) stiamo correggendo l'SDK in modo che restituisca esplicitamente un errore se il buffer dei dati è di dimensioni eccessive. Restituiamo `CKR_MECHANISM_INVALID` per le operazioni API `C_EncryptUpdate` e `C_DecryptUpdate`. Stiamo valutando alternative per supportare buffer più grandi senza dover ricorrere alla crittografia in più parti. Gli aggiornamenti verranno annunciati nel AWS CloudHSM forum e nella pagina della cronologia delle versioni.

Problema: la derivazione della chiave Elliptic-curve Diffie-Hellman (ECDH) viene eseguita parzialmente all'interno dell'HSM

La chiave privata EC rimane sempre all'interno dell'HSM, ma il processo di derivazione della chiave viene eseguito in più fasi. Pertanto, nel client sono disponibili i risultati intermedi di ciascuna fase.

- Impatto: in Client SDK 3, la chiave derivata utilizzando il `CKM_ECDH1_DERIVE` meccanismo è prima disponibile sul client e poi importata nell'HSM. Un handle della chiave viene quindi restituito all'applicazione.

- Soluzione alternativa: se stai implementando SSL/TLS Offload in AWS CloudHSM, questa limitazione potrebbe non essere un problema. Se l'applicazione richiede che la chiave rimanga sempre all'interno di un limite FIPS, prendere in considerazione l'utilizzo di un protocollo alternativo che non si basi sulla derivazione della chiave ECDH.
- Stato della risoluzione: SDK 5.16 ora supporta ECDH con Key Derivation, che viene eseguita interamente all'interno dell'HSM.

Problema: la verifica delle firme secp256k1 non riesce su piattaforme EL6 come CentOS6 e RHEL 6

Questo si verifica perché la libreria PKCS#11 di CloudHSM evita una chiamata di rete durante l'inizializzazione dell'operazione di verifica utilizzando OpenSSL per verificare i dati della curva CE. Poiché Secp256k1 non è supportato dal pacchetto OpenSSL predefinito sulle piattaforme EL6, l'inizializzazione non riesce.

- Impatto: la verifica della firma Secp256k1 non riuscirà sulle piattaforme EL6. La chiamata di verifica non riuscirà e restituirà un errore `CKR_HOST_MEMORY`.
- Soluzione alternativa: si consiglia di utilizzare Amazon Linux 1 o qualsiasi piattaforma EL7 se l'applicazione PKCS#11 richiede di verificare le firme secp256k1. In alternativa, eseguire l'aggiornamento del pacchetto OpenSSL a una versione che supporti la curva secp256k1.
- Stato della risoluzione: stiamo implementando correzioni per tornare a HSM se la convalida della curva locale non è disponibile. L'aggiornamento della libreria PKCS #11 sarà annunciato nella pagina della cronologia delle versioni.

Problema: una sequenza errata di chiamate di funzione fornisce risultati indefiniti anziché dare errore

- Impatto: se si chiama una sequenza errata di funzioni, il risultato finale non è corretto anche se le singole chiamate di funzione danno esito positivo. Ad esempio, i dati decrittografati potrebbero non corrispondere al testo in chiaro originale oppure potrebbe venire meno la verifica delle firme. Questo problema riguarda sia le operazioni a parte singola che quelle in più parti.

  Le tabelle seguenti mostrano le sequenze di chiamate di funzione corrette per le operazioni PKCS #11.

  Sequenze corrette per operazioni composte da una sola parte:

Operation	Sequenza corretta
Crittografa	`C_EncryptInit` → `C_Encrypt`
Decrypt	`C_DecryptInit` → `C_Decrypt`
Sign	`C_SignInit` → `C_Sign`
Verifica	`C_VerifyInit` → `C_Verify`
Trova oggetti	`C_FindObjectsInit` → `C_FindObjects` → `C_FindObjectsFinal`

Sequenze corrette per operazioni in più parti:

Operation	Sequenza corretta
Crittografa	`C_EncryptInit` → `C_EncryptUpdate` (uno o più) → `C_EncryptFinal`
Decrypt	`C_DecryptInit` → `C_DecryptUpdate` (uno o più) → `C_DecryptFinal`
Sign	`C_SignInit` → `C_SignUpdate` (uno o più) → `C_SignFinal`
Verifica	`C_VerifyInit` → `C_VerifyUpdate` (uno o più) → `C_VerifyFinal`

- Soluzione alternativa: l'applicazione deve, in conformità con la specifica PKCS #11, utilizzare la sequenza corretta di chiamate di funzione per le operazioni in una sola parte e in più parti, come mostrato nelle tabelle precedenti. L'applicazione non deve fare affidamento sulla libreria PKCS #11 di CloudHSM per restituire un errore in questa circostanza.

Problema: la sessione di sola lettura non è supportata in SDK 5

- Problema: SDK 5 non supporta l'apertura di sessioni con. Read-Only `C_OpenSession`

- Impatto: se tenti di chiamare `C_OpenSession` senza fornire `CKF_RW_SESSION`, la chiamata darà esito negativo con l'errore `CKR_FUNCTION_FAILED`.
- Soluzione alternativa: quando si apre una sessione, è necessario trasferire i flag `CKF_SERIAL_SESSION | CKF_RW_SESSION` alla chiamata di funzione `C_OpenSession`.

Problema: il file di intestazione **`cryptoki.h`** è Windows-only

- Problema: con le versioni di AWS CloudHSM Client SDK 5 da 5.0.0 a 5.4.0 su Linux, il file di intestazione è compatibile solo con i sistemi operativi Windows. `/opt/cloudhsm/include/pkcs11/cryptoki.h`
- Impatto: è possibile che si verifichino problemi quando si tenta di includere questo file di intestazione nell'applicazione sui sistemi operativi. Linux-based
- Stato della risoluzione: aggiornamento a AWS CloudHSM Client SDK 5 versione 5.4.1 o successiva, che include una Linux-compatible versione di questo file di intestazione.

Problema: **`C_ GetAttributeValue`** restituisce CKR_DEVICE_ERROR quando è limitato

Quando una chiamata viene limitata, viene restituita. `C_GetAttributeValue` `CKR_DEVICE_ERROR` Ritornano le altre operazioni limitate. `CKR_FUNCTION_FAILED`

- Impatto: se l'applicazione riprova`CKR_FUNCTION_FAILED`, non riprova queste richieste limitate.
- Soluzione alternativa: riprovare le operazioni limitate con un backoff esponenziale su entrambi i lati. `CKR_DEVICE_ERROR` `CKR_FUNCTION_FAILED` Se la limitazione persiste, aggiungi altri HSM al cluster per aumentare la velocità totale di lettura e scrittura o ridurre la frequenza di richieste. Per ulteriori informazioni, consulta Limitazione HSM.
- Stato della risoluzione: questo problema è stato risolto in Client SDK 5.18.0. `C_GetAttributeValue`ora restituisce `CKR_FUNCTION_FAILED` le letture limitate dell'archivio dei certificati e degli attributi chiave. Effettua l'aggiornamento alla versione 5.18.0 o successiva per beneficiare della correzione.

Problema: la libreria PKCS #11 non riprova le operazioni di archiviazione dei certificati con limitazioni

Quando si superano i limiti di velocità per le operazioni di archiviazione dei certificati, la libreria PKCS #11 non riprova automaticamente queste operazioni limitate.

- Impatto: un'operazione di archiviazione limitata dei certificati restituisce un errore all'applicazione anziché essere ritentata automaticamente.
- Soluzione alternativa: riprova le operazioni di archiviazione limitata dei certificati nell'applicazione con un backoff esponenziale. Se la limitazione persiste, aggiungi altri HSM al cluster per aumentare la velocità totale di lettura e scrittura o ridurre la frequenza di richieste. Per ulteriori informazioni, consulta Limitazione HSM.
- Stato della risoluzione: questo problema è stato risolto in Client SDK 5.18.0. La libreria PKCS #11 ora riprova automaticamente le operazioni di archiviazione dei certificati con limitazioni. Effettua l'aggiornamento alla versione 5.18.0 o successiva per beneficiare della correzione.

Problemi noti dell'SDK JCE per AWS CloudHSM

I seguenti problemi influiscono sull'SDK JCE per. AWS CloudHSM

Argomenti

- Issue: (Problema:) quando si utilizzano coppie di chiavi asimmetriche, viene visualizzata la capacità della chiave occupata anche quando non si creano o importano esplicitamente le chiavi
- Problema: il JCE KeyStore è di sola lettura
- Problema: i buffer per la AES-GCM crittografia non possono superare i 16.000 byte
- Problema: la derivazione della chiave Elliptic-curve Diffie-Hellman (ECDH) viene eseguita parzialmente all'interno dell'HSM
- Problema: KeyGenerator e interpreta KeyAttribute erroneamente il parametro della dimensione della chiave come numero di byte anziché di bit
- Problema: Client SDK 5 genera l'avviso "An illegal reflective access operation has occurred"
- Problema: il pool di sessioni JCE è esaurito
- Problema: perdita di memoria del Client SDK 5 con operazioni GetKey

Issue: (Problema:) quando si utilizzano coppie di chiavi asimmetriche, viene visualizzata la capacità della chiave occupata anche quando non si creano o importano esplicitamente le chiavi

- Impact: (Impatto:) questo problema può causare l'esaurirsi in modo inaspettato dello spazio delle chiavi nei moduli HSM e si verifica quando l'applicazione utilizza un oggetto chiave JCE standard per le operazioni di crittografia anziché un oggetto `CaviumKey`. Quando utilizzi un oggetto chiave JCE standard, `CaviumProvider` importa implicitamente tale chiave nell'HSM come chiave di sessione e non elimina questa chiave finché l'applicazione non viene chiusa. Di conseguenza, le chiavi si accumulano mentre l'applicazione è in esecuzione e ciò può causare l'esaurimento dello spazio libero delle chiavi nei moduli HSM, bloccando l'applicazione.
- Workaround: (Soluzione alternativa:) quando si utilizza la classe `CaviumSignature`, `CaviumCipher`, `CaviumMac` o la classe `CaviumKeyAgreement`, è necessario fornire la chiave come `CaviumKey` invece di un oggetto chiave JCE standard.

  È possibile convertire manualmente una chiave normale in `CaviumKey` utilizzando la classe `ImportKey` e quindi è possibile eliminare manualmente la chiave al termine dell'operazione.
- Resolution status: (Stato di risoluzione): stiamo aggiornando `CaviumProvider` per gestire correttamente le importazioni implicite. Non appena disponibile, la correzione sarà annunciata nella pagina della cronologia delle versioni.

Problema: il JCE KeyStore è di sola lettura

- Impact: (Impatto) non è possibile archiviare un tipo di oggetto che attualmente non è supportato dall'HSM nel keystore JCE. Nello specifico, non è possibile archiviare certificati nel keystore. Questo impedisce l'interoperabilità con strumenti come jarsigner, che si aspettano di trovare il certificato nel keystore.
- Workaround: (Soluzione) è possibile rilavorare il codice per caricare i certificati da file locali o da una posizione del bucket S3 anziché dal keystore.
- Stato della risoluzione: è possibile utilizzare AWS CloudHSM keystore per archiviare i certificati.

Problema: i buffer per la AES-GCM crittografia non possono superare i 16.000 byte

Multi-part AES-GCM la crittografia non è supportata.

- Impatto: non è possibile utilizzare AES-GCM per crittografare dati di dimensioni superiori a 16.000 byte.
- Soluzione alternativa: puoi utilizzare un meccanismo alternativo, ad esempioAES-CBC, oppure puoi dividere i dati in parti e crittografare ogni parte singolarmente. Se si dividono i dati, è necessario gestire il testo cifrato diviso e la sua decrittografia. Poiché FIPS richiede la generazione del vettore di inizializzazione (IV) sull'HSM, l'IV per ogni dato AES-GCM sarà AES-GCM-encrypted diverso.
- Resolution status: (Stato di risoluzione) stiamo correggendo l'SDK in modo che restituisca esplicitamente un errore se il buffer dei dati è di dimensioni eccessive. Stiamo valutando alternative che supportino buffer più grandi senza dover ricorrere alla crittografia in più parti. Gli aggiornamenti saranno annunciati nel forum AWS CloudHSM e nella pagina della cronologia delle versioni.

Problema: la derivazione della chiave Elliptic-curve Diffie-Hellman (ECDH) viene eseguita parzialmente all'interno dell'HSM

La chiave privata EC rimane sempre all'interno dell'HSM, ma il processo di derivazione della chiave viene eseguito in più fasi. Pertanto, nel client sono disponibili i risultati intermedi di ciascuna fase. Un esempio di derivazione della chiave ECDH è disponibile negli esempi di codice Java.

- Impatto: Client SDK 3 aggiunge la funzionalità ECDH al JCE. Quando si utilizza la `KeyAgreement` classe per derivare a SecretKey, questa è prima disponibile sul client e quindi importata nell'HSM. Un handle della chiave viene quindi restituito all'applicazione.
- Soluzione alternativa: se stai implementando SSL/TLS Offload in AWS CloudHSM, questa limitazione potrebbe non essere un problema. Se l'applicazione richiede che la chiave rimanga sempre all'interno di un limite FIPS, prendere in considerazione l'utilizzo di un protocollo alternativo che non si basi sulla derivazione della chiave ECDH.
- Stato della risoluzione: SDK 5.16 ora supporta ECDH con Key Derivation, che viene eseguita interamente all'interno dell'HSM.

Problema: KeyGenerator e interpreta KeyAttribute erroneamente il parametro della dimensione della chiave come numero di byte anziché di bit

Quando si genera una chiave utilizzando la `init` funzione della KeyGenerator classe o l'`SIZE`attributo dell'AWS CloudHSM KeyAttribute enum, l'API si aspetta erroneamente che l'argomento sia il numero di byte della chiave, quando dovrebbe invece essere il numero di bit della chiave.

- Impatto: le versioni di Client SDK da 5.4.0 a 5.4.2 prevedono erroneamente che la dimensione della chiave venga fornita alle API specificate sotto forma di byte.
- Soluzione alternativa: convertire la dimensione della chiave da bit a byte prima di utilizzare la KeyGenerator class o l' KeyAttribute enum per generare le chiavi utilizzando il provider AWS CloudHSM JCE se si utilizzano le versioni Client SDK da 5.4.0 a 5.4.2.
- Stato della risoluzione: aggiorna la versione SDK del client alla 5.5.0 o successiva, che include una correzione per prevedere correttamente le dimensioni delle chiavi in bit quando si utilizza la classe o l'enum per generare le chiavi. KeyGenerator KeyAttribute

Problema: Client SDK 5 genera l'avviso "An illegal reflective access operation has occurred"

Quando si utilizza Client SDK 5 con Java 11, CloudHSM genera il seguente avviso Java:

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by
 com.amazonaws.cloudhsm.jce.provider.CloudHsmKeyStore (file:/opt/cloudhsm/java/
cloudhsm-jce-5.6.0.jar) to field java.security .KeyStore.keyStoreSpi
WARNING: Please consider reporting this to the maintainers of
 com.amazonaws.cloudhsm.jce.provider.CloudHsmKeyStore
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective
 access operations
WARNING: All illegal access operations will be denied in a future release
```

Questo problema è stato risolto nella versione 5.8 e successive di Client SDK.

> **Note**
>
> OpenJDK 11 non è più supportato. Client SDK 5.17.1 è l'ultima versione a supportare OpenJDK 11.

Problema: il pool di sessioni JCE è esaurito

Impatto: potresti non essere in grado di eseguire operazioni in JCE dopo aver visualizzato il seguente messaggio:

```
com.amazonaws.cloudhsm.jce.jni.exception.InternalException: There are too many
 operations
happening at the same time: Reached max number of sessions in session pool: 1000
```

Soluzioni alternative:

- Riavvia l'applicazione JCE se riscontri un impatto.
- Quando si esegue un'operazione, potrebbe essere necessario completare l'operazione JCE prima di perdere il riferimento all'operazione.

> **Note**
>
> A seconda dell'operazione, potrebbe essere necessario un metodo di completamento.

Operation	Metodo/i di completamento
Crittografia	`doFinal()` in modalità crittografia o decrittografia `wrap()` in modalità wrapping `unwrap()` in modalità annullamento del wrapping
KeyAgreement	`generateSecret()` o `generateSecret(String)`
KeyPairGenerator	`generateKeyPair()` , `genKeyPair()` o `reset()`
KeyStore	Nessun metodo necessario
MAC	`doFinal()` o `reset()`
MessageDigest	`digest()` o `reset()`
SecretKeyFactory	Nessun metodo necessario

Operation	Metodo/i di completamento
SecureRandom	Nessun metodo necessario
Firma	`sign()` in modalità firma `verify()` in modalità verifica

Stato della risoluzione: abbiamo risolto questo problema in Client SDK 5.9.0 e versioni successive. Per risolvere questo problema, aggiorna il tuo Client SDK a una di queste versioni.

Problema: perdita di memoria del Client SDK 5 con operazioni GetKey

- Impatto: l'`getKey`operazione API presenta una perdita di memoria in JCE nelle versioni Client SDK 5.10.0 e precedenti. Se utilizzi l'`getKey`API più volte nell'applicazione, ciò comporterà una maggiore crescita della memoria e, di conseguenza, un aumento dell'ingombro di memoria nell'applicazione. Nel tempo, ciò potrebbe causare errori di limitazione o richiedere il riavvio dell'applicazione.
- Soluzione alternativa: consigliamo di eseguire l'aggiornamento a Client SDK 5.11.0. Se ciò non è possibile, ti consigliamo di non chiamare l'`getKey`API più volte nell'applicazione. Piuttosto, riutilizza il più possibile la chiave restituita in precedenza dall'`getKey`operazione precedente.
- Stato della risoluzione: aggiorna la versione SDK del client alla 5.11.0 o successiva, che include una correzione per questo problema.

Problemi noti di OpenSSL Dynamic Engine per AWS CloudHSM

Questi sono i problemi noti di OpenSSL Dynamic Engine for. AWS CloudHSM

Argomenti

- Problema: non è possibile eseguire l'installazione AWS CloudHSM OpenSSL Dynamic Engine su RHEL 6 e CentOS6
- Problema: per impostazione predefinita, è supportato solo l'offload RSA sull'HSM
- Problema: la crittografia e la decrittografia RSA con riempimento OAEP tramite una chiave nell'HSM non sono supportate
- Problema: viene eseguito sull'HSM solo l'offload della generazione di chiavi private delle chiavi RSA ed ECC

- Problema: non è possibile installare OpenSSL Dynamic Engine per Client SDK 3 su RHEL 8, CentOS 8 o Ubuntu 18.04 LTS
- Problema: SHA-1 firma e verifica la deprecazione su RHEL 9 (9.2+)
- Problema: AWS CloudHSM OpenSSL Dynamic Engine non è compatibile con il provider FIPS per OpenSSL v3.x
- Problema: l' SSL/TLS offload non riesce con le suite di crittografia ECDSA in TLS 1.0 e TLS 1.1 a partire da SDK 5.16

Problema: non è possibile eseguire l'installazione AWS CloudHSM OpenSSL Dynamic Engine su RHEL 6 e CentOS6

- Impatto: OpenSSL Dynamic Engine supporta OpenSSL 1.0.2 [f+]. Per impostazione predefinita, RHEL 6 e CentOS 6 vengono forniti con OpenSSL 1.0.1.
- Soluzione alternativa: aggiornare la libreria OpenSSL su RHEL 6 e CentOS 6 alla versione 1.0.2 [f +].

Problema: per impostazione predefinita, è supportato solo l'offload RSA sull'HSM

- Impatto: per massimizzare le prestazioni, l'SDK non è configurato per scaricare funzioni aggiuntive come la generazione o le operazioni di numeri casuali. EC-DH
- Workaround: (Soluzione) contattarci attraverso l'apertura di un caso di supporto se si necessita dell'offload di operazioni aggiuntive.
- Resolution status: (Stato di risoluzione) stiamo aggiungendo il supporto all'SDK per configurare le opzioni di offload tramite un file di configurazione. Non appena disponibile, l'aggiornamento sarà annunciato nella pagina della cronologia delle versioni.

Problema: la crittografia e la decrittografia RSA con riempimento OAEP tramite una chiave nell'HSM non sono supportate

- Impact: (Impatto) qualsiasi chiamata alla crittografia e alla decrittografia RSA con riempimento OAEP ha esito negativo con un errore di divisione per zero. Questo avviene perché il motore dinamico OpenSSL chiama l'operazione a livello locale utilizzando il falso file PEM anziché eseguire l'offload dell'operazione sull'HSM.

- Workaround: (Soluzione) è possibile eseguire questa procedura utilizzando Libreria PKCS #11 per AWS CloudHSM Client SDK 5 o Provider JCE per AWS CloudHSM Client SDK 5.
- Resolution status: (Stato di risoluzione) stiamo aggiungendo il supporto all'SDK per eseguire correttamente l'offload di questa operazione. Non appena disponibile, l'aggiornamento sarà annunciato nella pagina della cronologia delle versioni.

Problema: viene eseguito sull'HSM solo l'offload della generazione di chiavi private delle chiavi RSA ed ECC

Per qualsiasi altro tipo di chiave, il AWS CloudHSM motore OpenSSL non viene utilizzato per l'elaborazione delle chiamate. Viene invece utilizzato il motore OpenSSL locale. Questo genera una chiave a livello locale nel software.

- Impact: (Impatto) poiché il failover è silenzioso, non vi sono indicazioni della mancata ricezione di una chiave che era stata generata in modo sicuro sull'HSM. Se la chiave è generata a livello locale da OpenSSL nel software, si visualizzerà una traccia di output contente la stringa `"..........+++++"`. Questa traccia è assente in caso di offload dell'operazione nell'HSM. Poiché la chiave non è generata o archiviata nell'HSM, non sarà disponibile per l'utilizzo futuro.
- Workaround: (Soluzione) utilizzare il motore OpenSSL solo per i tipi di chiavi che supporta. Per tutti gli altri tipi di chiave, usa PKCS #11 o JCE nelle applicazioni o `key_mgmt_util` usali nella CLI.

Problema: non è possibile installare OpenSSL Dynamic Engine per Client SDK 3 su RHEL 8, CentOS 8 o Ubuntu 18.04 LTS

- Impatto: per impostazione predefinita, RHEL 8, CentOS 8 e Ubuntu 18.04 LTS sono dotati di una versione di OpenSSL che non è compatibile con OpenSSL Dynamic Engine per Client SDK 3.
- Soluzione alternativa: utilizza una piattaforma Linux che fornisca supporto per OpenSSL Dynamic Engine. Per ulteriori informazioni sulle piattaforme supportate, consulta la pagina relativa alle piattaforme supportate.
- Stato della risoluzione: AWS CloudHSM supporta queste piattaforme con OpenSSL Dynamic Engine for Client SDK 5. Per ulteriori informazioni, consulta le pagine relative alle piattaforme supportate e a OpenSSL Dynamic Engine.

Problema: SHA-1 firma e verifica la deprecazione su RHEL 9 (9.2+)

- Impatto: l'uso del SHA-1 message digest per scopi crittografici è diventato obsoleto in RHEL 9 (9.2+). Di conseguenza, le operazioni di firma e verifica con l' SHA-1 utilizzo di OpenSSL Dynamic Engine falliranno.
- Soluzione alternativa: se il tuo scenario richiede l'uso di firme crittografiche signing/verifying esistenti o di SHA-1 terze parti, consulta Enhancing RHEL Security: Understanding SHA-1 deprecation on RHEL 9 (9.2+) e RHEL 9 (9.2+) Release Notes per ulteriori dettagli.

Problema: AWS CloudHSM OpenSSL Dynamic Engine non è compatibile con il provider FIPS per OpenSSL v3.x

- Impatto: riceverai un errore se tenti di utilizzare AWS CloudHSM OpenSSL Dynamic Engine quando il provider FIPS è abilitato per le versioni 3.x di OpenSSL.
- Soluzione alternativa: per utilizzare AWS CloudHSM OpenSSL Dynamic Engine con OpenSSL versioni 3.x, assicurati che il provider «predefinito» sia configurato. Per saperne di più sul provider predefinito, visita il sito Web di OpenSSL. https://www.openssl.org/docs/man3.0/man7/OSSL_PROVIDER-default.html

Problema: l' SSL/TLS offload non riesce con le suite di crittografia ECDSA in TLS 1.0 e TLS 1.1 a partire da SDK 5.16

- Impatto: i tentativi di connessione che utilizzano TLS 1.0 o TLS 1.1 falliscono perché queste versioni utilizzano SHA-1 il servizio di firma, che non soddisfa i requisiti FIPS 186-5. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf
- Soluzione alternativa: se non riesci ad aggiornare immediatamente le versioni TLS, puoi migrare a cluster non FIPS, che non applicano i requisiti di resistenza dell'hash. Tuttavia, consigliamo di eseguire l'aggiornamento a TLS 1.2 o TLS 1.3 per mantenere la conformità FIPS e le best practice di sicurezza.
- Risoluzione: aggiorna l'implementazione per utilizzare TLS 1.2 o TLS 1.3. L'Internet Engineering Task Force (IETF) ha reso obsoleti TLS 1.0 e TLS 1.1 a causa di problemi di sicurezza.

Problemi noti del Key Storage Provider (KSP) per AWS CloudHSM

Questi sono i problemi noti di Key Storage Provider (KSP) per. AWS CloudHSM

Argomenti

- Problema: la verifica di un archivio di certificati non riesce
- Problema: incoerenza del nome del contenitore nell'archivio dei certificati durante l'utilizzo della modalità di compatibilità SDK3 per Client SDK 5

Problema: la verifica di un archivio di certificati non riesce

Quando si utilizzano le versioni Client SDK 5.14 e 5.15, la chiamata `certutil -store my CERTIFICATE_SERIAL_NUMBER` genera il seguente errore:

```
ERROR: Could not verify certificate public key against private key
```

- Impatto: non è possibile utilizzare `certutil` per convalidare un archivio di certificati creato con Client SDK 5.
- Soluzione alternativa: convalida la coppia di chiavi associata al certificato firmando un file utilizzando la chiave privata e verificando la firma utilizzando la chiave pubblica. Questa operazione può essere eseguita utilizzando Microsoft SignTool seguendo i passaggi forniti qui. ???
- Stato della risoluzione: stiamo lavorando per aggiungere il supporto per la verifica dei certificati utilizzando`certutil`. Non appena disponibile, la correzione sarà annunciata nella pagina della cronologia delle versioni.

Problema: incoerenza del nome del contenitore nell'archivio dei certificati durante l'utilizzo della modalità di compatibilità SDK3 per Client SDK 5

Quando si utilizza il `certutil -store my CERTIFICATE_SERIAL_NUMBER` comando per visualizzare i certificati i cui file di riferimento chiave sono stati generati utilizzando il comando generate-file in 5.16.0, si verifica il seguente errore: AWS CLI

```
ERROR: Container name inconsistent: CONTAINER_NAME
```

Questo errore si verifica perché c'è una mancata corrispondenza tra il nome del contenitore memorizzato nel certificato e il nome del file di riferimento della chiave generato dalla CLI di CloudHSM.

- Impatto: nonostante questo errore, i certificati e le chiavi associate rimangono completamente funzionanti. Tutte le applicazioni che utilizzano questi certificati continueranno a funzionare normalmente.
- Soluzione alternativa: per risolvere questo errore, rinomina il nome del file di riferimento chiave in Nome contenitore semplice o univoco. Fate riferimento al seguente esempio di output del comando `certutil -store my`

