aws

Guida per l'utente

AWS CloudHSM

AWS CloudHSM: Guida per l'utente

Table of Contents

Cos’è AWS CloudHSM?

AWS CloudHSM unisce i vantaggi del cloud AWS con la sicurezza dei moduli di sicurezza hardware (HSM). 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 sugli HSM ad alta disponibilità presenti nel cloud AWS, usufruisci dell'accesso a bassa latenza e di una radice di affidabilità sicura che automatizza la gestione degli HSM (inclusi backup, provisioning, configurazione e manutenzione).

AWS CloudHSM offre ai clienti una serie di vantaggi:

Gli HSM sono convalidati secondo lo standard FIPS 140-2 di livello 3

AWS CloudHSM utilizza HSM generici conformi agli standard, single-tenant e convalidati secondo lo standard FIPS 140-2 di livello 3. Offrono maggiore flessibilità rispetto ai servizi AWS completamente gestiti che hanno algoritmi e lunghezze di chiave predeterminati per l'applicazione.

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 assicura il pieno controllo degli algoritmi e delle chiavi in uso. 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, gli SDK AWS CloudHSM ti assicurano il pieno controllo sullo sviluppo delle applicazioni, sul linguaggio applicativo, sul threading e sulla posizione fisica delle applicazioni.

Esegui la migrazione dei carichi di lavoro di crittografia nel cloud

I clienti che eseguono la migrazione di un'infrastruttura a chiave pubblica che utilizza Public Key Cryptography Standards #11 (PKCS #11), Java Cryptographic Extension (JCE), API di crittografia: Next Generation (CNG) o il provider di archiviazione delle chiavi (KSP) possono eseguire la migrazione in AWS CloudHSM apportando meno modifiche alla propria applicazione.

Accesso a cluster FIPS e non FIPS

Per ulteriori informazioni sulle operazioni che è possibile eseguire con AWS CloudHSM, consulta i seguenti argomenti. Quando sei pronto a utilizzare AWS CloudHSM, consulta Nozioni di base.

> **Note**
>
> Se desideri un servizio gestito per la creazione e il controllo delle chiavi di crittografia senza ricorrere al tuo HSM, ti consigliamo di utilizzare AWS Key Management Service.
> Se stai cercando un servizio elastico che gestisca gli HSM di pagamento e le chiavi per le applicazioni di elaborazione dei pagamenti nel cloud, valuta la possibilità di utilizzare la crittografia dei pagamenti AWS.

Indice

Casi d'uso AWS CloudHSM

AWS CloudHSM può essere utilizzato per raggiungere diversi obiettivi. Il contenuto di questo argomento fornisce una panoramica di ciò che è possibile fare con AWS CloudHSM.

Raggiungi la conformità alle normative

Le aziende che devono allinearsi agli standard di sicurezza aziendali possono utilizzare AWS CloudHSM per la gestione di chiavi private che proteggono dati altamente riservati. Gli HSM forniti da AWS CloudHSM sono certificati secondo lo standard FIPS 140-2 di livello 3 e sono conformi a PCI DSS. Inoltre, AWS CloudHSM è conforme agli standard PCI PIN e PCI-3DS. Per ulteriori informazioni, consulta Conformità.

Crittografia e decrittografia dei dati

Utilizza AWS CloudHSM per gestire le chiavi private che proteggono i dati altamente riservati, la crittografia in transito e la crittografia dei dati inattivi. Inoltre, AWS CloudHSM offre un'integrazione conforme agli standard con più SDK di crittografia.

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). Utilizza AWS CloudHSM per creare coppie di chiavi pubbliche e private asimmetriche progettate specificamente per questo scopo.

Autenticazione di messaggi utilizzando HMAC e CMAC

Nella crittografia, i codici di autenticazione dei messaggi cifrati (CMAC) e i codici di autenticazione dei messaggi basati su hash (HMAC) 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 HMAC e CMAC.

Sfrutta i vantaggi di AWS CloudHSM e AWS Key Management Service

I clienti possono combinare AWS CloudHSM e AWS KMS per archiviare il materiale delle chiavi in un ambiente single-tenant certificato secondo lo standard FIPS 140-2 di livello 3, ottenendo al contempo i principali vantaggi di gestione, scalabilità e integrazione cloud offerti da AWS KMS. Per ulteriori informazioni al riguardo, consulta gli archivi delle chiavi AWS CloudHSM nella Guida per gli sviluppatori di AWS Key Management Service.

Offload dell'elaborazione SSL/TLS per i server Web

Per inviare dati in modo sicuro su Internet, i server Web utilizzano coppie di chiavi pubbliche-private e un certificato con chiave pubblica SSL/TLS per stabilire sessioni HTTPS. Questo processo comporta molte operazioni di calcolo per i server Web, ma è possibile ridurre l'onere di calcolo fornendo al contempo una maggiore sicurezza trasferendone una parte al cluster AWS CloudHSM. Per informazioni sulla configurazione dell'offload SSL/TLS con AWS CloudHSM, consulta Offload 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 archiviando la chiave di crittografia principale TDE negli HSM in AWS

CloudHSM. Per informazioni sulla configurazione di Oracle TDE con AWS CloudHSM, consulta 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 archiviare tali chiavi private nel cluster AWS CloudHSM e quindi utilizzare gli HSM 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. È possibile utilizzare AWS CloudHSM per generare numeri casuali in modo sicuro negli HSM sotto il tuo controllo e che sono visibili solo a te.

Funzionamento di AWS CloudHSM

Questo argomento fornisce una panoramica dei concetti e dell'architettura di base che si utilizzano per crittografare in modo sicuro i dati ed eseguire operazioni crittografiche negli HSM. AWS CloudHSM opera nel tuo cloud privato virtuale (VPC) Amazon. Prima di poter utilizzare AWS CloudHSM, devi innanzitutto creare un cluster, aggiungervi HSM, creare utenti e chiavi e quindi utilizzare SDK del client per l'integrazione degli HSM con l'applicazione. Una volta completata questa operazione, si utilizzano i log dell'SDK del client AWS CloudTrail, log di audit e Amazon CloudWatch per monitorare AWS CloudHSM.

Scopri i concetti di base di AWS CloudHSM e il modo in cui collaborano per proteggere i dati.

Argomenti

Cluster AWS CloudHSM

Far funzionare insieme i singoli HSM in un cluster sincronizzato, ridondante e ad alta disponibilità può essere difficile, ma AWS CloudHSM si occupa del lavoro complicato fornendo moduli di sicurezza hardware (HSM) in cluster. Un cluster è una raccolta di singoli HSM che AWS CloudHSM mantiene sincronizzati. Quando si esegue un'attività o operazione su un HSM in un cluster, gli altri HSM nel cluster vengono aggiornati automaticamente. Per soddisfare gli obiettivi di disponibilità, durabilità e scalabilità, è necessario impostare il numero di HSM nel cluster in più zone di disponibilità.

È possibile creare un cluster che abbia da 1 a 28 HSM (il limite predefinito è 6 HSM per account AWS per regione AWS). È possibile posizionare gli HSM in zone di disponibilità differenti in una regione AWS. L'aggiunta di più HSM a un cluster fornisce prestazioni maggiori. La diffusione di cluster tra le zone di disponibilità fornisce ridondanza e disponibilità elevate.

Per ulteriori informazioni sui cluster, consulta Gestione dei cluster AWS CloudHSM.

Per creare un cluster, consulta Nozioni di base.

Utenti HSM

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

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. 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 HSM

AWS CloudHSM consente di generare, archiviare e gestire in modo sicuro le chiavi di crittografia in HSM single-tenant che si trovano 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.
- Utilizzare funzioni hash per elaborare i digest dei messaggi e i codici di autenticazione dei messaggi basati su hash (HMAC).
- Eseguire il wrapping di altre chiavi e proteggerle.
- Accedere a dati casuali protetti da crittografia.

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 consentirti di personalizzare le tue soluzioni, AWS CloudHSM offre molti tipi di chiavi e algoritmi tra cui scegliere, e gli algoritmi supportano una vasta gamma di dimensioni di chiavi. Per ulteriori informazioni, consulta le pagine relative agli attributi e ai meccanismi di ogni SDK del client AWS CloudHSM.

Come gestire le chiavi

Le chiavi AWS CloudHSM sono gestite tramite SDK e strumenti a riga di comando. Per ulteriori informazioni su come utilizzare questi strumenti per gestire le chiavi, consulta le pagine Gestione delle chiavi in AWS CloudHSM e Best practice per AWS CloudHSM.

Chi possiede le chiavi

In AWS CloudHSM, il crypto user (CU) che crea la chiave ne diventa proprietario. Il proprietario può utilizzare i comandi key share e key unshare per condividere e annullare la condivisione della chiave con altri CU. Per ulteriori informazioni, consulta Utilizzare la CLI di CloudHSM per condividere e annullare la condivisione delle chiavi.

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

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

SDK del client

Quando utilizzi AWS CloudHSM, esegui operazioni crittografiche con i Software Development Kit (SDK) del client AWS CloudHSM. AWS CloudHSM Gli SDK client includono:

- Public Key Cryptography Standard #11 (PKCS #11)
- Provider JCE
- OpenSSL Dynamic Engine
- API di crittografia: Next Generation (CNG) e provider di archiviazione delle chiavi (KSP) per Microsoft Windows

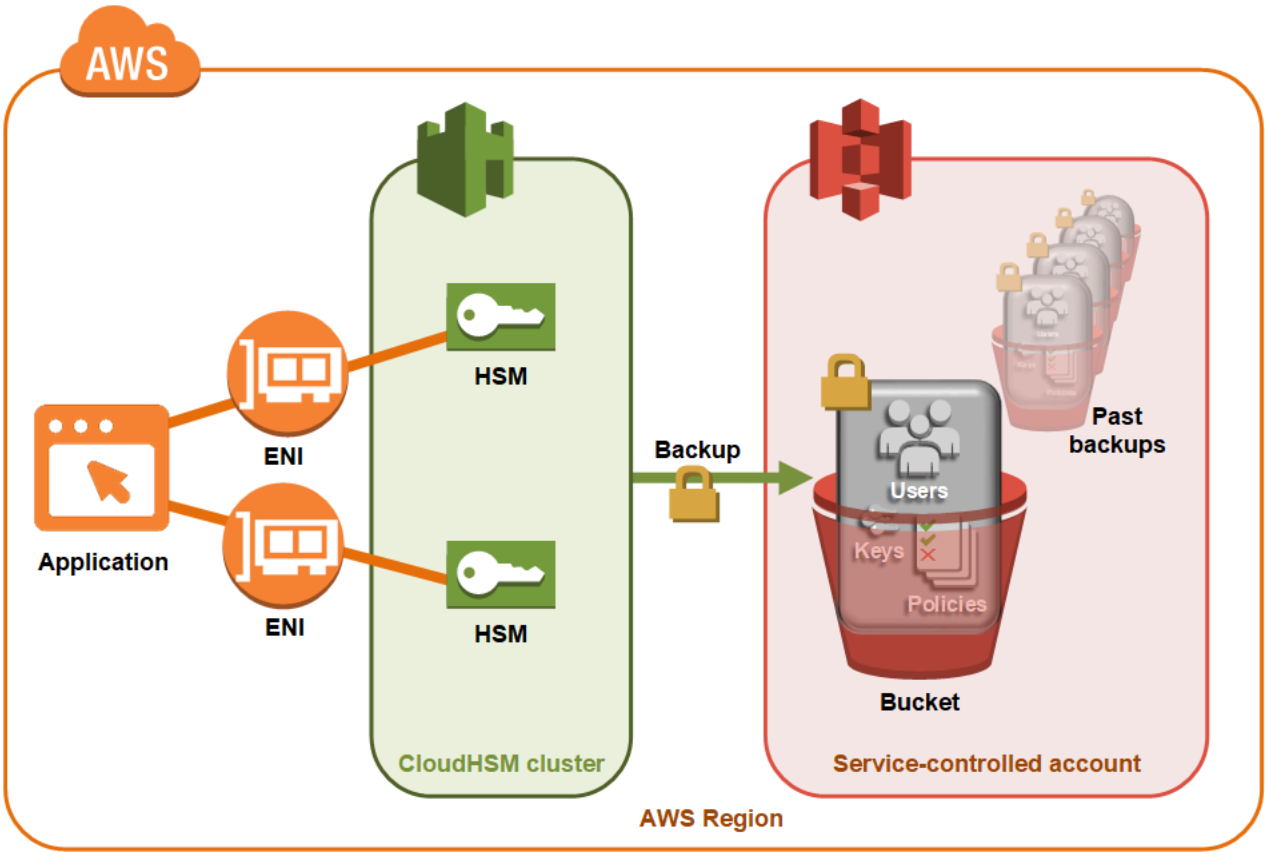
Puoi utilizzare uno o tutti questi SDK nel cluster AWS CloudHSM. Scrivi il codice dell'applicazione per utilizzare questi SDK per eseguire operazioni crittografiche negli HSM.

Gli strumenti a riga di comando e di utilità sono necessari non solo per utilizzare gli SDK, ma anche per configurare le credenziali, le policy e le impostazioni dell'applicazione. Per ulteriori informazioni, consulta Strumenti a riga di comando AWS CloudHSM.

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 nd-to-end crittografia E.

Backup del cluster AWS CloudHSM

AWS CloudHSM esegue backup periodici degli utenti, delle chiavi e delle policy 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 Gestione dei backup.

Sicurezza

Quando AWS CloudHSM effettua un backup dall'HSM, l'HSM effettua la crittografia di tutti i suoi dati prima dell'invio a AWS CloudHSM. I dati non lasciano mai l'HSM come testo normale. Inoltre, i backup non possono essere decrittografati da 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

Per informazioni sulle regioni supportate per AWS CloudHSM, consultare Regioni ed endpoint AWS CloudHSM nella Riferimenti generali di AWS o nella Tabella delle regioni.

AWS CloudHSM potrebbe non essere disponibile in tutte le zone di disponibilità di una determinata regione. Tuttavia, questo non dovrebbe avere ripercussioni sulle prestazioni, in quanto AWS CloudHSM bilancia automaticamente il carico su tutti gli HSM di un cluster.

Come per la maggior parte delle risorse AWS, cluster e HSM 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 Nozioni di base su AWS CloudHSM.

Ai fini del ripristino di emergenza, AWS CloudHSM consente di copiare i backup del cluster AWS CloudHSM da una regione all'altra. Per ulteriori informazioni, consulta Backup del cluster AWS CloudHSM.

Prezzi

AWS CloudHSM offre la possibilità di pagare a ore senza impegni a lungo termine o pagamenti anticipati. Per ulteriori informazioni, consulta la pagina relativa ai prezzi di AWS CloudHSM sul sito Web di AWS.

Nozioni di base su AWS CloudHSM

Di seguito viene illustrato come creare, inizializzare e attivare un cluster 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.

Indice

Creazione di un gruppo di amministratori IAM;

Come best practice, non utilizzare Utente root dell'account AWS per interagire con AWS, incluso AWS CloudHSM. Usa piuttosto 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 la AdministratorAccesspolicy ad esso. Quindi, crea un nuovo utente amministratore e aggiungilo al gruppo. Aggiungi altri utenti al gruppo, se necessario. Ogni utente aggiunto eredita la AdministratorAccesspolitica dal gruppo.

Un'altra best practice è quella di creare un gruppo di amministratori AWS CloudHSM che abbia solo le autorizzazioni necessarie per eseguire 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 sezione Politiche gestite dal cliente per AWS CloudHSM di seguito contiene la policy da collegare al gruppo di amministratori AWS CloudHSM.

AWS CloudHSM definisce un ruolo legato ai servizi per il tuo account AWS. Il ruolo collegato ai servizi definisce attualmente le autorizzazioni che consentono all'account di registrare eventi AWS CloudHSM. Il ruolo può essere creato automaticamente da AWS CloudHSM o manualmente da te. Non è possibile modificare il ruolo, ma è possibile eliminarlo. Per ulteriori informazioni, vedi Ruoli collegati ai servizi per AWS CloudHSM.

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

Inizia creando un utente IAM; insieme a un gruppo amministratore per l'utente.

Registrarsi per creare un Account AWS

Se non disponi di un Account AWS, completa la procedura seguente per crearne uno.

Per registrarsi a un Account AWS

1. Apri la pagina all'indirizzo https://portal.aws.amazon.com/billing/signup.
2. Segui le istruzioni online.

   Nel corso della procedura di registrazione riceverai una telefonata, durante la quale sarà necessario inserire un codice di verifica attraverso la tastiera del telefono.

   Durante la registrazione di un Account AWS, viene creato un Utente root dell'account AWS. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWSnell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente amministrativo e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

Al termine del processo di registrazione, riceverai un'e-mail di conferma da AWS. È possibile visualizzare l'attività corrente dell'account e gestire l'account in qualsiasi momento accedendo all'indirizzo https://aws.amazon.com/ e selezionando Il mio account.

Creazione di un utente amministratore

Dopo aver effettuato la registrazione di un Account AWS, proteggi Utente root dell'account AWS, abilita AWS IAM Identity Center e crea un utente amministratore in modo da non utilizzare l'utente root per le attività quotidiane.

Protezione dell'Utente root dell'account AWS

1. Accedi alla AWS Management Console come proprietario dell'account scegliendo Utente root e immettendo l'indirizzo email del Account AWS. Nella pagina successiva, inserisci la password.

   Per informazioni sull'accesso utilizzando un utente root, consulta la pagina Accesso come utente root della Guida per l'utente di Accedi ad AWS.
2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

   Per ricevere istruzioni, consulta Abilitazione di un dispositivo MFA virtuale per l'utente root dell'Account AWS (console) nella Guida per l'utente IAM.

Creazione di un utente amministratore

1. Abilita Centro identità IAM.

   Per istruzioni, consulta Abilitazione di AWS IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center.
2. In Centro identità IAM, assegna l'accesso amministrativo a un utente amministratore.

   Per un tutorial sull'utilizzo di IAM Identity Center directory come origine di identità, consulta Configure user access with the default IAM Identity Center directory nella Guida per l'utente di AWS IAM Identity Center.

Accesso come utente amministratore

- Per accedere con l'utente IAM Identity Center, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente IAM Identity Center.

  Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta Accedere al portale di accesso AWS nella Guida per l'utente Accedi ad AWS.

Per le policy di esempio di AWS CloudHSM che è possibile collegare al gruppo di utenti IAM, vedi Gestione delle identità e degli accessi per AWS CloudHSM.

Crea un cloud privato virtuale (VPC)

Se non disponi ancora di un cloud privato virtuale (VPC), segui i passaggi descritti in questo argomento per crearne uno.

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

Per creare un VPC

1. Apri alla console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Sulla barra di navigazione, usa lo strumento di selezione della regione per scegliere una delle regioni AWS in cui l'AWS CloudHSM è attualmente supportato.
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. Lascia tutte le altre opzioni impostate sui valori predefiniti.
7. Seleziona Crea VPC.
8. Dopo aver creato il VPC, seleziona Visualizza VPC per visualizzare il VPC appena creato.

Creazione di un cluster

Un cluster è una raccolta di singoli moduli HSM. AWS CloudHSM sincronizza i moduli HSM in ogni cluster affinché funzionino come unità logica.

Quando crei un cluster, AWS CloudHSM crea un gruppo di sicurezza del cluster a tuo nome. Questo gruppo di sicurezza controlla l'accesso di rete ai moduli HSM del cluster. Consente connessioni in entrata solo da istanze Amazon Elastic Compute Cloud (Amazon EC2) presenti nel 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.

> **Important**
>
> Quando crei un cluster, AWS CloudHSM crea un ruolo legato ai servizi denominato AWSServiceRoleForCloudHSM. Se AWS CloudHSM non è in grado di creare il ruolo o questo non esiste ancora, potresti non essere in grado di creare un cluster. Per ulteriori informazioni, vedi Risoluzione di problemi di creazione dei cluster. Per ulteriori informazioni sui ruoli legati al servizio, vedi Ruoli collegati ai servizi per AWS CloudHSM.

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

Per creare un cluster (console)

1. Apri la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
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 Create cluster (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).
   b. In Zone di disponibilità (AZ), a fianco di ciascuna zona di disponibilità, scegli la sottorete privata creata.

   > **Note**
   >
   > Anche se AWS CloudHSM non è supportato in una determinata zona di disponibilità, non dovrebbe verificarsi una riduzione delle prestazioni, in quanto AWS CloudHSM bilancia automaticamente il carico su tutti i moduli HSM di un cluster. Per informazioni sul supporto per le zone di disponibilità per AWS CloudHSM, vedi AWS CloudHSMregioni ed endpoint nella Riferimenti generali di AWS .
5. Seleziona Successivo.
6. Specifica per quanto tempo il servizio deve conservare i backup.

> **Note**
>
> 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 Configurazione della politica di conservazione dei backup .

7. Seleziona 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.

Per creare un cluster (AWS CLI)

- Al prompt dei comandi, esegui il comando create-cluster. Specificare il tipo di istanza HSM, il periodo di conservazione del backup e gli ID delle sottoreti in cui si intende creare i moduli HSM. Utilizza l'ID delle sottoreti private create. Specifica una sola sottorete per ogni zona di disponibilità.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm1.medium \
      --backup-retention-policy Type=DAYS,Value=<number of days> \
      --subnet-ids <subnet ID>

{
    "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",
        "Certificates": {},
        "State": "CREATE_IN_PROGRESS",
        "Hsms": [],
        "ClusterId": "cluster-igklspoyj5v",
        "CreateTimestamp": 1502423370.069
    }
}
```

Per creare un cluster (API AWS CloudHSM)

- Inviare una richiesta CreateCluster. Specificare il tipo di istanza HSM, il periodo di conservazione del backup e gli ID delle sottoreti in cui si intende creare i moduli HSM. Utilizza l'ID delle sottoreti private create. Specifica una sola sottorete per ogni zona di disponibilità.

Se non riesci a creare un cluster, il problema potrebbe essere correlato a problemi con i ruoli collegati ai servizi AWS CloudHSM. Per assistenza nella risoluzione del problema, vedi Risoluzione di problemi di creazione dei cluster.

Rivedi gruppo di sicurezza del cluster

Quando si crea un cluster, AWS CloudHSM crea un gruppo di sicurezza con il nome `cloudhsm-cluster-`*`clusterID`*`-sg`. 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 per comunicare con i moduli HSM del cluster.

> **Warning**
>
> - Non eliminare o modificare la regola TCP preconfigurata, che viene inserita in tale gruppo di sicurezza. Questa regola è in grado di evitare problemi di connettività e accessi non autorizzati ai moduli HSM.
> - Il gruppo di sicurezza del cluster impedisce gli accessi non autorizzati ai moduli HSM. Chiunque possa accedere alle istanze del gruppo di sicurezza può accedere ai tuoi moduli HSM. La maggior parte delle operazioni richiede un utente per l'accesso all'HSM. È tuttavia possibile azzerare i moduli HSM senza l'autenticazione, che distrugge il materiale di chiavi, certificati e 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.

Nella fase successiva, è possibile avviare un' istanza Amazon EC2 e collegarla ai moduli HSM collegando il gruppo di sicurezza del cluster.

Avviare un'istanza del client Amazon EC2

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
- AWS 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 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. Aprire la console 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 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. Scegli Seleziona un gruppo di sicurezza esistente.
12. In Gruppi di sicurezza comuni, seleziona il gruppo di sicurezza predefinito dal menu a tendina.
13. In Configurazione archiviazione, utilizza i menu a tendina per scegliere una configurazione di archiviazione.
14. 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**
>
> L'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 di Amazon EC2

Quando viene avviata un'istanza Amazon EC2, lo hai associato a un gruppo di sicurezza Amazon VPC di default. 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.

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. Aprire la console 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. Selezionare Salva.

Connect l'istanza Amazon EC2 al cluster AWS CloudHSM

Devi collegare il gruppo di sicurezza del cluster all'istanza EC2 in modo che l'istanza EC2 possa comunicare con i moduli HSM nel tuo 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 la Console 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

Dopo avere creato un cluster, puoi creare un 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 console dell'AWS CloudHSM, utilizza la AWS CLI per eseguire il comando describe-clusters o invia una richiesta Descrivi Clusters nell'API dell'AWS CloudHSM. È possibile creare un HSM dalla console dell'AWS CloudHSM, dalla AWS CLI o dall'API AWS CloudHSM.

Per creare un HSM (console)

1. Aprire la console dell'AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
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 l'HSM in fase di creazione.
5. Seleziona Crea.

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. Specificare 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",
        "AvailabilityZone": "us-west-2a",
        "ClusterId": "cluster-igklspoyj5v",
        "EniId": "eni-5d7ade72",
        "SubnetId": "subnet-fd54af9b",
        "State": "CREATE_IN_PROGRESS"
    }
}
```

Per creare un HSM (API dell'AWS CloudHSM)

- 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 dell'identità e dell'autenticità dell'HSM del cluster (facoltativo)

Per inizializzare il cluster, devi firmare una richiesta di firma del certificato generata dal primo 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.