```
Subject: CN=www.website.com, OU=Organizational-Unit, O=Organization, L=City, S=State,
 C=US
Non-root Certificate
Cert Hash(sha1): 1add52
Key Container = 7e3c-b2f5
Simple container name: tq-3daacd89
Unique container name: tq-3daacd89
ERROR: Container name inconsistent: 7e3c-b2f5
```

  Per impostazione predefinita, i file di riferimento chiave verranno archiviati in `C:\Users\Default\AppData\Roaming\Microsoft\Crypto\CaviumKSP\GlobalPartition`

  1. Rinomina il file di riferimento della chiave con il nome semplice del contenitore.
  2. Ripara l'archivio dei certificati con il nuovo nome del contenitore di chiavi. Per maggiori dettagli, consulta i passaggi da 12 a 14 in KSP Migration.
- Stato della risoluzione: questo problema è stato risolto nella versione 5.16.1 di Client SDK. Per risolvere questo problema, aggiorna il tuo Client SDK alla versione 5.16.1 o successiva.

Problemi noti del provider OpenSSL per AWS CloudHSM

Questi sono i problemi noti di OpenSSL Provider for. AWS CloudHSM

Argomenti

- Problema: errori nella CLI di OpenSSL se utilizzata con OpenSSL Provider
- Problema: l'offload TLS Ed25519 non è supportato sui cluster FIPS
- Problema: ML-DSA le operazioni non riescono su piattaforme con OpenSSL precedenti alla 3.5
- Problema: l'handshake ML-DSA TLS non riesce su Amazon Linux 2023 e RHEL senza «algoritmi di firma condivisi»

Problema: errori nella CLI di OpenSSL se utilizzata con OpenSSL Provider

- Impatto: il provider AWS CloudHSM OpenSSL non supportava le operazioni CLI OpenSSL (creazione CSR, firma dei certificati). È stato necessario utilizzare l'OpenSSL Dynamic Engine per le operazioni CSR e relative ai certificati.
- Stato della risoluzione: Client SDK 5.18.0 risolve questo problema. L'OpenSSL Provider ora supporta le operazioni CLI OpenSSL per tutti i tipi di chiave (RSA, EC, Ed25519 e). ML-DSA Esegui l'aggiornamento alla versione 5.18.0 o successiva per beneficiare della correzione.

Problema: l'offload TLS Ed25519 non è supportato sui cluster FIPS

- Impatto: i tipi di chiave Ed25519 non sono disponibili sui cluster. FIPS-mode I tentativi di utilizzare questo tipo di chiave per l'offload TLS su un cluster FIPS falliscono.
- Risoluzione: utilizzare Ed25519 solo su cluster non FIPS. Per i cluster FIPS, utilizza i tipi di chiave RSA o EC per l'offload TLS.

Problema: ML-DSA le operazioni non riescono su piattaforme con OpenSSL precedenti alla 3.5

- Impatto: i tipi di ML-DSA chiave (ML-DSA-44, ML-DSA-65, ML-DSA-87) richiedono OpenSSL 3.5 o successivo per la creazione di CSR, la creazione di certificati e l'offload TLS. Sulle piattaforme con un vecchio sistema OpenSSL, ML-DSA le operazioni falliscono con un errore di «algoritmo non supportato».
- Risoluzione: usa una piattaforma con OpenSSL 3.5 o versione successiva o crea un binario OpenSSL 3.5+ personalizzato per le operazioni. ML-DSA

Problema: l'handshake ML-DSA TLS non riesce su Amazon Linux 2023 e RHEL senza «algoritmi di firma condivisi»

- Impatto: le connessioni TLS che utilizzano ML-DSA certificati non riescono sulle piattaforme Amazon Linux 2023, RHEL 9 e RHEL 10 con l'errore. `tls1_set_server_sigalgs:no shared signature algorithms` Ciò si verifica perché il framework delle politiche crittografiche a livello di sistema non include algoritmi di ML-DSA firma (`mldsa44`,,) nell'elenco consentito predefinito. `mldsa65` `mldsa87` `SignatureAlgorithms` Non-TLS le operazioni (generazione delle chiavi,

firma, verifica) non sono influenzate. Ubuntu 26.04 LTS non è interessato perché non utilizza il framework delle politiche crittografiche.

- Risoluzione: abilita la sotto-policy crittografica post-quantum (PQ) sulla tua piattaforma:
  - Amazon Linux 2023 (richiede AL2023.12 +): Esegui. `sudo update-crypto-policies --set DEFAULT:PQ` Per ulteriori informazioni, consulta le politiche di Post-quantum crittografia nella Guida per l'utente di Amazon Linux 2023.
  - RHEL 9 (richiede RHEL 9.8+): Esegui, allora. `sudo dnf update crypto-policies` `sudo update-crypto-policies --set DEFAULT:PQ` Per ulteriori informazioni, consulta Utilizzo delle politiche crittografiche a livello di sistema nella documentazione di RHEL 9.
  - RHEL 10 (richiede RHEL 10.1+): viene abilitato automaticamente nella policy. ML-DSA `DEFAULT` Esegui `sudo dnf update` per assicurarti di disporre del pacchetto di crypto-policies più recente. Per ulteriori informazioni, consulta Utilizzo delle politiche crittografiche a livello di sistema nella documentazione di RHEL 10.

Problemi noti dell'interfaccia a riga di comando di CloudHSM per AWS CloudHSM

I seguenti problemi influiscono sulla CLI di CloudHSM per. AWS CloudHSM

Argomenti

- Il filtro key-reference non riesce a selezionare le chiavi di sessione

Il **`filtro`** key-reference non riesce a selezionare le chiavi di sessione

I comandi utilizzati `key-reference` per filtrare le chiavi di sessione (temporanee) hanno esito negativo e genera l'errore. `UX000: Ephemeral key is not expected because we cannot build it without HSM Connection`

Il comando key set-attribute non è interessato e può selezionare le chiavi di sessione tramite. `key-reference`

- Soluzione alternativa: utilizza filtri basati sugli attributi (ad esempio`attr.label`) per selezionare le chiavi di sessione. Se più chiavi di sessione condividono attributi identici, utilizza key set-attribute con il `key-reference` filtro per assegnare prima etichette univoche, quindi filtra per etichetta.

- Stato della risoluzione: questo problema è stato risolto in Client SDK 5.18.0. Il `key-reference` filtro ora seleziona le chiavi di sessione (temporanee) nella CLI di CloudHSM e in JCE. Effettua l'aggiornamento alla versione 5.18.0 o successiva per beneficiare della correzione.

Problemi noti per le istanze Amazon EC2 che eseguono Amazon Linux 2 con AWS CloudHSM

I seguenti problemi riguardano AWS CloudHSM le istanze Amazon EC2 in esecuzione su Amazon Linux 2.

Problema: Amazon Linux 2 versione 2018.07 utilizza un **`pacchetto`** ncurses aggiornato (versione 6) attualmente incompatibile con AWS CloudHSM SDK

Viene restituito il seguente errore durante l'esecuzione di cloudhsm_mgmt_util o key_mgmt_util: AWS CloudHSM AWS CloudHSM Utilità di gestione (CMU) AWS CloudHSM Utilità di gestione delle chiavi (KMU)

```
/opt/cloudhsm/bin/cloudhsm_mgmt_util: error while loading shared libraries:
 libncurses.so.5: cannot open shared object file: No such file or directory
```

- Impatto: le istanze in esecuzione su Amazon Linux 2 versione 2018.07 non saranno in grado di utilizzare tutte le utilità. AWS CloudHSM
- Soluzione alternativa: esegui il comando seguente nelle istanze Amazon Linux 2 EC2 per installare il pacchetto `ncurses` supportato (versione 5):

  ```
  sudo yum update && yum install ncurses-compat-libs
  ```

- Stato della risoluzione: questo problema è stato risolto nella versione 1.1.2 del client AWS CloudHSM . È necessario effettuare l'upgrade a questo client per sfruttare i vantaggi offerti dalla soluzione.

Problemi noti relativi all'integrazione di applicazioni di terze parti con AWS CloudHSM

I seguenti problemi influiscono sull' AWS CloudHSM integrazione con applicazioni di terze parti.

Problema: Client SDK 3 non supporta l'impostazione da parte di Oracle dell'attributo PKCS #11 CKA_MODIFIABLE durante la generazione della **`chiave`** master

Questo limite è definito nella libreria PKCS #11. Per ulteriori informazioni, consulta l'annotazione 1 su Attributi PKCS #11 supportati.

- Impatto: la creazione della chiave master Oracle non va a buon fine.
- Soluzione alternativa: impostare la variabile di ambiente speciale `CLOUDHSM_IGNORE_CKA_MODIFIABLE_FALSE` su TRUE durante la creazione di una nuova chiave master. Questa variabile di ambiente è necessaria solo per la generazione di chiavi master e non deve essere utilizzata per altri motivi. Ad esempio, si utilizzerà questa variabile per la prima chiave master creata e quindi si utilizzerà nuovamente questa variabile di ambiente solo se si desidera ruotare l'edizione della chiave master. Per ulteriori informazioni, consulta Creazione della chiave di crittografia principale di Oracle TDE.
- Stato di risoluzione: stiamo migliorando il firmware HSM per supportare completamente l'attributo CKA_MODIFIABLE. Gli aggiornamenti verranno annunciati nel forum e nella pagina della cronologia delle versioni AWS CloudHSM

Problemi noti per AWS CloudHSM modifica cluster

I seguenti problemi hanno un impatto sui clienti che tentano di utilizzare l'API modify-cluster per modificare il tipo HSM di un cluster.

Argomenti

- Problema: la latenza di accesso aumenta a causa dell'aumento delle iterazioni PBKDF2
- Problema: impossibile modificare il tipo HSM a causa della creazione della chiave token

Problema: la latenza di accesso aumenta a causa dell'aumento delle iterazioni PBKDF2

- Impatto: i cluster con un numero elevato di utenti subiranno un periodo di migrazione prolungato. Ciò è dovuto alle modifiche nel processo di ripristino del backup che esegue le operazioni PBKDF2 per utente durante il ripristino di un backup hsm1.medium su hsm2m.medium per la prima volta.
- Stato della risoluzione: se hai creato una nuova istanza hsm2m.medium o sei migrato a hsm2m.medium da hsm1.medium prima del 20 dicembre 2025, dovrai reimpostare la password per

sfruttare i miglioramenti delle prestazioni che abbiamo implementato per le operazioni di accesso. Fai riferimento alla modifica della password per le istruzioni.

Problema: impossibile modificare il tipo HSM a causa della creazione della chiave token

- Impatto: i clienti che eseguono carichi di lavoro basati su token key non saranno in grado di avviare la migrazione. Ciò avviene perché l'HSM verrà messo in modalità di scrittura limitata per prevenire scenari di perdita di dati durante la modifica del tipo di HSM.
- Soluzione alternativa: interrompi la creazione e l'eliminazione delle chiavi dei token e attendi 7 giorni. In alternativa, contatta l'assistenza se
  - Non è in grado di gestire il blocco delle migrazioni di chiavi token e non può eseguire una blue/green distribuzione.
  - È in grado di gestire il blocco delle operazioni con le chiavi token per la durata della migrazione, ma non può attendere l'intero periodo di 7 giorni.
- Stato della risoluzione: il problema è stato risolto. I clienti che eseguono carichi di lavoro basati su token key possono ora iniziare la migrazione. Le creazioni e le eliminazioni di token key verranno bloccate per tutta la durata della migrazione.

Problemi noti relativi al mancato funzionamento utilizzando AWS CloudHSM versione client 5.12.0 su hsm2.medium

I seguenti problemi hanno un impatto sull' AWS CloudHSM utilizzo della versione client 5.12.0 AWS CloudHSM

Problema: errore durante l'operazione get-attribute

Se stai migrando da hsm1.medium a hsm2m.medium e utilizzi CloudHSM Client SDK 5.12.0, potresti riscontrare errori relativi alla gestione degli attributi.

Potresti visualizzare il seguente messaggio di errore nei log del client: `Error in deserialization of data: Invalid integer conversion`

Impatto: le operazioni seguenti falliranno utilizzando la versione client 5.12.0

- In PKCS #11 SDK, le chiamate a C_ hanno esito negativo GetAttributeValue

- In CloudHSM CLI, il comando key list non mostra alcun attributo nell'output
- In CloudHSM CLI, il key generate-file potrebbe avere esito negativo per le chiavi generate utilizzando hsm1.medium

Risoluzione: consigliamo di eseguire l'aggiornamento alla versione più recente dell'SDK che risolve questo problema.

AWS CloudHSM Errori di sincronizzazione delle chiavi Client SDK 3

In Client SDK 3, se la sincronizzazione lato client fallisce, AWS CloudHSM fa il possibile per eliminare tutte le chiavi indesiderate che potrebbero essere state create (e che ora sono indesiderate). Questo processo prevede la rimozione immediata del materiale delle chiavi indesiderato oppure permette di contrassegnare il materiale indesiderato per la successiva rimozione. In entrambi i casi, non è richiesto alcun intervento per risolvere il problema. Nel raro caso in cui AWS CloudHSM non sia possibile rimuovere e contrassegnare materiale chiave indesiderato, è necessario eliminare il materiale chiave.

Problema: tenti di generare, importare o annullare il wrapping di una chiave token e vengono visualizzati degli errori che indicano la mancata "rimozione definitiva".

```
2018-12-24T18:28:54Z liquidSecurity ERR: print_node_ts_status:
[create_object_min_nodes]Key: 264617 failed to tombstone on node:1
```

Causa: AWS CloudHSM la rimozione e la marcatura del materiale chiave indesiderato non sono riuscite.

Risoluzione: un HSM nel cluster contiene materiale di chiave indesiderato che non è contrassegnato come indesiderato. È necessario rimuovere manualmente il materiale della chiave. Per eliminare manualmente il materiale della chiave indesiderato, utilizza key_mgmt_util (KMU) o un'API della libreria PKCS #11 o del provider JCE. Per ulteriori informazioni, consulta deleteKey o SDK del client.

Per rendere le chiavi token più durevoli, AWS CloudHSM fallisce le operazioni di creazione delle chiavi che non hanno esito positivo sul numero minimo HSMs specificato nelle impostazioni di sincronizzazione lato client. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi in AWS CloudHSM.

AWS CloudHSM Client SDK 3 verifica le prestazioni HSM con lo strumento pkpspeed

Questo argomento descrive come verificare le prestazioni del modulo di sicurezza AWS CloudHSM hardware (HSM) con Client SDK 3.

Per verificare le prestazioni del HSMs AWS CloudHSM cluster, è possibile utilizzare lo strumento pkpspeed (Linux) o pkpspeed_blocking (Windows) incluso in Client SDK 3. Lo strumento pkpspeed viene eseguito in condizioni ideali e chiama direttamente l'HSM per eseguire le operazioni senza passare attraverso un SDK come. PKCS11 Si consiglia di testare il carico dell'applicazione in modo indipendente per determinare le esigenze di scalabilità specifiche. Non è consigliabile eseguire i seguenti test: Random (I), ModExp (R) ed EC point mul (Y).

Per ulteriori informazioni sull'installazione del client in un'istanza EC2 per Linux, consulta Installa e configura il AWS CloudHSM client per CMU (Linux). Per ulteriori informazioni sull'installazione del client in un'istanza di Windows, consulta Installare e configurare il AWS CloudHSM client per CMU (Windows).

Dopo aver installato e configurato il AWS CloudHSM client, esegui il comando seguente per avviarlo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe C:
\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Se il software client è già stato installato, potrebbe essere necessario scaricare e installare la versione più recente per ottenere pkpspeed. Lo strumento pkpspeed è disponibile in `/opt/cloudhsm/bin/pkpspeed` in Linux o in `C:\Program Files\Amazon\CloudHSM\` in Windows.

Per usare pkpspeed eseguire il comando pkpspeed o pkpspeed_blocking.exe, specificando il nome utente e la password di un utente di crittografia (CU) nel modulo HSM. Impostare quindi le opzioni per l'utilizzo tenendo presente le seguenti raccomandazioni.

Consigli sui test

- Per testare le prestazioni delle operazioni di firma e verifica RSA, scegliere il tipo di crittografia `RSA_CRT` in Linux o l'opzione B in Windows. Non scegliere `RSA` (opzione A in Windows). I tipi di crittografia sono equivalenti, ma `RSA_CRT` è ottimizzato per le prestazioni.
- Iniziare con un numero ridotto di thread. Per testare le prestazioni AES, un thread è in genere sufficiente a mostrare le prestazioni massime. Per testare le prestazioni RSA (`RSA_CRT`), tre o quattro thread sono in genere sufficienti.

Opzioni configurabili per lo strumento pkpspeed

- Modalità FIPS: AWS CloudHSM è sempre in modalità FIPS (vedi AWS CloudHSM FAQsper i dettagli). Ciò può essere verificato utilizzando gli strumenti CLI come documentato nella Guida per l' AWS CloudHSM utente ed eseguendo il Ottieni informazioni sull'hardware per ogni HSM in un AWS CloudHSM cluster con CMU comando che indicherà lo stato della modalità FIPS.
- Tipo di test (con o senza blocchi): specifica come vengono eseguite le operazioni con thread. Molto probabilmente si ottengono risultati migliori usando test senza blocchi. in quanto utilizzano thread e concorrenza.
- Numero di thread: il numero di thread con cui eseguire il test.
- Tempo in secondi per eseguire il test (max = 600): lo strumento pkpspeed genera risultati misurati in OPERAZIONI/secondo e riferisce questo valore per ogni secondo di esecuzione del test. Ad esempio, se il test viene eseguito per 5 secondi, l'output potrebbe essere simile ai seguenti valori di esempio:
  - `OPERATIONS/second 821/1`
  - `OPERATIONS/second 833/1`
  - `OPERATIONS/second 845/1`
  - `OPERATIONS/second 835/1`
  - `OPERATIONS/second 837/1`

Test che possono essere eseguiti con lo strumento pkpspeed

- AES GCM: testa la crittografia di tipo AES GCM.
- Basic 3DES CBC: testa la crittografia di tipo 3DES CBC. Vedi la nota 1 di seguito per una modifica imminente.
- AES di base: verifica la crittografia AES. CBC/ECB
- Digest: testa l'hash digest.
- Firma ECDSA: testa la firma ECDSA.
- Verifica ECDSA: testa la verifica ECDSA.
- FIPS Random: testa la generazione di un numero casuale conforme a FIPS (nota: il test può essere utilizzato solo in modalità con blocco).
- HMAC: testa HMAC.
- Random: questo test non è rilevante perché sono in uso HSM FIPS 140-2.

- RSA non-CRT rispetto a RSA_CRT: testa le operazioni di firma e verifica RSA.
- RSA OAEP Enc: testa la crittografia RSA OAEP.
- RSA OAEP Dec: testa la decrittografia RSA OAEP.
- Decrittografia a chiave privata RSA non-CRT: testa la crittografia a chiave privata RSA (non ottimizzata).
- Decrittografia a chiave privata RSA CRT: testa la crittografia a chiave privata RSA (ottimizzata).
- Firma RSA PSS: testa la firma RSA PSS.
- Verifica RSA PSS: testa la verifica RSA PSS.
- Crittografia a chiave pubblica RSA: testa la crittografia a chiave pubblica RSA.

La crittografia a chiave pubblica RSA, la decrittografia privata RSA non-CRT e la decrittografia a chiave privata RSA CRT richiedono inoltre all'utente di rispondere a quanto segue:

```
Do you want to use static key [y/n]
```

Se si inserisce y, una chiave precalcolata viene importata nell'HSM.

Se si inserisce n, viene generata una nuova chiave.

[1] In conformità con le linee guida del NIST, ciò non è consentito per i cluster in modalità FIPS dopo il 2023. Per i cluster in modalità non FIPS, è ancora consentito dopo il 2023. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Esempi

Gli esempi seguenti mostrano le opzioni che è possibile scegliere con pkpspeed (Linux) o pkpspeed_blocking (Windows) per testare le prestazioni del modulo HSM per le operazioni RSA e AES.

Example- Uso di pkpspeed per testare le prestazioni RSA

È possibile eseguire questo esempio in Windows, Linux e in sistemi operativi compatibili.

Linux

Utilizzare queste istruzioni per Linux e sistemi operativi compatibili.

```
/opt/cloudhsm/bin/pkpspeed -s CU user name -p password

SDK Version: 2.03

        Available Ciphers:
                AES_128
                AES_256
                3DES
                RSA  (non-CRT. modulus size can be 2048/3072)
                RSA_CRT (same as RSA)
For RSA, Exponent will be 65537

Current FIPS mode is: 00002
Enter the number of thread [1-10]: 3
Enter the cipher: RSA_CRT
Enter modulus length: 2048
Enter time duration in Secs: 60
Starting non-blocking speed test using data length of 245 bytes...
[Test duration is 60 seconds]

Do you want to use static key[y/n] (Make sure that KEK is available)?n
```

Windows

```
c:\Program Files\Amazon\CloudHSM>pkpspeed_blocking.exe -s CU user name -p password

Please select the test you want to run

RSA non-CRT------------------->A
RSA CRT----------------------->B
Basic 3DES CBC---------------->C
Basic AES--------------------->D
FIPS Random------------------->H
Random------------------------>I
AES GCM ---------------------->K

eXit------------------------->X
B

Running 4 threads for 25 sec

Enter mod size(2048/3072):2048
Do you want to use Token key[y/n]n
```

```
Do you want to use static key[y/n] (Make sure that KEK is available)?  n
OPERATIONS/second                821/1
OPERATIONS/second                833/1
OPERATIONS/second                845/1
OPERATIONS/second                835/1
OPERATIONS/second                837/1
OPERATIONS/second                836/1
OPERATIONS/second                837/1
OPERATIONS/second                849/1
OPERATIONS/second                841/1
OPERATIONS/second                856/1
OPERATIONS/second                841/1
OPERATIONS/second                847/1
OPERATIONS/second                838/1
OPERATIONS/second                843/1
OPERATIONS/second                852/1
OPERATIONS/second                837/
```

Example- Uso di pkpspeed per testare le prestazioni AES

Linux

Utilizzare queste istruzioni per Linux e sistemi operativi compatibili.

```
/opt/cloudhsm/bin/pkpspeed -s <CU user name> -p <password>

SDK Version: 2.03

        Available Ciphers:
                AES_128
                AES_256
                3DES
                RSA  (non-CRT. modulus size can be 2048/3072)
                RSA_CRT (same as RSA)
For RSA, Exponent will be 65537

Current FIPS mode is: 00000002
Enter the number of thread [1-10]: 1
Enter the cipher: AES_256
Enter the data size [1-16200]: 8192
Enter time duration in Secs: 60
Starting non-blocking speed test using data length of 8192 bytes...
```

Windows

```
c:\Program Files\Amazon\CloudHSM>pkpspeed_blocking.exe -s CU user name -p password
login as USER
Initializing Cfm2 library
        SDK Version: 2.03

 Current FIPS mode is: 00000002
Please enter the number of threads [MAX=400] : 1
Please enter the time in seconds to run the test [MAX=600]: 20


Please select the test you want to run

RSA non-CRT------------------->A
RSA CRT----------------------->B
Basic 3DES CBC---------------->C
Basic AES--------------------->D
FIPS Random------------------->H
Random------------------------>I
AES GCM ---------------------->K

eXit------------------------->X
D

Running 1 threads for 20 sec

Enter the key size(128/192/256):256
Enter the size of the packet in bytes[1-16200]:8192
OPERATIONS/second                9/1
OPERATIONS/second                10/1
OPERATIONS/second                11/1
OPERATIONS/second                10/1
OPERATIONS/second                10/1
OPERATIONS/second                10/...
```

AWS CloudHSM L'utente o la policy di Client SDK 5 contiene valori non coerenti

AWS CloudHSM non sincronizza automaticamente gli utenti o le politiche (come le impostazioni mTLS) tra gli HSM di un cluster. La CLI di CloudHSM esegue la sincronizzazione ottimale di queste

operazioni tra gli HSM, ma possono comunque verificarsi incongruenze. Questa pagina descrive come identificare e risolvere le incoerenze sia per gli utenti che per le policy.

Risolvi i valori utente incoerenti

Il `user list` comando in AWS CloudHSM Client SDK 5 restituisce un elenco di tutti gli utenti e le proprietà degli utenti nel cluster. Se una delle proprietà di un utente ha il valore "inconsistent «, questo utente non è sincronizzato nel cluster. Ciò significa che l'utente esiste con proprietà diverse su diversi HSM del cluster. In base a quale proprietà è incoerente, è possibile effettuare diverse azioni di riparazione.

La tabella seguente descrive la procedura per risolvere le incoerenze di un singolo utente. Se un singolo utente presenta varie incoerenze, risolvile seguendo questi passaggi dall'alto verso il basso. Se vari utenti presentano incoerenze, effettua i passaggi per ciascun utente, risolvendo interamente le incoerenze per un utente prima di passare a quello successivo.

> **Note**
>
> Per eseguire la procedura, l'ideale sarebbe effettuare l'accesso come amministratore. Se il tuo account amministratore è incoerente, effettua l'accesso come amministratore e segui la procedura, poi ripeti i passaggi finché tutte le proprietà non saranno coerenti. Una volta che il tuo account amministratore è coerente, puoi continuare a utilizzarlo per sincronizzare altri utenti del cluster.

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
Il "ruolo" dell'utente è "incoerente"	`{ "username": "test_user", "role": "inconsistent ", "locked": "false", "mfa": [],`	Questo utente è un utente CryptoUser su alcuni moduli di protezione hardware e un amministratore su altri moduli di protezione hardware. Questo può accadere se due SDK tentano	1. Effettua l'accesso come amministratore. 2. Elimina l'utente su tutti gli HSM: user delete --username **<user's name>** --role admin

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
	"cluster-coverage": "full"	di creare contemporaneamente lo stesso utente con ruoli diversi. È necessario rimuovere l'utente e ricrearlo con il ruolo desiderato.	user delete --username **<user's name>** --role crypto-user 3. Crea l'utente con il ruolo desiderato: user create --username **<user's name>** --role **<desired role>**

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino

La "cluster-coverage" dell'utente è "incoerente"	{
	"username":
	"test_user",
	
	"role": "crypto-user",
	"locked":
	"false",
	"mfa": [],
	"cluster-coverage":
	"**inconsistent** "
	}
		Questo utente esiste in un sottoinsieme di HSM nel cluster. Questo può accadere se l'operazione user create o l'operazione user delete sono riuscite parzialmente.
		
		È necessario completare l'operazione precedente, creando o rimuovendo l'utente dal cluster.
			Se l'utente non dovrebbe esistere, segui questa procedura:
			
			1. Effettua l'accesso come amministratore.
			2. Eseguire il comando:
			
			user delete --username**<user's name>** --role admin
			3. Ora esegui il comando seguente:
			
			user delete --username**<user's name>** --role crypto-user
			
			Se l'utente dovrebbe esistere, segui questa procedura:
			
			1. Effettua l'accesso come amministratore.
			2. Esegui il comando seguente:

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
			user create --username **<user's name>** --role **<desired role>**

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
Il parametro "bloccato " dell'utente è "incoerente" o "true"	{ "username": "test_user", "role": "crypto-user", "locked": **inconsistent**, "mfa": [], "cluster-coverage": "full" }	Questo utente è bloccato in un sottoinsieme di HSM. Questo può accadere se un utente utilizza la password errata e si connette solo a un sottoinsieme di HSM nel cluster. È necessario modificare le credenziali dell'utente per garantire la coerenza in tutto il cluster.	Se l'utente ha attivato l'autenticazione a più fattori, segui questa procedura: 1. Effettua l'accesso come amministratore. 2. Esegui il comando a seguire per disattivare temporaneamente l'autenticazione a più fattori: user change-mfa token-sign --username **<user's name>** --role **<desired role>** --disable 3. Modifica la password dell'utente in modo che possa effettuare l'accesso a tutti gli HSM: user change-password --username **<user's name>** --role **<desired role>**

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
			Se l'autenticazione a più fattori deve essere attiva per l'utente, segui questa procedura: 1. Chiedi all'utent e di effettuare l'accesso e riattivar e l'autenticazione a più fattori (per questa operazion e l'utente dovrà firmare i token e fornire la propria chiave pubblica in un file PEM): user change-mfa token-sig n --username **<user's name>** --role **<desired role>** —token <File>

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino

Lo stato dell'autenticazione a più fattori è "incoerente"	{
	"username":
	"test_user",
	
	"role": "crypto-user",
	"locked":
	"false",
	"mfa": [
	{
	"strategy":
	"token-sign",
	"status":
	"**inconsistent** "
	}
	],
	"cluster-coverage":
	"full"
	}
		Questo utente ha diversi flag relativi all'autenticazione a più fattori in diversi HSM del cluster.
		
		Questo può accadere se un'operazione relativa all'autenticazione a più fattori viene completata solo in un sottoinsieme di HSM.
		
		È necessario reimpostare la password dell'utente e consentirgli di riattivare l'autenticazione a più fattori.
			Se l'utente ha attivato l'autenticazione a più fattori, segui questa procedura:
			
			1. Effettua l'accesso come amministratore.
			2. Esegui il comando a seguire per disattivare temporaneamente l'autenticazione a più fattori:
			
			user change-mfa token-sign --username **<user's name>** --role **<desired role>** --disable
			
			3. In seguito sarà necessario modificare la password dell'utente in modo che possa effettuare l'accesso a tutti gli HSM:
			
			user change-password --username **<user's name>**

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
			--role **<desired role>** Se l'autenticazione a più fattori deve essere attiva per l'utente, segui questa procedura: 1. Chiedi all'utent e di effettuare l'accesso e riattivar e l'autenticazione a più fattori (per questa operazion e l'utente dovrà firmare i token e fornire la propria chiave pubblica in un file PEM): user change-mfa token-sig n --username **<user's name>** --role **<desired role>** —token <File>

Risolvi i valori delle policy MTL non coerenti

Analogamente agli utenti, le policy MTLS (trust anchors e Enforcement Level) non vengono sincronizzate automaticamente tra gli HSM. La CLI di CloudHSM esegue la sincronizzazione ottimale

quando si eseguono comandi MTLS, ma possono comunque verificarsi incongruenze. È possibile verificare lo stato di sincronizzazione della configurazione mTLS utilizzando i seguenti comandi.

Controlla la sincronizzazione degli MTL (trust & anchor)

Esegui il cluster mtls list-trust-anchors comando per verificare lo stato di sincronizzazione dei tuoi trust anchors. Nell'output, ogni trust anchor ha un campo. `cluster-coverage` Se il valore è "full «, il trust anchor è presente su tutti gli HSM. Se il valore non è «completo», il trust anchor non è sincronizzato tra tutti gli HSM del cluster.

```
{
  "error_code": 0,
  "data": {
    "trust_anchors": [
      {
        "certificate-reference": "0x01",
        "certificate": "<PEM Encoded Certificate>",
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Se un trust anchor ha una copertura del cluster incoerente, esegui nuovamente il comando di registrazione o annullamento della registrazione per completare l'operazione:

- Per completare la registrazione di un trust anchor che non è presente in alcuni moduli di protezione hardware:

  cluster mtls register-trust-anchor --path **<path-to-certificate>**
- Per completare l'annullamento della registrazione di un trust anchor che deve essere rimosso:

  cluster mtls deregister-trust-anchor --certificate-reference **<certificate-reference>**

Per ulteriori informazioni sulla configurazione di MTL, consulta Configurare il TLS reciproco tra client e. AWS CloudHSM

AWS CloudHSM Errori di replica degli utenti di Client SDK 5

Il `user replicate` comando nella CLI CloudHSM replica un utente tra cluster CloudHSM clonati. AWS Questa guida affronta gli errori dovuti a incongruenze degli utenti all'interno del cluster di origine o tra i cluster di origine e di destinazione. User replicate verifica la coerenza degli utenti controllando i seguenti attributi:

- Ruolo utente
- Stato di blocco dell'account
- Stato del quorum
- Stato dell'autenticazione Multi-Factor (MFA)

Problema: l'utente selezionato non è sincronizzato in tutto il cluster

Il processo di replica degli utenti verifica la sincronizzazione degli utenti in tutto il cluster di origine. Se l'attributo di un utente ha il valore «inconsistente», significa che l'utente non è sincronizzato nel cluster. La replica utente non riesce e viene visualizzato il seguente messaggio di errore:

```
{
  "error_code": 1,
  "data": "Specified user is inconsistent across the cluster"
}
```

Per verificare la desincronizzazione degli utenti nel cluster di origine:

- Esegui il `user list` comando nella CLI di CloudHSM.

```
aws-cloudhsm > user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
```

```
        "cluster-coverage": "full"
      },
      {
        "username": "example-inconsistent-user",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "inconsistent"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Risoluzione: sincronizza gli attributi utente in tutto il cluster di origine

- Per sincronizzare le informazioni sugli utenti in tutto il cluster di origine, fare riferimento a quanto segue:. AWS CloudHSM L'utente o la policy di Client SDK 5 contiene valori non coerenti

Problema: Nel cluster di destinazione esiste un utente con attributi diversi

Se un utente esiste già con lo stesso riferimento esiste in uno o più HSMs nel cluster di destinazione ma ha attributi utente diversi, può verificarsi il seguente errore:

```
{
  "error_code": 1,
  "data": "User replicate failed on 1 of 3 connections"
}
```

Risoluzione

1. Determina quale versione dell'utente deve essere conservata.

2. Eliminare l'utente indesiderato nel cluster appropriato eseguendo il `user delete` comando. Per ulteriori informazioni, consulta Eliminare un AWS CloudHSM utente con CloudHSM CLI.
3. Replica l'utente eseguendo il comando. `user replicate`

Problema: la replica utente da hsm2m.medium a hsm1.medium non riesce

La replica utente da hsm2m.medium a hsm1.medium non è supportata. Se si replica un utente da un cluster di origine hsm2m.medium a un cluster di destinazione hsm1.medium, si verificherà il seguente errore:

```
{
  "error_code": 1,
  "data": "User replicate failed on 1 of 1 connections"
}
```

Risoluzione

- Utilizza la gestione degli utenti con CloudHSM CLI per ricreare manualmente gli utenti mancanti.

AWS CloudHSM Errori di replica chiave di Client SDK 5

Il `key replicate` comando nella CLI di CloudHSM replica una chiave da un cluster di origine a un cluster di destinazione. AWS CloudHSM AWS CloudHSM Questa guida risolve i guasti causati da incongruenze all'interno del cluster di origine o tra i cluster di origine e di destinazione.

Problema: la chiave selezionata non è sincronizzata in tutto il cluster

Il processo di replica delle chiavi verifica la sincronizzazione delle chiavi in tutto il cluster di origine. Se le informazioni o gli attributi chiave hanno il valore «incoerente», significa che la chiave non è sincronizzata nel cluster. La replica delle chiavi non riesce e viene visualizzato il seguente messaggio di errore:

```
{
  "error_code": 1,
  "data": "The selected key is not synchronized throughout the cluster"
}
```

Per verificare la desincronizzazione delle chiavi nel cluster di origine:

1. Esegui il `key list` comando nella CLI di CloudHSM.
2. Usa il `--filter` flag per specificare la chiave.
3. Aggiungi il `--verbose` contrassegno per visualizzare l'output completo con le principali informazioni sulla copertura.

```
aws-cloudhsm > key list --filter attr.label=example-desynchronized-key-label --verbose
{
  "error_code": 0,
  "data": {
    "matched_keys": [
      {
        "key-reference": "0x000000000048000f",
        "key-info": {
          "key-owners": [
            {
              "username": "cu1",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
        "key-quorum-values": {
          "manage-key-quorum-value": 0,
          "use-key-quorum-value": 0
        },
          "cluster-coverage": "full"
        },
        "attributes": {
          "key-type": "aes",
          "label": "example-desynchronized-key-label",
          "id": "0x",
          "check-value": "0xbe79db",
          "class": "secret-key",
          "encrypt": false,
          "decrypt": false,
          "token": true,
          "always-sensitive": true,
          "derive": false,
          "destroyable": true,
          "extractable": true,
          "local": true,
          "modifiable": true,
```

```
          "never-extractable": false,
          "private": true,
          "sensitive": true,
          "sign": "inconsistent",
          "trusted": false,
          "unwrap": false,
          "verify": true,
          "wrap": false,
          "wrap-with-trusted": false,
          "key-length-bytes": 16
        }
      }
    ],
    "total_key_count": 1,
    "returned_key_count": 1
  }
}
```

Risoluzione: sincronizza le informazioni e gli attributi chiave in tutto il cluster di origine

Per sincronizzare le informazioni e gli attributi chiave in tutto il cluster di origine:

1. Per attributi chiave non coerenti: utilizzate il `key set-attribute` comando per impostare l'attributo desiderato per la chiave specifica.
2. Per una copertura utente condivisa non coerente: utilizza i `key unshare` comandi `key share` o per modificare la condivisione delle chiavi con gli utenti desiderati.

Problema: una chiave con lo stesso riferimento esiste nel cluster di destinazione con informazioni o attributi diversi

Se nel cluster di destinazione esiste una chiave con lo stesso riferimento ma ha informazioni o attributi diversi, può verificarsi il seguente errore:

```
{
  "error_code": 1,
  "data": "Key replicate failed on 1 of 3 connections"
}
```

Risoluzione

1. Determina quale versione della chiave deve essere conservata.
2. Elimina la versione della chiave indesiderata utilizzando il `key delete` comando nel cluster appropriato.
3. Replica la chiave dal cluster con la versione corretta.

AWS CloudHSM errore rilevato durante il controllo della disponibilità delle chiavi

Problema: un modulo di sicurezza AWS CloudHSM hardware (HSM) restituisce il seguente errore:

```
Key <KEY HANDLE> does not meet the availability requirements - The key must be
 available on at least 2 HSMs before being used.
```

Causa: i controlli di disponibilità delle chiavi cercano chiavi che, in condizioni rare ma possibili, potrebbero andare perse. Questo errore si verifica in genere nei cluster con un solo HSM o nei cluster con due HSMs durante il periodo in cui uno di essi viene sostituito. In queste situazioni, è probabile che l'errore riportato sopra sia stato causato dalle seguenti operazioni del cliente:

- È stata generata una nuova chiave utilizzando un comando come o. La categoria generate-symmetric nella CLI di CloudhSM La categoria di generazione di coppie asimmetriche in CloudHSM CLI
- È stata avviata un'operazione Elenca le chiavi per un utente con CLI CloudhSM.
- È stata avviata una nuova istanza dell'SDK.

> **Note**
>
> OpenSSL crea spesso un fork di nuove istanze dell'SDK.

Risoluzione/consiglio: effettua una delle seguenti azioni per evitare che si verifichi questo errore:

- Utilizza il parametro --disable-key-availability-check per impostare la disponibilità delle chiavi su false nel file di configurazione dello strumento di configurazione. Per ulteriori informazioni, consulta la sezione AWS CloudHSM Parametri di configurazione di Client SDK 5 dello Strumento di configurazione.

- Se utilizzi un cluster con due HSMs, evita di utilizzare le operazioni che hanno causato l'errore, tranne durante il codice di inizializzazione.
- Aumenta la quantità di HSMs dati presenti nel cluster ad almeno tre.

AWS CloudHSM estrazione di chiavi usando JCE

Utilizza le seguenti sezioni per risolvere i problemi relativi all'estrazione AWS CloudHSM delle chiavi con JCE.

getEncoded o getS getPrivateExponent restituisce null

`getEncoded`, `getPrivateExponent` e `getS` restituiranno un valore null perché sono disabilitati per impostazione predefinita. Per abilitarli, fai riferimento alla pagina Estrazione delle chiavi tramite JCE per AWS CloudHSM.

Se `getEncoded`, `getPrivateExponent` e `getS` restituiscono un valore null dopo essere stati abilitati, vuol dire che la chiave non soddisfa i prerequisiti corretti. Per ulteriori informazioni, vedi Estrazione delle chiavi tramite JCE per AWS CloudHSM.

getEncoded getPrivateExponent o GETS restituiscono byte della chiave al di fuori dell'HSM

Tu o qualcuno con accesso al tuo sistema ha abilitato l'estrazione in chiaro delle chiavi. Consulta le pagine seguenti per ulteriori informazioni, comprese le istruzioni su come ripristinare questa configurazione allo stato disabilitato predefinito.

- Estrazione delle chiavi tramite JCE per AWS CloudHSM
- Protezione ed estrazione delle chiavi da un HSM

Limitazione HSM

Quando il carico di lavoro supera la capacità del modulo di sicurezza hardware (HSM) del AWS CloudHSM cluster, riceverai messaggi di errore che indicano che gli HSM sono occupati o limitati. In questo caso, la velocità di trasmissione effettiva potrebbe essere ridotta oppure potrebbe verificarsi un aumento del tasso di richieste di rifiuto da parte degli HSM. Inoltre, gli HSM possono inviare i seguenti errori di congestione.

Per Client SDK 5

- In PKCS11, gli errori di congestione sono mappati in `CKR_FUNCTION_FAILED`. Questo errore può verificarsi per diversi motivi, ma se è la limitazione (della larghezza di banda della rete) HSM a causare questo errore, nel log verranno visualizzate le seguenti righe di log:
  - `[cloudhsm_provider::hsm1::hsm_connection::e2e_encryption::error] Failed to prepare E2E response. Error: Received error response code from Server. Response Code: 187`
  - `[cloudhsm_pkcs11::decryption::aes_gcm] Received error from the server. Error: This operation is already in progress. Internal error code: 0x000000BB`
- In JCE, gli errori di congestione sono mappati in `com.amazonaws.cloudhsm.jce.jni.exception.InternalException: Unexpected error with the Provider: The HSM could not queue the request for processing.`
- Altri errori di congestione degli SDK riportano il seguente messaggio: `Received error response code from Server. Response Code: 187`.

Per Client SDK 3

- In PKCS11, gli errori di congestione sono mappati in errori `CKR_OPERATION_ACTIVE`.
- In JCE, gli errori di congestione sono mappati in `CFM2Exception` con lo stato `0xBB (187)`. Le applicazioni possono utilizzare la funzione `getStatus()` su `CFM2Exception` per verificare quale stato restituisce l'HSM.
- Altri errori di congestione degli SDK riportano il seguente messaggio: `HSM Error: HSM is already busy generating the keys(or random bytes) for another request.`

Risoluzione

È possibile risolvere questi problemi eseguendo una o più delle azioni a seguire:

- Aggiungi i comandi di ripetizione dei tentativi per le operazioni HSM rifiutate a livello dell'applicazione. Prima di abilitare i comandi di ripetizione dei tentativi, assicurati che il cluster sia di dimensioni adeguate per soddisfare i picchi di carico.

> **Note**
>
> Per Client SDK 5.8.0 e versioni successive, i comandi di ripetizione dei tentativi sono attivati per impostazione predefinita. Per i dettagli sulla configurazione del comando Nuovo tentativo di ciascun SDK, consulta la pagina Configurazioni avanzate per lo strumento di configurazione del Client SDK 5.

- Aggiungi altri HSM al cluster seguendo le istruzioni riportate in Scalabilità degli HSM in un AWS CloudHSM cluster.

> **Important**
>
> Ti consigliamo di effettuare test di carico sul cluster per determinare il carico massimo da prevedere, quindi di aggiungere un altro modulo HSM per garantire un'elevata disponibilità.

- Riduci il numero di richieste inviate dall'applicazione. Con la libreria PKCS #11 di Client SDK 5, ogni `C_GetAttributeValue` chiamata invia una richiesta, indipendentemente dal numero di attributi contenuti nel modello. Richiedi tutti gli attributi di cui hai bisogno in un modello e memorizza nella cache i valori nella tua applicazione. Per ulteriori informazioni, consulta Recupera gli attributi con la libreria PKCS #11 per AWS CloudHSM Client SDK 5.

  Con JCE, ogni attributo richiesto costa una richiesta, quindi richiedi solo gli attributi necessari alla tua applicazione.

Connessione persa al AWS CloudHSM cluster

Quando hai configurato il AWS CloudHSM client, hai fornito l'indirizzo IP del primo HSM del cluster. Questo indirizzo IP viene salvato nel file di configurazione del AWS CloudHSM client. Quando il client viene avviato, cerca di connettersi a questo indirizzo IP. Se non ci riesce, ad esempio perché l'HSM ha dato errore oppure lo hai eliminato, potrebbero essere visualizzati errori come il seguente:

```
LIQUIDSECURITY: Daemon socket connection error
```

```
LIQUIDSECURITY: Invalid Operation
```

Per risolvere tali errori, aggiorna il file di configurazione con l'indirizzo IP di un HSM attivo e raggiungibile nel cluster.

Per aggiornare il file di configurazione per il AWS CloudHSM client

1. Utilizzare uno dei seguenti modi per trovare l'indirizzo IP di un HSM attivo nel cluster.
   - Visualizza la HSMsscheda nella pagina dei dettagli del cluster nella AWS CloudHSM console.
   - Usa il AWS Command Line Interface (AWS CLI) per emettere il describe-clusterscomando.

   Sarà necessario disporre di questo indirizzo IP in una fase successiva.
2. Utilizzare il seguente comando per arrestare il client.

   Amazon Linux

   ```
   $ sudo stop cloudhsm-client
   ```

   Amazon Linux 2

   ```
   $ sudo service cloudhsm-client stop
   ```

   CentOS 7

   ```
   $ sudo service cloudhsm-client stop
   ```

   CentOS 8

   ```
   $ sudo service cloudhsm-client stop
   ```

   RHEL 7

   ```
   $ sudo service cloudhsm-client stop
   ```

   RHEL 8

   ```
   $ sudo service cloudhsm-client stop
   ```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client stop
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client stop
```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe stop AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

  Usa Ctrl + C nella finestra di comando in cui hai avviato il AWS CloudHSM client.

3. Utilizzare il comando seguente per aggiornare il file di configurazione del client, fornendo l'indirizzo IP indicato in una fase precedente.

```
$ sudo /opt/cloudhsm/bin/configure -a <IP address>
```

4. Utilizzare il seguente comando per avviare il client .

   Amazon Linux

```
$ sudo start cloudhsm-client
```

   Amazon Linux 2

```
$ sudo service cloudhsm-client start
```

   CentOS 7

```
$ sudo service cloudhsm-client start
```

CentOS 8

```
$ sudo service cloudhsm-client start
```

RHEL 7

```
$ sudo service cloudhsm-client start
```

RHEL 8

```
$ sudo service cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo service cloudhsm-client start
```

Windows

- Per client Windows dalla versione 1.1.2+:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per client Windows 1.1.1 e versioni precedenti:

```
C:\Program Files\Amazon\CloudHSM>start "cloudhsm_client" cloudhsm_client.exe
 C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_client.cfg
```

Log AWS CloudHSM di controllo mancanti CloudWatch

Se hai creato un AWS CloudHSM cluster prima del 20 gennaio 2018, dovrai configurare manualmente un ruolo collegato ai servizi per consentire la consegna dei log di controllo di quel cluster. Per le istruzioni su come abilitare un ruolo legato al servizio su un cluster HSM, consulta

la pagina relative alle informazioni sui ruoli legati al servizio e la pagina sulla creazione di un ruolo legato al servizio nella Guida per l'utente IAM.

IV personalizzati con lunghezza non conforme per l'inserimento delle chiavi AES AWS CloudHSM

Questo argomento sulla risoluzione dei problemi ti aiuta a determinare se l'applicazione genera chiavi con wrapping irrecuperabili. Se riscontri delle ripercussioni per via di questo problema, fai riferimento a questo argomento per risolverlo.

Argomenti

- Determina se il codice genera chiavi con wrapping irrecuperabili
- Azioni da intraprendere se il codice genera chiavi con wrapping irrecuperabili

Determina se il codice genera chiavi con wrapping irrecuperabili

Il problema ha delle ripercussioni solo se riscontri tutte le condizioni seguenti:

Condizione	Come saperlo
L'applicazione utilizza la libreria PKCS #11	La libreria PKCS #11 viene installata come file `libpkcs11.so` nella cartella `/opt/cloudhsm/lib` . In genere, le applicazioni scritte in linguaggio C utilizzano direttamente la libreria PKCS #11, mentre le applicazioni scritte in Java possono utilizzare la libreria indirettamente tramite un livello di astrazione Java. Se utilizzi Windows, il problema NON ti riguarda poiché la libreria PKCS #11 non è attualmente disponibile per Windows.
L'applicazione utilizza nello specifico la versione 3.0.0 della libreria PKCS #11	Se hai ricevuto un'email dal AWS CloudHSM team, probabilmente stai utilizzando la versione 3.0.0 della libreria PKCS #11.

Condizione	Come saperlo
	Per verificare la versione del software sulle istanze dell'applicazione, utilizza questo comando: `rpm -qa | grep ^cloudhsm`
Esegui il wrapping delle chiavi utilizzando il wrapping di chiavi AES	Per wrapping di chiavi AES si intende che l'uso di una chiave AES per eseguire il wrapping di un'altra chiave. Il nome del meccanismo corrispondente è `CKM_AES_KEY_WRAP` . Viene utilizzato con la funzione `C_WrapKey` . Altri meccanismi di wrapping basati su AES che utilizzano vettori di inizializzazione (IV), come `CKM_AES_GCM` e `CKM_CLOUDHSM_AES_GCM` , non sono interessati da questo problema. Scopri di più su funzioni e meccanismi.
Specifichi un IV personalizzato durante una chiamata al wrapping di chiavi AES e la lunghezza dell'IV è inferiore a 8	Il wrapping di chiavi AES viene generalmente inizializzato utilizzando una struttura `CK_MECHANISM` come indicato di seguito: `CK_MECHANISM mech = {CKM_AES_KEY_WRAP, IV_POINTER, IV_LENGTH };` Questo problema ti riguarda solo se: • IV_POINTER non è NULL • IV_LENGTH è inferiore a 8 byte

Se non riscontri tutte le condizioni di cui sopra, puoi smettere di leggere. È possibile rimuovere adeguatamente il wrapping dalle chiavi e questo problema non ha alcuna ripercussione. In caso contrario, consulta the section called "Azioni da intraprendere se il codice genera chiavi con wrapping irrecuperabili".

Azioni da intraprendere se il codice genera chiavi con wrapping irrecuperabili

È necessario eseguire i tre passaggi seguenti:

1. Aggiorna immediatamente la libreria PKCS #11 a una versione più recente
   - Libreria PKCS #11 più recente per Amazon Linux, CentOS 6 e RHEL 6
   - Libreria PKCS #11 più recente per Amazon Linux 2, CentOS 7 e RHEL 7
   - Libreria PKCS #11 più recente per Ubuntu 16.04 LTS
2. Aggiorna il software per utilizzare un IV conforme agli standard

   Si consiglia vivamente di seguire il codice di esempio fornito e di specificare semplicemente un IV NULL, che induce l'HSM a utilizzare l'IV predefinito conforme agli standard. In alternativa, puoi specificare esplicitamente l'IV come `0xA6A6A6A6A6A6A6A6` con una lunghezza IV corrispondente di 8. Non è consigliabile utilizzare nessun altro IV per il wrapping di chiavi AES e gli IV personalizzati per il wrapping di chiavi AES verranno disabilitati in una versione futura della libreria PKCS #11.

   Il codice di esempio per specificare correttamente l'IV viene visualizzato in aes_wrapping.c on. https://github.com/aws-samples/aws-cloudhsm-pkcs11-examples/blob/master/src/wrapping/aes_wrapping.c#L72 GitHub
3. Identifica e recupera le chiavi con wrapping esistenti

   È necessario identificare le chiavi salvate utilizzando la versione 3.0.0 della libreria PKCS #11, quindi contattare l'assistenza per ricevere assistenza () nel recupero di tali chiavi. https://aws.amazon.com/support

> **Important**
>
> Questo problema riguarda solo le chiavi con wrapping con la versione 3.0.0 della libreria PKCS #11. È possibile eseguire il wrapping delle chiavi utilizzando versioni precedenti (2.0.4 e pacchetti di numero inferiore) o versioni successive (3.0.1 e pacchetti di numero superiore) della libreria PKCS #11.

Risoluzione degli errori di creazione dei AWS CloudHSM cluster

Quando si crea un cluster, AWS CloudHSM crea il ruolo collegato al servizio AWSService RoleForCloud HSM, se il ruolo non esiste già. Se AWS CloudHSM non è possibile creare il ruolo collegato al servizio, il tentativo di creare un cluster potrebbe fallire.

Questo argomento spiega come risolvere i problemi più comuni per creare un cluster correttamente. Questo ruolo deve essere creato solo una volta. Dopo aver creato il ruolo legato al servizio nel tuo account, puoi utilizzare uno qualsiasi dei metodi supportati per creare e gestire ulteriori cluster.

Le sezioni che seguono propongono dei suggerimenti per risolvere i problemi di mancata creazione dei cluster correlati al ruolo legato al servizio. Se anche seguendo i suggerimenti non dovessi riuscire a creare un cluster, contatta Supporto. Per ulteriori informazioni sul ruolo collegato al servizio AWSService RoleForCloud HSM, vedere. Service-linked ruoli per AWS CloudHSM

Argomenti

- Aggiungere l'autorizzazione mancante
- Creare manualmente il ruolo legato al servizio
- Utilizza un utente non federato

Aggiungere l'autorizzazione mancante

Per creare un ruolo legato al servizio, l'utente deve disporre dell'autorizzazione `iam:CreateServiceLinkedRole`. Se l'utente IAM che sta creando il cluster non dispone di questa autorizzazione, il processo di creazione del cluster fallisce quando tenta di creare il ruolo collegato al servizio nel tuo account. AWS

Quando la mancanza di un'autorizzazione determina un errore, il messaggio di errore ripoterà il seguente testo.

```
This operation requires that the caller have permission to call
 iam:CreateServiceLinkedRole to create the CloudHSM Service Linked Role.
```

Per risolvere l'errore, assegnare all'utente IAM che sta tentando di creare il cluster l'autorizzazione `AdministratorAccess` o aggiungere l'autorizzazione `iam:CreateServiceLinkedRole` alla policy IAM dell'utente. Per istruzioni, consulta Aggiunta di autorizzazioni a un utente nuovo o esistente.

Quindi tenta nuovamente di creare il cluster.

Creare manualmente il ruolo legato al servizio

Puoi utilizzare la console IAM, la CLI o l'API per creare il ruolo collegato al servizio AWSService RoleForCloud HSM. Per ulteriori informazioni, consulta Creazione di un ruolo collegato ai servizi nella Guida per l'utente IAM.

Utilizza un utente non federato

Gli utenti federati, le cui credenziali hanno origine esterna a AWS, possono eseguire molte delle attività di un utente non federato. Tuttavia, AWS non consente agli utenti di effettuare chiamate API per creare un ruolo legato al servizio da un endpoint federato.

Per risolvere questo problema, crea un utente non federato con l'autorizzazione `iam:CreateServiceLinkedRole` oppure concedi a un utente non federato esistente l'autorizzazione `iam:CreateServiceLinkedRole`. Quindi invita l'utente a creare un cluster da AWS CLI. Questa operazione crea un ruolo collegato al servizio nel tuo account.

Una volta creato il ruolo collegato al servizio, se preferisci, puoi eliminare il cluster creato dall'utente non federato. L'eliminazione del cluster non ha effetti sul ruolo. Successivamente, qualsiasi utente con le autorizzazioni richieste, inclusi gli utenti federati, può creare cluster nel tuo account. AWS CloudHSM

Per verificare che il ruolo sia stato creato, apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/e scegli Ruoli. Altrimenti, utilizza il comando get-role in AWS CLI.

```
$  aws iam get-role --role-name AWSServiceRoleForCloudHSM
 {
     "Role": {
         "Description": "Role for CloudHSM service operations",
         role policy statement
         "RoleId": "AROAJ4I6WN5QVGG5G7CBY",
         "CreateDate": "2017-12-19T20:53:12Z",
         "RoleName": "AWSServiceRoleForCloudHSM",
         "Path": "/aws-service-role/cloudhsm.amazonaws.com/",
         "Arn": "arn:aws:iam::111122223333:role/aws-service-role/
cloudhsm.amazonaws.com/AWSServiceRoleForCloudHSM"
     }
 }
```

Recupero dei log di configurazione AWS CloudHSM del client

AWS CloudHSM offre strumenti per Client SDK 3 e Client SDK 5 per raccogliere informazioni sull'ambiente in uso per consentire a AWS Support di risolvere i problemi.

Argomenti

- AWS CloudHSM Strumento di supporto Client SDK 5
- AWS CloudHSM Strumento di supporto Client SDK 3

AWS CloudHSM Strumento di supporto Client SDK 5

Lo script per AWS CloudHSM Client SDK 5 estrae le seguenti informazioni:

- File di configurazione per il componente Client SDK 5
- File di log disponibili
- Versione attuale del sistema operativo
- Informazioni sul pacchetto

Esecuzione dello strumento informativo per Client SDK 5

Client SDK 5 include uno strumento di supporto client per ciascun componente, ma tutti gli strumenti funzionano allo stesso modo. Esegui lo strumento per creare un file di output con tutte le informazioni raccolte.

Gli strumenti utilizzano una sintassi come questa:

```
[ pkcs11 | dyn | jce | ksp | cli]_info
```

Ad esempio, per raccogliere informazioni per il supporto da un host Linux che esegue la libreria PKCS #11 e fare in modo che il sistema scriva nella directory predefinita, è necessario eseguire questo comando:

```
/opt/cloudhsm/bin/pkcs11_info
```

Lo strumento crea il file di output all'interno della directory `/tmp`.

AWS CloudHSM CLI

Per raccogliere dati di supporto per la AWS CloudHSM CLI su Linux

- Utilizza lo strumento di supporto per raccogliere dati.

```
/opt/cloudhsm/bin/cli_info
```

Per raccogliere dati di supporto per AWS CloudHSM Windows

- Utilizza lo strumento di supporto per raccogliere dati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\cli_info.exe"
```

PKCS #11 library

Come raccogliere dati di supporto per la libreria PKCS #11 su Linux

- Utilizza lo strumento di supporto per raccogliere dati.

```
/opt/cloudhsm/bin/pkcs11_info
```

Come raccogliere dati di supporto per la libreria PKCS #11 su Windows

- Utilizza lo strumento di supporto per raccogliere dati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\pkcs11_info.exe"
```

OpenSSL Dynamic Engine

Come raccogliere dati di supporto per OpenSSL Dynamic Engine su Linux

- Utilizza lo strumento di supporto per raccogliere dati.

```
/opt/cloudhsm/bin/dyn_info
```

JCE provider

Come raccogliere dati di supporto per il provider JCE su Linux

- Utilizza lo strumento di supporto per raccogliere dati.

```
/opt/cloudhsm/bin/jce_info
```

Come raccogliere dati di supporto per il provider JCE su Windows

- Utilizza lo strumento di supporto per raccogliere dati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\jce_info.exe"
```

Key Storage Provider

Per raccogliere dati di supporto per Key Storage Provider su Windows

- Utilizza lo strumento di supporto per raccogliere dati.