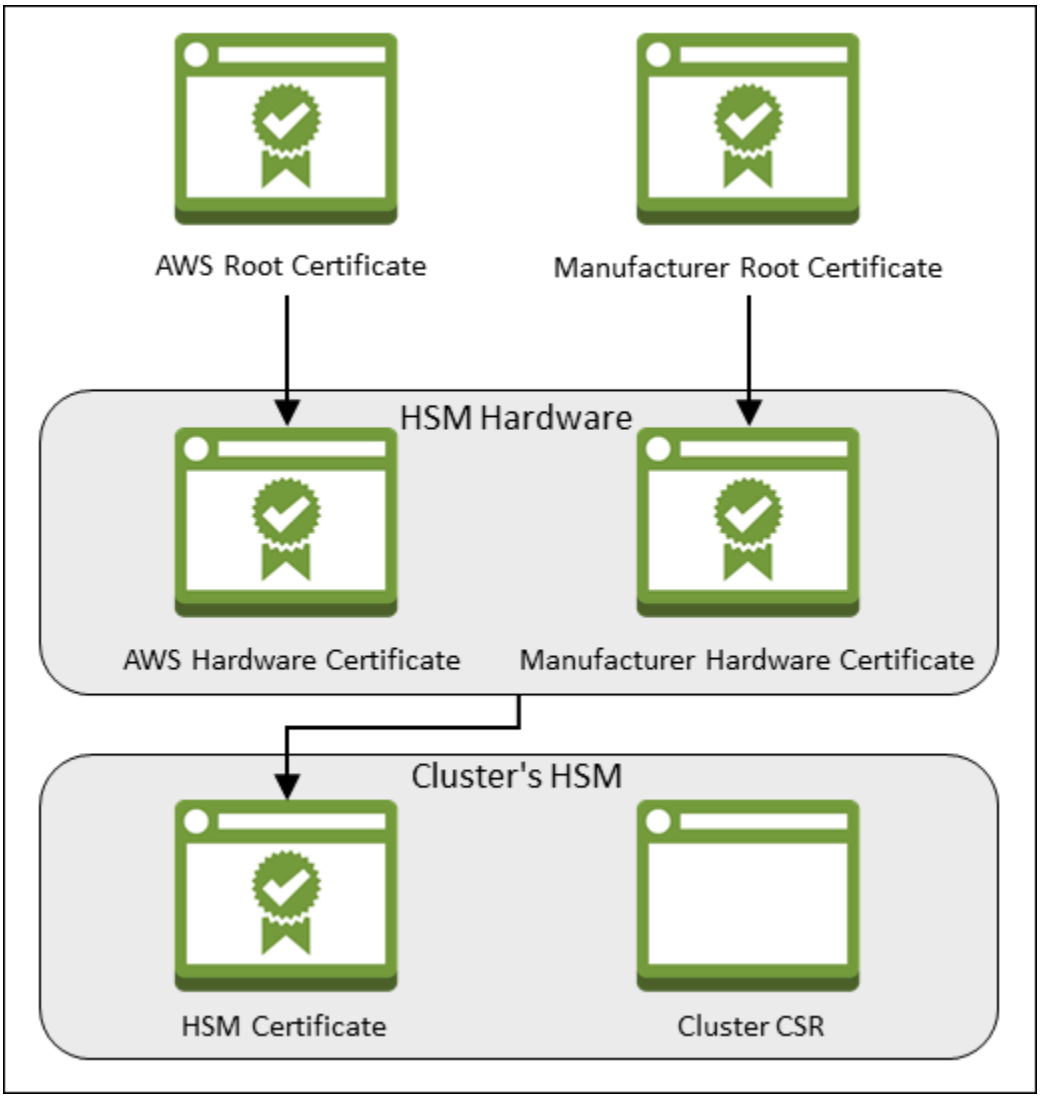
Argomenti

Panoramica

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 da AWS CloudHSM e uno dal provider dell'hardware HSM.
2. Verifica le catene di certificati in questa fase, costruisci due catene di certificati, una per il certificato root AWS CloudHSM e una per il certificato root del produttore. Quindi è possibile verificare il certificato HSM con queste catene di certificati per determinare che AWS CloudHSM e il produttore di hardware attestino entrambi 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.



Certificato root AWS

Questo è il certificato root AWS CloudHSM.

Certificato root del produttore

Questo è il certificato root del produttore dell'hardware.

Certificato hardware AWS

AWS CloudHSM ha creato questo certificato quando l'hardware HSM è stato aggiunto al parco istanze. Questo certificato attesta che AWS CloudHSM possiede l'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'hardware convalidato da FIPS quando si crea il primo HSM nel cluster. Questo certificato attesta che l'hardware HSM ha creato il modulo HSM.

CSR del cluster

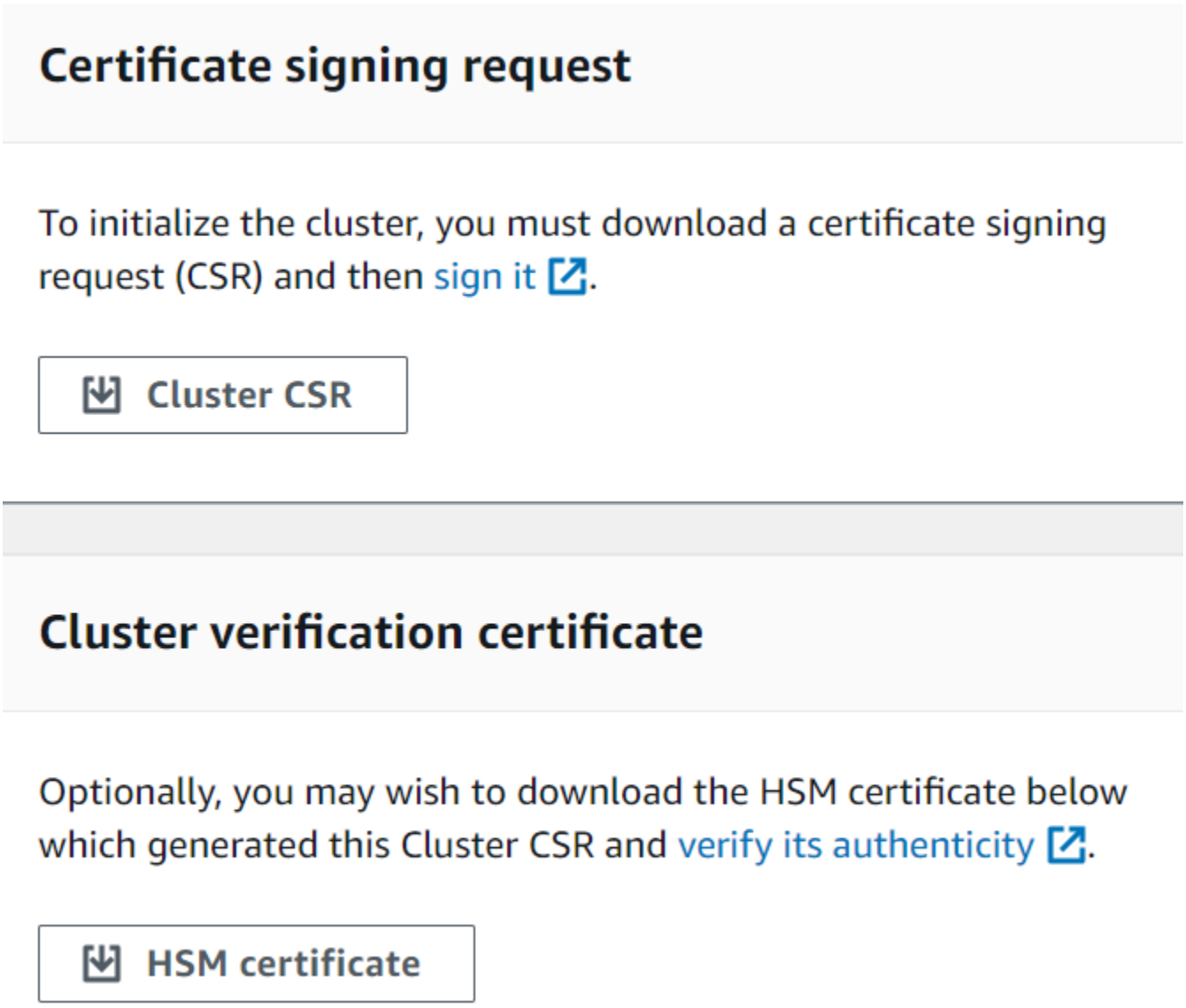
Il primo HSM crea la CSR del cluster. Quando firmi la CSR del cluster, rivendichi il cluster. Quindi, puoi utilizzare la CSR firmata per inizializzare il cluster.

Ottieni i certificati da HSM

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

Per ottenere i certificati CSR e HSM (della console)

1. Apri la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Seleziona il pulsante di opzione accanto all'ID del cluster 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.

Per ottenere i certificati HSM e la CSR (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 *<ID cluster>* 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 *<ID cluster>* 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 hardware di AWS. Sostituisci *<ID cluster>* 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 *<ID cluster>* 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
```

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

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

Ottenere i certificati root

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

Per ottenere i certificati root AWS CloudHSM e produttore

1. Scarica il certificato root AWS CloudHSM: AWS_CloudHSM_Root-G1.zip.
2. Scarica il certificato root del produttore: liquid_security_certificate.zip

   Per scaricare il certificato dalla pagina iniziale, https://www.marvell.com/products/security-solutions/liquid-security-hsm-adapters-and-appliances/liquidsecurity-certificate.html, quindi scegli Scarica certificato.

   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.

Verifica le catene di certificato

In questa fase, è possibile costruire due catene di certificati, una per il certificato root AWS CloudHSM e una per il certificato root 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, per questa fase non è necessaria la AWS CLI e non occorre associare la shell all'account AWS.

Per verificare il certificato HSM con il certificato root AWS CloudHSM

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.

   Utilizzare il comando seguente per creare una catena di certificati che include il certificato hardware del produttore AWS e il certificato root AWS CloudHSM del produttore, in questo ordine. Sostituisci *<ID cluster>* 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 *<ID cluster>* 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 *<ID cluster>* 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 *<ID cluster>* 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
```

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 disporre di OpenSSL, disponibile nella maggior parte delle shell di Linux, ma non è necessaria la AWS CLI per questa fase. 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.

Inizializzazione del cluster

Completa le fasi descritte negli argomenti seguenti per inizializzare il cluster AWS CloudHSM.

> **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

- Ottenere la CSR del cluster
- Firma la CSR
- Inizializzazione del cluster

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. In caso contrario, puoi scaricare subito la CSR tramite la console AWS CloudHSM, AWS Command Line Interface (AWS CLI) o l'API AWS CloudHSM.

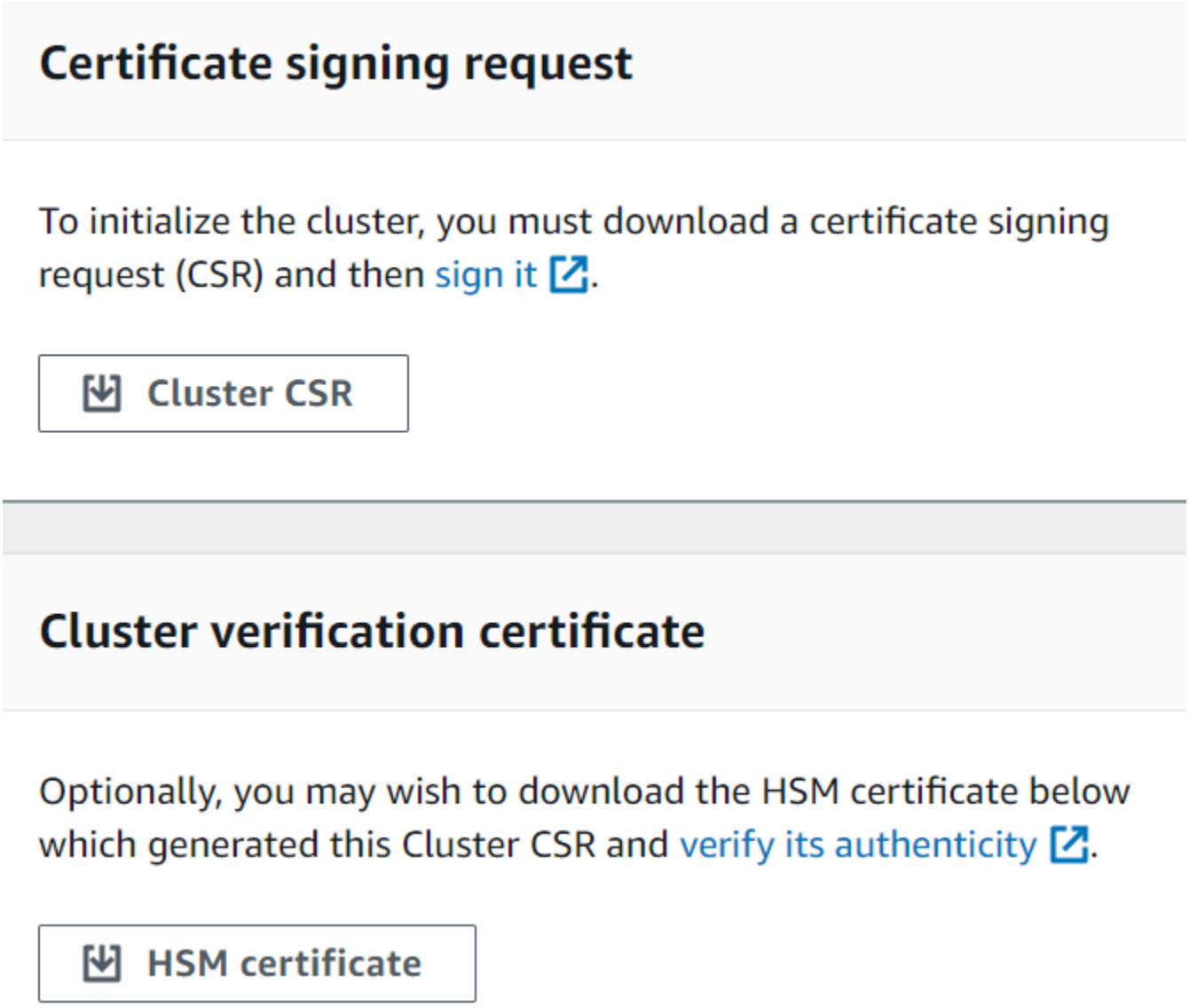
Important

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.

Per ottenere la CSR (console)

1. Apri la console AWS CloudHSM 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.

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. Sostituire l'*<ID del cluster>* con l'ID del cluster che hai creato in precedenza.

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

Per ottenere la CSR (API AWS CloudHSM)

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

Firma la CSR

Al momento, per firmare la CSR per il cluster, devi creare un certificato di firma autofirmato e utilizzarlo. Per questa fase non serve la AWS CLI e non occorre associare la shell all'account AWS. Per firmare la CSR, completa le attività seguenti:

1. Completa la sezione precedente (vediOttenere la CSR del cluster).
2. Crea una chiave privata.
3. Utilizza la chiave privata per creare un certificato di firma.
4. Firma la CSR del cluster.

Crea una chiave privata

> **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 l'accesso all'istanza AWS CloudHSM, deve essere presente il certificato, ma non la chiave privata.

Utilizza il comando seguente per creare una chiave privata. Quando si inizializza un cluster AWS CloudHSM, è necessario utilizzare il certificato RSA 2048 o il certificato RSA 4096.

```
$ openssl genrsa -aes256 -out customerCA.key 2048
Generating RSA private key, 2048 bit long modulus
```

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

Utilizza la chiave privata per creare un 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. Nell'esempio seguente vengono utilizzati OpenSSL e la chiave privata creata nella fase precedente per creare un certificato di firma. 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 customerCA.key -out customerCA.crt
Enter pass phrase for customerCA.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 `customerCA.crt`. Inserisci questo certificato su ogni host da cui ti connetterai al 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 della CSR del cluster

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. Nell'esempio vengono utilizzati la chiave privata e il certificato autofirmato creati nella fase precedente.

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

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

Inizializzazione del cluster

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

Per inizializzare un cluster (console)

1. Apri la console AWS CloudHSM 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, selezionare il file denominato *<cluster ID>*_CustomerHsmCertificate.crt.

b. Accanto a Certificazione, scegli Carica file. Quindi, seleziona il certificato di firma. Se sono state effettuate le fasi riportate nella sezione precedente, seleziona il file denominato customerCA.crt.

c. Scegli Carica e inizializza.

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 certificato di firma. Se sono state effettuate le fasi riportate nella sezione precedente, il certificato di firma è salvato in un file denominato customerCA.crt.

```
$ aws cloudhsmv2 initialize-cluster --cluster-id <cluster ID> \
                                    --signed-cert file://<cluster
 ID>_CustomerHsmCertificate.crt \
                                    --trust-anchor file://customerCA.crt
{
    "State": "INITIALIZE_IN_PROGRESS",
    "StateMessage": "Cluster is initializing. State will change to INITIALIZED upon
 completion."
}
```

Per inizializzare un cluster (API AWS CloudHSM)

- Invia una richiesta InitializeCluster con quanto segue:
  - L'ID del cluster creato in precedenza.
  - Il certificato dell'HSM che hai firmato in precedenza.

- Il certificato di firma.

Installa e configura la CLI di CloudHSM

Per interagire con il modulo HSM nel cluster AWS CloudHSM, è necessaria la CLI di CloudHSM.

Attività

- Installa gli strumenti da AWS CloudHSM riga di comando

Installa gli strumenti da AWS CloudHSM riga di comando

Connect all'istanza client ed esegui i seguenti comandi per scaricare e installare gli strumenti da riga di AWS CloudHSM comando. Per ulteriori informazioni, consulta Avviare un'istanza del client Amazon EC2.

Utilizza i seguenti comandi per scaricare e installare la CLI di CloudHSM.

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
```

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
```

CentOS 7 (7.8+)

CentOS 7 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
```

RHEL7 (7.8+)

RHEL 7 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
```

RHEL8 (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
```

RHEL9 (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
```

Ubuntu 20.04 LTS

Ubuntu 20.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-cli_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u20.04_amd64.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
```

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
```

Windows Server 2019

Per Windows Server 2019 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 IP 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.

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

Attivazione del cluster

Quando attivi un cluster AWS CloudHSM, il suo stato passa 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, devi prima copiare il certificato di emissione nella posizione predefinita per la piattaforma su ogni istanza EC2 che si connette al cluster (il certificato di emissione viene creato quando inizializzi il cluster).

Linux

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

Windows

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

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.

Attivare un cluster

1. Connettersi all'istanza del client avviata in precedenza. Per ulteriori informazioni, vedi Avviare un'istanza del client Amazon EC2. 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

   ```
   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.

Riconfigurazione SSL con un nuovo certificato e una chiave privata (opzionale)

AWS CloudHSM utilizza un certificato SSL per stabilire una connessione a un HSM. Quando installi il client, sono inclusi una chiave predefinita e un certificato SSL. Tuttavia, puoi creare e utilizzare un certificato personale. Dovrai utilizzare il certificato autofirmato (*customerCA.crt*) creato al momento dell'inizializzazione del cluster.

Ad alto livello, si tratta di un processo in due fasi:

1. In primo luogo, devi creare una chiave privata, quindi utilizzare tale chiave per creare una richiesta di firma del certificato (CSR). Utilizza il certificato di emissione, il certificato che hai creato quando hai inizializzato il cluster, per firmare la CSR.
2. Successivamente, utilizza lo strumento di configurazione per copiare la chiave e il certificato nelle directory appropriate.

Crea una chiave, una CSR, quindi firma la CSR

I passaggi sono gli stessi per Client SDK 3 o Client SDK 5.

Per riconfigurare SSL con un nuovo certificato e una chiave privata

1. Crea una chiave privata con il seguente comando OpenSSL:

```
openssl genrsa -out ssl-client.key 2048
Generating RSA private key, 2048 bit long modulus
........+++
............+++
e is 65537 (0x10001)
```

2. Utilizza il seguente comando OpenSSL per creare una richiesta di firma del certificato (CSR). Ti verranno posta una serie di domande per il certificato.

```
openssl req -new -sha256 -key ssl-client.key -out ssl-client.csr
Enter pass phrase for ssl-client.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) [XX]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

3. Accedi al CSR con il certificato *`customerCA.crt`* creato al momento dell'inizializzazione del cluster.

```
openssl x509 -req -days 3652 -in ssl-client.csr \
        -CA customerCA.crt \
        -CAkey customerCA.key \
        -CAcreateserial \
        -out ssl-client.crt
Signature ok
subject=/C=US/ST=WA/L=Seattle/O=Example Company/OU=sales
Getting CA Private Key
```

Abilita SSL personalizzato per AWS CloudHSM

I passaggi sono diversi per Client SDK 3 o Client SDK 5. Per ulteriori informazioni sull'utilizzo dello strumento della riga di comando configurazione, vedi ???.

Argomenti

- SSL personalizzato per Client SDK 3
- SSL personalizzato per Client SDK 5

SSL personalizzato per Client SDK 3

Utilizza lo strumento di configurazione per Client SDK 3 per abilitare SSL personalizzato. Per ulteriori informazioni sullo strumento di configurazione per Client SDK 3, vedi ???.

Per utilizzare un certificato e una chiave personalizzati per l'autenticazione reciproca client-server TLS con Client SDK 3 su Linux

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

```
sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
sudo /opt/cloudhsm/bin/configure --ssl \
 --pkey /opt/cloudhsm/etc/ssl-client.key \
 --cert /opt/cloudhsm/etc/ssl-client.crt
```

3. Aggiungi il certificato `customerCA.crt` all'archivio di fiducia. Crea un hash del nome dell'oggetto del certificato. In questo modo viene creato un indice per consentire la ricerca del certificato con tale nome.

```
openssl x509 -in /opt/cloudhsm/etc/customerCA.crt -hash | head -n 1
1234abcd
```

Crea una directory.

```
mkdir /opt/cloudhsm/etc/certs
```

Crea un file che contiene il certificato con il nome hash.

```
sudo cp /opt/cloudhsm/etc/customerCA.crt /opt/cloudhsm/etc/certs/1234abcd.0
```

SSL personalizzato per Client SDK 5

Utilizza uno qualsiasi degli strumenti di configurazione di Client SDK 5 per abilitare SSL personalizzato. Per ulteriori informazioni sullo strumento di configurazione per Client SDK 5, vedi ???.

PKCS #11 library

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

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

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
cp ssl-client.key C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

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

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

OpenSSL Dynamic Engine

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

JCE provider

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
cp ssl-client.key C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

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

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

CloudHSM CLI

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
cp ssl-client.key C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

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

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

Per creare l'applicazione

Crea applicazioni e lavora con le chiavi utilizzando AWS CloudHSM.

Per iniziare a creare e utilizzare le chiavi nel nuovo cluster, devi prima creare un utente Hardware Security Module (HSM) con CloudHSM Management Utility (CMU). Per ulteriori informazioni, vedi Comprendere le attività di gestione degli utenti HSM, Guida introduttiva all'interfaccia a riga di comando (CLI) di AWS CloudHSM e Come gestire gli utenti HSM.

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

Con gli utenti HSM attivi, puoi accedere all'HSM e creare e utilizzare chiavi con una delle seguenti 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 i provider CNG e KSP per usare AWS CloudHSM con Microsoft Windows Server Certificate Authority (CA)
- Usa i provider CNG e KSP per usare AWS CloudHSM con lo strumento Microsoft Sign
- Utilizza i provider CNG e KSP per l'offload TLS con il web server Internet Information Serber (IIS)

Best practice per AWS CloudHSM

Esegui le best practice riportate in questo argomento per un utilizzo efficaceAWS CloudHSM.

Indice

Gestione dei cluster

Segui le best practice riportate in questa sezione durante la creazione, l'accesso e la gestione AWS CloudHSM 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 Prestazioni per le stime delle prestazioni sulle dimensioni e le configurazioni più comuni dei cluster. Ti consigliamo di testare il carico 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 un'elevata disponibilità

Aggiungi la ridondanza per tenere conto della manutenzione: AWS puoi sostituire l'HSM per la manutenzione programmata o se rileva un problema. Come regola generale, la dimensione del cluster dovrebbe avere almeno +1 di ridondanza. Ad esempio, se sono necessari due moduli di protezione hardware per il funzionamento del servizio nelle ore di punta, la dimensione ideale del cluster sarà quindi tre. Se si seguono le migliori pratiche relative alla disponibilità, queste sostituzioni HSM non dovrebbero influire sul servizio. Tuttavia, le operazioni in corso sull'HSM sostituito potrebbero non riuscire e devono essere ritentate.

Distribuisci i tuoi HSM in molte zone di disponibilità: considera come sarà in grado di funzionare il tuo servizio durante un'interruzione della zona di disponibilità. AWSconsiglia 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.

Disponete di almeno tre moduli di protezione hardware per garantire la durabilità 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 cluster

Usa le sottoreti private per limitare l'accesso alla tua istanza: avvia gli HSM e le istanze client nelle sottoreti private del tuo VPC. Ciò limita l'accesso ai tuoi moduli di protezione hardware 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 client richiede l'accesso all'AWS CloudHSMAPI, 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.

Riconfigura SSL per proteggere la comunicazione client-server: AWS CloudHSM utilizza TLS per stabilire una connessione al tuo HSM. Dopo aver inizializzato il cluster, puoi sostituire il certificato TLS e la chiave predefiniti utilizzati per stabilire la connessione TLS esterna. Per ulteriori informazioni, consulta Migliorare la sicurezza del server Web con l'offload SSL/TLS in AWS CloudHSM.

Riduci i costi adattandolo alle tue esigenze

Non sono previsti costi iniziali di utilizzo. AWS CloudHSM Paghi una tariffa oraria per ogni HSM avviato fino alla chiusura dell'HSM. Se il servizio non richiede l'uso continuo diAWS CloudHSM, è possibile ridurre i costi riducendo (eliminando) gli HSM fino a zero quando non sono necessari. Quando sono nuovamente necessari gli HSM, è possibile ripristinare gli HSM da un backup. Se, ad esempio, hai un carico di lavoro che richiede di firmare il codice una volta al mese, in particolare l'ultimo giorno del mese, puoi prima scalare il cluster, ridimensionarlo eliminando gli HSM al termine

del lavoro e quindi ripristinare il cluster per eseguire nuovamente le operazioni di firma alla fine del mese successivo.

AWS CloudHSMesegue automaticamente backup periodici degli HSM del 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 dalla stessa posizione in cui l'hai lasciato. Per calcolare i costi AWS CloudHSM dell'architettura, consulta la sezione Prezzi. AWS CloudHSM

Risorse correlate:

- Panoramica generale dei backup
- Politica di conservazione dei backup
- Copiare un backup tra regioniAWS

Gestione degli utenti HSM

Segui le best practice riportate in questa sezione per gestire efficacemente gli utenti del 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. Gestione degli utenti HSM nell'AWS CloudHSM

Proteggi le credenziali dei tuoi utenti HSM

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

Disponi di almeno due amministratori per prevenire 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 questo caso, puoi utilizzare l'altro amministratore per reimpostare la password.

> **Note**
>
> Gli amministratori in Client SDK 5 sono sinonimo di funzionari crittografici (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. Ciò può limitare il potenziale impatto se una delle password di amministrazione 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 attribuendo a diversi utenti di criptovalute responsabilità e azioni nettamente diverse (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:

- Condivisione chiave
- Annullare condivisione chiave

Gestione delle chiavi HSM

Segui le migliori pratiche riportate in questa sezione per la gestione delle chiavi inAWS CloudHSM.

Scegli il tipo di chiave giusto

Quando si utilizza una chiave di sessione, le transazioni al secondo (TPS) saranno limitate a un HSM laddove 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 su tutti gli HSM disponibili nel cluster. Per ulteriori informazioni, consulta Impostazioni di sincronizzazione e durabilità delle chiavi in AWS CloudHSM.

Gestisci i limiti di archiviazione delle chiavi

Gli HSM hanno dei 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. Quote AWS CloudHSM Se l'applicazione richiede più del limite, è possibile utilizzare una o più delle seguenti strategie per gestire efficacemente le chiavi:

Utilizzate il Trusted Wrapping per archiviare le chiavi in un archivio dati esterno: utilizzando il Trusted Key Wrapping, potete superare il limite di archiviazione delle chiavi archiviando tutte le chiavi in un archivio dati esterno. Quando è necessario utilizzare questa chiave, è possibile estrarre la chiave nell'HSM come chiave di sessione, utilizzare la chiave per l'operazione richiesta e quindi scartare la chiave di sessione. I dati chiave originali rimangono archiviati in modo sicuro nell'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'archiviazione delle chiavi in più cluster. In questo approccio, si mantiene una mappatura delle chiavi archiviate in ogni cluster. Utilizzate questa mappatura per indirizzare le richieste dei client 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 cluster con il provider JCE
- Connessione a più slot con PKCS #11

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 impacchettate sono crittografate e devono essere aperte utilizzando la stessa chiave di avvolgimento prima di poter essere utilizzate. Le chiavi non estraibili non possono essere esportate dall'HSM in nessuna circostanza. 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 Wrap per limitare la capacità degli utenti HSM di avvolgere/scartare solo le chiavi che sono state esplicitamente contrassegnate come attendibili da un amministratore. Per ulteriori informazioni, consulta gli argomenti sul Trusted Key Wrapping in. Gestione delle chiavi in AWS CloudHSM

Risorse correlate

- Funzioni di wrapping e annullamento del wrapping
- Funzioni di cifratura per JCE
- Attributi chiave Java supportati
- Attributi chiavi per la CLI di CloudHSM

Integrazione di applicazioni

Segui le best practice riportate in questa sezione per ottimizzare l'integrazione dell'applicazione con il cluster. AWS CloudHSM

Avvia il tuo Client SDK

Prima che l'SDK del client 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 del cluster senza dover tenere traccia di ogni singolo indirizzo. In questo modo si aggiunge una maggiore resilienza all'inizializzazione dell'applicazione nel caso in cui un HSM sia in fase di manutenzione o durante un'interruzione della zona di disponibilità. Per ulteriori dettagli, consulta Esegui il bootstrap di Client SDK.

Effettua l'autenticazione per eseguire operazioni

InAWS CloudHSM, è necessario autenticarsi nel cluster prima di poter eseguire la maggior parte delle operazioni, come le operazioni crittografiche.

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

Autenticazione con PKCS #11: in PKCS #11, si accede 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 operazioni di accesso aggiuntive. Per esempi sull'autenticazione con PKCS #11, vedere. Codici di esempio per la libreria PKCS #11

Autenticazione con JCE: il provider JCE supporta l'accesso sia AWS CloudHSM implicito che esplicito. Il metodo che funziona per 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 si disconnette 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, consulta. Fornire le credenziali al provider JCE

Autenticazione con OpenSSL: con OpenSSL Dynamic Engine, fornisci le credenziali tramite variabili di ambiente. AWS CloudHSMconsiglia di archiviare in modo sicuro le credenziali HSM quando non vengono utilizzate dall'applicazione. Se possibile, è necessario configurare l'ambiente per recuperare e impostare sistematicamente queste variabili di ambiente senza immetterle manualmente. Per i dettagli sull'autenticazione con OpenSSL, consulta. Installazione di OpenSSL Dynamic Engine

Gestisci in modo efficace le chiavi nella tua applicazione

Usa gli attributi chiave per controllare cosa possono fare le chiavi: quando generi una chiave, usa gli attributi chiave per definire una serie di autorizzazioni che consentiranno o negheranno tipi specifici di operazioni per quella chiave. Si consiglia di generare le chiavi con il minor numero di attributi necessari per completare l'attività. Ad esempio, non si dovrebbe consentire a una chiave AES utilizzata per la crittografia anche di estrarre le chiavi dall'HSM. Per ulteriori informazioni, consulta le nostre pagine sugli attributi per i seguenti Client SDK:

- Attributi chiave PKCS #11
- Attributi chiave JCE

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

- Con PKCS #11, è possibile trovare le chiavi utilizzando l'`C_FindObjects`API.

- Con JCE, è possibile trovare le chiavi utilizzando il. KeyStore

Per prestazioni ottimali, si AWS consiglia di utilizzare i comandi key find (come findKey eElenco delle chiavi) una sola volta durante l'avvio dell'applicazione e di memorizzare nella cache l'oggetto chiave restituito nella memoria dell'applicazione. Se avete bisogno di questo oggetto chiave in un secondo momento, dovreste recuperare l'oggetto dalla cache invece di interrogarlo per ogni operazione, il che comporterebbe un notevole sovraccarico di prestazioni.

Usa il multithreading

AWS CloudHSMsupporta 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 dovrebbe essere assegnata la 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 condividete 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

È possibile che si verifichino 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 +1 durante gli eventi di manutenzione.
- Le interruzioni della zona di disponibilità comportano una riduzione del numero di moduli di protezione hardware disponibili nel cluster.

Limitazione (della larghezza di banda della rete) HSMPer informazioni su come gestire al meglio questo scenario, consulta.

Per garantire che il cluster sia di dimensioni adeguate e non subisca limitazioni, consigliamo di AWS eseguire un test di carico nel proprio ambiente con il traffico di picco previsto.

Integra nuovi tentativi nelle operazioni del cluster

AWSpuò sostituire l'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 su tutte le operazioni instradate verso il cluster. Si prevede che i tentativi successivi di operazioni non riuscite a causa di sostituzioni abbiano esito positivo.

Implementa strategie di disaster recovery

In risposta a un evento, potrebbe essere necessario spostare il traffico lontano da un intero cluster o regione. 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, consulta 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 e la libreria PKCS #11 in Client SDK 5 supportano la connessione a più cluster dalla stessa applicazione. Ad esempio, è possibile avere due cluster attivi, ciascuno in regioni diverse, e l'applicazione può connettersi a entrambi contemporaneamente e bilanciare il carico tra i due come parte 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. Consulta le rispettive pagine per i dettagli:

- Connessione a più slot con PKCS #11
- Connessione a più cluster con il provider JCE

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

Monitoraggio

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

Monitora i log dei client

Ogni Client SDK scrive registri che puoi monitorare. Per informazioni sulla registrazione dei client, consulta. Utilizzo dei log SDK del client

Su piattaforme progettate per essere effimere, come Amazon ECS eAWS Lambda, la raccolta dei log dei client da un file può essere difficile. In queste situazioni, è consigliabile configurare la registrazione 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 per consentirti di conservarlo e visualizzarlo.

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

Strumento di configurazione Client SDK 5Per informazioni su come configurare le opzioni di registrazione nell'applicazione, consulta la sezione.

Monitora i registri di controllo

AWS CloudHSMpubblica i log di controllo sul tuo account Amazon CloudWatch . I log di controllo provengono dall'HSM e tengono traccia di determinate operazioni a scopo di controllo.

È possibile utilizzare i registri 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 di AWS CloudTrail

AWS CloudHSM è integrato con AWS CloudTrail, un servizio che offre un record delle operazioni eseguite da un utente, ruolo o servizio AWS in AWS CloudHSM. AWS CloudTrail acquisisce tutte le chiamate API per AWS CloudHSM come eventi. Le chiamate acquisite includono le chiamate dalla console di AWS CloudHSM e le chiamate di codice alle operazioni delle API AWS CloudHSM.

È possibile AWS CloudTrail utilizzarlo per controllare qualsiasi chiamata API effettuata al piano di AWS CloudHSM controllo per garantire che non si verifichino attività indesiderate nel proprio account.

Per informazioni dettagliate, vedi Uso di AWS CloudTrail e AWS CloudHSM.

Monitora i CloudWatch parametri di Amazon

Puoi utilizzare i CloudWatch parametri 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 i CloudWatch parametri di Amazon, puoi configurare gli CloudWatch allarmi Amazon per avvisarti di qualsiasi potenziale problema che potrebbe influire sul tuo servizio. Ti consigliamo di configurare gli allarmi per monitorare quanto segue:

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

Per ulteriori dettagli, consulta Utilizzo dei file di log Amazon CloudWatch e Audit Logs AWS CloudHSM.

Gestione dei cluster AWS CloudHSM

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

Per creare un cluster, consulta Nozioni di base.

Architettura cluster

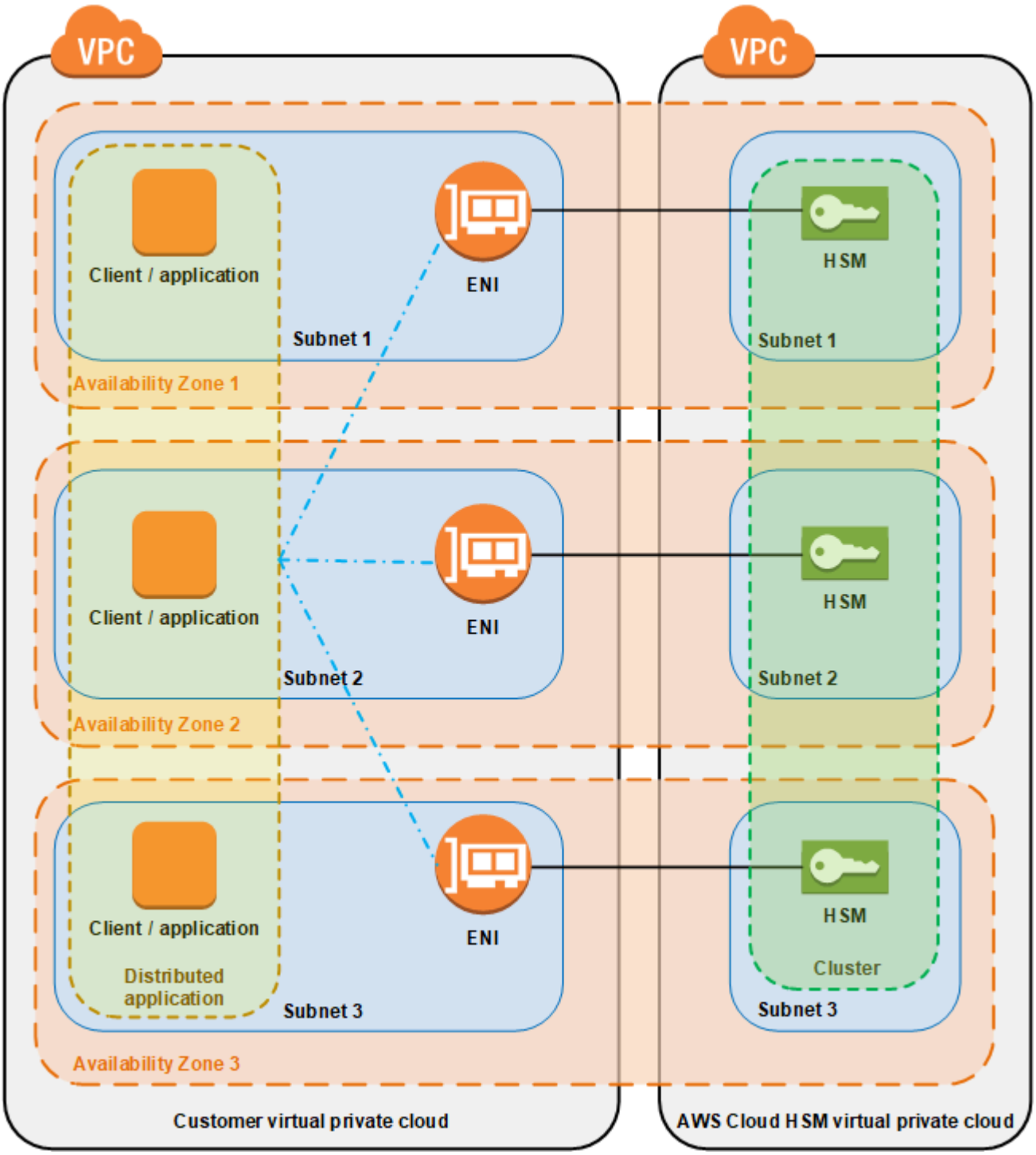
Quando crei un cluster, devi specificare un cloud privato virtuale (VPC) nell'account AWS e una o più sottoreti all'interno del VPC. È consigliabile creare una sottorete in ciascuna zona di disponibilità scelta nella tua regione AWS di appartenenza. Quando crei un VPC, puoi creare le sottoreti private. Per ulteriori informazioni, vedi Crea un cloud privato virtuale (VPC).

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 si crea un HSM, AWS CloudHSM offre un'interfaccia di rete elastica (ENI) nella sottorete specificata nell'account AWS. L'interfaccia di rete elastica è l'interfaccia per l'interazione con l'HSM. L'HSM risiede in un VPC separato, in un account AWS 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 in 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 cluster AWS CloudHSM con tre HSM, ognuno in una zona di disponibilità differente nel VPC.



Sincronizzazione del cluster

In un cluster AWS CloudHSM, AWS CloudHSM mantiene le chiavi sui singoli HSM sincronizzati. Non è necessaria alcuna operazione per sincronizzare le chiavi sugli HSM. Per mantenere gli utenti e le

policy in ciascun HSM sincronizzati, aggiornare il file di configurazione del client AWS CloudHSM prima di gestire gli utenti HSM. Per ulteriori informazioni, consulta Sincronizzazione degli utenti HSM.

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

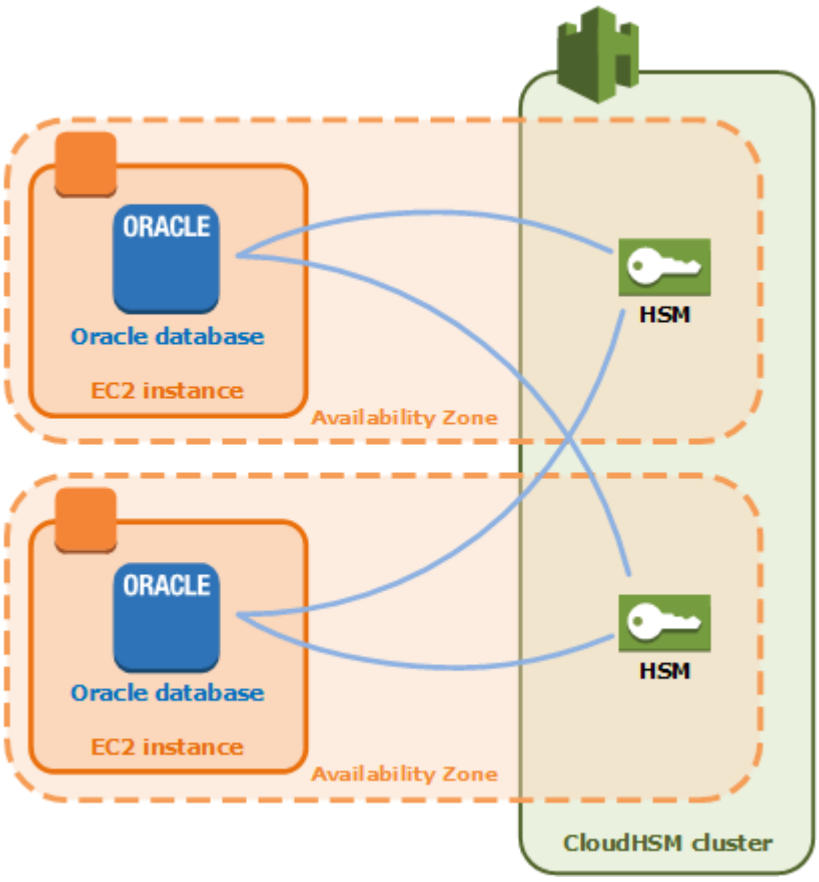
Se gli HSM in un cluster perdono la sincronizzazione, AWS CloudHSM li risincronizza automaticamente. Per abilitare questa operazione, AWS CloudHSM utilizza le credenziali dell'utente dell'appliance. Questo utente esiste su tutti gli HSM forniti da AWS CloudHSM e ha autorizzazioni limitate. È possibile 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.

Cluster a elevata disponibilità e sistema di bilanciamento del carico

Quando si crea un cluster AWS CloudHSM 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 zone di disponibilità AWS differenti, si ottiene automaticamente una disponibilità elevata. Elevata disponibilità significa che è possibile ottenere maggiore affidabilità, perché nessun HSM singolo rappresenta un singolo punto di errore. È consigliabile disporre di un minimo di due HSM in ogni cluster, con ogni HSM in diverse zone di disponibilità 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 di database memorizzano le chiavi master in un cluster che include un HSM in ciascuna zona di disponibilità. AWS CloudHSM sincronizza automaticamente le chiavi su entrambi gli HSM, in modo che siano immediatamente accessibili e ridondanti.



Connessione di client SDK al cluster AWS CloudHSM

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.

```
"C:\Program Files\Amazon\CloudHSM\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>
```

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.

```
"C:\Program Files\Amazon\CloudHSM\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.

```
"C:\Program Files\Amazon\CloudHSM\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 console dell'AWS CloudHSM 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, nella scheda HSM, scegli uno degli indirizzi IP elencati in Indirizzo IP ENI.

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

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters da AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
```

```
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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.

```
"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>
```

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.

```
"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 IP 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.

```
"C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI IP
 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 Strumento di configurazione 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.

```
C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe -a <HSM IP address>
```

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

Aggiunta o rimozione di moduli HSM in un cluster AWS CloudHSM

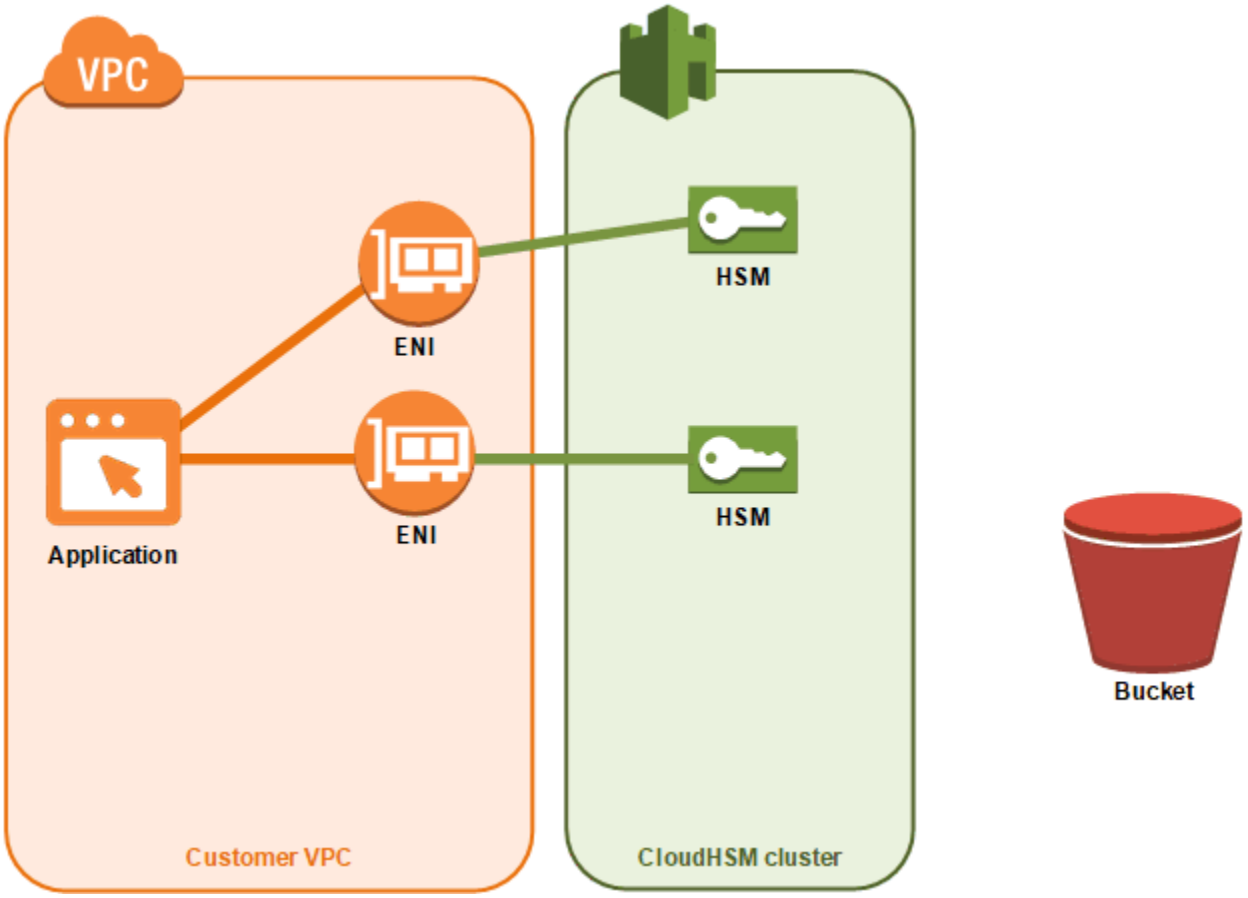
Per aumentare o ridurre il tuo cluster dell'AWS CloudHSM, puoi aggiungere o rimuovere i moduli HSM utilizzando la console dell'AWS CloudHSM oppure uno degli SDK dell'AWS o strumenti a riga di comando di. Ti consigliamo di effettuare test di carico sul cluster per determinare il carico massimo da prevedere,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

Argomenti

- Aggiunta di un modulo HSM
- Rimozione di un modulo HSM

Aggiunta di un modulo HSM

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 modulo HSM esistente nel cluster. Per ulteriori informazioni, consulta Backup.