```
PS C:\> & "C:\Program Files\Amazon\CloudHSM\bin\ksp_info.exe"
```

Recupero dei log da un ambiente serverless

Per la configurazione per ambienti serverless, come Fargate o Lambda, ti consigliamo di configurare AWS CloudHSM il tipo di registro su. `term` Una volta configurato`term`, l'ambiente serverless sarà in grado di eseguire l'output su. CloudWatch

Per ottenere i log del client CloudWatch, consulta Working with log groups and log stream nella Amazon CloudWatch Logs User Guide.

AWS CloudHSM Strumento di supporto Client SDK 3

Lo script per AWS CloudHSM Client SDK 3 estrae le seguenti informazioni:

- Sistema operativo e la sua versione corrente
- Informazioni sulla configurazione client dai file `cloudhsm_client.cfg`, `cloudhsm_mgmt_util.cfg` e `application.cfg`

- Registri client dalla posizione specifica della piattaforma
- Informazioni su cluster e HSM utilizzando cloudhsm_mgmt_util
- Informazioni su OpenSSL
- Versione corrente del client e della build
- Versione del programma di installazione

Esecuzione dello strumento informativo per Client SDK 3

Lo script crea un file di output con tutte le informazioni raccolte. Lo script crea il file di output all'interno della directory `/tmp`.

Linux: `/opt/cloudhsm/bin/client_info`

Windows: `C:\Program Files\Amazon\CloudHSM\client_info`

> **Warning**
>
> Questo script presenta un problema noto per le versioni di Client SDK 3 da 3.1.0 a 3.3.1. Si consiglia vivamente di eseguire l'aggiornamento alla versione 3.3.2, che include una correzione per questo problema. Per ulteriori informazioni, consulta la pagina Problemi noti prima di utilizzare questo strumento.

AWS CloudHSM quote

Le quote, precedentemente note come limiti, sono i valori assegnati alle risorse. AWS Le quote seguenti si applicano alle AWS CloudHSM risorse per AWS regione e account. AWS La quota predefinita è il valore iniziale applicato da AWS, e questi valori sono elencati nella tabella seguente. Una quota regolabile può essere aumentata al di sopra della quota predefinita.

Service Quotas

Risorsa	Quota predefinita	Modificabile?
Cluster	4	Sì
Soluzioni HSM	6	Sì
HSM per cluster	28	No

Il modo consigliato per richiedere un aumento delle quote consiste nell'aprire la console Quote di servizio. Nella console, scegli il servizio e la quota e invia la richiesta. Per ulteriori informazioni, consulta la pagina relativa alla Documentazione sulle quote di servizio.

Le quote nella tabella Quote di sistema seguente non sono regolabili.

Quote di sistema

Risorsa	Quota per hsm1.medium	Quota per hsm2m.medium
Numero massimo di chiavi per cluster	3.300	16.666 chiavi in totale, con chiavi asimmetriche con un massimo di 3.333
Numero massimo di utenti per cluster	250	1,024
Lunghezza massima del nome utente	31 caratteri	31 caratteri
Lunghezza richiesta della password	Da 8 a 32 caratteri	da 8 a 32 caratteri

Risorsa	Quota per hsm1.medium	Quota per hsm2m.medium
Numero massimo di connessioni client simultanee per cluster [1]	900	900
Numero massimo di sessioni PKCS #11 per applicazione	1,024	1,024
Numero massimo di certificati archiviati per cluster [2]	Non supportata	60
Operazioni di lettura dell'archiviazione dei certificati al secondo per HSM [2]	Non supportata	10
Operazioni di scrittura di archiviazione dei certificati al secondo per HSM [2]	Non supportata	1

[1] Una connessione client per Client SDK 3 è un client daemon. Per Client SDK 5, una connessione client è un'applicazione.

[2] L'archiviazione dei certificati è disponibile solo sui cluster hsm2m.medium. Per ulteriori informazioni, consulta Limiti di archiviazione dei certificati.

Download per AWS CloudHSM Client SDK

I seguenti argomenti forniscono i download per gli SDK AWS CloudHSM client.

> **Note**
>
> Per informazioni sulle piattaforme supportate da ciascun Client SDK, consulta e. AWS CloudHSM Piattaforme supportate da Client SDK 5 AWS CloudHSM Piattaforme supportate da Client SDK 3

Argomenti

- AWS CloudHSM ultima versione di Client SDK
- AWS CloudHSM versioni precedenti di Client SDK
- AWS CloudHSM versioni Client SDK obsolete
- AWS CloudHSM Versioni Client SDK a fine ciclo di vita

AWS CloudHSM ultima versione di Client SDK

A marzo 2021, è AWS CloudHSM stata rilasciata la versione 5.0.0 di Client SDK, che introduce un nuovissimo Client SDK con requisiti, funzionalità e supporto per la piattaforma diversi.

Client SDK 5 è completamente supportato per gli ambienti di produzione e offre gli stessi componenti e lo stesso livello di supporto di Client SDK 3. Per ulteriori informazioni, consulta Confronta AWS CloudHSM Supporto per i componenti Client SDK.

> **Note**
>
> A partire da CloudHSM SDK 5.17, stiamo aggiornando il nostro contratto sul livello di servizio per supportare fino a 3 versioni secondarie precedenti e un anno dalla data di rilascio di una versione SDK. Rilasceremo nuove versioni di CloudHSM SDK per offrire nuove funzionalità e correggere sicurezza, stabilità e prestazioni. Disattiveremo i link per il download per le versioni precedenti e non supportate dell'SDK CloudHSM non appena saranno disponibili nuove versioni.

Questa sezione include l'ultima versione di Client SDK. Per informazioni sulle versioni precedenti, consulta Versioni precedenti dell'SDK. Per le versioni obsolete, consulta Versioni SDK obsolete.

Versione Client SDK 5: versione 5.18.0

Amazon Linux 2023

Scarica il software versione 5.18.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 8d321bd3d74b7365e40107e43e5aa4680f4938513ca1c33f1c739369d54cc466)
- OpenSSL Dynamic Engine (SHA256 checksum 107d522466e83e6f9a9d92d189271180ed60b2bf0d7343e0d46225106a485480)
- Provider OpenSSL (SHA256 checksum 0d6470a0083273ae1d0922ab7b8b240f929762764bd7b0c84d95f87b3c38fca4)
- Provider JCE (SHA256 checksum 2ee2076bea49b970e1b9f6a0edee11dc9cbd139ef11a2687e4e6f54c9cc3ef66)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 0c77a4014d77df64ea3faaf1cc8d8f67dacd8a5ca53baa0026ab0b65d4fe6b69)

Scarica il software versione 5.18.0 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 8206a783676c2dc737872f8d536d2233e76197505b80c5bbff11690cc975d105)
- OpenSSL Dynamic Engine (SHA256 checksum 5eb84a69a9f28218660ec6d879b86964da3402dc27ecaacf205316723346023d)
- Provider OpenSSL (SHA256 checksum a6cafef403e45c19faf6977906ffbf06b4d12b79ddb5fb1f26383c9d1e3dd4ec)
- Provider JCE (SHA256 checksum 8323afdfd2f080dad51c92fe6fd1b802c7db76f6b29949d6c5349472e1377a06)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum a709088dd8857988c893475d85b9a1b3d7a3545006b119156d1e45bebe77021e)

Amazon Linux 2

Scarica il software versione 5.18.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 3d6de0aac5e9f49e96f8aace1e2515c050934f76952ae2a74b3a15952fe9b01e)
- OpenSSL Dynamic Engine (SHA256 checksum c68bbffc1b26926072f0b4e066c03e6f4721d36f4f2ea184efae053dd5088f30)

- Provider JCE (SHA256 checksum a1157f1a66a43c90ba3bd79a1dabbb2435c66f8ae50d096644aecf40a4392205)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum f29d3ad3fb8084c2a6491003907f494876af800b3a925a000ef11702b05be306)

Scarica il software versione 5.18.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 5fb27847ed54c4ec19dbe395eab417b5fe79071f3560a5e0200240bb540542e7)
- OpenSSL Dynamic Engine (SHA256 checksum e93c7ac45d754ba9217ed68cd9f2c09c3581128518119a3edada75c36fedd18a)
- Provider JCE (SHA256 checksum 19741d2bd33e9b540379878baf2917b0fe95fa14a103e3192dad0050028bb032)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 8e9427916230ee4b6357620c310a5463d001342a2955c436e93e314a369fbc26)

RHEL 10 (10.0+)

Scarica il software versione 5.18.0 per RHEL 10 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 56c835581b68fbf6fff052f43fad93c60986c82cc74ee012dcaba25a4153ca06)
- Provider OpenSSL (SHA256 checksum a06765c7e10ac46125a27ae4466464efb28d722decadb90881d6b002a7949954)
- Provider JCE (SHA256 checksum 86b2e64b8d5b354b2ea0e41126f2cccea1be124160621d04e7ee534d2570172b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 5eabb49d6be32f2dc6e9c9a377bd0d6fc2354eb3b51412e0002fa061625ff206)

Scarica il software versione 5.18.0 per RHEL 10 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 96b3ae6cf52b63369fe1420ae4ed95a2bf2f42b7c1543d87894ff6014bece1ea)
- Provider OpenSSL (SHA256 checksum dc6b4e852f8a079ad98fc6af11a177cd8e738ab89bd6ccee0392c25898cf3ef2)
- Provider JCE (SHA256 checksum 6f93577ca3dd2f5ae0980be8811753f294df193d36583f42a92d859c9a6143e2)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum a4dd39dbdf4ffd6e589b485635e535a3d0fbf4cab815378ff7d4fa1576c3de17)

RHEL 9 (9.2+)

Scarica il software versione 5.18.0 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ef2d3a0d427f6d246a69db06a07b520d98721f24fa1180af2a70a6cb90f175b8)
- OpenSSL Dynamic Engine (SHA256 checksum 56e67217c93bcaca96aafede8049a83018740edcaedb72cc7801350be715b28c)
- Provider OpenSSL (SHA256 checksum b4fe26cc8159f77188430238660a64c6cb8ff6db21c7945ce62ba2832d3c3610)
- Provider JCE (SHA256 checksum 481e2cda3c26b5297f91402e80176350feccbb9f1b58001f1f577d1c800340dd)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum d37cabf2373a9935394339f704b6f92bf183fbe8abcb2ab92deb756e0fec71ee)

Scarica il software versione 5.18.0 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum b0d8ec4871887ad40c51e725d510137ed7311eecfc3981060eea74d99fda4ab5)
- OpenSSL Dynamic Engine (SHA256 checksum 475404fd30d008f952e2d07c8fc39aaeb6de200248e3cfa07711a3dbcb2d7d04)
- Provider OpenSSL (SHA256 checksum 2985c6f422a2dc21578cd6f15876c147543776ba7da91db5990621543efc5ff8)
- Provider JCE (SHA256 checksum 45a035f5005459431a68786ea4db0184b71f8e8d5277db8b8ccbe251416b10ca)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 803a867e3df2f2aeb5230f6939db8ffd202991406bce2c626b987465f802fcfb)

RHEL 8 (8.3+)

Scarica il software versione 5.18.0 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 912d5d2a7376180124283309 3472d1365cbcfc3551b1fe93743dce0cbdf8c727)
- OpenSSL Dynamic Engine (SHA256 checksum 70b9aedde526fed0d65114f9b4d8a9576b1ad0efe0bc271c7ef6765345ef66ce)
- Provider JCE (SHA256 checksum ecc9146e3ba72af288834bffe5c4a93922e9ae379f9663fac7d86ffbb36785a8)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)

- CLI CloudHSM (SHA256 checksum 784009c529ff86f357b20c62f1680f35a4710e22d1fd40f9018dab81fa0030c7)

Scarica il software versione 5.18.0 per RHEL 8 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 9ee50fd41904ade515bfab6073fa2d8ba2e1e1c373116420d9125049e3bcdd1f)
- OpenSSL Dynamic Engine (SHA256 checksum 26ac4952f6363ba5a5fcebd45b9709343ed31d05fc5c21240b0e530520295caa)
- Provider JCE (SHA256 checksum ec2c5e4aa8ee0307cf393ad14a890dab5ed904aa4e0de7f7d9ec0806c243a704)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum edbeec5c1f8ac877fbd53f70dbe906bf69839a0493802bf9b1ef1c3c574a07e3)

Ubuntu 26.04 LTS

Scarica il software versione 5.18.0 per Ubuntu 26.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 4b3864459b482f8d3862582b52809315b7653b8b63523f47920c3812643cbab0)
- OpenSSL Dynamic Engine (SHA256 checksum 3e53d5da4ab44384b90bcdc5a4b9b120a630f407df633765414dc3146d9e6e28)
- Provider OpenSSL (SHA256 checksum 4967595c4ea3538f13c647469f9a7ba4583b54bb81ca761bbe1488bcfca78395)
- Provider JCE (SHA256 checksum bc24b988be625826a18353dd0d2c2b1366e28d6d57386865bb8dc6b1c38ed794)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 70b18fad5e3f150695a6e476c13b3e5efa29a5c0c38010522737e0ad36f4cba0)

Scarica il software versione 5.18.0 per Ubuntu 26.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 23d562db2c6fc6a9253a66c32eb11283f7300aada9c4a714516f1dce029a731a)
- OpenSSL Dynamic Engine (SHA256 checksum dfc2777e34d6016c9f39edf97553cd23591b34856bfae56cfead25a4969e9eb7)
- Provider OpenSSL (SHA256 checksum 9f2e8ffbd92334942aea1df515c0f987c88c698acd4a74e03f7a14fe98c682be)
- Provider JCE (SHA256 checksum c85a67409cbbbd38f54b6ac2f2ab8a67335452d42b03628681750b08bf938ebe)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)

- CLI CloudHSM (SHA256 checksum 6fa5d24febe3e3f6c8649f3bd75e2b090660cd82fbedac0e5491a726988f99bc)

Ubuntu 24.04 LTS

Scarica il software versione 5.18.0 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 72ab424e34ee2780ae770679180f456374d8be1b66a82aa441defa603c8ec3a7)
- OpenSSL Dynamic Engine (SHA256 checksum e9ebd0cada051ff21218655f1431171ddd69d219d3cdea13fd9b72ac30fb0a0c)
- Provider JCE (SHA256 checksum 551277453f0bf3a8e78a0e0fec31eab3341dd40adceff3225e606800e0c9c12e)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 28cba68e9721a1683f9a394704bd786b234cb44a8acc10dbe696aaee351a6f8d)

Scarica il software versione 5.18.0 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum c17653c12df1448cba9f1714c6325e35d51df4015557db26b239b82cc9d50db1)
- OpenSSL Dynamic Engine (SHA256 checksum 253c3b8b3e40461889e5101d66f88fa7d21866ee9af1155e014890f6bb795ad1)
- Provider JCE (SHA256 checksum 1e79f34dbc840ec605cb85b4dcd61a01aa2490200e13c98f5021191f4a3bbb41)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 150a3ce22f9a2e7e6ac1b2415b94edc09c7589a64044856409071f0a55360b2d)

Ubuntu 22.04 LTS

Scarica il software versione 5.18.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7c1f6c82edfa7ecbd98ee8e27d4fca205aacdd770ebfe962511214d511beb904)
- OpenSSL Dynamic Engine (SHA256 checksum 014106115154a1d47a44be7dbbeb0483b8c2ed7e676e728c1bb0d73323a94d00)
- Provider JCE (SHA256 checksum 0ff395b8ff58832b2b50f82066e8979329cb95ae4097748632a9b590e88e352a)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 8e5cc97c69bea415a272eae5b85d16566342cb581441a5061042a3998dc5600f)

Scarica il software versione 5.18.0 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 7c5e51bb931f54cd524d5c02487b3c2e0c26d802cf52c1d6dee8f6f1dde5513b)
- OpenSSL Dynamic Engine (SHA256 checksum 7a3a1b696fe1fcb34b3a3c4f4556fa89038d27165a1265458289f9b67694422a)
- Provider JCE (SHA256 checksum ce9740e58424e6e2f2cc98d840755e440339465da323e85049ab7dd2edc4be46)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum 20ec9d5dd380be3f77fec871b45e1e7739b00b9438b95213666d0e3173a99d76)

Windows Server 2025

Scarica il software versione 5.18.0 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7a6486f73ad63312ef4b09b5284185f666d68abf5c5cd23c0a234a071e1d59e6)
- Provider JCE (SHA256 checksum 89dc221ffce8510b7b1365d5655e4b6823341986d4ecb159350e9e1e8a02e823)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum b536e61880080695dc705234eb45abef5f34bc20273ae9cdca25acf9163d86a6)
- Key Storage Provider (KSP) (SHA256 checksum 8dd827b51e471e08094c963b745cd6960d4d398b0ee26930981bafccaf4258a6)

Windows Server 2022

Scarica il software versione 5.18.0 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7a6486f73ad63312ef4b09b5284185f666d68abf5c5cd23c0a234a071e1d59e6)
- Provider JCE (SHA256 checksum 89dc221ffce8510b7b1365d5655e4b6823341986d4ecb159350e9e1e8a02e823)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum b536e61880080695dc705234eb45abef5f34bc20273ae9cdca25acf9163d86a6)
- Key Storage Provider (KSP) (SHA256 checksum 8dd827b51e471e08094c963b745cd6960d4d398b0ee26930981bafccaf4258a6)

Windows Server 2019

Scarica il software versione 5.18.0 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7a6486f73ad63312ef4b09b5284185f666d68abf5c5cd23c0a234a071e1d59e6)
- Provider JCE (SHA256 checksum 89dc221ffce8510b7b1365d5655e4b6823341986d4ecb159350e9e1e8a02e823)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum b536e61880080695dc705234eb45abef5f34bc20273ae9cdca25acf9163d86a6)
- Key Storage Provider (KSP) (SHA256 checksum 8dd827b51e471e08094c963b745cd6960d4d398b0ee26930981bafccaf4258a6)

Windows Server 2016

Scarica il software versione 5.18.0 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7a6486f73ad63312ef4b09b5284185f666d68abf5c5cd23c0a234a071e1d59e6)
- Provider JCE (SHA256 checksum 89dc221ffce8510b7b1365d5655e4b6823341986d4ecb159350e9e1e8a02e823)
  - Javadocs per AWS CloudHSM (SHA256 checksum 741a56da7358118b091ed8cdcecbccc445574688249b1d5cfba4ae5d153d16c2)
- CLI CloudHSM (SHA256 checksum b536e61880080695dc705234eb45abef5f34bc20273ae9cdca25acf9163d86a6)
- Key Storage Provider (KSP) (SHA256 checksum 8dd827b51e471e08094c963b745cd6960d4d398b0ee26930981bafccaf4258a6)

Client SDK 5.18.0 introduce nuove funzionalità e miglioramenti su più componenti, tra cui le firme ML-DSA post-quantistiche e il supporto EdDSA.

Supporto piattaforme

- È stato aggiunto il supporto per Ubuntu 26.04 LTS.

Nuove funzionalità

- È stato aggiunto il supporto per la generazione, la firma e la verifica di chiavi post-quantistiche ML-DSA (Module-Lattice-Based Digital Signature Algorithm, FIPS 204).

- È stato aggiunto il supporto per la firma e la verifica EdDSA (Edwards-curve Digital Signature Algorithm).

CLI CloudHSM

- È stato aggiunto il supporto per ML-DSA la generazione di coppie di chiavi (ML-DSA-44 ML-DSA-65,, e ML-DSA-87), la firma e la verifica. Per ulteriori informazioni, consultare Genera una coppia di ML-DSA chiavi asimmetrica con CloudHSM CLI, Genera una firma con il ML-DSA meccanismo della CLI di CloudHSM e Verifica una firma firmata con il ML-DSA meccanismo nella CLI di CloudHSM.
- È stata aggiunta la firma e la verifica PureEd DSA (Ed25519), estendendo il supporto EdDSA nella CLI di CloudHSM. Per ulteriori informazioni, consulta Genera una firma con il meccanismo PureEd DSA nella CLI di CloudHSM e Verifica una firma firmata con il meccanismo PureEd DSA nella CLI di CloudHSM.

Libreria PKCS #11

- È stato aggiunto il supporto per la generazione, la firma e la verifica delle ML-DSA chiavi e per la firma e la verifica EdDSA (Ed25519 e Ed25519ph). Per ulteriori informazioni, consulta Meccanismi supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5.

Provider JCE

- È stato aggiunto il supporto per la generazione, la firma e la verifica delle ML-DSA chiavi e per EdDSA (Ed25519 e Ed25519ph). Per ulteriori informazioni, consulta Meccanismi supportati per il provider JCE per AWS CloudHSM Client SDK 5.

Provider OpenSSL

- È stato aggiunto il supporto per i tipi di chiave PureEd DSA (Ed25519) per l'offload TLS su cluster non FIPS e i tipi di chiave ML-DSA (ML-DSA-44,, e ML-DSA-87) per l'offload TLS su cluster FIPS e non FIPS. ML-DSA-65 Per ulteriori informazioni, consulta Tipi di chiave supportati per OpenSSL Provider per AWS CloudHSM Client SDK 5 e Meccanismi supportati dal provider OpenSSL.
- Il provider OpenSSL ora supporta le operazioni CLI OpenSSL, tra cui la creazione di richieste di firma dei certificati (CSR) e la firma dei certificati, per tutti i tipi di chiave supportati.

Bug fixes/Improvements

- È stato risolto un problema noto a causa del quale il `key-reference` filtro non riusciva a selezionare le chiavi di sessione (temporanee) nella CLI di CloudHSM e nel provider JCE.
- Migliore gestione della limitazione e comportamento automatico dei tentativi negli SDK del client, insieme a correzioni di bug aggiuntive e miglioramenti della stabilità.

AWS CloudHSM versioni precedenti di Client SDK

Questa sezione elenca le versioni precedenti di Client SDK.

Versione 5.17.2

Amazon Linux 2023

Scarica il software versione 5.17.2 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 61bed6310a97119ef1bd7da177fc8f3fdf3ea688468148c08e186428a500b6b2)
- OpenSSL Dynamic Engine (SHA256 checksum 22e7636ca6ef41cbfaad6c226b58a162ba611a2ed950d29a27fde60f7836cb73)
- Provider OpenSSL (SHA256 checksum 1940e89e71f9099dfc6febf27e9afed9824c9255959c752d6d2c953b35229298)
- Provider JCE (SHA256 checksum 78d744b4f10698603f5ea483641b05a35076c4b7627dee245eed777fa5a205e2)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum b4bce34c0b74d5d8ae2b59c377246cd176f01cd13a3f771bd048a6eeeed2d695)

Scarica il software versione 5.17.2 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 01dda2dbda06216d63447e3119e87e2c729e50326910736fa97d95134a7ccaac)
- OpenSSL Dynamic Engine (SHA256 checksum c0eedc2803f16e0cbbc5c024763b24940860afd70e4c561a40769ab7597847ef)
- Provider OpenSSL (SHA256 checksum b46e600f1c8210283fe77fe66ebc50483590d65c54f38d0d4f8c6caa72a1f02a)
- Provider JCE (SHA256 checksum f3309a72f045e63a61a05b1a1c133099438a06dda6e57e1cb013e90cb383bfaa)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum b35aa1c023d6ea3d1b4699aa6e1a96da998f10b9d2aa201bd02b3e7b906cd6b4)

Amazon Linux 2

Scarica il software versione 5.17.2 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b5bcb6977373d3affa6dab18d05921e665ff8a4c72ff9396711d2817e27080a7)
- OpenSSL Dynamic Engine (SHA256 checksum 603c8c034b4e154c6e022d1a9455d9f0d5dd86f9a25479c7499ed6f680668199)
- Provider JCE (SHA256 checksum 9bcdc6580e4c3c3cf5b762a8b3a5a722693b1b784b487ebe34e625b6615d47d4)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 33af8335abe8eeb00540123cb5a120d3533099b69812038ba02da2e316327c83)

Scarica il software versione 5.17.2 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 1839a2c51dcd7f4b81ffd7e493d2aecfd70276d0392cef32aa421e1a8e9d7819)
- OpenSSL Dynamic Engine (SHA256 checksum bcf34ead09ba23debf2aea7859b4afd650732f8dbbfdddec675b68690ea0d775)
- Provider JCE (SHA256 checksum 5afb78d35db24d7c954aac98b69504b7857d834d3c8db0d27a5bcab834cbe44f)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum b2940f697d8d4341f9f0b9d8619d9e7663d3a3cb24c6fd4aedc167e1c5acf22d)

RHEL 10 (10.0+)

Scarica il software versione 5.17.2 per RHEL 10 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 0a679176689208e74d90db0b9d3096389617ec6fe7dafefcaa7ca96ad885c3d1)
- Provider OpenSSL (SHA256 checksum c497ff802f377ef15d5f6c320e59f122852521847268c9cac667beb0d52bb1d0)
- Provider JCE (SHA256 checksum 5333438d932b16240291c8d0d96dde19d76487c0c741c2e52764b5a2386b0ce4)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 7fd322439bd23b5eddd2881cd6a30ab43e81fef3659fad8245eaba64f8cf83da)

Scarica il software versione 5.17.2 per RHEL 10 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum d724b1093fb6bf28141ef380708fe0422597af8c285dbeac5118f78d58d92d0a)
- Provider OpenSSL (SHA256 checksum 0f95006198bc6b1fbf98b3fd83f73cf03db98387f3fc0fcf5bc3b92afdf3ecff)
- Provider JCE (SHA256 checksum 9ff17247153659b6a3e0e4d14aebdcc3c74ccf6de213e914525b2ae83b6b8279)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 7b6e9351b51e6cdbb4711d55e15a8f8a617ddaf68375df580dc5806b662e95fd)

RHEL 9 (9.2+)

Scarica il software versione 5.17.2 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 1e65fdbd6a413a160fb0d0927b80ff3b106c287ee43485bda7a7344997319cf0)
- OpenSSL Dynamic Engine (SHA256 checksum 059d5e03badda05e57792f863edb931cfd5b25ac7a8c2087760173b90e476284)
- Provider OpenSSL (SHA256 checksum e3f0e2ef1c3330597b90567e339f99a88512cf9b6c750de64b58efe6a58ee3cb)
- Provider JCE (SHA256 checksum 76cdb538108a56b3c03bc451f66364928c5823e9dd7d8e66f08429dd275fca16)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 748132c6a26567b0ce2889e72f327bc7c1c462361f9db992ed2e61490b5339d8)

Scarica il software versione 5.17.2 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 79f5c59fe85f19f16d1863994f19a52fa42f572ff8cb2484421cbf8e231be43c)
- OpenSSL Dynamic Engine (SHA256 checksum d3eaf1fb4d26cb60853930e437d1398d538ce9f4a16b4df9f9af5580f8993123)
- Provider OpenSSL (SHA256 checksum 21f1a0231784e64866b9bf77d2ffa442d1d0d76d86042ba43d38b7154a029b62)
- Provider JCE (SHA256 checksum d4691de51f3ba88563e30bbff8e785cc440d397839891e95655eb7c545dea492)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 80c7511a82dab0bd8f88325bed55e2eedd346f69ab56d4d9b3f68d3ff9b99897)

RHEL 8 (8.3+)

Scarica il software versione 5.17.2 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b3c8a8edfdf7896b011cf0ea323131bd2c1cccaba973abe1b66e5086f9ed571c)
- OpenSSL Dynamic Engine (SHA256 checksum da00cda535bb2959f5927f9671ee33d8c33bbb0e3b1cb092bdf9c001c061d529)
- Provider JCE (SHA256 checksum ede280e95c5635e3410c5a953b751e76af4d24dc392c52ef6126411b30cc6e02)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum dfbf3b26ab1e674943bcb1244f59f93309eb14d2b1f69375385198cd5d946589)

Scarica il software versione 5.17.2 per RHEL 8 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 52b7834d0a383f4dd266eb24606088f2918124b3b131684d249239a8a33ccdef)
- OpenSSL Dynamic Engine (SHA256 checksum 16c145cbb0ecaea05ebb0990c8732f1c75316ca0a77c4d5951fada6e27be17df)
- Provider JCE (SHA256 checksum f2e4e094f9cf7f0b05e3f3607b2613ab713c8893bd8292b116389c2be774b797)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 2594a3c18e0b38dc3239720d4e475dbf7fb91aa360696029477327e2d4242b2c)

Ubuntu 24.04 LTS

Scarica il software versione 5.17.2 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 3569462455b3e1ba6d2297f00615d1da62913376016584681f813d4b6c671ce3)
- OpenSSL Dynamic Engine (SHA256 checksum d805bace899b9d595152c4d44286a797b48b1d47b4a6002beac923d249dd8c8f)
- Provider OpenSSL (SHA256 checksum 5a931ef08394c08681913a3d8daed58ee4b1bb55fdf109a8de366cea19fcc561)
- Provider JCE (SHA256 checksum 8343b7b26ddc2dde6daaf382f1d09c46f4b0633ce17112ff3839c717ad8d6335)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum ef5e6d07c71f8d370207e5ca93425e07d1821bc57153589014542240 1a43752d)

Scarica il software versione 5.17.2 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 2c9ed8f60fe8ea5d98753778f2a92384047e18a113ab0dec03a2f5007c03805a)

- OpenSSL Dynamic Engine (SHA256 checksum fd7eda127d4c9d26d9aa880b4fad93947a9f012aa6f89debed7138f77a0ff6cc)
- Provider OpenSSL (SHA256 checksum 97fecfddbcf348840398fb793a1b84e432a7dd8e300161180b4568ea06c087b8)
- Provider JCE (SHA256 checksum 069111cc8757e63ac80d0e20715751291f27effb47938f185d5c21ca60c01892)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 89833ec8e0fa0bcadc92f2784003fa030d26db8aa338f2e8a384e168dfa31a63)

Ubuntu 22.04 LTS

Scarica il software versione 5.17.2 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 05e48c9625ed6ea5b86527058dee6282034a36865017991fbd114df23bfea759)
- OpenSSL Dynamic Engine (SHA256 checksum 63f28d02069b312aeb917b8182c9c6e618ab752a4ea86449411d85bfa5c29d17)