3. AWS CloudHSM ripristina il backup nel nuovo modulo HSM. in modo da garantirne la sincronizzazione con gli altri moduli HSM nel cluster.

4. I moduli HSM esistenti nel cluster avvisano il client AWS CloudHSM della presenza di un nuovo modulo HSM.
5. Il client stabilisce una connessione con il nuovo modulo HSM.

Per aggiungere un modulo HSM (console)

1. Apri la console AWS CloudHSM 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 modulo 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"
    }
}
```

Per aggiungere un modulo HSM (API AWS CloudHSM)

- 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 modulo HSM

Puoi rimuovere un modulo HSM utilizzando la console AWS CloudHSM, AWS CLI o l'API dell'AWS CloudHSM.

Per rimuovere un modulo HSM (console)

1. Apri la console AWS CloudHSM 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 modulo 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 modulo HSM (API dell'AWS CloudHSM)

- 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 eliminare un cluster, devi rimuovere tutti i moduli HSM in esso contenuti. Per ulteriori informazioni, vedi Rimozione di un modulo HSM.

Dopo aver rimosso tutti i moduli HSM, è possibile eliminare un cluster utilizzando la console AWS CloudHSM, AWS Command Line Interface (AWS CLI) oppure l'API AWS CloudHSM.

Per eliminare un cluster (console)

1. Apri la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
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 (API AWS CloudHSM)

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

Creazione di cluster AWS CloudHSM dai backup

Per ripristinare un cluster AWS CloudHSM 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 Gestione dei backup.

Crea cluster dai backup (console)

1. Apri la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Scegli Create cluster (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.
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 Configurazione della politica di conservazione dei backup .
8. Seleziona 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. Specifica il tipo di istanza HSM, gli ID delle sottoreti in cui si desidera creare i moduli HSM e l'ID del backup che si sta ripristinando.

```
$ aws cloudhsmv2 create-cluster --hsm-type hsm1.medium \
                                --subnet-ids <subnet ID 1> <subnet ID 2> <subnet ID
 N> \
                                --source-backup-id <backup ID>
{
    "Cluster": {
        "HsmType": "hsm1.medium",
        "VpcId": "vpc-641d3c0d",
        "Hsms": [],
        "State": "CREATE_IN_PROGRESS",
        "SourceBackupId": "backup-rtq2dwi2gq6",
        "BackupPolicy": "DEFAULT",
        "BackupRetentionPolicy": {
            "Type": "DAYS",
            "Value": 90
         },
        "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

Gestione dei backup AWS CloudHSM

AWS CloudHSM esegue backup periodici del cluster almeno una volta ogni 24 ore. Un backup contiene copie crittografate dei seguenti dati:

- Utenti (CO, CU e AU)
- 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 Configurazione della politica di conservazione dei backup .

Argomenti

Utilizzo dei backup

Quando aggiungi un HSM a un cluster che in precedenza conteneva uno o più HSM attivi, il servizio ripristina l'ultimo backup sul nuovo HSM. Usa i backup per gestire gli HSM che usi meno frequentemente. 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 “Eliminazione e ripristino di 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.

Eliminazione e ripristino di backup

Dopo aver eliminato un backup, 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 Gestione dei backup.

Eliminazione e ripristino di backup (console)

Per eliminare un backup (console)

1. Apri la console dell'AWS CloudHSM 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. 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. Scegliere Delete (Elimina).

   Lo stato del backup diventa `PENDING_DELETE`. Puoi ripristinare un backup in attesa di eliminazione per un massimo di 7 giorni dopo la richiesta di eliminazione.

Per ripristinare un backup (console)

1. Apri la console dell'AWS CloudHSM 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. 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).

Eliminazione e ripristino di backup (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
    }
}
```

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"
    }
}
```

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",
            "CreateTimestamp": 1534461854.64,
            "ClusterId": "cluster-dygnwhmscg5",
            "DeleteTimestap": 1536339805.522,
        }
}
```

Eliminazione e ripristino di backup (API AWS CloudHSM)

Consulta i seguenti argomenti su come eliminare e ripristinare backup utilizzando l'API.

- DeleteBackup
- RestoreBackup

Configurazione della politica di conservazione dei backup AWS CloudHSM

Ad eccezione dei cluster creati prima del 18 novembre 2020, la politica di conservazione dei backup predefinita per i cluster è di 90 giorni. È possibile 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 Gestione dei backup.

Informazioni sulla 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 CloudHSMnon elimina l'ultimo backup di un cluster.

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.

Esclusione per i cluster esistenti

AWS CloudHSM ha lanciato la conservazione gestita dei backup il 18 novembre 2020. I cluster creati prima del 18 novembre 2020 prevedono 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 assegna al cluster una politica di conservazione dei backup di un anno più 90 giorni (455 giorni).

> **Note**
>
> È possibile disattivare completamente la conservazione dei backup gestiti contattando l'assistenza (https://aws.amazon.com/support).

Configurare la conservazione dei backup (console)

Per configurare la politica di conservazione dei backup (console)

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

Configurazione della politica di conservazione dei backup (AWS CLI)

Controlla lo stato di un backup o individuane l'ID utilizzando il comando describe-backups da 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. Specifica l'ID di backup e impostate il flag no-never-expires per includere il backup nella politica di conservazione dei backup, ad indicare 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
   }
}
```

Configurazione della politica di conservazione dei backup (API AWS CloudHSM)

Consulta i seguenti argomenti su come gestire la conservazione dei backup utilizzando l'API.

- ModifyCluster
- ModifyBackupAttributes

Copiare un backup tra regioniAWS

È possibile copiare i backup tra aree geografiche per molte ragioni, tra cui resilienza interregionale, carichi di lavoro globali e ripristino di emergenza. 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 Gestione dei backup o Creazione di cluster dai backup.

Tieni 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, vedi Gestione dei cluster .
- I backup non possono essere copiati tra le regioni con limitazioni AWS e le regioni standard. I backup possono essere copiati tra le regioni GovCloud (US-East) AWS e GovCloud (US-West) AWS.

Copia i backup in diverse regioni (console)

Per copiare i backup in diverse regioni (console)

1. Aprire la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
2. Per modificare la regione AWS, utilizza il selettore di regioni 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>
```

Copiare i backup in diverse regioni (API AWS CloudHSM)

Fai riferimento al seguente argomento per scoprire come copiare i backup in diverse regioni utilizzando l'API.

- CopyBackupToRegion

Tagging delle risorse AWS CloudHSM

Un tag è un'etichetta che assegni a una risorsa AWS. 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 i tag alle risorse AWS, AWS genera un report di allocazione dei costi in cui l'utilizzo e i costi sono aggregati in base ai tag. Puoi utilizzare questo report per visualizzare i costi di AWS CloudHSM in termini di progetti o applicazioni, invece di visualizzare tutti i costi di AWS CloudHSM come una voce singola.

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.

Puoi utilizzare la console AWS CloudHSM oppure uno degli SDK o degli strumenti a riga di comando di AWS per aggiungere, aggiornare, elencare e rimuovere i tag.

Argomenti

Aggiunta o aggiornamento dei tag

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. Aprire la console AWS CloudHSM all'indirizzo https://console.aws.amazon.com/cloudhsm/home.
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. Seleziona 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. Seleziona 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 i tag (API AWS CloudHSM)

- Inviare una richiesta TagResource. Specificare i tag e l'ID del cluster per cui si sta effettuando il tagging.

Come elencare i tag

Puoi creare un elenco dei tag di un cluster dalla console AWS CloudHSM, da AWS CLI o dall'API AWS CloudHSM.

Per elencare i tag (console)

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

- Invia una richiesta ListTags, specificando l'ID del cluster di cui elencare i tag.

Rimozione dei tag

Puoi rimuovere i tag da un cluster tramite la console AWS CloudHSM, AWS CLI o l'API AWS CloudHSM.

Per rimuovere i tag (console)

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

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. Seleziona 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. Se si utilizza la AWS CLI per rimuovere i tag, specificare solo le chiavi del tag e non i valori.

```
$ aws cloudhsmv2 untag-resource --resource-id <cluster ID> \
                                --tag-key-list "<tag key>"
```

Per rimuovere i tag (API AWS CloudHSM)

- Invia una richiesta UntagResource nell'API AWS CloudHSM, specificando l'ID del cluster e i tag da rimuovere.

Gestione di chiavi e utenti HSM in AWS CloudHSM

Prima di utilizzare il cluster AWS CloudHSM per l'elaborazione crittografica, è necessario creare utenti e chiavi nei moduli HSM del cluster. Vedi i seguenti argomenti per ulteriori informazioni sull'uso degli strumenti a riga di comando di AWS CloudHSM per gestire chiavi e utenti HSM. È inoltre possibile imparare a utilizzare l'autenticazione del quorum (anche detta controllo accesso "M of N").

Argomenti

Gestione degli utenti HSM nell'AWS CloudHSM

Nell'AWS CloudHSM, devi utilizzare gli strumenti a riga di comando CLI CloudHSM 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.

Argomenti

Gestione degli utenti HSM con la CLI di CloudHSM

Usa gli strumenti a riga di comando della CLI di CloudHSM per creare e gestire gli utenti sul tuo HSM con l'SDK più recente.

Argomenti

- Utilizzo della CLI di CloudHSM per gestire l'autenticazione del quorum (controllo dell'accesso "M of N")

Informazioni sugli utenti HSM

La maggior parte delle operazioni che puoi eseguire sull'HSM richiede le credenziali di un utente HSM. L'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

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

Amministratore 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 autorizzazioni degli utenti HSM.

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 autorizzazioni degli utenti HSM.

Utente dell'appliance (AU)

L'utente dell'applicazione (AU) è in grado di eseguire operazioni di clonazione e sincronizzazione sugli HSM del cluster. AWS CloudHSM usa l'AU per sincronizzare gli HSM in un cluster AWS CloudHSM. L'AU esiste su tutti gli HSM forniti da AWS CloudHSM e ha autorizzazioni limitate. Per ulteriori informazioni, consulta Tabella autorizzazioni degli utenti HSM.

AWS non è in grado di eseguire operazioni sugli HSM. AWS non può visualizzare o modificare gli utenti o le chiavi e non è in grado di eseguire alcuna operazione di crittografia con tali chiavi.

Tabella autorizzazioni degli utenti HSM

La tabella seguente elenca le operazioni HSM ordinate in base al tipo di utente o sessione HSM che può eseguire l'operazione.

	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

	Admin	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
Modifica della password di qualsiasi utente	 Sì	 No	 No	 No
Aggiunta, rimozione di utenti	 Sì	 No	 No	 No
Ottenimento stato sincronizzazione[3]	 Sì	 Sì	 Sì	 No
Estrazione, inserimento di oggetti nascosti[4]	 Sì	 Sì	 Sì	 No
Funzioni di gestione Key[5]	 No	 Sì	 No	 No
Crittografia, decrittografia	 No	 Sì	 No	 No
Firma, verifica	 No	 Sì	 No	 No

	Admin	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
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.

Utilizzo della CLI di CloudHSM per gestire gli utenti

Questo argomento fornisce step-by-step istruzioni sulla gestione degli utenti dei moduli di sicurezza hardware (HSM) con CloudHSM CLI. Per ulteriori informazioni sulla CLI di CloudHSM o sugli utenti HSM, consulta CLI di CloudHSM e Uso della CLI di CloudHSM.

Sections

- Informazioni sulla gestione degli utenti HSM con CLI di CloudHSM
- Download della CLI di CloudHSM
- Gestione degli utenti HSM con la CLI di CloudHSM

Informazioni sulla gestione degli utenti HSM con CLI di CloudHSM

Per gestire gli utenti HSM, 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.

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 sull'interfaccia a riga di comando di CloudHSM (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. Apri la console dell'AWS CloudHSM 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, nella scheda HSM, scegli uno degli indirizzi IP elencati in Indirizzo IP ENI.

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

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters da AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
```

```
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

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.

Gestione degli utenti HSM con la CLI di CloudHSM

Questa sezione include i comandi di base per gestire gli utenti HSM con la CLI di CloudHSM.

> **Note**
>
> Nota: i comandi utente della CLI di CloudHSM sono elencati nel riferimento ai comandi utente della CLI di CloudHSM

Argomenti

- Per creare un amministratore
- Per creare un crypto user
- Per elencare tutti gli utenti HSM sul cluster
- Per modificare le password dell'utente HSM
- Per eliminare utenti dell'HSM

Per creare un amministratore

Segui questi passaggi per creare un amministratore.

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

Linux

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

Windows

```
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": "admin",
    "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. Inserisci nuovamente la password per confermare che la password inserita è corretta.

Per creare un crypto user

Seguire la procedura indicata di seguito per creare un utente.

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

Linux

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

Windows

```
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": "admin",
    "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. Inserisci nuovamente la password per confermare che la password inserita è corretta.

Per elencare tutti gli utenti HSM sul cluster

Usa il comando user list per elencare tutti gli utenti sul cluster. 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

```
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 elenco utenti.

Per modificare le password dell'utente HSM

Usa il comando user change-password per cambiare una password.

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

   ```
   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": "admin1",
    "role": "admin"
  }
}
```

4. Immettere il comando user change-password.

```
aws-cloudhsm > user change-password --username <USERNAME> --role <ROLE>
```

5. Immetti la nuova password.
6. Immetti di nuovo 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

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

2. Utilizza il comando login e accedi come amministratore.

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

3. Inserisci la password dell'amministratore.

```
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin1",
```

```
    "role": "admin"
  }
}
```

4. 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>
```

5. Immetti la nuova password.
6. Immetti di nuovo la nuova password.

Per ulteriori informazioni su user change-password, consulta modifica password dell'utente.

Per eliminare utenti dell'HSM

Usa user delete per eliminare un utente. Per eliminare un altro utente devi accedere come amministratore.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

Per 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

   ```
   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": "admin",
    "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.

Uso della CLI di CloudHSM per gestire la MFA

Per maggiore sicurezza, ti consigliamo di configurare l'autenticazione a più fattori (MFA) per agevolare la protezione del cluster. Per ulteriori informazioni, consulta gli argomenti riportati di seguito.

Argomenti

- Informazioni sulla MFA per gli utenti HSM
- Uso dell'autenticazione MFA per gli utenti HSM

Informazioni sulla MFA per gli utenti HSM

Quando accedi a un cluster con un account utente HSM abilitato per la MFA, fornisci alla CLI di CloudHSM la tua password (il primo fattore, quello che conosci) e la CLI di CloudHSM 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 MFA per un utente, consulta Configurazione della MFA per la CLI di CloudHSM

Autenticazione quorum e MFA

Il cluster utilizza la stessa chiave per l'autenticazione del quorum e per l'autenticazione MFA. Questo significa che un utente con MFA abilitata è effettivamente registrato per il controllo degli accessi MoFN o 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 con autenticazione quorum, registri quell'utente come utente registrato 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 Gestione del quorum ("M of N").

Uso dell’autenticazione MFA per gli utenti HSM

Questo argomento fornisce informazioni e istruzioni per l'utilizzo della CLI di CloudHSM per gestire l'autenticazione a più fattori (MFA). Per ulteriori informazioni sulla CLI di CloudHSM, consulta Interfaccia a riga di comando (CLI) di CloudHSM.

Argomenti

- Requisiti per la coppia di chiavi MFA
- Configurazione della MFA per la CLI di CloudHSM
- Creare utenti con la MFA abilitata
- Accesso degli utenti con MFA abilitata
- Rotazione delle chiavi per gli utenti con MFA abilitata
- Annullare la registrazione di una chiave pubblica MFA per gli utenti amministratori quando la chiave pubblica MFA è registrata

- Informazioni di riferimento sul file token

Per ulteriori informazioni sulla gestione di utenti HSM, consulta Interfaccia a riga di comando (CLI) di CloudHSM

Requisiti per la coppia di chiavi MFA

Per abilitare l'autenticazione MFA per un utente HSM, 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

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

Completa la procedura riportata di seguito per configurare la MFA per la CLI di CloudHSM.

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. Utilizzando la CLI di CloudHSM, accedi al tuo account utente.

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

3. 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:
```

4. 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": ""
    }
  ]
}
```

5. 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
```

6. 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
```

7. Adesso, hai le firme binarie dei token. Devi codificarli usando la base64 e reinserirli nel tuo 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
```

8. 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
    }
  ]
}
```

9. 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"
},
```

Creare utenti con la MFA abilitata

Segui la procedura per creare utenti con l'autenticazione 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.

Accesso degli utenti con MFA abilitata

Segui la procedura per l'accesso di utenti con l'autenticazione 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 non firmati/firmati, dove i 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>"
  }
}
```

Rotazione delle chiavi per gli utenti con MFA abilitata

Segui la procedura per ruotare le chiavi per utenti con l'autenticazione 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 chiavi pubblica/privata RSA utile 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 per gli utenti amministratori quando la chiave pubblica MFA è registrata

Segui la procedura seguente per annullare la registrazione di una chiave pubblica MFA per 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"
  }
```

```
}
```

Informazioni di riferimento sul file token

Il file token generato durante la registrazione di una chiave pubblica MFA o quando si tenta di accedere tramite MFA è costituito da quanto segue:

- Token: un array di coppie di token non firmati/firmati codificati in base64 sotto forma di oggetti letterali JSON.
- 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
    }
```

```
  ]
}
```

Utilizzo della CLI di CloudHSM per gestire l'autenticazione del quorum (controllo dell'accesso "M of N")

Gli HSM nel tuo cluster AWS CloudHSM supportano l'autenticazione del quorum, nota anche come controllo dell'accesso "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 dell'amministratore: creazione ed eliminazione di utenti HSM e modifica della password di un utente HSM diverso. Per ulteriori informazioni, consulta Utilizzo dell'autenticazione del quorum per gli amministratori.

I seguenti argomenti forniscono ulteriori informazioni sull'autenticazione del quorum in AWS CloudHSM.

Argomenti

- Panoramica dell'autenticazione del quorum con strategia token-sign
- Ulteriori informazioni sull'autenticazione del quorum
- Nomi e tipi di servizi che supportano l'autenticazione del quorum
- Utilizzo dell'autenticazione del quorum per gli amministratori: prima configurazione
- Utilizzo dell'autenticazione del quorum per gli amministratori
- Modifica del valore minimo del quorum per gli amministratori

Panoramica dell'autenticazione del quorum con strategia token-sign

Le seguenti operazioni riepilogano i processi di autenticazione del quorum. Per le operazioni e gli strumenti specifici, consultare Utilizzo dell'autenticazione del quorum per gli amministratori.

1. Ciascun utente 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.

Ulteriori informazioni sull'autenticazione del quorum

Ricorda le seguenti informazioni aggiuntive sull'utilizzo dell'autenticazione del quorum in AWS CloudHSM.

- Un utente HSM può firmare il proprio token del quorum, vale a dire che 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.
- Se l'autenticazione a più fattori è abilitata, il cluster utilizza la stessa chiave per l'autenticazione del quorum e per l'autenticazione a più fattori (MFA). Per ulteriori informazioni sull'utilizzo dell'autenticazione del quorum e dell'autenticazione a due fattori, consulta la pagina relativa all'utilizzo della CLI di CloudHSM CLI per gestire l'autenticazione a più fattori.
- Ciascun HSM può contenere un solo token alla volta per servizio.

Nomi e tipi di servizi che supportano l'autenticazione del quorum

Servizi di amministrazione: l'autenticazione del quorum è utilizzata per servizi di amministrazione con privilegi, 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 MFA e del quorum.

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 servizio	Tipo servizio	Operazioni di servizio
Utente	Admin	• Crea utente • Elimina utente • Modifica utente-password • Modifica utente-mfa
Quorum	Admin	• segno del token del quorum set-quorum-value

Utilizzo dell'autenticazione del quorum per gli amministratori: prima configurazione

I seguenti argomenti descrivono la procedura da completare per configurare il modulo di sicurezza hardware (HSM), in modo che gli amministratori possano utilizzare l'autenticazione del quorum. 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 Utilizzo dell'autenticazione del quorum per gli amministratori.

Argomenti

- Prerequisiti
- Creazione e registrazione di una chiave per la firma
- Impostazione del valore minimo del quorum sull'HSM

Prerequisiti

Per comprendere questo esempio, è bene avere familiarità con la CLI di CloudHSM. In questo esempio, il cluster AWS CloudHSM ha due HSM, ognuno con gli stessi amministratori, come illustrato nel seguente output del comando user list. Per ulteriori informazioni sulla creazione degli utenti, vedere Uso della CLI di CloudHSM.

```
aws-cloudhsm>user list
{
  "error_code": 0,
  "data": {
    "users": [
      {
        "username": "admin",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin2",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin3",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "admin4",
        "role": "admin",
        "locked": "false",
        "mfa": [],
        "quorum": [],
        "cluster-coverage": "full"
      },
      {
        "username": "app_user",
        "role": "internal(APPLIANCE_USER)",
        "locked": "false",
        "mfa": [],
        "quorum": [],
```

```
        "cluster-coverage": "full"
      }
    ]
  }
}
```

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 spiega come utilizzare OpenSSL per creare una chiave RSA a 2.048 bit protetta da una passphrase. Per usare questo esempio, sostituisci *<admin.key>* con il nome del file in cui intendi archiviare la chiave.

```
$ openssl genrsa -out <admin.key> -aes256 2048
Generating RSA private key, 2048 bit long modulus
.....................................+++
.+++
e is 65537 (0x10001)
Enter pass phrase for admin.key:
Verifying - Enter pass phrase for admin.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 admin.key -outform PEM -pubout -out admin1.pub
Enter pass phrase for admin.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 di registrazione

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

   Linux

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

   Windows

   ```
   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 popolato 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

   ```
   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 Modifica utente-quorum token-firma registra per registrare una chiave pubblica. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help user change-quorum token-sign register.

Example - Registrazione di una chiave pubblica con il cluster AWS CloudHSM

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 al file PEM della chiave pubblica
> Campo obbligatorio: sì
> /path/tokenfile: il percorso del file con il token firmato dalla chiave privata dell'utente
> Campo 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"
      }
    ]
  }
}
```

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. È possibile modificare il valore minimo del quorum in qualsiasi momento; per ulteriori informazioni, consulta la pagina 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

```
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 Quorum token-firma Impostazione-valore-quorum per impostare il valore minimo del quorum. Per ulteriori informazioni, vedi l'esempio seguente oppure utilizza il comando help quorum token-sign set-quorum-value.

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 seguente comando di esempio, sostituisci l'ultimo numero (*<2>*) con il valore minimo del quorum desiderato.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service user --value <2>
{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

Nell'esempio precedente, il servizio identifica il servizio HSM di cui si sta impostando il valore minimo del quorum. Il comando Quorum token-firma elenco-valori-quorum elenca i tipi, i nomi e le descrizioni dei servizi HSM inclusi nel servizio.

Servizi di amministrazione: l'autenticazione del quorum è utilizzata per servizi di amministrazione con privilegi, 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 MFA e del quorum.

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 servizio	Tipo servizio	Operazioni di servizio
Utente	Admin	• Crea utente • Elimina utente • Modifica utente-password • Modifica utente-mfa
Quorum	Admin	• segno del token del quorum set-quorum-value

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 Utilizzo del quorum ("M of N").

Utilizzo dell'autenticazione del quorum per gli amministratori

Un amministratore dell'HSM può configurare l'autenticazione del quorum per le seguenti operazioni nel cluster AWS CloudHSM:

- Crea utente

- Elimina utente
- Modifica utente-password
- Modifica utente-mfa

Dopo che il cluster AWS CloudHSM è stato configurato per l'autenticazione del quorum, gli amministratori non possono svolgere operazioni di gestione degli utenti HSM in modo autonomo. 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

- Ottenere un token del quorum
- Ottenere le firme dagli amministratori di approvazione
- Approva il token sul cluster AWS CloudHSM ed esegui un'operazione di gestione degli utenti

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

```
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 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 usare per memorizzare 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",
  "approval_data": "AAEAAwAAABgAAAAAAAAAAJ9eFkfcP3mNzJAlfK
+OWbNhZG1pbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABj5vbeAAAAAAAAAAAAAQADAAAAFQAAAAAAAAAW/
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAABAAMAAAAUA
+b23gAAAAAAAAA",
  "token": "0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": []
}
```

Il file del token comprende:

- 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 serie di token con codifica base64 firmati (firme) del token non firmato, in cui ogni firma di un approvatore è sotto forma di valore letterale di un oggetto JSON:

  ```
  {
        "username": "<APPROVER_USERNAME>",
        "signature": "<APPROVER_RSA2048_BIT_SIGNATURE>"
  }
  ```

  Ogni firma viene creata come conseguenza di un approvatore che utilizza la propria chiave privata RSA a 2048 bit corrispondente la cui chiave pubblica è stata registrata nell'HSM.

È possibile confermare l'esistenza del token del quorum per il servizio utente generato nel cluster CloudHSM eseguendo il comando quorum token-sign list:

```
aws-cloudhsm > quorum token-sign list
{
  "error_code": 0,
```

```
  "data": {
    "tokens": [
      {
        "username": "admin",
        "service": "user",
        "approvals-required": {
          "value": 2
        },
        "number-of-approvals": {
          "value": 0
        },
        "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.

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 un altro strumento di firma, assicurati che lo strumento utilizzi la chiave privata dell'amministratore (chiave di firma) per firmare un digest SHA-256 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
v5Euk83amq1fij0zyvD2FkbWluAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPm9t4AAAAAAAAAAAABAAMAA
+b23gAAAAAAAAA",
  "token": "0l2LZkmAHZyAc1hPhyckOoVW33aGrgG77qmDHWQ3CJ8=",
  "signatures": [
    {
      "username": "admin2",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw=="
    },
    {
      "username": "admin3",
      "signature": "O6qx7/mUaVkYYVr1PW7l8JJko+Kh3e8zBIqdk3tAiNy+1rW
+OsDtvYujhEU4aOFVLcrUFmyB/CX9OQmgJLgx/pyK+ZPEH+GoJGqk9YZ7X1nOXwZRP9g7hKV
+7XCtg9TuDFtHYWDpBfz2jWiu2fXfX4/
jTs4f2xIfFPIDKcSP8fhxjQ63xEcCf1jzGha6rDQMu4xUWWdtDgfT7um7EJ9dXNoHqLB7cTzphaubNaEFbFPXQ1siGm
ssktwyruGFLpXs1n0tJOEglGhx2qbYTs+omKWZdORl5WIWEXW3IXw/
Dg5vVObrNpvG0eZKO8nSMc27+cyPySc+ZbNw==
    }
```

```
  ]
}
```

Approva il token sul cluster AWS CloudHSM ed esegui un'operazione di gestione degli utenti

Dopo aver ottenuto le approvazioni/firme necessarie, come descritto nella sezione precedente, l'amministratore può fornire quel token al cluster AWS CloudHSM ed effettuare una delle seguenti operazioni di gestione degli utenti:

- Crea
- Elimina
- Modifica-password
- user change-mfa

Per ulteriori informazioni sull'utilizzo di questi comandi, consulta Uso della CLI di CloudHSM.

Durante la transazione, il token verrà approvato all'interno del cluster AWS CloudHSM 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 del valore minimo del quorum per gli amministratori

Dopo aver impostato il valore minimo del quorum in modo che gli amministratori possano utilizzare l'autenticazione del quorum, se lo desideri puoi cambiare tale valore minimo. 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 (2), almeno due (2) amministratori dovranno approvare la modifica del valore minimo.

> **Note**
>
> Il valore del quorum del servizio utente deve sempre essere inferiore al valore del quorum del servizio quorum. Per maggiori informazioni sui nomi dei servizi, come il servizio quorum e il

servizio utente, consulta la pagina Nomi e tipi di servizi che supportano l'autenticazione del quorum.

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.

Per modificare il valore minimo del quorum per gli amministratori

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

   Linux

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

   Windows

   ```
   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 <admin> --role admin
   ```

3. Utilizza il comando quorum token-sign list-quorum-values per ottenere i valori minimi del quorum per tutti i nomi del servizio. Per ulteriori informazioni, consulta l'esempio di seguito.
4. Se il valore minimo del quorum per il servizio quorum è inferiore a quello del servizio utente, utilizza il comando quorum token-sign set-quorum-value per modificare il valore del servizio quorum. Modifica il valore del servizio quorum impostandolo su uno (1) uguale o superiore a quello del servizio utente. Per maggiori informazioni, consulta il seguente esempio:
5. Genera un token del quorum accertandoti di specificare il servizio quorum come servizio per il quale è possibile utilizzare il token.
6. Ottieni le approvazioni (firme) dagli altri amministratori.
7. Approva il token sul cluster AWS CloudHSM ed esegui un'operazione di gestione degli utenti.

8. Utilizza il comando quorum token-sign set-quorum-value per modificare il valore minimo del quorum per il servizio utente.

Example - Ottenimento dei valori minimi del quorum e modifica del valore per il servizio quorum

Il seguente comando di esempio mostra che il valore minimo corrente del quorum per il servizio utente è due (2).

```
aws-cloudhsm > quorum token-sign list-quorum-values{
  "error_code": 0,
  "data": {
    "user": 2,
    "quorum": 1
  }
}
```

Per modificare il valore minimo del quorum del servizio quorum, utilizza il comando quorum token-sign set-quorum-value impostando un valore uguale o superiore al valore del servizio utente. L'esempio seguente imposta il valore minimo del quorum per il servizio quorum su due (2), lo stesso valore impostato per il servizio utente.

```
aws-cloudhsm > quorum token-sign set-quorum-value --service quorum --value 2{
  "error_code": 0,
  "data": "Set quorum value successful"
}
```

Il comando seguente mostra che il valore minimo del quorum adesso è due (2) per il servizio utente e per 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)

In AWS CloudHSM, devi utilizzare la CLI CloudHSM o la CloudHSM Management Utility (CMU) per creare e gestire gli utenti sul tuo HSM. La CLI di CloudhSM è progettata per essere utilizzata con l'SDK più recente, mentre la CMU è progettata per essere utilizzata con gli SDK precedenti.

Argomenti

Informazioni sugli utenti HSM

La maggior parte delle operazioni che puoi eseguire sull'HSM richiede le credenziali di un utente HSM. L'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

- 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. Accedi all'HSM e modifica la password dell'utente 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 autorizzazioni degli utenti HSM. Quando si attiva un nuovo cluster, lo stato dell'utente cambia 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 autorizzazioni degli utenti HSM.

Utente dell'appliance (AU)

L'utente dell'applicazione (AU) è in grado di eseguire operazioni di clonazione e sincronizzazione sugli HSM del cluster. AWS CloudHSM usa l'AU per sincronizzare gli HSM in un cluster AWS CloudHSM. L'AU esiste su tutti gli HSM forniti da AWS CloudHSM e ha autorizzazioni limitate. Per ulteriori informazioni, consulta Tabella autorizzazioni degli utenti HSM.

AWS non è in grado di eseguire operazioni sugli HSM. AWS non può visualizzare o modificare gli utenti o le chiavi e non è in grado di eseguire alcuna operazione di crittografia con tali chiavi.

Tabella autorizzazioni degli utenti HSM

La tabella seguente elenca le operazioni HSM ordinate in base al tipo di utente o sessione HSM che può eseguire l'operazione.

	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[3]	 Sì	 Sì	 Sì	 No
Estrazione, inserimento di oggetti nascosti[4]	 Sì	 Sì	 Sì	 No

	Crypto officer (CO)	Utente di crittografia (CU)	Utente dell'appliance (AU)	Sessione autenticata
Funzioni di gestione Key[5]	 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.

Uso della CloudHSM Management Utility (CMU) per gestire gli utenti

Questo argomento fornisce step-by-step istruzioni sulla gestione degli utenti dei moduli di sicurezza hardware (HSM) con CloudHSM Management Utility (CMU), uno strumento a riga di comando fornito con Client SDK. Per ulteriori informazioni su CMU e utenti HSM, consulta Utility di gestione CloudHSM e Informazioni sugli utenti HSM.

Sections

Informazioni sulla gestione degli utenti HSM con la CMU

Per gestire gli utenti HSM, devi accedere all'HSM con il nome utente e la password di un crypto officer (CO). Solo i CO sono in grado di gestire gli utenti. L'HSM contiene un CO predefinito denominato admin. Hai impostato la password per admin quando hai attivato il cluster.

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 hai più HSM, puoi utilizzare 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 CMU.

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la console dell'AWS CloudHSM 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, nella scheda HSM, scegli uno degli indirizzi IP elencati in Indirizzo IP ENI.

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

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters da AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

Uso della CMU con i Client SDK 3.2.1 e versioni precedenti

Con il Client SDK 3.3.0, AWS CloudHSM ha aggiunto il supporto per il parametro `--cmu`, che semplifica il processo di aggiornamento del file di configurazione per la 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 7.9+

```
$ 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 8.3+

```
$ 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. Eseguite 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.

Come gestire gli utenti HSM con la CMU

Questa sezione include i comandi di base per gestire gli utenti HSM con la CMU.

Per creare utenti HSM

Usa createUser per creare nuovi utenti sull'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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

Windows

```
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 negli elenchi CMU corrisponda al numero di HSM 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

```
C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 negli elenchi CMU corrisponda al numero di HSM 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.

Per elencare tutti gli utenti HSM del cluster

Usa il comando listUsers per elencare tutti gli utenti del cluster. 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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.

Per modificare le password dell'utente HSM

Usa changePswd per modificare una password.

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, devi 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 negli elenchi CMU corrisponda al numero di HSM 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

   ```
   C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
   ```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 negli elenchi CMU corrisponda al numero di HSM 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.

Per eliminare utenti dell'HSM

Usa deleteUser per eliminare un utente. Per eliminare un altro utente devi accedere come CO.

> **Tip**
>
> Non puoi eliminare i crypto user (CU) che possiedono chiavi.

Per 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

```
C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
```

2. Avvia la CMU.

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg
   ```

   Windows

   ```
   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 negli elenchi CMU corrisponda al numero di HSM 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.

Utilizzo della CloudHSM Management Utility (CMU) per gestire l'autenticazione a due fattori (2FA) per i responsabili della crittografia

Per una maggiore sicurezza, puoi configurare l'autenticazione a due fattori (2FA) per proteggere il cluster. Puoi abilitare la 2FA solo per i crypto officer (CO).

> **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

Capire la 2FA per gli utenti HSM

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.

Autenticazione del quorum e 2FA

Il cluster utilizza la stessa chiave per l'autenticazione del quorum e per la 2FA. Ciò significa che un utente con 2FA abilitata viene effettivamente registrato per il controllo degli accessi M-of-N. 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.

Utilizzo della 2FA per utenti HSM

Questa sezione descrive come utilizzare la 2FA per gli utenti HSM, inclusa la creazione di utenti HSM con 2FA, la rotazione delle chiavi e l'accesso all'HSM come utenti abilitati alla 2FA. Per ulteriori informazioni su come lavorare con gli utenti HSM, vedi ???, ???, ???, ???, e ???.

Creazione di utenti 2FA

Per abilitare la 2FA per un utente HSM, utilizza una chiave che soddisfi i seguenti requisiti.

Requisiti della coppia di chiavi per la 2FA

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 del quorum e 2FA”.

Usa CMU e la coppia di chiavi per creare un nuovo utente CO con 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 NAME> -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)?

   yCreating 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)
```

Gestione della 2FA per gli utenti HSM

Usa cambia password per cambiare la password di un utente 2FA, per abilitare o disabilitare la 2FA o per ruotare la chiave 2FA. Ogni volta che abiliti la 2FA, devi fornire una chiave pubblica per gli accessi 2FA.

La modifica della password comporta 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 del quorum e 2FA”.

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 del quorum e 2FA”.

3. Tipo **y**.

CMU conferma l'operazione di modifica della password.

Informazioni di riferimento sulla configurazione

Di seguito è riportato un esempio delle proprietà della 2FA nel file `authdata` sia per la richiesta generata dalla CMU che per le risposte dell'utente.

```
{
    "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.

id HSM

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")

Gli HSM nel tuo cluster AWS CloudHSM supportano l'autenticazione del quorum, nota anche come controllo dell'accesso "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:

- La gestione degli utenti HSM da parte dei crypto officer (CO): creazione ed eliminazione di utenti HSM e modifica della password di un utente HSM diverso. Per ulteriori informazioni, consulta Utilizzo dell'autenticazione del quorum per crypto officer.

I seguenti argomenti forniscono ulteriori informazioni sull'autenticazione del quorum in AWS CloudHSM.

Argomenti

Panoramica sull'autenticazione del quorum

Le seguenti operazioni riepilogano i processi di autenticazione del quorum. Per le operazioni e gli strumenti specifici, consultare Utilizzo dell'autenticazione del quorum per crypto officer.

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.

Ulteriori informazioni sull'autenticazione del quorum

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, consulta la pagina su autenticazione del quorum e autenticazione a due fattori.

Utilizzo dell'autenticazione del quorum per i crypto officer: prima configurazione

I seguenti argomenti descrivono la procedura da completare per configurare il modulo di sicurezza hardware (HSM), in modo che i crypto officer (CO) possano utilizzare l'autenticazione del quorum. È necessario eseguire questa procedura una sola volta quando si configura l'autenticazione del quorum per i CO. Una volta completata questa procedura, consultare Utilizzo dell'autenticazione del quorum per crypto officer.

Argomenti

- Prerequisiti
- Creazione e registrazione di una chiave per la firma
- 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 cluster AWS CloudHSM ha due HSM, ognuno con gli stessi CO, come illustrato nel seguente output del comando listUsers. Per ulteriori informazioni sulla creazione degli utenti, vedere Gestione degli 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
```

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 usare questo esempio, sostituire *officer1.key* con il nome del file in cui archiviare 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.

Campo obbligatorio: sì

<officer1.token.sig>

Il percorso a un file che contiene l'hash SHA256_PKCS firmato dal meccanismo del token di registrazione.

Campo 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.

Campo obbligatorio: sì

Quando tutti i CO hanno registrato le proprie chiavi pubbliche, l'output del comando listUsers mostra questo nella colonna `MofnPubKey`, come illustrato 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
```

Impostazione del valore minimo del quorum sull'HSM

Per utilizzare l'autenticazione del quorum per i CO, 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 nell'HSM può impostare il valore minimo del quorum, compresi i CO 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 setMValue 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 compreso tra due (2) e otto (8), fino al numero totale di CO sull'HSM. In questo esempio, l'HSM ha sei CO, pertanto il valore massimo possibile è sei.

Per utilizzare il seguente comando di esempio, sostituire l'ultimo numero (*2*) con il valore minimo del quorum desiderato.

```
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 Utilizzo dell'autenticazione del quorum per crypto officer.

Utilizzo dell'autenticazione del quorum per crypto officer

Un responsabile della crittografia (CO) nell'HSM può configurare l'autenticazione del quorum per le seguenti operazioni sull'HSM:

- Creazione di utenti HSM

- Eliminazione di utenti HSM
- Modifica della password di un altro utente HSM

Dopo che l'HSM è stato configurato per l'autenticazione del quorum, i CO non possono svolgere operazioni di gestione degli utenti HSM in modo autonomo. 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. Ottenere le approvazioni (firme) dagli altri CO.
3. Approvare il token sul modulo HSM.
4. Svolgere l'operazione di gestione degli utenti HSM.

Configura l'HSM per l'autenticazione del quorum per i CO, se non hai ancora effettuato questa operazione. Per ulteriori informazioni, consulta Prima configurazione.

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 - Ottenimento di 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`*: nome del CO che sta ottenendo il token. Deve essere lo stesso CO che ha eseguito l'accesso all'HSM e sta eseguendo il comando.
- *`officer1.token`*: nome del file da usare per memorizzare 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 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
```

Ottenimento delle firme dai CO di approvazione

Un CO che dispone di un token del quorum devono ottenerne l'approvazione da parte di altri CO. Per concedere l'approvazione, gli altri CO 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 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 - Ottenimento delle firme dai CO di approvazione

In questo esempio, il CO che dispone del token (officer1) necessita di almeno due approvazioni. I seguenti comandi di esempio mostrano come due CO 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*: nome del file che contiene la chiave di firma del CO.
- *officer1.token.sig1* e *officer1.token.sig2*: nome del file da usare per memorizzare la firma. Assicurati di salvare ogni firma in un file diverso.
- *officer1.token*: 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:
```

Approvazione del token firmato nell'HSM

Dopo che un CO ottiene il numero minimo di approvazioni (firme) da altri CO, deve approvare il token firmato nell'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. Sostituisci *approval.txt* 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
```

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 Gestione degli 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 del valore minimo del quorum per i crypto officer

Dopo avere impostato il valore minimo del quorum in modo che i responsabili della crittografia (CO) possano utilizzare l'autenticazione del quorum, se lo desideri puoi cambiare tale valore minimo. 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 CO dovranno approvare la modifica del valore minimo.

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 i CO (servizio 3), potresti dover modificare il valore minimo del 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 getMValue per ottenere il valore minimo del quorum per il servizio 3. Per maggiori informazioni, consulta il seguente esempio:
4. Utilizzare il comando getMValue 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 setMValue 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. Ottenere le approvazioni (firme) dagli altri CO.
8. Approvare il token sul modulo HSM.
9. Utilizzare il comando setMValue per modificare il valore minimo del quorum per il servizio 3 (operazioni di gestione degli utenti eseguite dai CO).

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]
```

Gestione delle chiavi in AWS CloudHSM

In AWS CloudHSM, utilizza una delle opzioni seguenti per gestire le chiavi sugli HSM del 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.

Argomenti

Impostazioni di sincronizzazione e durabilità delle chiavi in AWS CloudHSM

Questo argomento descrive le impostazioni di sincronizzazione delle chiavi in AWS CloudHSM, i problemi comuni che i clienti riscontrano durante l'utilizzo delle chiavi in un cluster e le strategie per incrementare la durabilità delle chiavi.

Argomenti

Concetti

Chiavi token

Chiavi persistenti create durante le operazioni di generazione, importazione o annullamento del wrapping delle chiavi. AWS CloudHSM sincronizza le chiavi dei token di un cluster.

Chiavi di sessione

Chiavi effimere che esistono solo su un modulo di sicurezza hardware (HSM, Hardware Security Module) nel cluster. AWS CloudHSM non sincronizza le chiavi di sessione di un cluster.

Sincronizzazione delle chiavi lato client

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.

Sincronizzazione delle chiavi lato server

Clona periodicamente le chiavi su ogni HSM del cluster. Non richiede alcuna gestione.

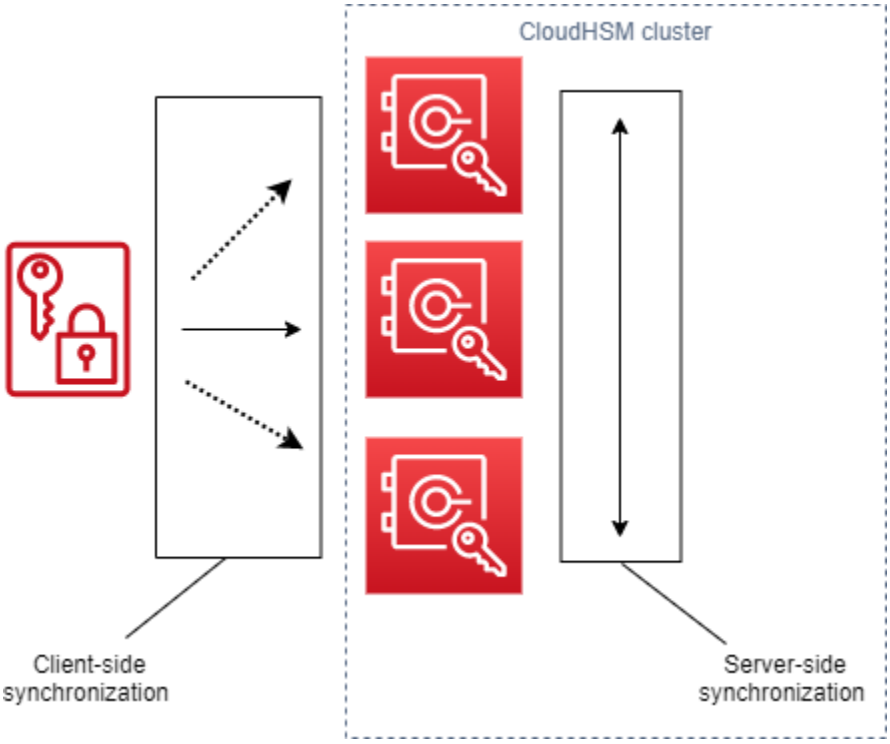
Impostazioni di durabilità delle chiavi del client

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.

Informazioni sulla sincronizzazione delle chiavi

AWS CloudHSM utilizza la sincronizzazione delle chiavi per clonare le chiavi token su tutti gli 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, consulta Metriche. 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.

Utilizzo delle impostazioni di durabilità delle chiavi del client

La sincronizzazione delle chiavi è un processo prevalentemente automatico, ma è possibile gestire le impostazioni di durabilità delle chiavi lato client. Il funzionamento delle impostazioni di durabilità delle chiavi lato client è 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 relativi a Client SDK 5”.
- 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 HSM di un cluster su cui devono essere presenti delle 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.

```
"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
```

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.

```
"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.

```
"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. Si stabilisce il numero di HSM su cui la creazione della chiave deve avere esito positivo per la riuscita dell'operazione complessiva. La sincronizzazione lato client tenta sempre, al massimo delle proprie capacità, di 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. Si noti che, indipendentemente dal numero di HSM specificato, il carico di lavoro sul cluster rimane invariato. La sincronizzazione lato client tenta sempre, al massimo delle proprie capacità, di 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 nodo `client` del file, aggiungi `create_object_minimum_nodes` e specifica un valore relativo al numero minimo di HSM su cui AWS CloudHSM deve creare 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, consultare 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 si specifica il parametro della riga di comando e un valore nel file di configurazione, AWS CloudHSM rispetta il valore MAGGIORE.