- Provider JCE (SHA256 checksum 7fd8bb608d4d08c1f40d4e2b816fe405d50b931840d8d185988059608bb617bd)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 9873c6f73832ac49010376544620ac29862d71bbf23df8d12dcb642fe805d1df)

Scarica il software versione 5.17.2 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 4d48f1185fad4a50611ac28ce35e0e1741c3392700f40f63d483c424ec05cd3f)
- OpenSSL Dynamic Engine (SHA256 checksum 1d72a33854c5a88662490bb3ff6794bf238662d42869523a7eb4b31ddcdb97d5)
- Provider JCE (SHA256 checksum 681875ab46c8d738781df35970c99bbc064a1a7155bfd15d4048e53eea8e3e8f)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum 2b9acb97054a55fb64b791b09b738bd1c5d433a8a1724f0677f704f8b8df888d)

Windows Server 2025

Scarica il software versione 5.17.2 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e08552078e12792ddda780fb2bb3ac98c9f60b3f1b3fec410d627114cc1dbb24)

- Provider JCE (SHA256 checksum 323fc72e3df00f2ec65d17c04ca51acec4b3097c39624e5012019f9af07b09b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum cc6a183e6c00f0713ab64d1fbfd01d4828ffba086f407025b5b7140c42012d1a)
- Key Storage Provider (KSP) (SHA256 checksum 5165f44f10eb8be3e99ce89d11206cec1b006d61df2004a022ddf97afe2c98c3)

Windows Server 2022

Scarica il software versione 5.17.2 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e08552078e12792ddda780fb2bb3ac98c9f60b3f1b3fec410d627114cc1dbb24)
- Provider JCE (SHA256 checksum 323fc72e3df00f2ec65d17c04ca51acec4b3097c39624e5012019f9af07b09b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum cc6a183e6c00f0713ab64d1fbfd01d4828ffba086f407025b5b7140c42012d1a)
- Key Storage Provider (KSP) (SHA256 checksum 5165f44f10eb8be3e99ce89d11206cec1b006d61df2004a022ddf97afe2c98c3)

Windows Server 2019

Scarica il software versione 5.17.2 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e08552078e12792ddda780fb2bb3ac98c9f60b3f1b3fec410d627114cc1dbb24)
- Provider JCE (SHA256 checksum 323fc72e3df00f2ec65d17c04ca51acec4b3097c39624e5012019f9af07b09b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum cc6a183e6c00f0713ab64d1fbfd01d4828ffba086f407025b5b7140c42012d1a)
- Key Storage Provider (KSP) (SHA256 checksum 5165f44f10eb8be3e99ce89d11206cec1b006d61df2004a022ddf97afe2c98c3)

Windows Server 2016

Scarica il software versione 5.17.2 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e08552078e12792ddda780fb2bb3ac98c9f60b3f1b3fec410d627114cc1dbb24)

- Provider JCE (SHA256 checksum 323fc72e3df00f2ec65d17c04ca51acec4b3097c39624e5012019f9af07b09b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum ddfa01e827443b56362ec32344d605f67755e332b66184a1b0b3b4242a23f087)
- CLI CloudHSM (SHA256 checksum cc6a183e6c00f0713ab64d1fbfd01d4828ffba086f407025b5b7140c42012d1a)
- Key Storage Provider (KSP) (SHA256 checksum 5165f44f10eb8be3e99ce89d11206cec1b006d61df2004a022ddf97afe2c98c3)

Client SDK 5.17.2 include correzioni di bug e miglioramenti.

Versione 5.17.1

Amazon Linux 2023

Scarica il software versione 5.17.1 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e5d37660312c8d5b9c36a9b74555222a755276e2c1f360418c34125b2b715b97)
- OpenSSL Dynamic Engine (SHA256 checksum 9b84d756e56f19b830adbd46aa5ac5096ae28944e34dd8dac3bd534e487d7aee)
- Provider OpenSSL (SHA256 checksum 5df0647ea5fe6da453bbf9a9578de9674ba93b722ec17dfe79825006211dc7a9)
- Provider JCE (SHA256 checksum ac820e95618bf6cecf76b6ae0c5fb59198041f1d583178105a506d255d886fc9)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 3b2f966c05690c14c325ea37cf9799bad477b2573f1077a5319d92933af9e590)

Scarica il software versione 5.17.1 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 8cf4f7e9bed86ed15198e0870075fa01cb8243d3646ae9a1417ee3eba0cb3111)
- OpenSSL Dynamic Engine (SHA256 checksum 831c7565caaea18c9c239b060347ec1b0e5121ec94d5bfe4cecda0fd4b1aa9ab)
- Provider OpenSSL (SHA256 checksum 73e91db43c98fa1eac6ce4d39f211d0dafa8445941b9f3adf77876cbeb443561)
- Provider JCE (SHA256 checksum 1cc6bd9226ac98ebf898498f3b51b41b3ed58304f0290bd2045ab472e3bf3e14)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum d716108dfc8f0e89e082603dbd5c95379b9f593628ab30bc611c37c169fae8d0)

Amazon Linux 2

Scarica il software versione 5.17.1 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum c22875d9a81b2a5814d949b06d1467d4fbb111e67a6b29d83e961d95bdebd9e9)
- OpenSSL Dynamic Engine (SHA256 checksum f54d264ede8624867c3010d6b3979c51cd05a34d90a4169208400013c8de76c5)
- Provider JCE (SHA256 checksum c46297760a1d306541498e6a0295d7143273286dbb798c1c65bee38799b68e6b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 87162212c93d8536d9247632221d855e841637a786c405eb47bc1410d62987d9)

Scarica il software versione 5.17.1 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 0abe258732376b20730c2d2e089458e83eff4e751df8e60c0c70d3d92e75d7a4)
- OpenSSL Dynamic Engine (SHA256 checksum 4f8101debd678327a26246e50df10fe79663fc264fefd49deef63e036221fc0b)
- Provider JCE (SHA256 checksum dfc83b8f2dbee97eab98bff431c3374181a0051884abe7e9424621fcdba9c8c5)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum bc43b221610d1c220ed402ee14bb0275ad79fb21a342639d1266579881da953a)

RHEL 10 (10.0+)

Scarica il software versione 5.17.1 per RHEL 10 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 11076479568454ad1083bb6a81953a73f9175f6d8626e247e1e16439a5c105ec)
- Provider OpenSSL (SHA256 checksum 6c3d7a58b38ef39c7b5310958eb684feed20edf6731603a8b727b4bd978f465f)
- Provider JCE (SHA256 checksum 921883948344a06716d6b3b30f67708dedff550978efce6d5817b36c83e83b35)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 4d69481cc4e725e65811e031fe344cb178016cd268759bb910acdb14df9be413)

Scarica il software versione 5.17.1 per RHEL 10 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 7a20af9fbe7a9874132c1314841ef2ead0daef92ced3ba2289938a88d0ee929e)
- Provider OpenSSL (SHA256 checksum e762bb21086131725e9ca13b7d25e6fcb474888ae49317e1b045ddf826a8338c)
- Provider JCE (SHA256 checksum c8bb96ded1abdfb54cd710fade3c91ce6030f5dc9ddeea639cc0a0cac5c26bb1)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum d48ccaeccc9d4247f0cf4c81a6f092704630933dbc89e0b0ba9a75629fd863b7)

RHEL 9 (9.2+)

Scarica il software versione 5.17.1 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum df4f1de8e68e363b351770dd74782d204f04d4db646c554fa96182b429006618)
- OpenSSL Dynamic Engine (SHA256 checksum 06d2e65adb076e65b81b51bcea34b64f718b6aa29c173eebcaf277de2c8d03ec)
- Provider OpenSSL (SHA256 checksum 802e608484ab748eccdce8da0854dc668e90578d4c2bc6ede4caf4121d10171f)
- Provider JCE (SHA256 checksum 1a4fdfe82cc79a4de85a19d6645916b6b388c919a71a22efae7bafaba195db0f)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum daab5b758c61f49b34b6f7e288debb121adc3d63b9652db086855d160c988570)

Scarica il software versione 5.17.1 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 2fe6549339b743200c6be91b8abf8017c3f67c90f7db25ffd0ab1915b91d0f78)
- OpenSSL Dynamic Engine (SHA256 checksum a0a964dcb817251faafcdd432e6260dff1d6e1bcdd7e7ffbd54a691cbe366b58)
- Provider OpenSSL (SHA256 checksum bbcc9b3a2455328e6e7b11ff64b7e7653757960a74bfd9d6b3cc998b2a955fb2)
- Provider JCE (SHA256 checksum f9c74e2ededb9d2fdb7076789223c4cde7a0845b6f999b006fa741e9428195c7)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 0c4c1c7f4e06ba3b5b5dfe619abb0fa9d4729c87c668361fdb99b2a62f4b00fc)

RHEL 8 (8.3+)

Scarica il software versione 5.17.1 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 0f8465ab44f7388980b22bc1f8c6c3c3e09e809ba2712170c85db86918817803)
- OpenSSL Dynamic Engine (SHA256 checksum 8fe4f9fb6088d2a32e210696775d395b918ee867dede8b2152ff37d0d99cc8f3)
- Provider JCE (SHA256 checksum 948d37f435e7b03ec8409ee48f89c265009c7e7a76e7a18921c5ef88f3504391)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 2168377badf708047ee14d2bc57b9234f01ce2c9f42568dbd1185723a63eeeb8)

Scarica il software versione 5.17.1 per RHEL 8 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 69db8b832edba06fe055ce9772bfad645a86e5e530d20e8f14738fc7cb06a5c7)
- OpenSSL Dynamic Engine (SHA256 checksum ac7ad7b11e63d1fb7af57ff1e61b421aa0fccbca99ed15b0ddec1602fea2acd3)
- Provider JCE (SHA256 checksum 7520a95076a7a794219677489d2c764502a6dcfeea484f10059e283f1aee6975)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 596acc3164695a262b15edbb92eda89ffd4b8d06b91617ce91058c2d99ae2a53)

Ubuntu 24.04 LTS

Scarica il software versione 5.17.1 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum bc3146d5dc7aabe5359c41d2bd0fd97ed4bc2f389be6a4b94423065af9d60bd0)
- OpenSSL Dynamic Engine (SHA256 checksum 05d6afa19949b69af4325f55d285a89255ab2c7487aba898c74cbd4ee8e30fdf)
- Provider OpenSSL (SHA256 checksum 3deba48c356478fdbb0236ad435cee8f7f8cfe3cec1a894aea3f79d80d398cce)
- Provider JCE (SHA256 checksum ebacb95d8f92528e236a5342cb5c1ecbf74c228428fb4a080c0c972b697292b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 575f54610d26561bb98b69548de2f61a17b5b93ed49be13821be536d76ffdb75)

Scarica il software versione 5.17.1 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum b0c918575e611f486902ebbcc3ee84e15217247f5577b06764cfc620660c6ec4)

- OpenSSL Dynamic Engine (SHA256 checksum 57f0afb66c34a91e96bef4964dd3fb487fe5b9b06d0818ee9a2c90ad60068321)
- Provider OpenSSL (SHA256 checksum dcf5128c6913e43e0a304d0221c243fd5228f235b78fac5ffd43478bb0d96062)
- Provider JCE (SHA256 checksum 21d43992ace3aec9ec3711e7bd64ec716d3281f627e78a6f7a328a8e533beec4)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 096844ef80a7d573843682cfb3797a0c45608d004639ab0e671065f3964c06e3)

Ubuntu 22.04 LTS

Scarica il software versione 5.17.1 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ec8c275832f9936e698106454e930268d7f2f310faffa2b7d03c3c03c6e297e1)
- OpenSSL Dynamic Engine (SHA256 checksum 316c3929363ad1d764f356bbf56dff121b3bccefa5060b94c5d9d7955f828ff4)
- Provider JCE (SHA256 checksum 4c53b61bcdada0a684b605c0630f52b6b3bf6e9ed64f4c623f35fa4283e9770d)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 6b4ba5d3d06e5234b64b7c60310d5b477c2d8732fbbd3e9c60dcb56f2ef69a6e)

Scarica il software versione 5.17.1 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum d26d10425572f0624d2205e1409b88ccc5d793bcfeef6cd26fdcb5c41f5c8129)
- OpenSSL Dynamic Engine (SHA256 checksum e3b6cd3e93f2645d9a23add1a514e314ad3d48a93b253e6b984577b8e502e1f0)
- Provider JCE (SHA256 checksum 7ab3ea04d34f92c585121b110f620b632616a5585a4ed153fbb61fedef667b06)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum 9cf4a05ebf2bce4a0c9e57dd2eef87ba47ac05c4a303d4465cbf41536736c685)

Windows Server 2025

Scarica il software versione 5.17.1 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 145ea74ff3ab70c0f5f1ccd8ced90b86f0b4f3ee029d2d9452e9b0724ae2f64e)

- Provider JCE (SHA256 checksum 074bc0a990d51debea16e8691171c1cfc4e164795eb839be1d311858b157b0a6)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum ddc11eee7b884ddaa203ded85f36553a9a636d35bf9dd4b6557c23a167ed7c4a)
- Key Storage Provider (KSP) (SHA256 checksum 5bef4732d23861795a8463c3a3511e0a235a29972d584c2f65384e87ba4bb48c)

Windows Server 2022

Scarica il software versione 5.17.1 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 145ea74ff3ab70c0f5f1ccd8ced90b86f0b4f3ee029d2d9452e9b0724ae2f64e)
- Provider JCE (SHA256 checksum 074bc0a990d51debea16e8691171c1cfc4e164795eb839be1d311858b157b0a6)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum ddc11eee7b884ddaa203ded85f36553a9a636d35bf9dd4b6557c23a167ed7c4a)
- Key Storage Provider (KSP) (SHA256 checksum 5bef4732d23861795a8463c3a3511e0a235a29972d584c2f65384e87ba4bb48c)

Windows Server 2019

Scarica il software versione 5.17.1 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 145ea74ff3ab70c0f5f1ccd8ced90b86f0b4f3ee029d2d9452e9b0724ae2f64e)
- Provider JCE (SHA256 checksum 074bc0a990d51debea16e8691171c1cfc4e164795eb839be1d311858b157b0a6)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum ddc11eee7b884ddaa203ded85f36553a9a636d35bf9dd4b6557c23a167ed7c4a)
- Key Storage Provider (KSP) (SHA256 checksum 5bef4732d23861795a8463c3a3511e0a235a29972d584c2f65384e87ba4bb48c)

Windows Server 2016

Scarica il software versione 5.17.1 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 145ea74ff3ab70c0f5f1ccd8ced90b86f0b4f3ee029d2d9452e9b0724ae2f64e)
- Provider JCE (SHA256 checksum 074bc0a990d51debea16e8691171c1cfc4e164795eb839be1d311858b157b0a6)
  - Javadocs per AWS CloudHSM (SHA256 checksum 89b5c2232c1739d61813d16f6d3177984bfffb66754367883e2d7e997a363f30)
- CLI CloudHSM (SHA256 checksum ddc11eee7b884ddaa203ded85f36553a9a636d35bf9dd4b6557c23a167ed7c4a)
- Key Storage Provider (KSP) (SHA256 checksum 5bef4732d23861795a8463c3a3511e0a235a29972d584c2f65384e87ba4bb48c)

Client SDK 5.17.1 include correzioni di bug e miglioramenti.

JCE

- Client SDK 5.17.1 è l'ultima versione a supportare OpenJDK 11.

Versione 5.17.0

Amazon Linux 2023

Scarica il software versione 5.17.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e0656d606c1f75b8ecf11df79f2d4aa17c0b68983bb20d271dabe02f7b97ca47)
- OpenSSL Dynamic Engine (SHA256 checksum b8917abe48799035018bc044fbe2e9db8d5568ebf29663ecc12d53ec2be5efd2)
- Provider OpenSSL (SHA256 checksum 61e49cb0f1f9eb7bcf76f5ef36f01cf1fadfa56bf5eab94b05efdd93214c9cf7)
- Provider JCE (SHA256 checksum 4ba2548fa27f3662c56fc140386366cfb40cde899695412c6d4731a7d0a70e43)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 0a9de06b3f15d38f3aa02dd0288d3a9ad41f525f692f539664168b9e6b95e18d)

Scarica il software versione 5.17.0 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 9bd40e13bf777a4651721ebe5613cf2e4f0f339679d50f06b81f7f1b73c1fef5)
- OpenSSL Dynamic Engine (SHA256 checksum e592419bc0449664f0707281a46ebb775091386ac5b297e35ff0e4160468cbe0)
- Provider OpenSSL (SHA256 checksum 46358ef68388928702d780564811814aa826c143ccc0624a2a45d8995226d33e)

- Provider JCE (SHA256 checksum db5bd1494c742a64ca3f2bc57253b17655ad38ecc27d45422965466c1caf740f)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum e3d21b4460a421058f65a597b0bb30a9fdda890363ebee107b124a1212d5a042)

Amazon Linux 2

Scarica il software versione 5.17.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 17d96d5f4ad25de74fadcc7272fac825d43c496b0b7912e4d32b1ccfb67250bc)
- OpenSSL Dynamic Engine (SHA256 checksum 3d3de61bd16ef849e53827709b8c8bd9eaf89bd1b7d56b1a02455bca38fe1eb6)
- Provider JCE (SHA256 checksum 5c268313ebf16cf762619a1e75f1d44efdbd6eaf5ba4bbce67ecfe96bbed9775)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 43b678eff0225590cc59c02645332f4459c8161a6742e562eb95902d1bec9134)

Scarica il software versione 5.17.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 7910ec55f8b106d804290b77e6d13cefb85ef79cbc728f69bb21eebeb7245c6b)
- OpenSSL Dynamic Engine (SHA256 checksum 399c16e0fbfa372710a83836517cfeaae0895513931427a458e7994ce91a321d)
- Provider JCE (SHA256 checksum 1678581e31fcfde58625ca1e3c0762c4b96095065a868a5b7f0872214e5ab9e8)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 52257b580d73d902e9fd29f2c334f88cabee484c7eec8af28f7b2aa7864466e7)

RHEL 10 (10.0+)

Scarica il software versione 5.17.0 per RHEL 10 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 037102ee2b7f9c244d647cfb06da282b99349e208d27b7b0aa80bd32e382a2d4)
- Provider OpenSSL (SHA256 checksum 3d6d97df03a003bae579dabbf9abeefc1bd2fa6d799a6db01c3ef98d02145393)
- Provider JCE (SHA256 checksum 4e69ec89acfefc19f922c68988d4674718d49d5cda2e0a288f7f0343ebfc270c)

    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 38a67191949cc8cb0c69ef2fe4694514b2ca6af4cf5f8763bd7f6860c501675a)

Scarica il software versione 5.17.0 per RHEL 10 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum bc383aff236d4b2a0fdbc5fd81907cf9212ad8f3cfc91fadeb6cdc14b9d3d581)
- Provider OpenSSL (SHA256 checksum d9ead84f8d2e6a09044646d62fed293ca082431e5e0338bcbbfcf8acd023ecc0)
- Provider JCE (SHA256 checksum c61fdb5f05cf486619deb3e23ecb171ce4485bb64c891bf4480fbe6a4276c3bc)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum fbd4387ef3fab177de8c888d98d1b24bc042d23ea2f93834dfb5ee918518f406)

RHEL 9 (9.2+)

Scarica il software versione 5.17.0 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7fc68025493fc4a0bc4d3b2fd11b17da4481b6bee51fc86a40f95fe4c206aa60)
- OpenSSL Dynamic Engine (SHA256 checksum 3e786e9218f657c93e1bf1b54c81f0f573e39d10147cf99485da73677b0d67ae)
- Provider OpenSSL (SHA256 checksum c52b366f9c6c960d697794bd7840888bc51e1ee5aa654b8d2c83ec1626797c56)
- Provider JCE (SHA256 checksum 427e9c68815188b59a0f9b2fd14793791d9a4ce4d3e5ddc34fce916af18a45f1)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 49d128639eea0e588c95281635a33b0e62837cf5a9d992e0bacb5b598a0037ed)

Scarica il software versione 5.17.0 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 3ae8f5068abb3d9ca75ac21b160acc7ee84c7ded6107a889602448af51d5bd82)
- OpenSSL Dynamic Engine (SHA256 checksum 64c5feaf86fc40e28616156f17942cc46d8ac93babdd8af59dd98a1f2e73117d)
- Provider OpenSSL (SHA256 checksum 24df4296ba0ae15f6b7060ccffeca0d1a24f58d9e8da8eb6a4afa3a6a50770fd)
- Provider JCE (SHA256 checksum 648d7948ab8a1271d7cf0eddc2ca32e960979be481b99a1ce2ce8b9fa3a2b2ed)

    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum edd1f3707aeea285a55cf24d2e2385489458a69e5d7dab4014783f5cdc232d77)

RHEL 8 (8.3+)

Scarica il software versione 5.17.0 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 85e070ed3273ead9698711dfc2c3e590dda1a01f99074943459ddc5ac052872d)
- OpenSSL Dynamic Engine (SHA256 checksum d0c8e5603f4c99d0ba547c0ba40361c15977f3349ac120d514afed59abf32c83)
- Provider JCE (SHA256 checksum 4567b9a236d0dfd0410863beaced6d5d4e71acb06b2b136c7c40ce6e704f3698)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum f8a090752849fac9bf99ef67b59f571e476908dccec1285b466c25e6cc71793c)

Scarica il software versione 5.17.0 per RHEL 8 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 71ade413435947eb2093727cb0e7a5c5f084ca859cb38d728a8c3d23f3b49bd0)
- OpenSSL Dynamic Engine (SHA256 checksum 02210da06b3fd22813fce8b89b753d150c8f321df405c40dd9f0172dc03968a7)
- Provider JCE (SHA256 checksum 80ac5dbaad1f644d6b9551e6a1d95d4b4a51ae82d34e45593be77b1068f7c910)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum c3c56c3b9fed1dc51761d9412007c727d3b608377a089bd637a44bc374744d5c)

Ubuntu 24.04 LTS

Scarica il software versione 5.17.0 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b3869a6fe9f2031cb5f36ac8d2a5a5280451159e7a68f0d6f91f98b5f0100939)
- OpenSSL Dynamic Engine (SHA256 checksum 3e46443a3f8425f62cec8d8ca414c4b5446b86b42012877ebe44a17303403413)
- Provider OpenSSL (SHA256 checksum 69c2e61894e954ac1d0a785025902af5aa1922a7f6011a847f2111e153db706e)

- Provider JCE (SHA256 checksum 663da1a7b13a1055bb90fd7b168b69fcfb000eb91c1c3ce6062d8cc8db607fe4)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 0d57b4f08bd768ed595978d2a6d3fe6bbbf8398a784c66c4e30c19ecf6730b5d)

Scarica il software versione 5.17.0 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 94d9a36a3d3f2f04c10539713b98ddad68654c739572064e4c123005a9caf575)
- OpenSSL Dynamic Engine (SHA256 checksum a442950055a7051f50b4cdfa7ec347c1b207b0bbf7e159b31a3bf96896b66146)
- Provider OpenSSL (SHA256 checksum dcc4b5a370da176a86e47683a78703a5604d6b7b49ba687377460f48c3ddff26)
- Provider JCE (SHA256 checksum 1fe1fa59a3e5c32edd5668604e43f7ce152473514b25ee736dfb0e9cf87c92c9)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 2bf2c13bf1520eaf2a45eb55aaba3334fa54da686d22d1a03005ec8f62ea8895)

Ubuntu 22.04 LTS

Scarica il software versione 5.17.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 2dc1f00952f69b59b0498e85c2f2009f80c9a2f9f28104995b0291da5f45d805)
- OpenSSL Dynamic Engine (SHA256 checksum df0e7ff8852a93fb4c5807317d3438ab96ffe69f19ba144d872991cbcb4172f6)
- Provider JCE (SHA256 checksum 77a3cc37e066ae0ad5655ddc3dd38a0513bdcbd3c9d3d780e2973ac42a75035e)
  - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 88ca1614178771991649b8117ddf27ffb99c6842ee89f68e77c6df3470e6ba78)

Scarica il software versione 5.17.0 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 0e1f62783c16debf9188c9cf0085fe972671e5ea9bf0fc2fc3255f908885949c)
- OpenSSL Dynamic Engine (SHA256 checksum 109d25583e2d5ed0fe821a77098d8f2c815011f7873168e0110aeb857dc903ab)
- Provider JCE (SHA256 checksum 9151661a39f6394aa3ace682bd7964095ed5300244582d3f6b1f81255ebf22cf)

    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 66e408cea800a93becf378b8611595ad8cd1bf51e0ae7feebc548fe3099b477c)

Windows Server 2025

Scarica il software versione 5.17.0 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- Provider JCE (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- Key Storage Provider (KSP) (SHA256 checksum 3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2022

Scarica il software versione 5.17.0 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- Provider JCE (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- Key Storage Provider (KSP) (SHA256 checksum 3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2019

Scarica il software versione 5.17.0 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- Provider JCE (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)

    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- Key Storage Provider (KSP) (SHA256 checksum 3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Windows Server 2016

Scarica il software versione 5.17.0 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 77dd4ad5da0d4a21c82ff2ea03c09a6b8c2005e27b6aba6724e4bec75606b0e7)
- Provider JCE (SHA256 checksum c2c4766b67f02a93cca22b166e63fe6f4c71bfd9f7b45aa2da6a603d825e5c1a)
    - Javadocs per AWS CloudHSM (SHA256 checksum a89e6d7e4e1f4049fefa64c765e97928fc54c366249f315ac6a839704e1ffba8)
- CLI CloudHSM (SHA256 checksum 4feb0aeb7554ddd919eb7df301a4da537fffb0577a89a053fba802a2d3657a2a)
- Key Storage Provider (KSP) (SHA256 checksum 3766d4d4ab1bd7afa6b72e74bc80d8e5054481c8535f5a5c841cc320617c4e1e)

Client SDK 5.17.0 introduce nuove funzionalità e miglioramenti su più componenti.

Supporto piattaforme

- È stato aggiunto il supporto RHEL 8 per l'architettura ARM64.

JCE

- È stato aggiunto il supporto per OpenJDK 25 su tutte le piattaforme
- Client SDK 5.17.0 è l'ultima versione a supportare OpenJDK 8.

Nuove funzionalità

- È stato introdotto il nuovo OpenSSL Provider SDK per una maggiore compatibilità con OpenSSL 3.2+ e operazioni crittografiche aggiuntive.
- È stato aggiunto il supporto per la curva Ed25519 alla generazione di chiavi EC.

- È stato aggiunto il supporto per la firma e la verifica EdDSA (Edwards-curve Digital Signature Algorithm) nella CLI di CloudHSM.

CLI CloudHSM

- È stato aggiunto il supporto per la curva Ed25519 alla generazione di chiavi EC per tipi hsm2m.medium non FIPS.
- È stato aggiunto il supporto per la firma e la verifica EdDSA (Edwards-curve Digital Signature Algorithm) nella CLI di CloudHSM. Per ulteriori informazioni, consulta Genera una firma con il meccanismo HashEd DSA nella CLI di CloudHSM e Verifica una firma firmata con il meccanismo HashEd DSA nella CLI di CloudhSM.

Provider OpenSSL

- Il nuovo OpenSSL Provider SDK fornisce l'integrazione con l'architettura OpenSSL 3.2+ Provider per i tipi hsm2m.medium.

Bug fixes/Improvements

- Meccanismi di riprova migliorati per le operazioni crittografiche.

Versione 5.16.2

Amazon Linux 2023

Scarica il software versione 5.16.2 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e223649cbf689afaeca445b962cceaab3c143b901ee7974a7a5360b6424c409b)
- OpenSSL Dynamic Engine (SHA256 checksum 9b51e8b1429f22705399299c111385e9434873a1dd26dda2c70498646590148c)
- Provider JCE (SHA256 checksum 45ec10c2e3d592fb1fe56cc9dee1d8653e3a6dfa15b8185618fd3cf66c567314)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 8134a0e316286310eda1606b6a14e1c9fd6f0e27934a43e55e24d3094ef870a6)

Scarica il software versione 5.16.2 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 7556c69294d0c67b359cf9995ec7047cdc90e21542d2c9d5e110e35f13c329c5)
- OpenSSL Dynamic Engine (SHA256 checksum 6ab4f3b9731a6f26891658f5e3a0ff74255096390c42aa7f702ba07e0bd924b1)
- Provider JCE (SHA256 checksum 98b80696762a73f3f9fbe6c3c7af0f8f9db23ba63d66d4d941690b6698019702)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 649397606f527f330049cf7adb5ad667126dfd7f63f878f1f46b469e2f127814)

Amazon Linux 2

Scarica il software versione 5.16.2 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 073ee736ad9c7de6e6451afdecc4d77e6ed1a63f314a7fc09373ff35986fa0fc)
- OpenSSL Dynamic Engine (SHA256 checksum b174fb24a15f1e22dd520ea9ac4ac14229c2adce14ac6df8901f9d3c0e7e24f6)
- Provider JCE (SHA256 checksum f18e42f2679ab0fb51c421965a5dd6c8fe1361442d92403d1d230bedb003b27a)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 5a6c83833b60017dd494c31fd4abf3896f57177be7951674c85d0d4e909d2041)

Scarica il software versione 5.16.2 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum df2c97a27855916983f242eba05e0741335158432ac46ebbeae47c4a3b7b0bef)
- OpenSSL Dynamic Engine (SHA256 checksum d25456e3da39c63d9c0d4a928815f995239b146f4f0b26a8c50c742905543207)
- Provider JCE (SHA256 checksum 7e31734efbaae49203e57d7ad4d1e291faa5b7e16c469a3596da82a7380bceeb)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum b77eb1f9ff58ccc864854d7f261e331ce50836cd770efc8e8eaebb33d78c23e2)

RHEL 10 (10.0+)

Scarica il software versione 5.16.2 per RHEL 10 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 8044cb40c9c65ae944900c740308e68eeb0156d3b93051d60463cd3f6095ef67)

- Provider JCE (SHA256 checksum a1b7f110c2b15ba7ce9adee74baf1f108d85146269321991314f9153a074ca27)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 3ded29fdb74537b284cf871af91dd64fce8483770141a3e4470e94fb69524621)

Scarica il software versione 5.16.2 per RHEL 10 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 643fa23ffea47dfb21fd326cacf99e0e45f44d767ca6198ab06023239eda5d2c)
- Provider JCE (SHA256 checksum e6b5c2760c60f5c82a248d36f3b47cea9b1bbb743cf1c02a1c13554b70ab87f8)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum a6af281ff9f429f964ac9d2babc251b4423748d2b2e6da22fc51b9fd5c59714c)

RHEL 9 (9.2+)

Scarica il software versione 5.16.2 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e0a44849c9cb562cf803ff182e54a35e7eee682efa2a99fdfac51a996822d126)
- OpenSSL Dynamic Engine (SHA256 checksum 97463a8a88239d4a81854f2a4da7e0371dbcb375a662c7eae7753c371ef3c39f)
- Provider JCE (SHA256 checksum 10cd4b9826451b7e9a11390cabcc7df548f5b3197e8bc87f5efe04605e503dc1)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 6c7fc59dcce4d7bc0c4460ae7c793f13fcbb306924942bc6e1457f729602f76f)

Scarica il software versione 5.16.2 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 7b8d3f5e2bd09f30ddc1599619c63aa4452b16e1037bf10af070c055ae48cbd7)
- OpenSSL Dynamic Engine (SHA256 checksum 4edb0315c050b3b0699cb244fe36b987c9fd2921999cf7519fb669354d2efe2d)
- Provider JCE (SHA256 checksum eae2064d9523df9f819ef4c7950a0ad15ef8bb01d8dcec062ee2558f5bebf029)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum f9f4073ebcde2b654b706efb1c0f99c08fe24903576e760029097b0e3c783f6a)

RHEL 8 (8.3+)

Scarica il software versione 5.16.2 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 9a3e023fd5ac8c444ef07bb28b96e81b467e066c9dc2da19f648f38eb04d05c9)
- OpenSSL Dynamic Engine (SHA256 checksum 6fc6ca58f16fe2f486f4730cd0a6029eeeebae98b3aed41fd8c605fe5c36c7a0)
- Provider JCE (SHA256 checksum cb6e9eccfeebca7377b1fb815c2ae306b39b271513ce73a6cd5f1dccb55eac4c)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum e848f3f2221e7f6b984bd2001331f9e90f20e6d7351484e4e2f12431c00d87e5)

Ubuntu 24.04 LTS

Scarica il software versione 5.16.2 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum d993f8c84d15cf93432b73cb37693c4ab68b3e765c35742adef3f947744b9893)
- OpenSSL Dynamic Engine (SHA256 checksum 055f0e4cc74ec7ead77fff3970422b4f1729e26a2cf3cc2be530f0e6a06f887e)
- Provider JCE (SHA256 checksum 5bd62f8f5cb3ff4ae9430020aa4f5c8a29cb635674d2400d989f512c370c755c)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum de263f6035cf7280a982434448aa079164050e4a09508816fd06bf000cf15f0c)

Scarica il software versione 5.16.2 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum a6496f28f82f3e92b0acc8e3ae1fbe230a6d3ed82586e8ca6fe6d2f82c275e7c)
- OpenSSL Dynamic Engine (SHA256 checksum 82275d6ac2628b4291162e3609d6788b4cf01d6488a9b6a84ed8787a5a96e873)
- Provider JCE (SHA256 checksum 67800957c69efca0fe01623364cb3b152f2780d15624955130d507a3f591e8e5)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 088f53d75250aa0ce3db7a7ba3f06ad7804b3dbeed6226683a3252890a867ac6)

Ubuntu 22.04 LTS

Scarica il software versione 5.16.2 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b9799ba75fb8885782283abae7f49aa76a72d27f692654dd3a415157f0ea9043)
- OpenSSL Dynamic Engine (SHA256 checksum 3c7675e9b7fa4bb692865559f09fc1f9fd78dadb97d8bd6b5d1fc03367d75b86)
- Provider JCE (SHA256 checksum 37c21821ad4d9fde9551112e8a6d7ec6b7bb62a540c0c08b6edd650d5c87b5cf)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 7d3a724892e993aebfb06ad5c3d1ef8fc650d65dff0a7de3cb9c17c089badc58)

Scarica il software versione 5.16.2 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 8a2b392e48b7f9021966f6c457444ddd0964f9bb85b7ecb988bc0d372bbef6c9)
- OpenSSL Dynamic Engine (SHA256 checksum cca0736d208d59d070ced8b82181e24919f21fbf0497e5a8a2f40436e06a8dd9)
- Provider JCE (SHA256 checksum c12e1b9bfe1cc78ecf35892050dd06904c6a29736cc628ac1481dca4ec3e3bfe)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum 586ee7663d54e76652e9f073772fcbda218dc3c6432a221d664ffbabee3590cd)

Windows Server 2025

Scarica il software versione 5.16.2 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- Provider JCE (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- Key Storage Provider (KSP) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2022

Scarica il software versione 5.16.2 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- Provider JCE (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- Key Storage Provider (KSP) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2019

Scarica il software versione 5.16.2 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- Provider JCE (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- Key Storage Provider (KSP) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Windows Server 2016

Scarica il software versione 5.16.2 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b5f55bc72fc994244f09e0252c7fc4a3ddf9607b20860c8718ff1d8a3db0a177)
- Provider JCE (SHA256 checksum c28ea8164f277bf067f4630d433ab7090b370225116a2d594dd563dfb5f9f991)
  - Javadocs per AWS CloudHSM (SHA256 checksum 17029e68946b83d47c8512f824599338e7315a99317e4fc33a2058cdc5512d0d)
- CLI CloudHSM (SHA256 checksum db708fc33bdc91b9ad554aa63de86a572d3ccff4ba3ab10feac9c5fd9ed79b94)
- Key Storage Provider (KSP) (SHA256 checksum c060be59673f325240bd0f1ccc9d9d97639eb7937fa323d71e223451bc0d2e19)

Client SDK 5.16.2 aggiunge il supporto per la piattaforma RHEL 10 e rimuove il supporto per Ubuntu 20.04 LTS. Client SDK 5.16.2 aggiunge anche il supporto per JCE Provider e KSP con nuove funzionalità, fornendo correzioni di bug e miglioramenti rispetto ad altri SDK.

Supporto piattaforme

- È stato aggiunto il supporto RHEL 10 per le architetture x86 e ARM.
  - OpenSSL Engine non è supportato in RHEL 10 come parte della transizione di Red Hat alla nuova architettura introdotta in OpenSSL 3.0. Provider-based

CLI CloudHSM

- È stato corretto un bug per cui la firma con dati prehash non cancellava i token in modalità interattiva.
- È stato corretto un bug `quorum token-sign list` che in precedenza causava errori quando erano presenti tre o più token.

Provider JCE

- È stato aggiunto il supporto per la ricerca di chiavi Elliptic Curve (EC) utilizzando ECParameters ed ecPoint per i tipi hsm2m.medium.
- È stato aggiunto il supporto dei parametri per ECDH con KDF. `sharedInfo` X9.63

Libreria PKCS #11

- È stato aggiunto il supporto per la ricerca di chiavi Elliptic Curve (EC) utilizzando ECParameters ed ecPoint per i tipi hsm2m.medium.

Bug fixes/Improvements

- Funzionalità di elenco dei token migliorata per visualizzare più token in un'unica operazione.
- È stato risolto il problema di cancellazione dei token di utilizzo delle chiavi nella modalità interattiva CLI di CloudHSM durante la firma con dati prehashati.
- È stato risolto un problema noto a causa del quale alcune operazioni HSM limitate su hsm2m.medium non venivano ritentate automaticamente. Fare riferimento a Problema: le operazioni possono fallire durante la creazione del backup per ulteriori dettagli.

Versione 5.16.1

Amazon Linux 2023

Scarica il software versione 5.16.1 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 342e81846436708cfc1fb459a7ef1c7b065d8a68b50a5e63653c24918d0338a0)
- OpenSSL Dynamic Engine (SHA256 checksum 4e83bebcd20201c04629a03b70136df8d225d3056346789054c4e2ce8c9b3cba)
- Provider JCE (SHA256 checksum 014e3c804a56087f8855ed30e480f9c193def0e1132a4769eae3ecd235d76c4f)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 854c5c21f9fbef4f53a710340f213c272365c2bb1233106266eca5a5aa547e11)

Scarica il software versione 5.16.1 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum dbe9498d91e1e2e8f80fc870a4a6576b0dc801dc20cb7a82bfded79b3a362ac0)
- OpenSSL Dynamic Engine (SHA256 checksum 76cff941f36275163db146e05bf5fe64440248643ceb5dfcb3b64103f4f016a7)
- Provider JCE (SHA256 checksum 8c38082797172b5630c74a8fc0c9e2652f0898bd2232b72911fbbaaa0260b5a6)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 2a0640e32405a7db138afb2444a542eab11c0a7d9610d5406478c7ac4602a14b)

Amazon Linux 2

Scarica il software versione 5.16.1 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum cfa59458ec239553c86ac55773bebc69a5dc7cdc08a3927917fd5918e10abe93)
- OpenSSL Dynamic Engine (SHA256 checksum 49e50bece7d73f3b7dc95eed1df856a6dee40e27f24f40f015c6a4a2e8dee839)
- Provider JCE (SHA256 checksum 85c8853860aa36a6c54e75c94d607a334d64dd34683ba70430ff5be61eedcc56)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 0efc3783e8429331aa446e45776fef46c7f9726a4768763ba4881dc2b05e090e)

Scarica il software versione 5.16.1 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum dad60a0380ef0ea9c469cae4de10dc47124bbecedafb6956d009734cd49abceb)
- OpenSSL Dynamic Engine (SHA256 checksum 7ee303421d94544cbe9df03022c48af327833e631d8f1ec59a466673b6e9395d)
- Provider JCE (SHA256 checksum 5ebe8157abb042ee92a7edfbfcb98bf79fc3a9907684565176fd28c387c08e88)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 5e76b4a9021a3c92e59fc608c8263af731013835738977f376fb8ad9189add56)

RHEL 9 (9.2+)

Scarica il software versione 5.16.1 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 696e58f74d89bd8e39f85bddc547f8c94fa2fbca498318a7bb94f22e8be7a668)
- OpenSSL Dynamic Engine (SHA256 checksum 86197cd8bdd70db91331bb8380ea094352b4087c95a04768d2cefc3bba18dffa)
- Provider JCE (SHA256 checksum 37ccc2df3e8aaddda74dc060f7f5bbe63e4769311f2ccbf313a8ab5d8831f206)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 45a6db22db9ae64c239b49ea9281d29d2b4364dd63ee2408f463de17c4da1877)

Scarica il software versione 5.16.1 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum fd79fbccf504f53a6e70497135fcea4e20982fca75b628124c7510dcbaddf56c)
- OpenSSL Dynamic Engine (SHA256 checksum 23fe7407be9c5f8da0ae64e560a9741887bc31936f0b88c8d1490e3c6893a8a8)
- Provider JCE (SHA256 checksum bd1c43e3e28c2f71e78a41342c064ff873fd8e052f89f90227fe00eaa28f089e)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 3e877a232303052b8a6b2b869dcf228edbb0da913d3d41393f622831f4455a27)

RHEL 8 (8.3+)

Scarica il software versione 5.16.1 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum d396c53c229b1eaa7cb30d4fcd17addc9170c7942795d1c82b23a157ec379686)
- OpenSSL Dynamic Engine (SHA256 checksum c01f2cb66a6c5be839906b25c6a7f7990507b1b8ec3da34c128c1a90838e48df)
- Provider JCE (SHA256 checksum 1745ab78de0712111659e91301f80e771e1b7ebc2ff8c4cc5e37bce591058520)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum b3ca40ba66062856ef63848c71f6e9dfa0a46a2b18d44c2d96a798fc4a4fb9cf)

Ubuntu 24.04 LTS

Scarica il software versione 5.16.1 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ade113ee72547cb6a8fe91a4f3ac403f462d0acd0a306f01d16ac90699f9b59c)
- OpenSSL Dynamic Engine (SHA256 checksum 9e00cb32b16fbb286a853f65c5c5154c3e805ad67820d829a0c5343f12cc9e00)
- Provider JCE (SHA256 checksum 4fab322a138e14372aba394de29f444af640479dd338539be84e0cf659c8993b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum f8c1e369885eb77ba4517388dab47c067a617e860fb67a028a1ebde0f96acaef)

Scarica il software versione 5.16.1 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 5e15a69f8fb1429ed5c10429c91323fe82a63c0fe7bcc2f968820733d1737449)
- OpenSSL Dynamic Engine (SHA256 checksum f5136ee61bd34b74d59a01d37964b70a0f80ec5981f732b265af1c5466309e6f)
- Provider JCE (SHA256 checksum acf5bebe7009461cdc1e31b95caadcbf80da09ec78d8f69142072da9baf61b79)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 511376d2faf9f991f2193d08f80fc1edd53bf06daaa6fd8bd7a3fba0d6563ad1)

Ubuntu 22.04 LTS

Scarica il software versione 5.16.1 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e1d787f10ee51d94732732811e4d2110f1b8e448e67fd47df8b53a2f7e56e3c8)

- OpenSSL Dynamic Engine (SHA256 checksum 29a9c41379754ce098a025feadc026f2f75a8638981f5b95ed07007d5b3d8510)
- Provider JCE (SHA256 checksum 390d21e499ba181dc2774c3140433f6e60b49d9de3a892a89038c27e1aff157b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum cadadafaa9464f944e5c2446df92fb1777a9e1e5116cab1b359a8aa101cf7ae8)

Scarica il software versione 5.16.1 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 3b6a3d50b4a55c150f12308ee3914451897bc4cc4c8420accf1d2d706316fcce)
- OpenSSL Dynamic Engine (SHA256 checksum c366a3c5faf4de32a6dedc4613234ba3d331b0abac3241bed7d25e0109a44f64)
- Provider JCE (SHA256 checksum 705416b47bc62781b5a6dfc78dd02dadb4f18f842b569c3e210b9535813b9e70)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 811b501615cf34665b70f103905094ca84ed7c126f72d91b040aacda99dbf22b)

Ubuntu 20.04 LTS

Scarica il software versione 5.16.1 per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6a2c33c78ada33fb435bbf8939b7cde3efe968e9f03250083dc6024b7ebd45b8)
- OpenSSL Dynamic Engine (SHA256 checksum 6ee178454e78d88e0ab92cd7c5b056d0c04cd6de192aea731d7ebcbd4c5ed761)
- Provider JCE (SHA256 checksum 1e61ccd8d2b37c64467051d26e8d8bd592465ecef04282d6f4a9491707ba059d)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum 645693f2ede4fa0d7879eedaa41ae23120d902f24c9a637ba088f277e703cb96)

Windows Server 2025

Scarica il software versione 5.16.1 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- Provider JCE (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)

  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- Key Storage Provider (KSP) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2022

Scarica il software versione 5.16.1 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- Provider JCE (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- Key Storage Provider (KSP) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2019

Scarica il software versione 5.16.1 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)
- Provider JCE (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- Key Storage Provider (KSP) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Windows Server 2016

Scarica il software versione 5.16.1 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 25beadaf28323412c60ed6695d89b6261d34da9cccf08869923b979854aa0329)

- Provider JCE (SHA256 checksum 65f487f22c0786b80d81b387f5f4c8a6c9395c8d31a020c91fc0105829b8ec2c)
  - Javadocs per AWS CloudHSM (SHA256 checksum 70320f9aee48a250f9384f626fc16db00ea35c32af4c29cd63e9c82065a0fe1b)
- CLI CloudHSM (SHA256 checksum d4dc9f5efd2b5e2eb07a59fb237f7f8dfd5476c859cdb9f91841354ce88783c4)
- Key Storage Provider (KSP) (SHA256 checksum 80f70eb3ba22d34e49b5b5da3fa183c86c751bff46229ae16fb83a5fc69a4d0c)

Client SDK 5.16.1 aggiunge il supporto per la firma e la verifica dei dati prehashati nella CLI di CloudHSM.

Supporto piattaforme

- SDK 5.16.1 è l'ultima versione a fornire il supporto per la piattaforma Ubuntu 20.04 LTS. Per ulteriori informazioni, consulta il sito Web di Ubuntu. https://ubuntu.com/blog/ubuntu-20-04-lts-end-of-life-standard-support-is-coming-to-an-end-heres-how-to-prepare

CLI CloudHSM

- È stato aggiunto il supporto per la firma e la verifica dei dati prehashati nella CLI di CloudHSM. Per ulteriori informazioni, consulta La categoria delle firme crittografiche in CloudHSM CLI e La categoria di verifica crittografica in CloudHSM CLI.

Provider JCE

- Modalità di AES/CBC/Pkcs5Padding crittografia aggiornata per generare automaticamente un vettore di inizializzazione (IV) casuale quando viene fornito null IV. In precedenza, null IV comportava un errore operativo. Gli IV espliciti rimangono obbligatori per le operazioni di decrittografia.

Bug fixes/Improvements

- Latenza ridotta per le ripetute operazioni di aggiornamento dei digest in JCE.
- È stato aggiornato il comando `generate key-reference` per denominare correttamente i file di riferimento delle chiavi KSP in base ai relativi valori di ID di attributo durante l'esecuzione su Windows Server. Per ulteriori informazioni, consulta Generazione di riferimenti chiave KSP (Windows).

Versione 5.16.0

Amazon Linux 2023

Scarica il software versione 5.16.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum d63271304f32f49838390a58d94a2140ae8a744ac64efcf0e6a65983c858d862)
- OpenSSL Dynamic Engine (SHA256 checksum 25d78c1df82355601ed6887bb47d64a06380d001da15a070cfa89dce65417fb6)
- Provider JCE (SHA256 checksum fcd876813e5d114e2b042765ff98b256410e988a4e373ea1aae6f260aebdd554)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 7f1441eb08daa7ceab91161900d89974ee86e0a3c54d878a29e43db775bd04f5)

Scarica il software versione 5.16.0 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 0b42e4e89cddda10c5be5f295a52d7a541bba057a3a07d3d9b192cf9eb49c776)
- OpenSSL Dynamic Engine (SHA256 checksum e8ee949d44b9f9ba64ed36eb7944acacbd1d76d7cf09d5492deae5928a9953e9)
- Provider JCE (SHA256 checksum 67a6fd8f7b5dfc60ec6d6d380c31035675373b542205ec7e5a687175191cd275)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 1e6c3053b19aded54045a3baa9fe2545f81a35332366cf2d77dca92ee888e654)

Amazon Linux 2

Scarica il software versione 5.16.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 5a89ff9801f89f51e27e70869f2713cfe9bbd87d1198246cffaafe74aff9e809)
- OpenSSL Dynamic Engine (SHA256 checksum 991b6288289d07972915aa0be6dfd6b3c33c1d6312bc304225715e30832e688f)
- Provider JCE (SHA256 checksum c5e673f79c2efd83195a288a5c8860c4bf74dba5bc8e422741dd5cc1be230fb3)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum b9021b0b348c08f433e7cce247ab1710e2a79f33a82ac36e1d6a4672171028db)

Scarica il software versione 5.16.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 97677a7f9759637b548205d31c9a08a3e7b27dfa5a38e9a4e8e2398ae9869a5a)
- OpenSSL Dynamic Engine (SHA256 checksum dde369dc4efd9f59e5c9a5459a532ad5d273f87839fba6dd5dcde2c3a2b61517)
- Provider JCE (SHA256 checksum b7d96e7453e26ff07ccd9437b102f4c21857c79daa6315054728fc7ac66150bc)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum c7386ac016b14ea8cb96a6ed655a67f46826cdb925bd87902277b5ce9b593d51)

RHEL 9 (9.2+)

Scarica il software versione 5.16.0 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 178d3e1ff0ec8cdc7f8c8be9aab772b4195602bb1c48dea692b628689be5ea3d)
- OpenSSL Dynamic Engine (SHA256 checksum 0488ad66cd825fafee70d1a1ccb1c8045b089af7b7d6ccadbec75f535376048c)
- Provider JCE (SHA256 checksum cc9a60fa981a46c674de1e715d92ec1b8cd5c801394bfb3f0d101880589d4202)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum cb7e3ce6704f015d5b9462f12e6ad5489366fb3e80f554a2bdf7f04e688be1a9)

Scarica il software versione 5.16.0 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 9567fb016b641821869919d4d6e9754ae08fd549381446640ff135a1e225b809)
- OpenSSL Dynamic Engine (SHA256 checksum 8c46a9dd908e77a0d8412e7050621d0c570e091845480815c230e26d1917227b)
- Provider JCE (SHA256 checksum 78a7c714aedd068e5e2f12f5044a6ce1618e6dfacbcdd91b4a9b505842dcc6c9)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum d2340ef4092ca6c1fa1ac8b484550508a5d9d8670e663bef6e595a07f8151be3)

RHEL 8 (8.3+)

Scarica il software versione 5.16.0 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 47bf628a4a2663b8e8c7207b43e5af1d05ed2247e85e1371773596c132c9a4ec)
- OpenSSL Dynamic Engine (SHA256 checksum 49dd44243863121f7781e0f4ac556828c746d85db723864ccf16ea5fb042846a)
- Provider JCE (SHA256 checksum c5e8a6664e6a8e5338a98b3845bed029f5936b517e6043aeb65ad58c3364eed8)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 71f2cd6bd8892ccaea845bfe575cdf20016dd7176e5216b76290e499c822a2f2)

Ubuntu 24.04 LTS

Scarica il software versione 5.16.0 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum fd7e78050aba017b7dc859f0b1e6f354f56b22091cb262b328aad204064a2960)
- OpenSSL Dynamic Engine (SHA256 checksum 9d2a9e8049caa12c8027ad2963fa30f22e1032d4025e39a4e2d5985073c283f5)
- Provider JCE (SHA256 checksum ae19c13f55e01538e989e8693aaf3a5d998c60f1ed71e721a34404a228947c20)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 7ebedb3e8c4b8e01cc1e5dfe32ae18c9542d9c2940c77bd71ee102093aaf1d1c)

Scarica il software versione 5.16.0 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum cfd9526cf495e81a2477b464ca3c124a89d07b1a366a14b39eeb27d6b5e2404b)
- OpenSSL Dynamic Engine (SHA256 checksum 02a5cc2cd9442b7bf982bb3ae44c9b134758b46e68e46fe4729b4adf1320d75c)
- Provider JCE (SHA256 checksum 124d04f667a7222d624f528404a289405669f4e4a1983b8a7926758bf67e35ec)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum b874556b34ff5f0d082ca19f62822cf97d6e7074ce3861d7577090f99cba0cbd)

Ubuntu 22.04 LTS

Scarica il software versione 5.16.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e36d9bf900e195e417db261034d820cb935bec5135a0aa332d46019c70257cac)

- OpenSSL Dynamic Engine (SHA256 checksum 2c39129a5acce96b8693ce8c4b4aa174d9c13bc8ea3c2652fa505b4e0a933842)
- Provider JCE (SHA256 checksum 03b70d834985a2500b73eecd831bff4f30389eb67460bf1d799466e59a9f9cfd)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 3fc4fc662188ebbd430d9b5e01a57774567b926f3478d0a455fee4968c47460d)

Scarica il software versione 5.16.0 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum c835d9bb604b5c47a702fd45267ffb9dfa37d68dd0734391f060835ea680a169)
- OpenSSL Dynamic Engine (SHA256 checksum 943213a0f14b50fbdf3a90286e7f7ab219ab77750698f5062013873419484a04)
- Provider JCE (SHA256 checksum 0ecc3e65a3b94c2dd3913f44aba58764e5558a8bedd3b742a42e4769533f3734)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 8558328b224b850cf9e1b3c4f8773565a90e38ab43e2c9e90bc5b841f98898d2)

Ubuntu 20.04 LTS

Scarica il software versione 5.16.0 per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 64a01e84d898aca3cd03f22022cf8dbbc806a1735a84df5f820ba5bfa3339b0e)
- OpenSSL Dynamic Engine (SHA256 checksum 53a73cfc60c3fddd3ad9173bf6e602faf289f20935b3c469a137847148442954)
- Provider JCE (SHA256 checksum 09544cb8129af4d06a241e1fbc0b6e6c207e1a518e5cb202f7b6c9802507d1b8)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum e1fdf1e014541f57d3ba4688bad88e7557e32ba974e020b4da76f85f1fc6aa29)

Windows Server 2025

Scarica il software versione 5.16.0 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- Provider JCE (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)

  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- Key Storage Provider (KSP) (SHA256 checksum 9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2022

Scarica il software versione 5.16.0 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- Provider JCE (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- Key Storage Provider (KSP) (SHA256 checksum 9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2019

Scarica il software versione 5.16.0 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)
- Provider JCE (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- Key Storage Provider (KSP) (SHA256 checksum 9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Windows Server 2016

Scarica il software versione 5.16.0 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6d3142d178127372de9ab40e7af5dc81fa7c627a18286dd521ad786d0947f17d)

- Provider JCE (SHA256 checksum 9f5c28583b5127b90e753dd7ce6d081bbe4810c0c30c424192704efa6ad1be34)
  - Javadocs per AWS CloudHSM (SHA256 checksum bbbbe99e427b7b4d51d7018a97c4d44372db020e44fdbb76fd3954fb104010e3)
- CLI CloudHSM (SHA256 checksum 22cfed533f528e0975f452af51d334baa1b5c143e722182dca107622e703b1ca)
- Key Storage Provider (KSP) (SHA256 checksum 9045c4f3e81093c4b49b93f3ea3f5caf2a4e6980628e0db2d971c437dc203bfc)

Client SDK 5.16 aggiunge ECDH con supporto KDF X963 sul tipo di cluster hsm2m.medium per il provider JCE e la libreria PKCS #11. Client SDK 5.16 aggiunge anche il supporto per la generazione di file di riferimento chiave KSP su Windows Server con CloudHSM CLI.

CLI CloudHSM

- È stato aggiunto il supporto per la generazione di file di riferimento delle chiavi KSP su Windows Server, per ulteriori informazioni, vedere. Generazione di riferimenti chiave KSP (Windows)

Provider JCE

- È stato aggiunto l'ECDH con supporto KDF X963 sul tipo di cluster hsm2m.medium, vedere. Meccanismi supportati per il provider JCE per AWS CloudHSM Client SDK 5

Libreria PKCS #11

- È stato aggiunto l'ECDH con supporto KDF X963 sul tipo di cluster hsm2m.medium, vedere. Meccanismi supportati per la libreria PKCS #11 per AWS CloudHSM Client SDK 5

Bug fixes/Improvements

- È stato corretto un bug per cui le chiavi di sessione non venivano pulite correttamente durante la disconnessione.
- È stato risolto un problema per cui le operazioni di disconnessione MTL riuscite su hsm2m.medium restituivano erroneamente una risposta di errore.
- È stato risolto un problema relativo all'errore di connessione durante l'avvio di Windows.
- È stato risolto un problema per cui gli strumenti di informazione SDK mostravano un output errato sui sistemi Ubuntu.

- È stato risolto un problema che in precedenza consentiva l'uso di ECDSA con hash più deboli in modalità FIPS.

Versione 5.15.0

Amazon Linux 2023

Scarica il software versione 5.15.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 41ef3178811df1dbb03b2cbac83fe0f4768bdc9b17005c409f1c0229f93ef11c)
- OpenSSL Dynamic Engine (SHA256 checksum afa1f9f8bd99f54866dea1b8928c00b951a6e492f5f36d0d6c7c38fff341d609)
- Provider JCE (SHA256 checksum 6ea775e05570ef3497a4df5c35a6ec1c682aea73c48e7fecec3e541af995759e)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 78c10bb213dd14fcfc5836e358de6aedac61db05125fd61137b0082214fdecbe)

Scarica il software versione 5.15.0 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum cd0617d29b6d64c00c8e14fd9a92f604e14acda4746f4f0b86b9de42367192fb)
- OpenSSL Dynamic Engine (SHA256 checksum bc9acfdd04eb1246eb3d5b0a8f3736ec017c0d1699d5395f85868d4a1722cd83)
- Provider JCE (SHA256 checksum 96d05301486206577b7be05aac561649167b57272c9c06ff839fd8ff2b5d96d5)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum b870eadb27736a2cde98022d57e9704c67ae15878cf0b910738859cdabaa35a2)

Amazon Linux 2

Scarica il software versione 5.15.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum c70ae4f0181a8187c9380481c51c1d03e12236dd86863ec818ed3f210b294c8e)
- OpenSSL Dynamic Engine (SHA256 checksum 08e9fd1dd80efa637f9a1727bb0de205ba124a3776b2e8bc21008ee458063a42)
- Provider JCE (SHA256 checksum 7932ed060e72c53b2556f30694b0ffe5342b244b6628c7a9dc03966aa49c8fe6)

    - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 0a939e4d5d0a2ff308c7a1d9e73ebc865e426214d556bde1bd29dbe807fbb583)