Per ulteriori informazioni, consulta i seguenti argomenti:

- GenDe A KeyPair
- Gene C KeyPair
- genere RSA KeyPair
- genSymKey
- importPrivateKey
- importPubKey
- imSymKey
- insertMaskedObject
- unWrapKey

Sincronizzare le chiavi in cluster clonati

La sincronizzazione lato client e lato server serve solo a sincronizzare le chiavi all'interno dello stesso cluster. Se si copia un backup di un cluster in un'altra regione, è possibile utilizzare il comando SyncKey di cloudhsm_mgmt_util (CMU) per sincronizzare le chiavi tra i cluster. È possibile utilizzare cluster clonati per la ridondanza tra regioni o per semplificare il processo di ripristino di emergenza. Per ulteriori informazioni, consulta la pagina syncKey.

Confezionamento delle chiavi AES AWS CloudHSM

Questo argomento descrive le opzioni per il key wrapping in AWS CloudHSMAES. 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
AES Wrapping Chiavi 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 relativa all'utilizzo di Wrapping Chiavi AES 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.

Nella tabella seguente vengono descritti i valori consentiti per IV, specificati dall'algoritmo di wrapping.

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) Valore predefinito: `0xA6A6A6A6A6A6A6A6`	Consentito 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 portachiavi degli algoritmi scartare i tasti supportati, come mostrato nella tabella seguente.
- In key_mgmt_util (KMU), utilizza i comandi wrapKey e unWrapKey con valori m appropriati, come illustrato nella tabella seguente.

Algoritmo Wrapping Chiavi AES	Meccanismo PKCS #11	Metodo Java	AWS CLI Comando secondario	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
Wrapping Chiavi AES senza Padding	• `CKM_CLOUDHSM_AES_KEY_WRAP_NO_PAD` (Meccanismo definito dal fornitore)	`AESWrap/ECB/NoPadding`	aes-no-pad	m = 5

Algoritmo Wrapping Chiavi AES	Meccanismo PKCS #11	Metodo Java	AWS CLI Comando secondario	Argomento della Key Management Utility (KMU)
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 attendibili in AWS CloudHSM

AWS CloudHSM supporta il wrapping attendibile delle chiavi per proteggere le chiavi di dati da minacce interne. Questo argomento descrive come creare chiavi attendibili per proteggere i dati.

Argomenti

- Informazioni sulle chiavi attendibili
- Attributi delle chiavi attendibili
- Come utilizzare le chiavi attendibili per eseguire il wrapping delle chiavi di dati
- Come annullare il wrapping di una chiave di dati con una chiave attendibile

Informazioni sulle chiavi attendibili

Una chiave attendibile è una chiave utilizzata per il wrapping di altre chiavi e identificata specificamente come chiave attendibile dagli amministratori e dai crypto officer (CO) tramite l'attributo `CKA_TRUSTED`. Inoltre, gli amministratori e i crypto officer (CO) utilizzano `CKA_UNWRAP_TEMPLATE` e attributi correlati per specificare le azioni che le chiavi di dati possono effettuare una volta annullato il wrapping mediante una chiave attendibile. 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 delle chiavi attendibili

I seguenti attributi consentono di contrassegnare una chiave come attendibile, specificare che è possibile eseguire o annullare il wrapping di una chiave di dati solamente con una chiave attendibile e controllare l'uso di una chiave di dati in seguito all'annullamento del wrapping:

- `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 le chiavi attendibili per eseguire il wrapping delle chiavi di dati

Per utilizzare una chiave attendibile per eseguire il wrapping di una chiave di dati, è necessario 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 Generazione chiavi-asimmetriche-coppia .

Per ulteriori informazioni sulla generazione di chiavi simmetriche, consulta la pagina Generazione chiavi-simmetriche .

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 Login per accedere come crypto user (CU).

2. Utilizza il comando Set di chiavi-attributi 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 Come contrassegnare una chiave come attendibile con 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 CMU.

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.

- Esempi con AWS CloudHSM PKCS #11
- Esempi con AWS CloudHSM JCE

Come annullare il wrapping di una chiave di dati con una chiave attendibile

Per annullare il wrapping di una chiave di dati, è necessaria una chiave attendibile con l'attributo `CKA_UNWRAP` impostato 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 la CLI di CloudHSM

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.

- La pagina sull'utilizzo di chiavi attendibili descrive come utilizzare gli attributi della libreria PKCS #11 e la CLI di CloudHSM per creare chiavi attendibili 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.

Utilizzare la CLI di CloudHSM per generare le chiavi

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

- Generazione di chiavi simmetriche
- Generazione di chiavi asimmetriche
- Argomenti correlati

Generazione di chiavi simmetriche

Usa i comandi elencati in Generazione chiavi-simmetriche per generare chiavi simmetriche. Per visualizzare tutte le opzioni disponibili, utilizza il comando help key generate-symmetric.

Genera 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
```

Argomenti

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave AES.

Campo obbligatorio: sì

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- 16, 24 e 32

Campo 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 `token=true`)

Per un elenco degli attributi delle chiavi AWS CloudHSM supportati, consulta la pagina Attributi chiavi per la CLI di CloudHSM.

Campo 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 i dati che potrebbe essere necessario decrittografare 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 persistenti/token. È possibile cambiare questa impostazione utilizzando <SESSION>, che garantisce che le chiavi generate con questo argomento siano chiavi di sessione/effimere

Campo 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
```

Argomenti

<LABEL>

Specifica un'etichetta definita dall'utente per la chiave segreta generica.

Campo obbligatorio: sì

<KEY-LENGTH-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- Da 1 a 800

Campo 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 `token=true`)

Per un elenco degli attributi delle chiavi AWS CloudHSM supportati, consulta la pagina Attributi chiavi per la CLI di CloudHSM.

Campo 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 i dati che potrebbe essere necessario decrittografare 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 persistenti/token. È possibile cambiare questa impostazione utilizzando <SESSION>, che garantisce che le chiavi generate con questo argomento siano chiavi di sessione/effimere

Campo obbligatorio: no

Generazione di chiavi asimmetriche

Utilizza i comandi elencati in Generazione chiavi-asimmetriche-coppia per generare coppie di chiavi asimmetriche.

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
```

Argomenti

<PUBLIC_LABEL>

Specifica un'etichetta definita dall'utente per la chiave pubblica.

Campo obbligatorio: sì

<PRIVATE_LABEL>

Specifica un'etichetta definita dall'utente per la chiave privata.

Campo obbligatorio: sì

<MODULUS_SIZE_BITS>

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Campo obbligatorio: sì

<ESPONENTE_PUBBLICO>

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Campo 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 `token=true`).

Per un elenco degli attributi delle chiavi AWS CloudHSM supportati, consulta la pagina Attributi chiavi per la CLI di CloudHSM.

Campo 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 i dati che potrebbe essere necessario decrittografare 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 persistenti/token. È possibile cambiare questa impostazione utilizzando <SESSION>, che garantisce che le chiavi generate con questo argomento siano chiavi di sessione/effimere

Campo 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
```

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 hanno un limite di 126 caratteri.

Campo 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 hanno un limite di 126 caratteri.

Campo obbligatorio: sì

<CURVE>

Specifica l'identificatore per la curva ellittica.

Valori validi:

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1

Campo 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 `token=true`).

Per un elenco degli attributi delle chiavi AWS CloudHSM supportati, consulta la pagina Attributi chiavi per la CLI di CloudHSM.

Campo 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 `token=true`).

Per un elenco degli attributi delle chiavi AWS CloudHSM supportati, consulta la pagina Attributi chiavi per la CLI di CloudHSM.

Campo 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 i dati che potrebbe essere necessario decrittografare 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).

Campo obbligatorio: no

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM

- Generazione chiavi-asimmetriche-coppia
- Generazione chiavi-simmetriche

Utilizzare la CLI di CloudHSM per eliminare le chiavi

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": [],
          "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 chiavi per la CLI di CloudHSM
- Elimina chiave

Utilizzare la CLI di CloudHSM per condividere e annullare la condivisione delle chiavi

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 la possiede. Il proprietario può utilizzare i comandi key share e key unshare per condividere e annullare la condivisione della chiave con altri 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.

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"
            },
          ],
          "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"
            }
          ],
          "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"
            },
          ],
          "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 chiavi per la CLI di CloudHSM
- Condivisione chiave
- Annullare condivisione chiave
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

Utilizzare la CLI di CloudHSM per filtrare le chiavi

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 e/o filtrare le chiavi con la CLI di CloudHSM, i comandi delle chiavi impiegano un meccanismo di filtraggio standardizzato basato sugli Attributi chiavi per la CLI di CloudHSM. È possibile specificare una chiave o un set di chiavi nei comandi delle chiavi utilizzando uno o più attributi AWS CloudHSM 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 cluster AWS CloudHSM.

Argomenti

Requisiti

Per filtrare le chiavi, bisogna aver effettuato l'accesso come crypto user (CU).

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"
            }
          ],
          "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"
            }
          ],
          "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"
            },
          ],
          "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"
            }
          ],
          "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 chiavi per la CLI di CloudHSM

Come contrassegnare una chiave come attendibile con 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 `key_to_be_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 Logout, scollegati come crypto user (CU).
4. Utilizzando il comando Login, 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"
  }
}
```

Gestione delle chiavi con KMU e CMU

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 sugli HSM del cluster AWS CloudHSM utilizzando lo strumento a riga di comando key_mgmt_util. Prima di gestire le chiavi, dovrai avviare il client AWS CloudHSM, avviare key_mgmt_util e accedere agli HSM. 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.

Generazione delle chiavi

Per creare chiavi nell'HSM, utilizza il comando corrispondente al tipo di chiave da creare.

Argomenti

- Generazione di chiavi simmetriche
- Generazione di una coppia di chiavi RSA
- Generazione di coppie di chiavi Elliptic Curve Cryptography (ECC)

Generazione di chiavi simmetriche

Utilizza il comando genSymKey per generare chiavi AES e altri tipi di chiavi simmetriche. 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
```

Generazione di una coppia di chiavi RSA

Per creare una coppia di chiavi RSA, utilizza il comando genRSAKeyPair. 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
```

Generazione di coppie di chiavi Elliptic Curve Cryptography (ECC)

Per creare una coppia di chiavi ECC, utilizza il comando genECCKeyPair. 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
```

Importazione delle chiavi

Per importare le chiavi segrete, ovvero chiavi simmetriche e chiavi private asimmetriche, nell'HSM, dovrai creare innanzitutto una chiave di wrapping nell'HSM. Puoi importare le chiavi pubbliche direttamente senza una chiave di wrapping.

Argomenti

Importazione di chiavi segrete

Completa la seguente procedura per importare una chiave segreta. 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, utilizza il comando unWrapKey.

Per importare una chiave segreta

1. Utilizza il comando genSymKey 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
```

Importazione di chiavi pubbliche

Utilizza il comando importPubKey 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
```

Esportazione delle chiavi

Per esportare le chiavi segrete, ovvero chiavi simmetriche e chiavi private asimmetriche, dall'HSM, dovrai creare innanzitutto 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**
>
> Il comando exSymKey crea una copia in testo non crittografato 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

- Esportazione di chiavi segrete
- Esportazione di chiavi pubbliche

Esportazione di chiavi segrete

Completa la seguente procedura per esportare una chiave segreta.

Per esportare una chiave segreta

1. Utilizza il comando genSymKey per creare una chiave di wrapping. 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, utilizza 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
```

Esportazione di chiavi pubbliche

Utilizza il comando exportPubKey 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
```

Eliminazione delle chiavi

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
```

Condivisione e annullamento della condivisione delle chiavi

In AWS CloudHSM, il CU che crea la chiave ne diventa proprietario. 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 nel momento in cui viene creata la chiave, ad esempio utilizzando il parametro `-u` del comando genSymKey o genRSAKeyPair. 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 condividere una chiave, è necessario avviare cloudhsm_mgmt_util, abilitare la crittografia end-to-end e accedere agli HSM. Per condividere una chiave, accedi all'HSM come crypto user (CU) che possiede la chiave. Solo i proprietari di chiavi possono condividere una chiave.

Utilizza il comando shareKey per condividere o annullare la condivisione di una chiave, specificando l'handle della chiave e l'ID dell'utente o degli utenti. Per condividere o annullare la condivisione con più di un utente, specifica un elenco di ID utente separato da virgole. 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 CMU

Il contenuto di questa sezione fornisce istruzioni sull'uso di CMU per contrassegnare una chiave come attendibile.

1. Accedi come crypto officer (CO) utilizzando il comando loginHSM.
2. Usa il comando setAttribute con `OBJ_ATTR_TRUSTED` (valore `134`) impostato su true (1).

   ```
   setAttribute <Key Handle> 134 1
   ```

Gestione dei cluster clonati

Utilizza CloudHSM Management Utility (CMU) 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 utilizza CMU per sincronizzare i cluster. A tale scopo, è necessario creare un nuovo file di configurazione CMU, specificare i moduli di sicurezza hardware (HSM) di entrambi i cluster nel nuovo file e quindi utilizzare CMU per connettersi al cluster con quel file.

Per utilizzare CMU su cluster clonati

1. Crea una copia del tuo file di configurazione corrente e cambia il nome della copia.

   Ad esempio, utilizza le seguenti posizioni dei file per individuare e creare una copia del file di configurazione corrente, quindi modifica il nome della copia da `cloudhsm_mgmt_config.cfg` `asyncConfig.cfg`.

   - Linux: `/opt/cloudhsm/etc/cloudhsm_mgmt_config.cfg`
   - Windows: `C:\ProgramData\Amazon\CloudHSM\data\cloudhsm_mgmt_config.cfg`
2. Nella copia rinominata, aggiungi l'IP dell'interfaccia di rete elastica (ENI) dell'HSM di destinazione (l'HSM nella regione esterna che deve essere sincronizzato). Ti consigliamo di aggiungere l'HSM di destinazione sotto l'HSM di origine.

```
{
    ...
    "servers": [
        {
            ...
            "hostname": "<ENI Source IP>",
            ...
        },
        {
            ...
            "hostname": "<ENI Destination IP>",
            ...
        }
    ]
}
```

Per informazioni su come ottenere questo indirizzo IP, vedi the section called “Ottieni un indirizzo IP per un HSM”.

3. Inizializza CMU con il nuovo file di configurazione:

   Linux

   ```
   $ /opt/cloudhsm/bin/cloudhsm_mgmt_util /opt/cloudhsm/etc/userSync.cfg
   ```

   Windows

   ```
   C:\Program Files\Amazon\CloudHSM>cloudhsm_mgmt_util.exe C:\ProgramData\Amazon
   \CloudHSM\data\userSync.cfg
   ```

4. Controllare i messaggi di stato restituiti per assicurarti che CMU sia connesso a tutti i moduli HSM e determinare quale degli IP ENI restituiti corrisponde a ogni cluster. Usa SyncUser e SyncKey per sincronizzare manualmente utenti e chiavi. Per ulteriori informazioni, vedi SyncUser e SyncKey.

Ottieni un indirizzo IP per un HSM

Utilizza questa sezione per ottenere un indirizzo IP per un HSM.

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la console dell'AWS CloudHSM 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, nella scheda HSM, scegli uno degli indirizzi IP elencati in Indirizzo IP ENI.

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

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters da AWS CLI. Nell'output del comando, l'indirizzo IP dei moduli HSM sono i valori di `EniIp`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

Argomenti correlati

- syncUser
- syncKey
- Copiare un backup tra regioni

Strumenti a riga di comando AWS CloudHSM

Questo argomento descrive gli strumenti a riga di comando disponibili per la gestione e l'utilizzo di AWS CloudHSM.

Argomenti

Comprendere gli strumenti a riga di comando

Oltre all'interfaccia a riga di comando AWS (AWS CLI) che usi per gestire le tue risorse AWS, AWS CloudHSM offre strumenti a riga di comando per la creazione e la gestione di utenti e chiavi HSM nei tuoi HSM. In AWS CloudHSM usi i familiari AWS CLI per la gestione del cluster e gli strumenti a riga di comando CloudHSM per la gestione dell'HSM.

Questi sono i vari strumenti a riga di comando:

Per gestire cluster e moduli HSM

Comandi CloudHSMv2 in AWS CLI e cmdlet HSM2 PowerShell nel modulo AWSPowerShell

- Questi strumenti consentono di ottenere, creare, eliminare e applicare tag ai cluster AWS CloudHSM e ai moduli HSM:
- Per utilizzare i comandi contenuti nei comandi CloudHSMv2 in AWS CLI, è necessario installarli e configurarli AWS CLI.
- cmdlet di HSM2 PowerShell nel modulo AWSPowerShell sono disponibili in un modulo Windows PowerShell e in un modulo PowerShell Core multipiattaforma.

Gestire gli utenti HSM

CLI CloudhSM

- Utilizza la CLI di CloudhSM per creare, eliminare, 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

Questi strumenti consentono di utilizzare strumenti e librerie di software AWS CloudHSM.

- Lo strumento di configurazione aggiorna i file di configurazione del client CloudHSM. In questo modo AWS CloudHSM può sincronizzare i moduli HSM in un cluster.

  AWS CloudHSM offre due versioni principali e Client SDK 5 è la più recente. 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 gli SDK precedenti

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.

Strumento di Configurazione

AWS CloudHSM sincronizza automaticamente i dati tra tutti i moduli di sicurezza hardware (HSM) in un cluster. Lo strumento configure aggiorna i dati dell'HSM nei file di configurazione utilizzati dal meccanismo di sincronizzazione. Usa configure per aggiornare i dati prima di usare gli strumenti a riga di comando, soprattutto quando i moduli HSM nel cluster sono cambiati.

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 vantaggi e i vantaggi che offre, vedi Vantaggi del 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 Client SDK 3 a Client SDK 5

Argomenti

- Strumento di configurazione Client SDK 5
- Strumento di configurazione Client SDK 3

Strumento di configurazione Client SDK 5

Utilizza lo strumento di configurazione del 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 Windows. `configure-pkcs11.exe`

Sintassi

PKCS #11

```
configure-pkcs11[ .exe ]
             -a <ENI IP address>
             [--hsm-ca-cert <customerCA certificate file path>]
             [--cluster-id <cluster ID>]
             [--endpoint <endpoint>]
             [--region <region>]
             [--server-client-cert-file <client certificate file path>]
             [--server-client-key-file <client key file path>]
             [--log-level <error | warn | info | debug | trace>]
                  Default is <info>
             [--log-rotation <daily | weekly>]
```

```
                Default is <daily>
            [--log-file <file name with path>]
                Default is </opt/cloudhsm/run/cloudhsm-pkcs11.log>
                Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-pkcs11.log>
            [--log-type <file | term>]
                Default is <file>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
            [--enable-validate-key-at-init]
                This is the default for PKCS #11
```

OpenSSL

```
configure-dyn[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
            [--cluster-id <cluster ID>]
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                Default is <error>
            [--log-type <file | term>]
                Default is <term>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                This is the default for OpenSSL
            [--enable-validate-key-at-init]
```

JCE

```
configure-jce[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
            [--cluster-id <cluster ID>]
```

```
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                 Default is <info>
            [--log-rotation <daily | weekly>]
                 Default is <daily>
            [--log-file <file name with path>]
                 Default is </opt/cloudhsm/run/cloudhsm-jce.log>
                 Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-jce.log>
            [--log-type <file | term>]
                 Default is <file>
            [-h | --help]
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                 This is the default for JCE
            [--enable-validate-key-at-init]
```

CloudHSM CLI

```
configure-cli[ .exe ]
            -a <ENI IP address>
            [--hsm-ca-cert <customerCA certificate file path>]
            [--cluster-id <cluster ID>]
            [--endpoint <endpoint>]
            [--region <region>]
            [--server-client-cert-file <client certificate file path>]
            [--server-client-key-file <client key file path>]
            [--log-level <error | warn | info | debug | trace>]
                 Default is <info>
            [--log-rotation <daily | weekly>]
                 Default is <daily>
            [--log-file <file name with path>]
                 Default for Linux is </opt/cloudhsm/run/cloudhsm-cli.log>
                 Default for Windows is <C:\\Program Files\\Amazon\\CloudHSM\
\cloudhsm-cli.log>
            [--log-type <file | term>]
                 Default setting is <file>
            [-h | --help]
```

```
            [-V | --version]
            [--disable-key-availability-check]
            [--enable-key-availability-check]
            [--disable-validate-key-at-init]
                 This is the default for CloudHSM CLI
            [--enable-validate-key-at-init]
```

Configurazioni avanzate

Per un elenco di configurazioni avanzate specifiche dello strumento di configurazione Client SDK 5, vedi Configurazioni avanzate per lo strumento di configurazione Client SDK 5.

> **Important**
>
> Dopo avere apportato le modifiche alla configurazione, è necessario riavviare l'applicazione affinché le modifiche abbiano effetto.

Esempi

Questi esempi mostrano come utilizzare lo strumento di configurazione di Client SDK 5.

Client SDK 5 Bootstrap

Example

In questo esempio viene utilizzato il parametro `-a` per aggiornare i dati HSM per il client SDK 5. Per utilizzare il parametro `-a`, è necessario disporre dell'indirizzo IP di uno dei moduli HSM del cluster.

PKCS #11 library

Per effettuare il bootstrap di un'istanza EC2 Linux per 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.

```
"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>
```

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.

```
"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 IP 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.

```
"C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" -a <The ENI IP
 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 Strumento di configurazione Client SDK 5.

Per ulteriori informazioni sul parametro `-a`, vedi the section called “Parametri”.

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`**

- Utilizza l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
"C:\Program Files\Amazon\CloudHSM\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`**

- Utilizza 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
```

JCE provider

Per effettuare il bootstrap di un'istanza EC2 Linux per Client SDK 5 con **`cluster-id`**

- Utilizza 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`**

- Utilizza l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
"C:\Program Files\Amazon\CloudHSM\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`**

- Utilizza 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`**

- Utilizza l'ID del cluster `cluster-1234567` per specificare l'indirizzo IP di un HSM nel cluster.

```
"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 API AWS CloudHSM da utilizzare per la chiamata, cosa che potrebbe essere necessaria per varie configurazioni di rete, ad esempio l'utilizzo dell' interfaccia dell'endpoint 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 personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
C:\Program Files\Amazon\CloudHSM\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
```

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 personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
"C:\Program Files\Amazon\CloudHSM\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 personalizzati

- Utilizza lo strumento di configurazione per specificare l'indirizzo IP di un HSM nel tuo cluster con una regione e un endpoint personalizzati.

```
"C:\Program Files\Amazon\CloudHSM\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 “Parametri”.

Aggiorna il certificato del client e la chiave per l'autenticazione reciproca client-server TLS

Example

Questo esempio mostra come utilizzare i parametri `server-client-cert-file` e `--server-client-key-file` 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 client-server TLS con Client SDK 5 su Linux

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
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.crt` `ssl-client.key`

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-pkcs11.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

OpenSSL Dynamic Engine

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-dyn \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

JCE provider

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-jce \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
cp ssl-client.key C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

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

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-jce.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

CloudHSM CLI

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

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

```
$ sudo cp ssl-client.crt /opt/cloudhsm/etc
sudo cp ssl-client.key /opt/cloudhsm/etc
```

2. Utilizzate lo strumento di configurazione per specificare `ssl-client.crt` e `ssl-client.key`.

```
$ sudo /opt/cloudhsm/bin/configure-cli \
            --server-client-cert-file /opt/cloudhsm/etc/ssl-client.crt \
            --server-client-key-file /opt/cloudhsm/etc/ssl-client.key
```

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

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

```
cp ssl-client.crt C:\ProgramData\Amazon\CloudHSM\ssl-client.crt
cp ssl-client.key C:\ProgramData\Amazon\CloudHSM\ssl-client.key
```

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

```
& "C:\Program Files\Amazon\CloudHSM\bin\configure-cli.exe" `
            --server-client-cert-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.crt `
            --server-client-key-file C:\ProgramData\Amazon\CloudHSM\ssl-
client.key
```

Per ulteriori informazioni sui parametri `server-client-cert-file` e `--server-client-key-file`, vedi the section called “Parametri”.

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.

```
"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
```

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.