Scarica il software versione 5.15.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum bcf907a8fb4722b09c54b8e5785fe2b8cffcedd6ee3fdda2d21879a012138077)
- OpenSSL Dynamic Engine (SHA256 checksum 8f70edc3a6a4a1bf0264c6567b1dda1ac69055f206753f88eeadbb8bf3bf9f38)
- Provider JCE (SHA256 checksum 75dd67736bb08fe7e46e113af10803a255ba8edee3016ca963c1ee94fe59d43b)
    - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 1c2ecf90c955281d99bcd8d1956d63debb15bbc8419744c83f88821ef8b78aee)

RHEL 9 (9.2+)

Scarica il software versione 5.15.0 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 65bd0b815eebc806674a7bf7c54e9f884595881547f5fffd08ff6a38aabdccbe)
- OpenSSL Dynamic Engine (SHA256 checksum f2af9f5882ab2e5a11defecc660f8af5c4d9d6e2e2b89873e6833fc2976f44ac)
- Provider JCE (SHA256 checksum 5124bace5d1544775c891f13a0e309b30dd73699116c46ecc9a77bba5f9cf633)
    - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 1eec31fb2c0ab3c2839d9bdb37874cf8dbc74a383279068fce9e8613966d06e0)

Scarica il software versione 5.15.0 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum f498697519afa04b6b0362a4388a6a38e2cb6813b781d6ce3d97a7de89c5cbfe)
- OpenSSL Dynamic Engine (SHA256 checksum 5410da63108a1b209e567e9bccf8bd7e4035af88b0d58b9d78b10917be1b40c1)
- Provider JCE (SHA256 checksum b46f233e6994d2c0ed505dc5c717ee3009daaeb2063d260aa78e06273770bffa)
    - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)

- CLI CloudHSM (SHA256 checksum e9f93eaa58db2f7ea1174164ef96ab219700933d353243c3c6ab1aebac5ccffe)

RHEL 8 (8.3+)

Scarica il software versione 5.15.0 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 87131e179d0e60ade302ec07b22803cfb39294bf060b786c41f154d95791ac94)
- OpenSSL Dynamic Engine (SHA256 checksum f412a2f5cd761db5940288bb252ce060d44735c6b436bb6d4fa7d3687a44a026)
- Provider JCE (SHA256 checksum d0844f55c08f9ff9c393138a9041efe1b59dc3dce20c0b2c23406efe6acc43db)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 9fd8033e478ce6d7d640c063c4f007359cb04c19d519826a745ad0885f96a0f8)

Ubuntu 24.04 LTS

Scarica il software versione 5.15.0 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ca5f2f80ae921cfebdc5c8bc35c39d2b19cfabfd5981932409eaf2e7c00a9097)
- OpenSSL Dynamic Engine (SHA256 checksum e44cd7b678a421957c84e4fc0f70280360fd4e1e66f4cabdd1b20b955ee5fcca)
- Provider JCE (SHA256 checksum bc6382f12769c3d87c34522d0d616b7a8c108574c003814e1300938c386655ac)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum f786be86c204680e670f7817376ff376733ebd96b51c31e9c98213078596637f)

Scarica il software versione 5.15.0 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 1ecf057d67137c0e9e1bb0dda57d581690c647b0360c3a617a8dc668919d08de)
- OpenSSL Dynamic Engine (SHA256 checksum 4f3d23e6b798f88be587108d8e6a225d796a3d080aef61ea384eb74a1270612a)
- Provider JCE (SHA256 checksum b9fd16bdcc1fcf59fde0d3e0debee500b0b7edfdff69209e84d14393097fe9d2)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum a95922d9b44e64a64723db0c21ac89566515a5a6c87de990af4a9e1f40c7424f)

Ubuntu 22.04 LTS

Scarica il software versione 5.15.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 966be12eb32de813ca07e766abf7b5616c0d2e105e9296d920aadaca10e5afdf)
- OpenSSL Dynamic Engine (SHA256 checksum f2840151d87b7f9cbff68993c25397afd48a16f054abf0f2fd4624662d3087d6)
- Provider JCE (SHA256 checksum 01aebdda05640e50a82cae04c5cb6d33ab909dadd917bd834957d4f75ad8c577)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 2e5eba9409abc429828505779c512cd2424719766c0e488f50aee288966cf61d)

Scarica il software versione 5.15.0 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 614f74f101c9c64fff515128c5a59fa23de5047494fe248e8aabdab441092b3e)
- OpenSSL Dynamic Engine (SHA256 checksum c15d6db77b76bce690749b73b947567eb5f2d76669887843116d0ce56c1f8ea7)
- Provider JCE (SHA256 checksum bca8511d5c0a173b0fef326016ce5091b6e6829fa2b3cb45ed5621290ae3e42a)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 2f5b673148b682d7e34619c51b5e8799abe7dc7fd4f046158a0d05320ba24dc1)

Ubuntu 20.04 LTS

Scarica il software versione 5.15.0 per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 86c8394b5ddff91a71194fb87c327efde36baa2380e559c04f9d543a6e74d61b)
- OpenSSL Dynamic Engine (SHA256 checksum 0a4227389fea61e6e7ac7cfa715eb341f7a4eeae9ed10e4c96da2c0dd4a18f9e)
- Provider JCE (SHA256 checksum 65b2d926ff9dfbe6c7864bc3a41b3da2383bc731dd199bcef8805a0543fbe612)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 09c2e55bcf72f9e530717950d8c5fdfd48574ae6ccb09a049526ca5b2a3b8aa9)

Windows Server 2025

Scarica il software versione 5.15.0 per Windows Server 2025 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- Provider JCE (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- Key Storage Provider (KSP) (SHA256 checksum 52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2022

Scarica il software versione 5.15.0 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- Provider JCE (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- Key Storage Provider (KSP) (SHA256 checksum 52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2019

Scarica il software versione 5.15.0 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- Provider JCE (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- Key Storage Provider (KSP) (SHA256 checksum 52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Windows Server 2016

Scarica il software versione 5.15.0 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum a903b63fe286f15bf669c0555b1fa4d86b33592ed05af0809acac28c0d3ace16)
- Provider JCE (SHA256 checksum fdef6251f06d77d51fddbc2184d3eec87ddec4fe35b3ac620343eb66c95ddf64)
  - Javadocs per AWS CloudHSM (SHA256 checksum ce5d92731f2a9c7f46c92bba92f3d0adff4ceb06b9d653597994100dfc352fbe)
- CLI CloudHSM (SHA256 checksum 6cc9205fe1fc514ddd9774f824e512940d6b1bc4cc0e251265bc46ab99746c28)
- Key Storage Provider (KSP) (SHA256 checksum 52ed9b08cd0ce100b8dcd3d8e8f411b6201f9f1b27872b19d1136c0bf36a29b8)

Client SDK 5.15 aggiunge il supporto per la replica di un utente su cluster clonati con CloudHSM CLI. Client SDK 5.15 aggiunge anche pacchetti di installazione per la libreria PKCS #11, il provider JCE, CloudHSM CLI e Key Storage Provider (KSP) per Windows Server 2025.

Supporto piattaforme

- È stato aggiunto il supporto per Windows Server 2025 for PKCS #11 library, JCE provider, CloudHSM CLI e Key Storage Provider (KSP).

CLI CloudHSM

- È stato aggiunto il seguente comando:
  - Replica un utente con CloudHSM CLI

Versione 5.14.0

Amazon Linux 2023

Scarica il software versione 5.14.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 05e7a3882166c694a7a09bc735f08f91c8145a4215176665eacacdf3e509abe8)
- OpenSSL Dynamic Engine (SHA256 checksum f4dd9966988418e100c276dc0d521f91afdfc0e6c008dbf8eda446ebaca83c14)
- Provider JCE (SHA256 checksum c4dee5c1173f6a1c7683aedb58e61d329c36933435c416f288d94bc9a68a6b31)

- Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 1fad069fde305450254287e43750d597db9af0cb8fd168300cd5eaed9e2af33a)

Scarica il software versione 5.14.0 per Amazon Linux 2023 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 9f34163d02bce26c1280e589310cda891d27995c50cec0a7fe083100ecff2b69)
- OpenSSL Dynamic Engine (SHA256 checksum 7444a4daad6e4715c82d6c39a7b03a07ee0201a13fe0f98da96acbf9d24abf6c)
- Provider JCE (SHA256 checksum 3393fe3a0f5c3a9c92106d74b7de074c818e095f97d2cfd600dbd47779b90b37)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum d9a2edce48c5f6646d5a351cb431712f2d2fc62d21f8318e7bb1ce579819d7f4)

Amazon Linux 2

Scarica il software versione 5.14.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 9c47b90bfa0ad51627cdb0dd8f148a56090fbdeb2490f1ab4009170c7b9c1120)
- OpenSSL Dynamic Engine (SHA256 checksum 215f9768331565085a317585b3dbe0514b251fdc428c96ed32491c4abb9fea56)
- Provider JCE (SHA256 checksum a802f941e95fcbf0ef37775fc096c0d6ae4c916fa08330a9d56defc5f99ff2b7)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum ac10f23dd81264b1d10a0760f62b5006d2b2b66bf1c6378248ca9326afb65a83)

Scarica il software versione 5.14.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum f2d1550f043c655ef5d51e2ca14a0416886d99902e0702bd7f30f93b2c563d4d)
- OpenSSL Dynamic Engine (SHA256 checksum 2c8e2c81af53ba3646d1f947894d84b9b54780bbca79b58f354125e4ac9427c0)
- Provider JCE (SHA256 checksum 3f9c881056e6905d46358585db72143b59971958a708fa4ac75cb53994487213)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)

- CLI CloudHSM (SHA256 checksum 6d5985e44c9852409dd3d342239fbbf7d0f9ad43d445e0980153c9b9eafe2b6f)

RHEL 9 (9.2+)

Scarica il software versione 5.14.0 per RHEL 9 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum c4812210421ff2fa5dac8477a2e6b10552aabd88b1f03d717e044e7293823158)
- OpenSSL Dynamic Engine (SHA256 checksum 3f6aeaa6ae1faae7d8bba1596f358cad1eec9e562cf08aaab9ded92cabe94719)
- Provider JCE (SHA256 checksum bd8f120b08f738ad4d1534b0e32aad903758d5e75d3aba7cb9ff6a77dec533db)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 84f13a9babd767edf90dabb0b14034ec5b2208898123a13966dd6b8519961c27)

Scarica il software versione 5.14.0 per RHEL 9 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 3faa1de563aa9773939cab7b39907dcb7b981ed7225019f9070a9dd52db7ae70)
- OpenSSL Dynamic Engine (SHA256 checksum 13b41ac47ef7ee7bf78f585b8347ca4da9ebc296e4fc1e6a0c2ff5b333354ca6)
- Provider JCE (SHA256 checksum 06803655ebe54d59c180bb17ac6fe56337bdd58ad0f6fe87c50ff8df32f70258)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum e6b4e9688d0db9d72bbee3450fe19736d640c9931adbd4f4ef73cb7ac2a08cf4)

RHEL 8 (8.3+)

Scarica il software versione 5.14.0 per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e400aeea6dbf7721f97e71c643a0db4f5f1094fb197fc46dc0ab293de9d16f2d)
- OpenSSL Dynamic Engine (SHA256 checksum bfcc27d251e62f9eba0fd508e7d08dc62126642d4cdd0b5566183957768b8c54)
- Provider JCE (SHA256 checksum b5500031b572c918a8df4a0347e01c8ea00a7366b865b310bd92427fa1ed53e3)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 000712d0a691efc64a6c5d54bfeb1ab48b315ebb5dd6926b0502e326bf700291)

Ubuntu 24.04 LTS

Scarica il software versione 5.14.0 per Ubuntu 24.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum bbec70a198a4b173620b4018accc297ce6a6a80d372706e0101997d6bca35bca)
- OpenSSL Dynamic Engine (SHA256 checksum fcb77f75bd465b22401a09a20c4109858333402951012 63b7171cdcc4ac9f980)
- Provider JCE (SHA256 checksum e1fa16aae2f6095c89a8bf392ef2cb9ca4db8853c858904ff90abf4bb491d74b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum af32e0ac1c5f8c7f8cb40d255abe7c63ce9d981293187a3fb452562fa05756f4)

Scarica il software versione 5.14.0 per Ubuntu 24.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 1232318b084347889f92136536537ca519373683ce39bcec20f6ac3fa2f42d7c)
- OpenSSL Dynamic Engine (SHA256 checksum 1745ee3a33d8e6ea72644e903c55c6a206204cf0a8bea200bc4a7b15736ed801)
- Provider JCE (SHA256 checksum 7a44acabbc90c996594ed53661937f9242850823347f7c386f02fc041a97471a)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum e6778bd12c55fd152b50033833531fe569472f4f2bd9927a345eb126e8305739)

Ubuntu 22.04 LTS

Scarica il software versione 5.14.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 6b4b1620e9a85267950633b171dd188b7ac7094e371e188fabc1bef7a911a16f)
- OpenSSL Dynamic Engine (SHA256 checksum fee5f0a65fab0f46ad58689af5dc510721581f31364d3be5cbbf79f5d9a60db8)
- Provider JCE (SHA256 checksum 5883a3d15e160d65f8e26f185e1ee30f68becad5f6fcd16abc1c4586689800b5)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 7b1a8cd2962f4be7325154c080dabfb820beac5629e272c3a3ebc0e6cab11d27)

Scarica il software versione 5.14.0 per Ubuntu 22.04 LTS su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 8680f63fd74272ea0e4d4c32dfb96b3eb9b60e03483c202d4e9b65ee101a178f)
- OpenSSL Dynamic Engine (SHA256 checksum c757304e8fc5f38be3bab7ac6d37a35dcb56d31e62f7194da62b3a176593d1d8)
- Provider JCE (SHA256 checksum a78ee59341da56af315d126ec7ed9f0dafe6e99649f659db2436be3863cb035b)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 18931bff869a0bd54846b3296d870fa19beba5652f979352be8fba6307e6d1aa)

Ubuntu 20.04 LTS

Scarica il software versione 5.14.0 per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum f1461c16b135ebcc17deec46aab88bd113ea122b8942fc188d4f05cd03e919a8)
- OpenSSL Dynamic Engine (SHA256 checksum 89211a7a7ed50eda2dc385c31ea76f1fbabd389ca691204873531d983c3eb0f7)
- Provider JCE (SHA256 checksum f098c32d61a53b073459a75d88b68e377a9b16335874fae060cb10df0da00df0)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum ff91fb930717c917344af2ba344dc6e02bd5abc004dcb6147e9412b67e2aa7ab)

Windows Server 2022

Scarica il software versione 5.14.0 per Windows Server 2022 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- Provider JCE (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- Key Storage Provider (KSP) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

Windows Server 2019

Scarica il software versione 5.14.0 per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- Provider JCE (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- Key Storage Provider (KSP) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

Windows Server 2016

Scarica il software versione 5.14.0 per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ee5a8e8e85fff7acd0bbafa23740e3981f7dc52e708972b600c2b26603786838)
- Provider JCE (SHA256 checksum 2ae0274f09f66981c03fd1e3c264e896ba7cd211168ea31369335db1b3ea2e77)
  - Javadocs per AWS CloudHSM (SHA256 checksum 173087a8f286dc4f88dc915ecc00aa515bec0ae2faf219654df9f3422d8e83bb)
- CLI CloudHSM (SHA256 checksum 3938654f88ce010042e48909e23991e178e411e2bd9e3c9ec25fcb8157c0cd55)
- Key Storage Provider (KSP) (SHA256 checksum b026e4d8c11e9ff6f22a7b9e10b8bb29e7572665f0d7978a3cef7d2354b7693f)

Client SDK 5.14 aggiunge il supporto per l'utilizzo delle chiavi con controllo del quorum e le operazioni di gestione delle chiavi tramite CloudHSM CLI. Client SDK 5.14 aggiunge anche il supporto per piattaforme Windows. Provider di archiviazione delle chiavi (KSP) per AWS CloudHSM Client SDK 5 Inoltre, Client SDK 5.14 aggiunge pacchetti di installazione per la libreria PKCS #11, il provider JCE, l'interfaccia a riga di comando CloudHSM e il Key Storage Provider (KSP) per Windows Server 2022.

Supporto piattaforme

- È stato aggiunto il supporto per Windows Server 2022 per la libreria PKCS #11, il provider JCE, l'interfaccia a riga di comando CloudHSM e il Key Storage Provider (KSP).

CLI CloudHSM

- È stato aggiunto il supporto per l'utilizzo delle chiavi con controllo del quorum e le operazioni di gestione delle chiavi.

Key Storage Provider (KSP)

- È stato aggiunto il supporto per Key Storage Provider (KSP), un'API crittografica specifica per il sistema operativo Microsoft Windows. Per ulteriori informazioni, consulta Provider di archiviazione delle chiavi (KSP) per AWS CloudHSM Client SDK 5

Versione 5.13.0

Client SDK 5.13 aggiunge il supporto per la configurazione del TLS reciproco sui tipi di cluster hsm2m.medium. Per informazioni sull'utilizzo del TLS reciproco con CloudHSM, consulta. Configura TLS reciproco tra client e AWS CloudHSM (consigliato) Client SDK 5.13 aggiunge anche pacchetti di installazione per Ubuntu 24.04 LTS.

Supporto piattaforme

- È stato aggiunto il supporto per Ubuntu 24.04 LTS sulle architetture x86_64 e ARM64 per tutti gli SDK.

CLI CloudHSM

- È stato aggiunto il supporto per gli utenti amministratori per l'esecuzione del comando. Replica di una chiave con la CLI di CloudhSM Client SDK 5.12 ha introdotto il comando key replicate per l'utilizzo da parte degli utenti crittografici.
- È stato aggiunto il seguente comando:
  - La categoria cluster mtls nella CLI di CloudhSM

Bug fixes/Improvements

- È stato risolto un problema per ridurre il tempo necessario al client per rilevare connessioni HSM non intatte, il che aiuta a prevenire errori di interruzione della connessione durante gli avviamenti lambda a caldo.

Versione 5.12.0

Client SDK 5.12.0 aggiunge il supporto ARM a diverse piattaforme e migliora le prestazioni per tutti gli SDK. Sono state aggiunte nuove funzionalità alla CLI CloudHSM e al provider JCE.

Supporto piattaforme

- È stato aggiunto il supporto per Amazon Linux 2023 sull'architettura ARM64 per tutti gli SDK.
- È stato aggiunto il supporto per Red Hat Enterprise Linux 9 (9.2+) su architettura ARM64 per tutti gli SDK.
- È stato aggiunto il supporto per Ubuntu 22.04 LTS su architettura ARM64 per tutti gli SDK.

CLI CloudHSM

- Aggiunto il seguente comando:
  - Replica di una chiave con la CLI di CloudhSM
- È stato aggiunto il supporto per la connessione a più cluster. Per ulteriori informazioni, consulta Connessione a più cluster con la CLI CloudhSM.

Provider JCE

- Aggiunto `KeyReferenceSpec` per il recupero delle chiavi utilizzando. `KeyStoreWithAttributes`
- Aggiunto `getKeys` per recuperare più chiavi contemporaneamente utilizzando. `KeyStoreWithAttributes`

Miglioramenti in termini di prestazioni

- Miglioramenti delle prestazioni per il NoPadding funzionamento AES CBC per tutti gli SDK.

Versione 5.11.0

Client SDK 5.11.0 aggiunge nuove funzionalità, migliora la stabilità e include correzioni di bug per tutti gli SDK.

Supporto piattaforme

- È stato aggiunto il supporto per Amazon Linux 2023 e RHEL 9 (9.2+) per tutti gli SDK.
- È stato rimosso il supporto per Ubuntu 18.04 LTS a causa della sua recente fine di vita.
- È stato rimosso il supporto per Amazon Linux a causa della sua recente fine del ciclo di vita.

CLI CloudHSM

- Sono stati aggiunti i seguenti comandi:
  - La categoria delle firme crittografiche in CloudHSM CLI
  - La categoria di verifica crittografica in CloudHSM CLI
  - Importa una chiave in formato PEM con CloudHSM CLI
  - Il comando key unwrap nella CLI di CloudHSM
  - Il comando key wrap nella CLI di CloudHSM
- Esporta una chiave asimmetrica con CloudHSM CLIora supporta l'esportazione di chiavi pubbliche.

OpenSSL Dynamic Engine

- L' AWS CloudHSM OpenSSL Dynamic Engine è ora supportato su piattaforme installate con una versione 3.x della libreria OpenSSL. Ciò include Amazon Linux 2023, RHEL 9 (9.2+) e Ubuntu 22.04.

JCE

- È stato aggiunto il supporto per JDK 17 e JDK 21.
- È stato aggiunto il supporto per le chiavi AES da utilizzare per le operazioni HMAC.
- È stato aggiunto il nuovo attributo `ID` chiave.
- È stata introdotta una nuova `DataExceptionCause` variante per l'esaurimento delle chiavi:`DataExceptionCause.KEY_EXHAUSTED`.

Bug fixes/Improvements

- È stata aumentata la lunghezza massima dell'`label`attributo da 126 a 127 caratteri.
- È stato corretto un bug che impediva l'apertura delle chiavi EC con il meccanismo. RsaOaep
- È stato risolto un problema noto relativo all'operazione getKey nel provider JCE. Fare riferimento a Problema: perdita di memoria del Client SDK 5 con operazioni GetKey per ulteriori dettagli.
- È stato migliorato l'accesso a tutti gli SDK per le chiavi Triple DES che hanno raggiunto il limite massimo di blocco di crittografia, secondo FIPS 140-2.
- Sono stati aggiunti problemi noti per l'OpenSSL Dynamic Engine. Per informazioni dettagliate, vedi Problemi noti di OpenSSL Dynamic Engine per AWS CloudHSM.

Versione 5.10.0

Il Client SDK 5.10.0 migliora la stabilità e include correzioni di bug per tutti gli SDK.

CLI CloudHSM

- Sono stati aggiunti nuovi comandi che consentono ai clienti di gestire le chiavi utilizzando la CLI di CloudHSM, tra cui:
  - Creare coppie di chiavi simmetriche e asimmetriche
  - Condivisione e annullamento della condivisione delle chiavi
  - Elenco e filtro delle chiavi attraverso gli attributi delle chiavi
  - Elenco degli attributi della chiave
  - Generazione file di riferimento della chiave
  - Eliminazione delle chiavi
- Registrazione degli errori migliorata.
- Aggiunto supporto per i comandi unicode multilinea in modalità interattiva.

Insetto fixes/Improvements

- Prestazioni migliorate per importazione, decompressione, derivazione e creazione di chiavi di sessione per tutti gli SDK.
- Corretto un bug nel provider JCE che impediva la rimozione dei file temporanei all'uscita.
- Corretto un bug che causava un errore di connessione in determinate condizioni dopo la sostituzione dei moduli HSM nel cluster.
- Modificato formato di output di JCE `getVersion` per la gestione di un numero elevato di versioni minori e per l'inclusione del numero di patch.

Supporto delle piattaforme

- Aggiunto supporto per Ubuntu 22.04 con JCE, PKCS #11 e CLI CloudHSM (supporto per OpenSSL Dynamic Engine non ancora disponibile).

Versione 5.9.0

Il Client SDK 5.9.0 migliora la stabilità e include correzioni di bug per tutti gli SDK. È stata effettuata un'ottimizzazione per tutti gli SDK per informare immediatamente le applicazioni in caso di guasto operativo quando un HSM risulta non disponibile. Questa versione include miglioramenti delle prestazioni per JCE.

Provider JCE

- Prestazioni migliorate
- È stato risolto un problema noto relativo all'esaurimento del pool di sessioni

AWS CloudHSM versioni Client SDK obsolete

Le versioni 5.8.0 e precedenti sono obsolete. Si sconsiglia di utilizzare le versioni deprecate nei carichi di lavoro di produzione. Non forniamo aggiornamenti retrocompatibili per le versioni obsolete, né ospitiamo versioni obsolete da scaricare. Se noti un impatto sulla produzione utilizzando versioni obsolete, esegui l'aggiornamento per ottenere correzioni del software.

Versioni obsolete di Client SDK 5

Questa sezione elenca le versioni obsolete di Client SDK 5.

Versione 5.8.0

La versione 5.8.0 introduce l'autenticazione quorum per CloudHSM CLI, l' SSL/TLSoffload con JSSE, il supporto multi-slot per PKCS #11, il supporto cluster/multi multiutente per JCE, l'estrazione delle chiavi con JCE, il supporto di KeyFactory per JCE, nuove configurazioni di riprova per i codici restituiti non terminali e include una migliore stabilità e correzioni di bug per tutti gli SDK.

Libreria PKCS #11

- Aggiunto supporto per la configurazione multi-slot.

Provider JCE

- Aggiunta estrazione delle chiavi basata sulla configurazione.
- Aggiunto supporto per configurazioni multi-cluster e multiutente.

- Aggiunto supporto per l'offload SSL e TLS con JSSE.
- È stato aggiunto il supporto unwrap per. AES/CBC/NoPadding
- Aggiunti nuovi tipi di fabbriche chiave: SecretKeyFactory e. KeyFactory

CLI CloudHSM

- Aggiunto supporto per l'autenticazione del quorum

Versione 5.7.0

La versione 5.7.0 introduce la CLI CloudHSM e include un nuovo algoritmo di autenticazione dei messaggi basato su cifratura (CMAC). Questa versione aggiunge l'architettura ARM ad Amazon Linux 2. I Javadocs del provider JCE sono ora disponibili per l' AWS CloudHSM.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.
- Ora supportato su architettura ARM con Amazon Linux 2.
- Algoritmi
  - CKM_AES_CMAC (firma e verifica)

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.
- Ora supportato su architettura ARM con Amazon Linux 2.

Provider JCE

- Maggiore stabilità e correzione dei bug.
- Algoritmi
  - AESCMAC

Versione 5.6.0

La versione 5.6.0 include un nuovo meccanismo di supporto per la libreria PKCS #11 e il provider JCE. Inoltre, la versione 5.6 supporta Ubuntu 20.04.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.
- Meccanismi
  - CKM_RSA_X_509, per le modalità di crittografia, decodifica, firma e verifica

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Maggiore stabilità e correzione dei bug.
- Crittografie
  - RSA/ECB/NoPadding, per le modalità di crittografia e decrittografia

Chiavi supportate

- EC con curve secp224r1 e secp521r1

Supporto piattaforme

- È stato aggiunto il supporto per Ubuntu 20.04.

Versione 5.5.0

La versione 5.5.0 aggiunge il supporto per l'integrazione con OpenJDK 11, Keytool e Jarsigner e meccanismi aggiuntivi al provider JCE. Risolve un problema noto relativo a una KeyGenerator classe che interpreta erroneamente il parametro di dimensione della chiave come numero di byte anziché di bit.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Supporto per i programmi Keytool e Jarsigner
- Supporto per OpenJDK 11 su tutte le piattaforme
- Crittografie
  - AES/CBC/NoPadding Modalità di crittografia e decrittografia
  - AES/ECB/PKCS5Padding Modalità di crittografia e decrittografia
  - AES/CTR/NoPadding Modalità di crittografia e decrittografia
  - AES/GCM/NoPadding Modalità Wrap and Unwrap
  - DESede/ECB/PKCS5Padding Modalità di crittografia e decrittografia
  - DESede/CBC/NoPadding Modalità di crittografia e decrittografia
  - AESWrap/ECB/NoPadding Modalità Wrap and Unwrap
  - AESWrap/ECB/PKCS5Padding Modalità Wrap and Unwrap
  - AESWrap/ECB/ZeroPadding Modalità Wrap and Unwrap
  - RSA/ECB/PKCS1Padding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPPadding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPWithSHA-1ANDMGF1Padding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPWithSHA-224ANDMGF1Padding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPWithSHA-256ANDMGF1Padding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPWithSHA-384ANDMGF1Padding Modalità Wrap and Unwrap
  - RSA/ECB/OAEPWithSHA-512ANDMGF1Padding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPPadding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPWithSHA-1ANDMGF1Padding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPWithSHA-224ANDMGF1Padding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPWithSHA-256ANDMGF1Padding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPWithSHA-384ANDMGF1Padding Modalità Wrap and Unwrap
  - RSAAESWrap/ECB/OAEPWithSHA-512ANDMGF1Padding Modalità Wrap and Unwrap
- KeyFactory e SecretKeyFactory

  - RSA - chiavi RSA da 2048 a 4096 bit, con incrementi di 256 bit
  - AES - chiavi AES a 128, 192 e 256 bit
  - Coppie di chiavi EC per le curve NIST secp256r1 (), secp384r1 (P-256) e secp256k1 P-384
  - DESede (3DES)
  - GenericSecret
  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512
- Sign/Verify
  - RSASSA-PSS
  - SHA1withRSA/PSS
  - SHA224withRSA/PSS
  - SHA256withRSA/PSS
  - SHA384withRSA/PSS
  - SHA512withRSA/PSS
  - SHA1 con RSA e MGF1
  - SHA224 con RSA e MGF1
  - SHA256 con RSA e MGF1
  - SHA384 con RSA e MGF1
  - SHA512 con RSA e MGF1

versione 5.4.2

La versione 5.4.2 include una maggiore stabilità e correzione dei bug per tutti gli SDK. Questa è anche l'ultima versione per la piattaforma CentOS 8. Per ulteriori informazioni, vedi il sito web CentOS.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Maggiore stabilità e correzione dei bug.

Versione 5.4.1

La versione 5.4.1 risolve un problema noto con la libreria PKCS #11. Questa è anche l'ultima versione per la piattaforma CentOS 8. Per ulteriori informazioni, vedi il sito web CentOS.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Maggiore stabilità e correzione dei bug.

Versione 5.4.0

La versione 5.4.0 aggiunge il supporto iniziale per il provider JCE per tutte le piattaforme. Il provider JCE è compatibile con OpenJDK 8.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Tipi di chiave
  - RSA - chiavi RSA da 2048-bit a 4096-bit, con incrementi di 256 bit.

  - AES - chiavi AES a 128, 192 e 256 bit.
  - Coppie di chiavi ECC per le curve NIST secp256r1 (), secp384r1 () e secp256k1. P-256 P-384
  - DESede (3DES)
  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifre (solo crittografia e decodifica)
  - AES/GCM/NoPadding
  - AES/ECB/NoPadding
  - AES/CBC/PKCS5Padding
  - DESede/ECB/NoPadding
  - DESede/CBC/PKCS5Padding
  - AES/CTR/NoPadding
  - RSA/ECB/PKCS1Padding
  - RSA/ECB/OAEPPadding
  - RSA/ECB/OAEPWithSHA-1ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-224ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-256ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-384ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-512ANDMGF1Padding
- File digest
  - SHA-1
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- Sign/Verify
  - NONE con RSA
  - SHA1 con RSA
  - SHA224 con RSA
  - SHA256 con RSA
  - SHA384 con RSA

  - SHA512 con RSA
  - NONE con ECDSA
  - SHA1 con ECDSA
  - SHA224 con ECDSA
  - SHA256 con ECDSA
  - SHA384 con ECDSA
  - SHA512 con ECDSA
- Integrazione con Java KeyStore

Versione 5.3.0

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Aggiungi il supporto per ECDSA sign/verify con curve e P-256 P-384 secp256k1.
- Aggiungi il supporto per le piattaforme: Amazon Linux, Amazon Linux 2, CentOS 7.8+, RHEL 7 (7.8+).
- Aggiunto supporto per la versione OpenSSL 1.0.2.
- Maggiore stabilità e correzione dei bug.

Provider JCE

- Tipi di chiave
  - RSA - chiavi RSA da 2048-bit a 4096-bit, con incrementi di 256 bit.
  - AES - chiavi AES a 128, 192 e 256 bit.
  - Coppie di chiavi EC per le curve NIST secp256r1 (), secp384r1 () e secp256k1. P-256 P-384
  - DESede (3DES)
  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifre (solo crittografia e decodifica)
  - AES/GCM/NoPadding

  - AES/ECB/NoPadding
  - AES/CBC/PKCS5Padding
  - DESede/ECB/NoPadding
  - DESede/CBC/PKCS5Padding
  - AES/CTR/NoPadding
  - RSA/ECB/PKCS1Padding
  - RSA/ECB/OAEPPadding
  - RSA/ECB/OAEPWithSHA-1ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-224ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-256ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-384ANDMGF1Padding
  - RSA/ECB/OAEPWithSHA-512ANDMGF1Padding
- File digest
  - SHA-1
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- Sign/Verify
  - NONE con RSA
  - SHA1 con RSA
  - SHA224 con RSA
  - SHA256 con RSA
  - SHA384 con RSA
  - SHA512 con RSA
  - NONE con ECDSA
  - SHA1 con ECDSA
  - SHA224 con ECDSA
  - SHA256 con ECDSA
  - SHA384 con ECDSA

  - SHA512 con ECDSA
- Integrazione con Java KeyStore

Versione 5.2.1

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Versione 5.2.0

La versione 5.2.0 aggiunge il supporto di tipi di chiavi e meccanismi aggiuntivi alla libreria PKCS #11.

Libreria PKCS #11

Tipo di chiavi

- Curve ECDSA— P-224, P-256 P-384, P-521 e secp256k1
- Triple DES (3DES)

Meccanismi

- CKM_EC_KEY_PAIR_GEN
- CKM_DES3_KEY_GEN
- CKM_DES3_CBC
- CKM_DES3_CBC_PAD
- CKM_DES3_ECB
- CKM_ECDSA
- CKM_ECDSA_SHA1
- CKM_ECDSA_SHA224
- CKM_ECDSA_SHA256
- CKM_ECDSA_SHA384

- CKM_ECDSA_SHA512
- CKM_RSA_PKCS per Encrypt/Decrypt

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Versione 5.1.0

La versione 5.1.0 aggiunge il supporto per meccanismi aggiuntivi alla libreria PKCS #11.

Libreria PKCS #11

Meccanismi

- CKM_RSA_PKCS per Wrap/Unwrap
- CKM_RSA_PKCS_PSS
- CKM_SHA1_RSA_PKCS_PSS
- CKM_SHA224_RSA_PKCS_PSS
- CKM_SHA256_RSA_PKCS_PSS
- CKM_SHA384_RSA_PKCS_PSS
- CKM_SHA512_RSA_PKCS_PSS
- CKM_AES_ECB
- CKM_AES_CTR
- CKM_AES_CBC
- CKM_AES_CBC_PAD
- CKM_SP800_108_COUNTER_KDF
- CKM_GENERIC_SECRET_KEY_GEN
- CKM_SHA_1_HMAC
- CKM_SHA224_HMAC
- CKM_SHA256_HMAC
- CKM_SHA384_HMAC
- CKM_SHA512_HMAC

- Solo Wrap/Unwrap CKM_RSA_PKCS_OAEP
- CKM_RSA_AES_KEY_WRAP
- CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD
- CKM_CLOUDHSM_AES_KEY_WRAP_PKCS5_PAD
- CKM_CLOUDHSM_AES_KEY_WRAP_ZERO_PAD

Operazioni API

- C_ CreateObject
- C_ DeriveKey
- C_ WrapKey
- C_ UnWrapKey

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Versione 5.0.1

La versione 5.0.1 aggiunge il supporto iniziale per OpenSSL Dynamic Engine.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Versione iniziale di OpenSSL Dynamic Engine.
- Questa versione offre un supporto introduttivo per i tipi di chiave e gli API OpenSSL:
  - Generazione di chiavi RSA per chiavi a 2048, 3072 e 4096 bit
  - API OpenSSL:
    - RSA Sign utilizzando RSA PKCS con & RSA PSS SHA1/224/256/384/512
    - Generazione di chiavi RSA

  Per ulteriori informazioni, vedi OpenSSL Dynamic Engine.

- Piattaforme supportate: CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3+ e Ubuntu 18.04 LTS
  - Richiede: OpenSSL 1.1.1

  Per ulteriori informazioni, vedi Piattaforme supportate.
- Supporto per SSL/TLS Offload su CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3 e Ubuntu 18.04 LTS, incluso NGINX 1.19 (per alcune suite di crittografia).

  Per maggiori informazioni, vedi SSL/TLS Offload su Linux usando Tomcat o Offload su Linux usando NGINX o Apache. SSL/TLS

Versione 5.0.0

La versione 5.0.0 è la prima versione.

Libreria PKCS #11

- Questa è la versione iniziale.

Supporto introduttivo alla libreria PKCS #11 nella versione 5.0.0 del client SDK

Questa sezione descrive in dettaglio il supporto per i tipi di chiave, i meccanismi, le operazioni API e gli attributi della versione 5.0.0 del Client SDK.

Tipi di chiave:

- AES – chiavi AES a 128, 192 e 256 bit
- RSA – chiavi RSA da 2048 a 4096 bit, con incrementi di 256 bit

Meccanismi:

- CKM_AES_GCM
- CKM_AES_KEY_GEN
- CKM_CLOUDHSM_AES_GCM
- CKM_RSA_PKCS
- CKM_RSA_X9_31_KEY_PAIR_GEN
- CKM_SHA1
- CKM_SHA1_RSA_PKCS

- CKM_SHA224
- CKM_SHA224_RSA_PKCS
- CKM_SHA256
- CKM_SHA256_RSA_PKCS
- CKM_SHA384
- CKM_SHA384_RSA_PKCS
- CKM_SHA512
- CKM_SHA512_RSA_PKCS

Operazioni API:

- C_ CloseAllSessions
- C_ CloseSession
- C_Decodifica
- C_ DecryptFinal
- C_ DecryptInit
- C_ DecryptUpdate
- C_ DestroyObject
- C_Digest
- C_ DigestFinal
- C_ DigestInit
- C_ DigestUpdate
- C_Crittografia
- C_ EncryptFinal
- C_ EncryptInit
- C_ EncryptUpdate
- C_Finalizza
- C_ FindObjects
- C_ FindObjectsFinal
- C_ FindObjectsInit
- C_ GenerateKey

- C_ GenerateKeyPair
- C_ GenerateRandom
- C_ GetAttributeValue
- C_ GetFunctionList
- C_ GetInfo
- C_ GetMechanismInfo
- C_ GetMechanismList
- C_ GetSessionInfo
- C_ GetSlotInfo
- C_ GetSlotList
- C_ GetTokenInfo
- C_Inizializza
- C_Login
- C_Logout
- C_ OpenSession
- C_Firma
- C_ SignFinal
- C_ SignInit
- C_ SignUpdate
- C_Verifica
- C_ VerifyFinal
- C_ VerifyInit
- C_ VerifyUpdate

Attributi:

- GenerateKeyPair
  - Tutti gli attributi delle chiavi RSA
- GenerateKey
  - Tutti gli attributi delle chiavi AES
- GetAttributeValue

- Tutti gli attributi delle chiavi RSA
- Tutti gli attributi delle chiavi AES

Esempi:

- Generazione di chiavi (AES, RSA, EC)
- Elenco degli attributi chiave
- Crittografia e decodifica dei dati con AES GCM
- Firma e verifica dei dati con RSA

Versioni obsolete di Client SDK 3

Questa sezione elenca le versioni obsolete di Client SDK 3.

Versione 3.4.4

La versione 3.4.4 aggiunge aggiornamenti al provider JCE.

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Aggiorna la versione di log4j alla 2.17.1.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza.

Versione 3.4.3

La versione 3.4.3 aggiunge aggiornamenti al provider JCE.

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Aggiorna la versione di log4j alla 2.17.0.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza.

Versione 3.4.2

La versione 3.4.2 aggiunge aggiornamenti al provider JCE.

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Aggiorna la versione di log4j alla 2.16.0.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza.

Versione 3.4.1

La versione 3.4.1 aggiunge aggiornamenti al provider JCE.

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Aggiorna la versione di log4j alla 2.15.0.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza.

Versione 3.4.0

La versione 3.4.0 aggiunge aggiornamenti a tutti i componenti.

AWS CloudHSM Software client

- Maggiore stabilità e correzione dei bug.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Maggiore stabilità e correzione dei bug.

Windows (provider CNG e KSP)

- Maggiore stabilità e correzione dei bug.

Versione 3.3.2

La versione 3.3.2 risolve un problema con lo script client_info.

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Versione aggiornata per coerenza.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza.

Versione 3.3.1

La versione 3.3.1 aggiunge aggiornamenti a tutti i componenti.

AWS CloudHSM Software client

- Maggiore stabilità e correzione dei bug.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Maggiore stabilità e correzione dei bug.

Windows (provider CNG e KSP)

- Maggiore stabilità e correzione dei bug.

Versione 3.3.0

La versione 3.3.0 aggiunge l'autenticazione a due fattori (2FA) e altri miglioramenti.

AWS CloudHSM Software client

- Aggiunta l'autenticazione 2FA per crypto officer (CO). Per ulteriori informazioni, consulta Gestire Two-Factor l'autenticazione per i responsabili delle criptovalute.
- È stato rimosso il supporto della piattaforma per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto Linux.

- Aggiunta versione standalone di CMU da utilizzare con Client SDK 5 o Client SDK 3. Questa è la stessa versione di CMU inclusa nel client daemon della versione 3.3.0, ora puoi scaricare CMU senza scaricare il client daemon.

Libreria PKCS #11

- Maggiore stabilità e correzione dei bug.
- È stato rimosso il supporto della piattaforma per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto Linux.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza
- È stato rimosso il supporto della piattaforma per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto Linux.

Provider JCE

- Maggiore stabilità e correzione dei bug.
- È stato rimosso il supporto della piattaforma per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto Linux.

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza

Versione 3.2.1

La versione 3.2.1 aggiunge un'analisi di conformità tra l' AWS CloudHSM implementazione della libreria PKCS #11 e lo standard PKCS #11, nuove piattaforme e altri miglioramenti.

AWS CloudHSM Software client

- Aggiunta supporto della piattaforma per CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Per ulteriori informazioni, consulta ???.

Libreria PKCS #11

- Rapporto di conformità della libreria PKCS #11 per client SDK 3.2.1
- Aggiunta supporto della piattaforma per CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Per ulteriori informazioni, consulta ???.

OpenSSL Dynamic Engine

- Nessun supporto per CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Per ulteriori informazioni, vedi ???.

Provider JCE

- Aggiunta supporto della piattaforma per CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Per ulteriori informazioni, consulta ???.

Windows (provider CNG e KSP)

- Maggiore stabilità e correzione dei bug.

Versione 3.2.0

La versione 3.2.0 aggiunge il supporto per il mascheramento delle password e altri miglioramenti.

AWS CloudHSM Software client

- Aggiunta supporto per nascondere la password quando si utilizzano strumenti da riga di comando. Per ulteriori informazioni, vedi LoginHSM e LogoutHSM (cloudhsm_mgmt_util) e LoginHSM e LogouthSM (key_mgmt_util).

Libreria PKCS #11

- Aggiunge il supporto per l'hashing di dati di grandi dimensioni nel software per alcuni meccanismi del PKCS #11 che in precedenza non erano supportati. Per ulteriori informazioni, vedi Meccanismi supportati.

OpenSSL Dynamic Engine

- Maggiore stabilità e correzione dei bug.

Provider JCE

- Versione aggiornata per coerenza.

Windows (provider CNG e KSP)

- Maggiore stabilità e correzione dei bug.

Versione 3.1.2

La versione 3.1.2 aggiunge aggiornamenti al provider JCE.

AWS CloudHSM Software client

- Versione aggiornata per coerenza

Libreria PKCS #11

- Versione aggiornata per coerenza

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza

Provider JCE

- Aggiorna la versione di log4j alla 2.13.3

Windows (provider CNG e KSP)

- Versione aggiornata per coerenza

Versione 3.1.1

AWS CloudHSM Software client

- Versione aggiornata per coerenza.

Libreria PKCS #11

- Versione aggiornata per coerenza.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza.

Provider JCE

- Correzione dei bug e miglioramenti delle prestazioni.

Windows (CNG, KSP)

- Versione aggiornata per coerenza.

Versione 3.1.0

La versione 3.1.0 aggiunge il wrapping delle chiavi AES conforme agli standard.

AWS CloudHSM Software client

- Un nuovo requisito per l'aggiornamento: la tua versione del client deve corrispondere alla versione delle eventuali librerie del software in uso. Per eseguire l'aggiornamento, utilizza un comando batch che aggiorna contemporaneamente il client e tutte le librerie. Per ulteriori informazioni, vedi Aggiornamento del client SDK 3.
- Key_mgmt_util (KMU) include i seguenti aggiornamenti:
  - Sono stati aggiunti due nuovi metodi di wrapping delle chiavi AES - Wrap delle chiavi AES conforme agli standard zero padding e wrap delle chiavi AES senza padding. Per ulteriori informazioni, vedi wrapKey e unwrapKey.
  - Disabilitata la possibilità di specificare IV personalizzato quando si esegue il wrapping di una chiave mediante AES_KEY_WRAP_PAD_PKCS5. Per ulteriori informazioni, vedi wrapping delle chiavi AES.

Libreria PKCS #11

- Sono stati aggiunti due nuovi metodi di wrapping delle chiavi AES - Wrap delle chiavi AES conforme agli standard zero padding e wrap delle chiavi AES senza padding. Per ulteriori informazioni, vedi wrapping delle chiavi AES.
- È possibile configurare la lunghezza minima per le RSA-PSS firme. Per informazioni su come utilizzare questa funzionalità, consulta la sezione Lunghezza minima configurabile per le RSA-PSS firme su. GitHub

OpenSSL Dynamic Engine

- MODIFICA IMPORTANTE: le suite di crittografia TLS 1.0 e 1.2 con SHA1 non sono disponibili in OpenSSL Engine 3.1.0. Questo problema verrà risolto a breve.
- Se si intende installare la libreria OpenSSL Dynamic Engine su RHEL 6 o CentOS 6, vedi i Problemi noti relativi alla versione OpenSSL predefinita installata su tali sistemi operativi.
- Maggiore stabilità e correzione dei bug

Provider JCE

- MODIFICA IMPORTANTE: per risolvere un problema con la conformità Java Cryptography Extension (JCE), le operazioni di wrapping e unwrapping AES utilizzano ora correttamente l'algoritmo AESWrap anziché l'algoritmo AES. Questo significa `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE` non ha più successo per meccanismi AES/ECB . AES/CBC

  Per eseguire l'aggiornamento alla versione client 3.1.0, è necessario aggiornare il codice. Se disponi di chiavi sottoposte a wrapping, devi prestare particolare attenzione al meccanismo utilizzato per annullare il wrapping e al modo in cui le impostazioni predefinite di IV vengono modificate. Se hai inserito le chiavi con la versione client 3.0.0 o precedente, nella versione 3.1.1 devi usarle per AESWrap/ECB/PKCS5Padding scartare le chiavi esistenti. Per ulteriori informazioni, vedi wrapping delle chiavi AES.
- È possibile elencare più chiavi con la stessa etichetta nel provider JCE. Per informazioni su come scorrere tutte le chiavi disponibili, consulta Trova tutte le chiavi attive. GitHub
- È possibile impostare valori più restrittivi per gli attributi durante la creazione delle chiavi, inclusa la specifica di etichette diverse per chiavi pubbliche e private. Per ulteriori informazioni, vedi Attributi Java supportati.

Windows (CNG, KSP)

- Maggiore stabilità e correzione dei bug.

AWS CloudHSM Versioni Client SDK a fine ciclo di vita

Le seguenti versioni AWS CloudHSM client hanno raggiunto la fine del supporto. Queste versioni AWS CloudHSM client non sono più compatibili con il servizio e non riceveranno aggiornamenti. Per preservare la sicurezza dell'applicazione, AWS CloudHSM potrebbe rifiutare le connessioni delle versioni che hanno raggiunto la fine del supporto.

- Le versioni SDK 3.4.4 e precedenti hanno raggiunto la fine del supporto.
- Le versioni SDK 5.8.0 e precedenti hanno raggiunto la fine del supporto.

Cronologia dei documenti

Questo argomento descrive gli aggiornamenti importanti alla Guida per l'utente per AWS CloudHSM .

Argomenti

- Aggiornamenti recenti
- Aggiornamenti precedenti

Aggiornamenti recenti

La tabella seguente descrive le modifiche importanti apportate a questa documentazione a partire da aprile 2018. Oltre alle modifiche principali elencate qui, aggiorniamo la documentazione di frequente per migliorare le descrizioni e gli esempi e per dar spazio ai feedback inviatici. Per ricevere notifiche sulle modifiche rilevanti, utilizza il collegamento in alto a destra per iscriverti al feed RSS.

Per i dettagli sulle nuove versioni, vedi Download per AWS CloudHSM Client SDK

Modifica	Descrizione	Data
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.18.0.	18 agosto 2026
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.17.2.	22 maggio 2026
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.17.1.	19 marzo 2026
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.17.0.	6 gennaio 2026
Aggiunti i link per il download di OpenSSL Provider	Sono stati aggiunti i collegamenti per il download di OpenSSL Provider alla AWS CloudHSM versione client 5.17.	6 gennaio 2026

Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.16.2.	16 settembre 2025
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.16.1.	25 giugno 2025
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.16.0.	1 maggio 2025
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.15.0.	3 febbraio 2025
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.14.0.	26 novembre 2024
Nuovo tipo HSM e modalità cluster	È stato aggiunto il supporto per la creazione (hsm2m.medium) in modalità FIPS di cluster.	20 agosto 2024
Aggiunta nuova versione	AWS CloudHSM Rilasciata la versione client 5.13.0.	13 agosto 2024
Nuovo tipo HSM e modalità cluster	Lancio di un nuovo tipo di HSM (hsm2m.medium) e di una nuova modalità cluster (non FIPS).	10 giugno 2024
Aggiunta nuova versione	AWS CloudHSM Rilasciata la versione client 5.12.0.	20 marzo 2024
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.11.0.	17 gennaio 2024
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.10.0.	28 luglio 2023
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.9.0.	23 maggio 2023

Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.8.0.	16 marzo 2023
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.7.0.	16 novembre 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.6.0.	1 settembre 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.5.0.	13 maggio 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.4.2.	18 marzo 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.4.1.	10 febbraio 2022
Aggiunta nuova versione	Rilasciata la versione 5.4.0 del provider AWS CloudHSM JCE per piattaforme Windows.	1 febbraio 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.4.0, che aggiunge il supporto iniziale per il provider JCE per tutte le piattaforme Linux.	28 gennaio 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.3.0.	3 gennaio 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.4.4.	3 gennaio 2022
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.4.3.	20 dicembre 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.4.2.	15 dicembre 2021

Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.4.1.	10 dicembre 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.2.1.	4 ottobre 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.4.0.	25 agosto 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.2.0.	3 agosto 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.3.2.	2 luglio 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.1.0.	1 giugno 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.3.1.	26 Aprile 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.0.1.	8 aprile 2021
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.0.0.	12 marzo 2021
Aggiunti nuovi contenuti	È stata aggiunta l'interfaccia VPC Endpoint, una funzionalità di AWS che consente di creare una connessione privata tra il VPC AWS CloudHSM senza richiedere l'accesso su Internet o tramite un dispositivo NAT, una connessione VPN o una connessione. Direct Connect	10 febbraio 2021

Aggiunta nuova versione	Rilasciata la versione client 3.3.0 AWS CloudHSM .	3 febbraio 2021
Aggiunti nuovi contenuti	È stata aggiunta la conservazione gestita dei backup, una funzionalità che elimina automaticamente i vecchi backup.	18 novembre 2020
Aggiunti nuovi contenuti	È stato aggiunto un rapporto di conformità che analizza l'implementazione AWS CloudHSM Client SDK 3.2.1 della libreria PKCS #11 con lo standard PKCS #11.	29 ottobre 2020
Aggiunta nuova versione	Rilasciata la versione client 3.2.1. AWS CloudHSM	8 ottobre 2020
Aggiunti nuovi contenuti	È stata aggiunta la documentazione che descrive le impostazioni di sincronizzazione delle chiavi in AWS CloudHSM.	1 settembre 2020
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.2.0.	31 agosto 2020
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.1.2.	30 luglio 2020
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.1.1.	3 giugno 2020
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.1.0.	21 maggio 2020
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.0.1.	20 aprile 2020

Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.0.0 per la piattaforma Windows Server.	30 ottobre 2019
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 3.0.0 per tutte le piattaforme, ad eccezione di Windows.	22 ottobre 2019
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 2.0.4.	26 agosto 2019
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 2.0.3.	13 maggio 2019
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 2.0.1.	21 marzo 2019
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 2.0.0.	6 febbraio 2019
Aggiunta del supporto regionale	È stato aggiunto il AWS CloudHSM supporto per le regioni UE (Stoccolma) e AWS GovCloud (US-East).	19 dicembre 2018
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 1.1.2 per Windows.	20 novembre 2018
Problemi noti aggiornati	Nuovi contenuti sono stati aggiunti alla guida per la risoluzione dei problemi.	8 novembre 2018
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 1.1.2 per piattaforme Linux.	8 novembre 2018

Aggiunta del supporto regionale	È stato aggiunto il AWS CloudHSM supporto per le regioni UE (Parigi) e Asia Pacifico (Seoul).	24 ottobre 2018
Aggiunti nuovi contenuti	È stata aggiunta la possibili tà di eliminare e ripristinare i AWS CloudHSM backup.	10 settembre 2018
Aggiunti nuovi contenuti	È stata aggiunta la distribuz ione automatica dei log di controllo ad Amazon CloudWatch Logs.	13 agosto 2018
Aggiunti nuovi contenuti	È stata aggiunta la possibili tà di copiare un backup AWS CloudHSM del cluster in più regioni.	30 luglio 2018
Aggiunta del supporto regionale	È stato aggiunto il AWS CloudHSM supporto per la regione UE (Londra).	13 giugno 2018
Aggiunti nuovi contenuti	È stato aggiunto AWS CloudHSM il supporto per client e librerie per Amazon Linux 2, Red Hat Enterpris e Linux (RHEL) 6, Red Hat Enterprise Linux (RHEL) 7, CentOS 6, CentOS 7 e Ubuntu 16.04 LTS.	10 maggio 2018
Aggiunta nuova versione	AWS CloudHSM È stato aggiunto un client Windows.	30 Aprile 2018

Aggiornamenti precedenti

La tabella seguente descrive le modifiche importanti rispetto a quelle AWS CloudHSM precedenti al 2018.

Modifica	Descrizione	Data
Nuovo contenuto	Aggiunta dell'autenticazione del quorum (controllo accessi "M of N") per crypto officer (CO). Per ulteriori informazioni, vedi Utilizzo di CloudHSM Management Utility (CMU) per gestire l'autenticazione del quorum (controllo dell'accesso "M of N").	9 novembre 2017
Aggiorna	Aggiunta di documentazione sull'utilizzo dello strumento a riga di comando key_mgmt_util. Per ulteriori informazioni, vedi Riferimento per i comandi della AWS CloudHSM Key Management Utility.	9 novembre 2017
Nuovo contenuto	Aggiunta di Oracle Transparent Data Encryption. Per ulteriori informazioni, vedi Oracle Database Encryption.	25 ottobre 2017
Nuovo contenuto	Aggiunta dell'offload SSL. Per ulteriori informazioni, vediOffload SSL/TLS.	12 ottobre 2017
Nuova guida	Questa versione introduce AWS CloudHSM	14 agosto 2017

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.