```
"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.

```
"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 “Parametri”.

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 AWS CloudHSM su `term`. I log del client verranno inviati `stderr` e archiviati 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 name> --log-file file --
  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
  ```

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> --
  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> --
  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 “Parametri”.
  ```

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.

```
"C:\Program Files\Amazon\CloudHSM\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>
```

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.

```
"C:\Program Files\Amazon\CloudHSM\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.

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

Per ulteriori informazioni sul parametro `--hsm-ca-cert`, vedi the section called “Parametri”.

Parametri

-a ***<indirizzo IP ENI>***

Aggiunge l'indirizzo IP specificato ai file di configurazione di Client SDK 5. Immettere qualsiasi indirizzo IP ENI di un HSM dal cluster. Per ulteriori informazioni su come utilizzare questa opzione, vedi Bootstrap Client SDK 5.

Campo 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 ???.

Campo obbligatorio: no

--id-cluster***<ID cluster***

Effettua una chiamata `DescribeClusters` per trovare tutti gli indirizzi IP a interfaccia di rete elastica (ENI) HSM del cluster associati all'ID del cluster. Il sistema aggiunge gli indirizzi IP ENI ai file di configurazione AWS CloudHSM.

> **Note**
>
> Se utilizzi il parametro `--cluster-id` da un'istanza EC2 all'interno di un VPC che non ha accesso alla rete Internet pubblica, devi creare un endpoint VPC di interfaccia da collegare a AWS CloudHSM. Per ulteriori informazioni su endpoint VPC, vedi ???.

Campo obbligatorio: no

--endpoint***<endpoint>***

Specifica l'endpoint API AWS CloudHSM utilizzato per effettuare la chiamata `DescribeClusters`. È necessario impostare questa opzione insieme a `--cluster-id`.

Campo 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.

Campo obbligatorio: no

-- server-client-cert-file <client certificate file path>

Percorso al certificato client utilizzato per l'autenticazione reciproca client-server TLS.

Utilizza questa opzione solo se non desideri utilizzare la chiave predefinita e il certificato SSL/TLS inclusi in Client SDK 5. È necessario impostare questa opzione insieme a `--server-client-key-file`.

Campo obbligatorio: no

-- server-client-key-file <client key file path>

Percorso alla chiave del client utilizzato per l'autenticazione reciproca client-server TLS

Utilizza questa opzione solo se non desideri utilizzare la chiave predefinita e il certificato SSL/TLS inclusi in Client SDK 5. È necessario impostare questa opzione insieme a `--server-client-cert-file`.

Campo obbligatorio: no

--livello-log-*<errore | avviso | info | debug | tracciamento>*

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 Log Client SDK 5.

Campo obbligatorio: no

--rotazione-log*<giornaliero | settimanale>*

Specifica la frequenza con cui il sistema ruota i log. Per ulteriori informazioni, vedi Log Client SDK 5.

Campo obbligatorio: no

--file-log *<nome file con percorso>*

Specifica dove il sistema scriverà il file di log. Per ulteriori informazioni, vedi Log Client SDK 5.

Campo obbligatorio: no

--tipo-log*<terminale | file>*

Specifica se il sistema scriverà il log su un file o su un terminale. Per ulteriori informazioni, vedi Log Client SDK 5.

Campo obbligatorio: no

-h | --aiuto

Visualizza aiuto.

Campo obbligatorio: no

-v, --versione

Visualizza versione.

Campo obbligatorio: no

--disable-key-availability-check

Contrassegna per disabilitare il quorum per la disponibilità delle chiavi. Utilizza questo flag per indicare che AWS CloudHSM deve disabilitare il quorum per la disponibilità delle chiavi ed è possibile utilizzare le chiavi che esistono solo su un HSM del cluster. Per ulteriori informazioni sull'utilizzo di questo flag per impostare il quorum per la disponibilità delle chiavi, vedi ???.

Campo obbligatorio: no

--enable-key-availability-check

Contrassegna per abilitare il quorum della disponibilità delle chiavi. Utilizza questo flag per indicare che AWS CloudHSM deve utilizzare il quorum per la disponibilità delle chiavi e non consentire di utilizzare le chiavi finché tali chiavi non sono presenti su due HSM del cluster. Per ulteriori informazioni sull'utilizzo di questo flag per impostare il quorum di disponibilità delle chiavi, vedi ???.

Abilitata come impostazione predefinita.

Campo obbligatorio: no

-- disable-validate-key-at -inizializza

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.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- Operazione API DescribeClusters
- Descrivi-Cluster CLI AWS;
- Cmdlet Get-HSM2Cluster PowerShell
- Bootstrap Client SDK 5
- Endpoint VPC AWS CloudHSM
- Gestione delle impostazioni chiave di durabilità di Client SDK 5
- Log Client SDK 5

Configurazioni avanzate per lo strumento di configurazione del Client SDK 5

Lo strumento di configurazione del 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.

- Configurazioni avanzate per PKCS #11
  - Connessione a più slot con PKCS #11
  - Comandi di ripetizione dei tentativi per PKCS #11
- Configurazioni avanzate per JCE
  - Connessione a più cluster con il provider JCE
  - Comandi Nuovo tentativo per JCE
  - Estrazione delle chiavi tramite JCE
- Configurazioni avanzate per OpenSSL
  - Comandi di ripetizione dei tentativi per OpenSSL

Strumento di configurazione Client SDK 3

Utilizza lo strumento di configurazione Client SDK 3 per avviare il client daemon e configurare la CloudHSM Management Utility.

Sintassi

```
configure -h | --help
          -a <ENI IP address>
```

```
        -m [-i <daemon_id>]
        --ssl --pkey <private key file> --cert <certificate file>
        --cmu <ENI IP address>
```

Esempi

Questi esempi mostrano come utilizzare lo strumento configure.

Example : aggiornare i dati HSM per il client AWS CloudHSM e key_mgmt_util

In questo esempio viene utilizzato il parametro `-a` di configure per aggiornare i dati HSM per il client AWS CloudHSM e key_mgmt_util. Per utilizzare il parametro `-a`, è necessario disporre dell'indirizzo IP di uno degli HSM del cluster. Usa la console o la CLI AWS per ottenere l'indirizzo IP.

Per ottenere un indirizzo IP per un HSM (console)

1. Apri la console dell'AWS CloudHSM 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, nella scheda HSM, scegli uno degli indirizzi IP elencati in Indirizzo IP ENI.

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

- Ottieni l'indirizzo IP di un HSM utilizzando il comando describe-clusters da AWS CLI. Nell'output del comando, l'indirizzo IP degli HSM sono i valori di `EniIp`.

```
$  aws cloudhsmv2 describe-clusters


{
    "Clusters": [
        { ... }
            "Hsms": [
                {
...
                    "EniIp": "10.0.0.9",
...
```

```
                },
                {
...
                    "EniIp": "10.0.1.6",
...
```

: aggiornare i dati HSM

1. Prima di aggiornare il parametro `-a`, interrompere 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 client AWS CloudHSM.

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

```
C:\Program Files\Amazon\CloudHSM\bin\ configure.exe -a 10.0.0.9
```

3. Quindi, riavvia il client AWS CloudHSM. Quando viene avviato, il client utilizza l'indirizzo IP ENI nel proprio file di configurazione per eseguire query sul cluster. Quindi scrive gli indirizzi IP ENI di tutti i moduli HSM nel cluster sul file `cluster.info`.

   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 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
```

Quando il comando viene completato, i dati HSM che il client AWS CloudHSM e key_mgmt_util utilizzano risultano 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`, arresta il client AWS CloudHSM, esegui il comando `-a`, quindi riavvia il client AWS CloudHSM, come mostrato nel precedente esempio. 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

  ```
  C:\Program Files\Amazon\CloudHSM\bin\ 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

  ```
  C:\Program Files\Amazon\CloudHSM\bin\ configure.exe --cmu <IP address>
  ```

Parametri

-h | --aiuto

Visualizza la sintassi del comando.

Campo obbligatorio: sì

-a ***<indirizzo IP ENI>***

Aggiunge l'indirizzo IP dell'interfaccia di rete elastica (ENI) HSM specificata ai file di configurazione AWS CloudHSM. Inserisci l'indirizzo IP ENI di uno qualsiasi dei moduli HSM nel cluster. Non importa quale selezioni.

Per ottenere gli indirizzi IP ENI degli HSM nel cluster, utilizzare l'DescribeClustersoperazione, il AWS CLI comando describe-clusters o il cmdlet Get-HSM2Cluster. PowerShell

> **Note**
>
> Prima di eseguire il comando `-a` configure, arresta il client AWS CloudHSM. Quindi, al completamento del comando `-a`, riavvia il client AWS CloudHSM. Per ulteriori informazioni, vedi gli esempi.

Questo parametro modifica i seguenti file di configurazione:

- `/opt/cloudhsm/etc/cloudhsm_client.cfg`: utilizzato dal client AWS CloudHSM e da key_mgmt_util.
- `/opt/cloudhsm/etc/cloudhsm_mgmt_util.cfg`: utilizzato da cloudhsm_mgmt_util.

Quando viene avviato, il client AWS CloudHSM utilizza l'indirizzo IP ENI nel relativo file di configurazione per eseguire query sul cluster e aggiornare il file `cluster.info` (`/opt/cloudhsm/daemon/1/cluster.info`) con gli indirizzi IP ENI corretti per tutti i moduli HSM nel cluster.

Campo 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 viene aggiornato il parametro `-a` di configure e avviato il client AWS CloudHSM, il client daemon esegue query sul cluster e aggiorna i file `cluster.info` con gli indirizzi IP HSM corretti per tutti i moduli HSM nel 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 comando `-a` configure e di riavviare il client AWS CloudHSM prima dell'esecuzione del comando `-m`. In questo modo, i dati copiati nel file `cloudhsm_mgmt_util.cfg` dal file `cluster.info` saranno completi e accurati.

Campo obbligatorio: sì

-i

Specifica un client daemon alternativo. Il valore predefinito rappresenta il client AWS CloudHSM.

Impostazione predefinita: 1

Campo 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.

Campo obbligatorio: no

--pkey

Specifica la nuova chiave privata. Inserisci il percorso e il nome del file contenente la chiave privata.

Campo obbligatorio: sì se -ssl è 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.

Campo obbligatorio: sì se -ssl è specificato. Altrimenti non deve essere utilizzato.

-a ***<indirizzo IP ENI>***

Combina i parametri `-a` e `-m` in un unico parametro. Aggiunge l'indirizzo IP dell'interfaccia di rete elastica (ENI) HSM specificata ai file di configurazione AWS CloudHSM. 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.

Campo obbligatorio: sì

Argomenti correlati

- Configurazione di key_mgmt_util

Interfaccia a riga di comando (CLI) di CloudHSM

La CLI di CloudhSM aiuta gli amministratori a gestire gli utenti e gli utenti di criptovalute a gestire le chiavi nel proprio cluster. Include strumenti che possono essere utilizzati per creare, eliminare ed elencare utenti, modificare le password degli utenti, aggiornare l'autenticazione a più fattori (MFA) degli utenti. 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 la CLI di CloudHSM . Per un elenco definito di attributi chiave per la CLI di CloudHSM, vedere. Attributi chiavi per la CLI di CloudHSM Per informazioni su come utilizzare la CLI di CloudHSM per gestire le chiavi, consulta. Gestione delle chiavi con la CLI di CloudHSM

Per una guida rapida, vedi Nozioni di base sull'interfaccia a riga di comando di CloudHSM (CLI). Per informazioni dettagliate sui comandi di CloudHSM ed esempi di utilizzo dei comandi, vedi Riferimento per i comandi della CLI di CloudHSM.

Argomenti

Piattaforme supportate dalla CloudHSM Command Line Interface (CLI) di CloudHSM

Supporto Linux

Piattaforme supportate	Architettura x86_64	Architettura ARM
Amazon Linux 2	Sì	Sì
Amazon Linux 2023	Sì	No
CentOS 7 (7.8+)	Sì	No
Red Hat Enterprise Linux 7 (7.8+)	Sì	No
Red Hat Enterprise Linux 8 (8.3+)	Sì	No
Red Hat Enterprise Linux 9 (9.2+)	Sì	No
Ubuntu 20.04 LTS	Sì	No
Ubuntu 22.04 LTS	Sì	No

Nota: 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

Nozioni di base sull'interfaccia a riga di comando di CloudHSM (CLI)

L'interfaccia a riga di comando (CLI) di CloudHSM ti permette di gestire gli utenti del tuo cluster AWS CloudHSM. Utilizza questo argomento per iniziare con le attività di base di gestione degli utenti dell'HSM, come la creazione di utenti, l'elenco degli utenti e la connessione della CLI di CloudHSM al cluster.

Installa la CLI CloudhSM

Utilizza i seguenti comandi per scaricare e installare la CLI di CloudHSM.

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
```

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
```

CentOS 7 (7.8+)

CentOS 7 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
```

RHEL7 (7.8+)

RHEL 7 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
```

RHEL8 (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
```

RHEL9 (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
```

Ubuntu 20.04 LTS

Ubuntu 20.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-cli_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-cli_latest_u20.04_amd64.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
```

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
```

Windows Server 2019

Per Windows Server 2019 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 IP 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.

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

Utilizzo della CLI di CloudHSM

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

   Linux

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

   Windows

   ```
   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. Utilizza il comandologout per disconnetterti dal cluster dell'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 interattive e a comando singolo

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

```
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.

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 nei moduli HSM, digita user list.

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

Per terminare la sessione della CLI di CloudHSM, esegui il seguente comando:

```
aws-cloudhsm > quit
```

Modalità di comando singolo

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 chiavi 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 Chiave.

Attributi supportati

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.

La tabella seguente elenca gli attributi delle chiavi, i valori possibili, i valori predefiniti e le note correlate. 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 set chiavi-attributo	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` ,	No	Sì

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
	`public-key` e `private-key` .		
`curve`	Curva ellittica usata per generare la coppia di chiavi EC. Valori validi: `secp224r1` , `secp256r1` , `prime256v1` , `secp384r1` , `secp256k1` , and `secp521r1`	No	Da impostare con RSA, non può essere impostato con EC
`decrypt`	Impostazione predefinita: `False`	Sì	Sì
`derive`	Impostazione predefinita: `False`	Sì	Sì
`destroyable`	Impostazione predefinita: `True`	Sì	Sì
`ec-point`	Per le chiavi EC, codifica DER del valore ANSI X9.62 ECPoint "Q" in formato esadecimale. Per altri tipi di chiavi, questo attributo non esiste.	No	No

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
`encrypt`	Impostazione predefinita: `False`	Sì	Sì
`extractable`	Impostazione predefinita: `True`	No	Sì
`id`	Impostazione predefinita: Vuoto	No	Sì
`key-length-bytes`	Necessario per generare una chiave AES. Valori validi: `16`, `24`, e `32` byte.	No	No
`key-type`	Valori possibili: `aes`, `rsa` e `ec`	No	Sì
`label`	Impostazione predefinita: Vuoto	Sì	Sì
`local`	Impostazione predefinita: `True` per le chiavi generate nell'HSM, `False` per le chiavi importate nell'HSM.	No	No
`modifiable`	Impostazione predefinita: `True`	No	No

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
`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
`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 mai stata impostata su. `True`	No	No
`private`	Impostazione predefinita: `True`	No	Sì

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
`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.
`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: `False`	No	Sì

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
`trusted`	Impostazione predefinita: `False`	Sì	No
`unwrap`	Impostazione predefinita: `False`	Sì	Sì
`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
`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
`wrap-template`	I valori devono utilizzare il modello di attributo per abbinare la chiave sottoposta a wrapping usando questa chiave di wrapping.	Sì	No

Attributo della CLI del CloudHSM	Valore	Modificabile con set chiavi-attributo	Da impostare al momento della creazione della chiave
`wrap-with-trusted`	Impostazione predefinita: `False`	Sì	Sì

Ulteriori dettagli

Valore di controllo

Il valore di controllo 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.

L'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'hash SHA-1 della chiave pubblica.
- Chiavi HMAC: KCV per le chiavi HMAC non è attualmente supportato.

Argomenti correlati

- Chiave
- Riferimento per i comandi della CLI di CloudHSM

Migrazione da Client SDK 3 CMU e KMU a Client SDK 5 CloudHSM CLI

Utilizza questo argomento per migrare i flussi di lavoro che utilizzano gli strumenti da riga di comando di Client SDK 3, la CloudHSM Management Utility (CMU) e la Key Management Utility (KMU), per utilizzare invece lo strumento da riga di comando di Client SDK 5, CloudHSM CLI.

InAWS CloudHSM, le applicazioni dei clienti eseguono operazioni crittografiche utilizzando il Client Software Development Kit (SDK). AWS CloudHSM Client SDK 5 è l'SDK principale che continua ad

avere nuove funzionalità e supporto per la piattaforma. Questo argomento fornisce dettagli specifici sulla migrazione da Client SDK 3 a Client SDK 5 per gli strumenti da riga di comando.

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. Interfaccia a riga di comando (CLI) di CloudHSM Le operazioni di gestione degli utenti sono disponibili nei sottocomandi e. Utente Quorum 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.

> **Note**
>
> Se in Client SDK 3 hai fatto affidamento sulle syncUser funzionalità per syncKey la sincronizzazione tra cluster, continua a utilizzare la CMU. La CLI di CloudhSM in Client SDK 5 attualmente non supporta questa funzionalità.

Per istruzioni sulla migrazione a Client SDK 5, consulta. Migrazione da Client SDK 3 a Client SDK 5 Per i vantaggi della migrazione, consulta. Vantaggi del Client SDK 5

Riferimento per i comandi della CLI di CloudHSM

La CLI di CloudHSM aiuta gli admin 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 sull'interfaccia a riga di comando di CloudHSM (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	Descrizione	Tipo di utente
attivazione del cluster	Attiva un cluster CloudHSM e conferma che il cluster è nuovo. Questa deve essere	Amministratore non attivato

Comando	Descrizione	Tipo di utente
	fatto prima di poter eseguire qualsiasi altra operazione.	
segno crittografico ecdsa	Genera una firma utilizzando una chiave privata EC e il meccanismo di firma ECDSA.	Crypto user (CU)
segno crittografico rsa-pkcs	Genera una firma utilizzando una chiave privata RSA e il meccanismo di firma RSA-PKCS.	CU
segno crittografico rsa-pkcs-pss	Genera una firma utilizzando una chiave privata RSA e il meccanismo di firma RSA-PKCS-PSS.	CU
verifica crittografica 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

Comando	Descrizione	Tipo di utente
verifica crittografica rsa-pkcs	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 RSA-PKCS. 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 determina to meccanismo di firma.	CU
verifica crittografica 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 firma RSA-PKCS-PSS. 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 determina to meccanismo di firma.	CU
Eliminazione chiave	Elimina una chiave dal cluster dell'AWS CloudHSM.	CU
Generazione chiavi-file	Genera un file chiave nel cluster dell'AWS CloudHSM.	CU

Comando	Descrizione	Tipo di utente
chiave rsa generate-asymmetric-pair	Consente di generare una coppia di chiavi asimmetri che RSA nel cluster dell'AWS CloudHSM.	CU
chiave ec generate-asymmetric-pair	Genera una coppia di chiavi asimmetriche a curva ellittica (EC) nel cluster dell'AWS CloudHSM.	CU
Generazione chiavi-simmetriche-coppia aes	Genera una chiave AES simmetrica nel cluster dell'AWS CloudHSM.	CU
Generazione chiavi-simmetriche generiche-segrete	Genera una chiave simmetrica generica segreta nel cluster dell'AWS CloudHSM.	CU
chiave di importazione 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 presenti nel cluster dell'AWS CloudHSM.	CU
Impostazione chiavi-attributi	Imposta gli attributi delle chiavi nel cluster dell'AWS CloudHSM.	I CU possono eseguire questo comando, gli admin possono impostare l'attributo affidabile.
Condivisione chiave	Condivide una chiave con altri CU nel cluster dell'AWS CloudHSM.	CU

Comando	Descrizione	Tipo di utente
Annulla condivisione chiave	Annulla la condivisione di una chiave con altri CU del cluster dell'AWS CloudHSM.	CU
chiave unwrap aes-gcm	Scompone una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping AES-GCM.	CU
scartare le chiavi 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
chiave unwrap aes-pkcs5-pad	Scompone una chiave di payload utilizzando la chiave di wrapping AES e il meccanismo di srotolamento AES-PKCS5-PAD.	CU
scartare le chiavi 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
scartare i tasti 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

Comando	Descrizione	Tipo di utente
chiave unwrap rsa-aes	Scompone una chiave di payload utilizzando una chiave privata RSA e il meccanismo di unwrapping RSA-AES.	CU
chiave unwrap rsa-oaep	Scompone una chiave di payload utilizzando la chiave privata RSA e il meccanism o di decompressione RSA-OAEP.	CU
chiave unwrap rsa-pkcs	Scompone una chiave di payload utilizzando la chiave privata RSA e il meccanism o di decompressione RSA-PKCS.	CU
portachiavi aes-gcm	Racchiude una chiave di payload utilizzando una chiave AES sull'HSM e sul meccanismo di wrapping AES-GCM.	CU
portachiavi aes-no-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e il meccanismo di wrapping AES-NO-PAD.	CU
portachiavi aes-pkcs5-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e il meccanismo di wrapping AES-PKCS5-PAD.	CU

Comando	Descrizione	Tipo di utente
portachiavi aes-zero-pad	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e il meccanismo di wrapping AES-ZERO-PAD.	CU
involucro per chiavi cloudhsm-aes-gcm	Avvolge una chiave di payload utilizzando una chiave AES sull'HSM e il meccanismo di wrapping CLOUDHSM-AES-GCM.	UC
portachiavi rsa-aes	Racchiude una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e il meccanismo di wrapping RSA-AES.	CU
portachiavi rsa-oaep	Racchiude una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e il meccanismo di wrapping RSA-OAEP.	CU

Comando	Descrizione	Tipo di utente
Il key wrap rsa-pkcs comando esegue il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sull'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 clusterAWS CloudHSM. È possibile generare una coppia di chiavi RSA utilizzando il Generazione chiavi-asimmetriche-coppia comando e l'`wrap`attributo impostati su. `true` Tipo di utente I seguenti tipi di utenti possono eseguire questo comando. • Utenti di crittografia (CU) Requisiti • Per eseguire questo	Racchiude una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e il meccanismo di wrapping RSA-PKCS.	CU

Comando	Descrizione	Tipo di utente
Login	Accedi al tuo cluster AWS CloudHSM.	Admin, crypto user (CU) e utente dell'applicazione (AU)
Disconnessione	Disconnettiti dal cluster dell'AWS CloudHSM.	Admin, CU e utente appliance (AU)
Quorum token-firma elimina	Elimina uno o più token per un servizio autorizzato dal quorum.	Admin
Quorum token-firma generazione	Genera un token per un servizio autorizzato dal quorum.	Admin
Quorum token-firma elenco	Elenca tutti i token-firma del quorum presenti nel cluster di CloudHSM.	Tutti [1], compresi gli utenti non autenticati. L'accesso non è necessario.
firma del token del quorum 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-firma elenco-timeout	Ottiene il periodo di timeout del token in secondi per tutti i tipi di token.	Admin e utente crittografico
firma-token del quorum set-quorum-value	Imposta un nuovo valore di quorum per un servizio autorizzato dal quorum.	Admin
Quorum token-firma impostazione-timeout	Imposta il periodo di timeout del token in secondi per ogni tipo di token.	Admin

Comando	Descrizione	Tipo di utente
Modifica utente-mfa	Modifica la strategia di autenticazione a più fattori (MFA) di un utente.	Admin, CU
Modifica utente-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
Crea utente	Crea un utente nel tuo cluster dell'AWS CloudHSM.	Admin
Elimina utente	Elimina un utente dal cluster dell'AWS CloudHSM.	Admin
Elenco utenti	Elenca gli utenti del cluster dell'AWS CloudHSM.	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.

cluster

cluster 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:

- attivazione del cluster

- cluster hsm-info

attivazione del cluster

Utilizza il comando cluster activate nella CLI di CloudHSM per attivare un nuovo cluster. È 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.

- amministratore 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:
        --password <PASSWORD>
            Optional: Plaintext activation password If you do not include this argument
 you will be
            prompted for it.
```

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

- creazione utente
- Elimina utente
- Modifica della password di un utente

cluster hsm-info

Usa il comando cluster hsm-info nella CLI di CloudHSM per elencare gli HSM nel tuo cluster. Non è necessario che tu abbia eseguito l'accesso alla CLI di CloudHSM per eseguire questo comando.

> **Note**
>
> Se aggiungi o elimini moduli HSM, aggiorna i file di configurazione utilizzati dal client AWS CloudHSM e dagli strumenti a riga di comando. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM presenti nel cluster.

Tipo 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:
--config <CONFIG> [default: /opt/cloudhsm/etc/cloudhsm-cli.cfg]
```

Esempio

Questo comando elenca i moduli HSM presenti nel cluster dell'AWS CloudHSM.

```
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.
- Hardware-version-major: La versione principale dell'hardware.
- Hardware-version-minor: La versione secondaria dell'hardware.
- Firmware-Version-Major: La versione principale del firmware.
- Firmware-Version-minor: La versione secondaria del firmware.
- Firmware-build-number: il numero di build del firmware.
- Firmware-id: l'ID del firmware, che include le versioni principali e secondarie insieme al build.

Argomenti correlati

- attivazione del cluster

crypto

cryptoè 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:

- segno crittografico
  - segno crittografico ecdsa
  - segno crittografico rsa-pkcs
  - segno crittografico rsa-pkcs-pss
- verifica crittografica
  - verifica crittografica ecdsa
  - verifica crittografica rsa-pkcs
  - verifica crittografica rsa-pkcs-pss

segno crittografico

crypto signè 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:

- segno crittografico ecdsa
- segno crittografico rsa-pkcs
- segno crittografico rsa-pkcs-pss

Per utilizzarlocrypto sign, è necessario disporre di una chiave privata nell'HSM. È possibile generare una chiave privata con i seguenti comandi:

- chiave generate-asymmetric-pair ec
- chiave generate-asymmetric-pair rsa

segno crittografico ecdsa

Il crypto sign ecdsa comando genera 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 clusterAWS CloudHSM. È possibile generare una chiave privata EC utilizzando il chiave generate-asymmetric-pair ec comando con l'`sign`attributo impostato su`true`.

> **Note**
>
> Le firme possono essere verificate AWS CloudHSM con verifica crittografica sottocomandi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come generare una firma utilizzando crypto sign ecdsa il meccanismo di firma e la funzione hash ECDSA. SHA256 Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per 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: generazione di 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=="
  }
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specificate 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

Campo obbligatorio: sì

<KEY_FILTER>

Riferimento 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 CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo obbligatorio: sì

Argomenti correlati

- segno crittografico
- verifica crittografica

segno crittografico rsa-pkcs

Il crypto sign rsa-pkcs comando genera 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 chiave generate-asymmetric-pair rsa comando con l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate AWS CloudHSM con verifica crittografica sottocomandi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come generare una firma utilizzando crypto sign rsa-pkcs il meccanismo di firma e la funzione hash RSA-PKCS. SHA256 Questo comando utilizza una chiave privata nell'HSM.

Example Esempio: generare una firma per 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": "0x0000000000007008db",
    "signature": "XJ7mRyHnDRYrDWTQuuNb
+5mhoXx7VTsPMjgOQW4iMN7E42eNHj2Q0oovMmBdHUEH0F4HYG8FBJOBhvGuM8J/
z6y41GbowVpUT6WzjnIQs79K9i7i6oR1TYjLnIS3r/zkimuXcS8/ZxyDzru+GO9BUT9FFU/
of9cvu4Oyn6a5+IXuCbKNQs19uASuFARUTZ0a0Ny1CB1MulxUpqGTmI91J6evlP7k/2khwDmJ5E8FEar5/
Cvbn9t21p3Uj561ngTXrYbIZ2KHpef9jQh/cEIvFLG61sexJjQi8EdTxeDA
+I3ITO0qrvvESvA9+Sj7kdG2ceIicFS8/8LwyxiIC31UHQ=="
  }
}
```

Example Esempio: generazione di 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=="
  }
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specificate la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite i dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Campo obbligatorio: sì

<KEY_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.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo obbligatorio: sì

Argomenti correlati

- segno crittografico
- verifica crittografica

segno crittografico rsa-pkcs-pss

Il crypto sign rsa-pkcs-pss comando genera una firma utilizzando una chiave privata RSA e il meccanismo di `RSA-PKCS-PSS` firma.

Per utilizzare il crypto sign rsa-pkcs-pss comando, è necessario innanzitutto disporre di una chiave privata RSA nel clusterAWS CloudHSM. È possibile generare una chiave privata RSA utilizzando il chiave generate-asymmetric-pair rsa comando con l'`sign`attributo impostato su. `true`

> **Note**
>
> Le firme possono essere verificate AWS CloudHSM con verifica crittografica sottocomandi.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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 --key-filter [<KEY_FILTER>...] --hash-function
 <HASH_FUNCTION> --mgf <MGF> --salt-length <SALT_LENGTH> <--data-path <DATA_PATH>|--
data <DATA>>

Options:
      --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
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come 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 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": "0x00000000007008db",
```

```
    "signature": "H/z1rYVMzNAa31K4amE5MTiwGxDdCTgQXCJXRBKVOVm7ZuyI0fGE4sT/BUN
+977mQEV2TqtWpTsiF2IpwGM1VfSBRt7h/g4o6YERm1tTQLl7q+AJ7uGGK37zCsWQrAo7Vy8NzPShxekePo/
ZegrB1aHWN1fE8H3IPUKqLuMDI9o1Jq6kM986ExS7YmeOIclcZkyykTWqHLQVL2C3+A2bHJZBqRcM5XoIpk8HkPypjpN
+m4FNUds30GAemoOMl6asSrEJSthaZWV53OBsDOqzA8Rt8JdhXS+GZp3vNLdL1OTBELDPweXVgAu4dBX0FOvpw/
gg6sNvuaDK4YOBv2fqKg=="
  }
}
```

Example Esempio: generazione di 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=="
  }
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specificate la posizione dei dati da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite i dati)

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Campo obbligatorio: sì

<KEY_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.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo 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

Campo obbligatorio: sì

<SALT_LENGTH>

Specificate la lunghezza del sale.

Campo obbligatorio: sì

Argomenti correlati

- segno crittografico
- verifica crittografica

Argomenti correlati

- verifica crittografica

verifica crittografica

crypto verifyè 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 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 a un determinato meccanismo di firma.

> **Note**
>
> I file possono essere firmati in AWS CloudHSM con l'operazione segno crittografico.

verifica crittografica ecdsa

Il crypto verify ecdsa comando viene utilizzato per completare le seguenti operazioni:

- 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 crittograficamente in base a una determinata chiave pubblica ecdsa e a un determinato meccanismo di firma.

Per utilizzare il crypto verify ecdsa comando, è necessario innanzitutto disporre di una chiave pubblica EC nel cluster. AWS CloudHSM È possibile importare una chiave pubblica EC utilizzando il chiave di importazione pem comando con l'`verify`attributo impostato su`true`.

> **Note**
>
> È possibile generare una firma nella CLI di CloudHSM con sottocomandi. segno crittografico

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come 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 relazione con falsi firmi

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 che si trova 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"
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specificate 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

Campo obbligatorio: sì

<KEY_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.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatorio: Sì (a meno che non sia fornita tramite il percorso della firma)

<SIGNATURE_PATH>

Specificate la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

Argomenti correlati

- segno crittografico
- verifica crittografica

verifica crittografica rsa-pkcs

Il crypto verify rsa-pkcs comando viene utilizzato per completare le seguenti operazioni:

- 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` 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.

Per utilizzare il crypto verify rsa-pkcs comando, è necessario innanzitutto disporre di una chiave pubblica RSA nel cluster. AWS CloudHSM

> **Note**
>
> È possibile generare una firma utilizzando la CLI CloudhSM con i sottocomandi. segno crittografico

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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
  -h, --help
```

```
          Print help
```

Esempio

Questi esempi mostrano come verificare una firma generata utilizzando il meccanismo di firma e la funzione hash RSA-PKCS. crypto verify rsa-pkcs 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 relazione con falsi firmi

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 che si trova 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"
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Speciifica 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

Campo obbligatorio: sì

<KEY_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.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatorio: Sì (a meno che non sia fornita tramite il percorso della firma)

<SIGNATURE_PATH>

Specificate la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

Argomenti correlati

- segno crittografico
- verifica crittografica

verifica crittografica rsa-pkcs-pss

Il crypto sign rsa-pkcs-pss comando viene utilizzato per completare le seguenti operazioni.

- 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 RSA-PKCS-PSS.
- 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.

Per utilizzare il crypto verify rsa-pkcs-pss comando, è necessario innanzitutto disporre di una chiave pubblica RSA nel cluster. AWS CloudHSM È possibile importare una chiave pubblica RSA utilizzando il comando key import pem (ADD UNWRAP LINK HERE) con l'`verify`attributo impostato su. `true`

> **Note**
>
> È possibile generare una firma utilizzando la CLI CloudhSM con i sottocomandi. segno crittografico

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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
      --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 verificare una firma generata utilizzando crypto verify rsa-pkcs-pss il meccanismo di firma e la funzione hash RSA-PKCS-PSS. 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
 signature-file
{
  "error_code": 0,
  "data": {
    "message": "Signature verified successfully"
  }
}
```

Example Esempio: dimostrare una relazione con falsi firmi

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 che si trova 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"
}
```

Argomenti

<DATA>

Dati codificati Base64 da firmare.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso dati)

<DATA_PATH>

Specificate 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

Campo obbligatorio: sì

<KEY_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.

Per un elenco degli attributi chiave CLI di CloudHSM supportati, consulta Attributi chiave per CloudHSM CLI.

Campo 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

Campo obbligatorio: sì

<SIGNATURE>

Firma codificata Base64.

Obbligatorio: Sì (a meno che non sia fornita tramite il percorso della firma)

<SIGNATURE_PATH>

Specificate la posizione della firma.

Obbligatorio: Sì (a meno che non sia fornito tramite il percorso della firma)

Argomenti correlati

- segno crittografico
- verifica crittografica

Chiave

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 chiave
- Generazione chiavi-file
- chiave generate-asymmetric-pair
  - chiave generate-asymmetric-pair rsa
  - chiave ec generate-asymmetric-pair
- Generazione chiavi-simmetriche
  - Generazione chiavi-simmetriche aes
  - Generazione chiavi-simmetriche generiche-segrete
- chiave di importazione pem
- Elenco delle chiavi
- Set di chiavi-attributi
- Condivisione chiave
- Annullare condivisione chiave
- scartare i tasti
- portachiavi

Elimina chiave

Utilizza il comando key delete nella CLI di CloudHSM per eliminare una chiave da un cluster dell'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 di crittografia (CU)

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 --filter [<FILTER>...]

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 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"
  }
}
```

Argomenti

<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 delle chiavi supportati dalla CLI di CloudHSM, vedi Attributi chiavi per la CLI di CloudHSM

Campo obbligatorio: sì

Argomenti correlati

- Elenco delle chiavi
- Generazione chiavi-file
- Annullare condivisione chiave
- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

Generazione chiavi-file

Il key generate-file comando esporta una chiave asimmetrica dall'HSM. Se la destinazione è una chiave privata, il riferimento alla chiave privata verrà esportato in formato PEM falso. Se la destinazione è una chiave pubblica, i byte della chiave pubblica verranno esportati in formato PEM.

Il file PEM falso, che non contiene l'effettivo materiale della chiave privata ma fa invece riferimento alla chiave privata nell'HSM, può essere utilizzato per stabilire l'offload SSL/TLS dal server Web a. AWS CloudHSM Per ulteriori informazioni, vedi Offload SSL/TLS.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (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:
      --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)

      --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')
```

Esempio

Questo esempio mostra come utilizzare key generate-file per generare un file di chiavi nel cluster dell'AWS CloudHSM.

Example

```
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"
  }
}
```

Argomenti

<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 chiavi per la CLI di CloudHSM

Campo obbligatorio: no

<CODIFICA>

Specifica il formato di codifica per il file di chiavi

Campo obbligatorio: sì

<PERCORSO>

Specifica il percorso del file in cui verrà scritto il file di chiavi

Campo obbligatorio: sì

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi
- Generazione chiavi-asimmetriche-coppia
- Generazione chiavi-simmetriche

Generazione chiavi-asimmetriche-coppia

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
- Generazione chiavi-asimmetriche-coppia rsa

chiave generate-asymmetric-pair ec

Utilizza il comando key asymmetric-pair ec nella CLI di CloudHSM per generare una coppia di chiavi asimmetriche a curva ellittica (EC) nel cluster dell'AWS CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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]
      --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
  -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": [],
       "cluster-coverage": "session"
     },
     "attributes": {
       "key-type": "ec",
       "label": "ec-public-key-example",
       "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":
"0x047096513df542250a6b228fd9cb67fd0c903abc93488467681974d6f371083fce1d79da8ad1e9ede745fb9f38a
```

```
        "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": "ec-private-key-example",
        "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": 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 token=true encrypt=true \
    --private-attributes token=true decrypt=true
 {
  "error_code": 0,
  "data": {
    "public_key": {
      "key-reference": "0x00000000002806eb",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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": [],
        "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"
      }
    }
  }
}
```

Argomenti

<CURVA>

Specifica l'identificatore per la curva ellittica.

- prime256v1
- secp256r1
- secp224r1
- secp384r1
- secp256k1
- secp521r1

Campo obbligatorio: sì

<ATTRIBUTO_CHIAVE_PUBBLICA>

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 `token=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<ETICHETTA_PUBBLICA>

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.

Campo obbligatorio: sì

<ATTRIBUTI_CHIAVE_PRIVATA>

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 `token=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<ETICHETTA_PRIVATA>

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.

Campo 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 dopo la fine 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).

Campo obbligatorio: no

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

chiave generate-asymmetric-pair rsa

L'utilizzo del comando key generate-asymmetric-pair rsa genera una coppia di chiavi asimmetriche RSA nel tuo cluster dell'AWS CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-public-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "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": 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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "rsa-private-key-example",
        "id": "",
        "check-value": "0x498e1f",
        "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": 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 token=true encrypt=true \
--private-attributes token=true decrypt=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": "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": [],
        "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
      }
    }
```

```
  }
}
```

Argomenti

<DIMENSIONI_MODULO_BITS>

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Campo obbligatorio: sì

<ATTRIBUTI_CHIAVE_PRIVATA>

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 `token=true`)

Per un elenco degli attributi di chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<ETICHETTA_PRIVATA>

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.

Campo obbligatorio: sì

<ESPONENTE_PUBBLICO>

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Campo obbligatorio: sì

<ATTRIBUTO_CHIAVE_PUBBLICA>

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 `token=true`)

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<ETICHETTA_PUBBLICA>

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.

Campo 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 dopo la fine 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).

Campo obbligatorio: no

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

Generazione chiavi-simmetriche

key generate-symmetric è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando che genera chiavi simmetriche. Attualmente, questa categoria comprende i seguenti comandi:

- Generazione chiavi-simmetriche aes
- Generazione chiavi-simmetriche generiche-segrete

Generazione chiavi-simmetriche aes

Il comando key generate-symmetric aes genera una chiave AES simmetrica nel cluster dell'AWS CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "aes",
        "label": "example-aes",
        "id": "",
        "check-value": "0x9b94bd",
        "class": "secret-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": 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": [],
        "cluster-coverage": "session"
      },
      "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
      }
    }
  }
}
```

Argomenti

<ATTRIBUTI_CHIAVE>

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 `token=true`).

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<DIMENSIONI-CHIAVE-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- 16, 24 e 32

Campo obbligatorio: sì

<ETICHETTA>

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.

Campo 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 dopo la fine 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).

Campo obbligatorio: no

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

Generazione chiavi-simmetriche generiche-segrete

Il comando key generate-asymmetric-pair genera una chiave simmetrica generica segreta nel cluster AWS CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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": [],
        "cluster-coverage": "session"
      },
```

```
      "attributes": {
        "key-type": "generic-secret",
        "label": "example-generic-secret",
        "id": "",
        "class": "secret-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": 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 token=true encrypt=true
{
  "error_code": 0,
  "data": {
    "key": {
      "key-reference": "0x00000000002e08fd",
      "key-info": {
        "key-owners": [
          {
```

```
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "cluster-coverage": "session"
      },
      "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
      }
    }
  }
}
```

Argomenti

<ATTRIBUTI_CHIAVE>

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 `token=true`).

Per un elenco degli attributi delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

<DIMENSIONI-CHIAVE-BYTES>

Specifica le dimensioni della chiave in byte.

Valori validi:

- Da 1 a 800

Campo obbligatorio: sì

<ETICHETTA>

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.

Campo 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 dopo la fine 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).

Campo obbligatorio: no

Argomenti correlati

- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

chiave di importazione pem

Il key import pem comando in AWS CloudHSM importa una chiave di formato PEM in un HSM. È possibile utilizzarlo per importare le chiavi pubbliche generate al di fuori del HSM.

> **Note**
>
> Utilizzate il Generazione chiavi-file 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.

- Utenti di crittografia (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:
      --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: importare 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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
        "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": [],
        "cluster-coverage": "session"
      },
      "attributes": {
        "key-type": "rsa",
        "label": "example-imported-key-with-attributes",
        "id": "0x",
        "check-value": "0x99fe93",
        "class": "public-key",
        "encrypt": false,
        "decrypt": false,
        "token": false,
```

```
        "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"
  }
}
```

Argomenti

<PERCORSO>

Speciifica il percorso del file in cui si trova il file chiave.

Campo obbligatorio: sì

<ETICHETTA>

Specifica un'etichetta definita dall'utente per la chiave importata. La dimensione massima per l'`label` è di 126 caratteri.

Campo obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave avvolta.

Valori possibili:

- ec-public
- rsa-pubblico

Campo obbligatorio: sì

<IMPORT_KEY_ATTRIBUTES>

Specificate 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,`token=true`). Per un elenco degli attributi di chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: no

Argomenti correlati

- segno crittografico
- verifica crittografica

Elenco delle chiavi

Il comando key list trova tutte le chiavi per l'utente corrente presente nel cluster dell'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.

- Utenti di crittografia (CU)

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:
      --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
 -h, --help
        Print help
```

Esempi

Nell'esempio seguente vengono illustrati tipi diversi di esecuzione del comando key list.

Example Esempio: ricerca di tutte le chiavi - impostazione predefinita

Questo comando elenca le chiavi dell'utente connesso presente nel cluster dell'AWS CloudHSM.

> **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: ricerca di tutte le chiavi - verboso

L'output include le chiavi che l'utente possiede e condivide e tutte le chiavi pubbliche dei moduli HSM.

> **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": [],
          "cluster-coverage": "session"
        },
```

```
        "attributes": {
          "key-type": "ec",
          "label": "ec-test-private-key",
          "id": "",
          "check-value": "0x2a737d",
          "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": 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": [],
          "cluster-coverage": "session"
        },
        "attributes": {
          "key-type": "ec",
```

```
          "label": "ec-test-public-key",
          "id": "",
          "check-value": "0x2a737d",
          "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":
 "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": [],
          "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": [],
          "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": [],
          "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": [],
          "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 Utilizzare la CLI di CloudHSM per filtrare le chiavi.

Argomenti

<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 chiavi per la CLI di CloudHSM

Campo obbligatorio: no

<ELEMENTI_MASSIMO>

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.

Campo obbligatorio: no

<TOKEN_INIZIALE>

Token per specificare dove iniziare l'impaginazione. Si tratta del token-successivo da una risposta precedentemente troncata.

Campo obbligatorio: no

<VERBOSO>

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.

Campo obbligatorio: no

Argomenti correlati

- Elimina chiave
- Generazione chiavi-file
- Annullare condivisione chiave
- Attributi chiavi per la CLI di CloudHSM
- Utilizzare la CLI di CloudHSM per filtrare le chiavi

Set di chiavi-attributi

Usa il comando key set-attribute per impostare gli attributi delle chiavi nel cluster dell'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 chiavi per la CLI di CloudHSM.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Solo gli utenti di crittografia (CU) possono eseguire il 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:
```

```
      --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
 -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": "0x0000000000006400ec",
        "key-info": {
          "key-owners": [
            {
              "username": "bob",
              "key-coverage": "full"
            }
          ],
          "shared-users": [],
          "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
  }
}
```

Argomenti

<ATTRIBUTI_CHIAVE>

Specifica il nome dell'attributo della chiave.

Campo obbligatorio: sì

<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 chiavi per la CLI di CloudHSM

Campo obbligatorio: no

<VALORE_ATTRIBUTO_CHIAVE>

Specifica il valore dell'attributo della chiave.

Campo obbligatorio: sì

<RIFERIMENTO_CHIAVE>

Una rappresentazione esadecimale o decimale della chiave. (ad esempio un handle della chiave).

Campo obbligatorio: no

Argomenti correlati

- Utilizzare la CLI di CloudHSM per filtrare le chiavi
- Attributi chiavi per la CLI di CloudHSM

Condivisione chiave

Il comando key share condivide una chiave con altri CU del cluster dell'AWS CloudHSM.

Solo il CU che ha creato la chiave e di conseguenza la possiede può condividere la chiave. Gli utenti con cui è condivisa la chiave possono utilizzarla in operazioni di crittografia, ma non possono eliminarla, esportarla, 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 di crittografia (CU)

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:
      --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

  -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"
            }
          ],
          "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`

Argomenti

<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 delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: sì

<NOME UTENTE>

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.

Campo obbligatorio: sì

<RUOLO>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. Per ottenere il ruolo dell'utente, utilizzare il comando lista utenti. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Informazioni sugli utenti HSM.

Campo obbligatorio: sì

Argomenti correlati

- Utilizzare la CLI di CloudHSM per filtrare le chiavi
- Attributi chiavi per la CLI di CloudHSM

Annullare condivisione chiave

Il comando key unshare annulla la condivisione di una chiave con altri CU del cluster dell'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 di crittografia (CU)

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:
      --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

  -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"
          }
        ],
        "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"
          },
        ],
        "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

<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 delle chiavi supportati, vedi Attributi chiavi per la CLI di CloudHSM.

Campo obbligatorio: sì

<NOME UTENTE>

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.

Campo obbligatorio: sì

<RUOLO>

Specifica il ruolo assegnato a questo utente. Questo parametro è obbligatorio. Per ottenere il ruolo dell'utente, utilizzare il comando lista utenti. Per informazioni dettagliate sui tipi di utente su un HSM, vedi Informazioni sugli utenti HSM.

Campo obbligatorio: sì

Argomenti correlati

- Utilizzare la CLI di CloudHSM per filtrare le chiavi
- Attributi chiavi per la CLI di CloudHSM

scartare i tasti

Il comando key unwrap parent nella CLI di CloudHSM importa una chiave privata criptata (avvolta) simmetrica o asimmetrica da un file e nell'HSM. Questo comando è progettato per importare chiavi crittografate che sono state racchiuse dal portachiavi comando, ma può anche essere usato per scartare 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.

- chiave unwrap aes-gcm
- scartare le chiavi aes-no-pad
- chiave unwrap aes-pkcs5-pad
- scartare le chiavi aes-zero-pad
- scartare i tasti cloudhsm-aes-gcm
- chiave unwrap rsa-aes
- chiave unwrap rsa-oaep
- chiave unwrap rsa-pkcs

chiave unwrap aes-gcm

Il key unwrap aes-gcm comando decomprime 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 di crittografia (CU)

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:
      --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
      --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
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap aes-gcm comando utilizzando una chiave AES con il valore dell'unwrapattributo `true` impostato su.

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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<AAD>

Come valore AAD (Additional Authenticated Data) di GCM, in esadecimale.

Campo obbligatorio: no

<TAG_LENGTH_BITS>

Come lunghezza del tag GCM in bit.

Campo obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

<IV>

Valore iniziale usato per avvolgere la chiave, in esadecimale.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

scartare le chiavi aes-no-pad

Il key unwrap aes-no-pad comando scarta una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `AES-NO-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-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 di crittografia (CU)

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:
      --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
      --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
  -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 `true` impostato su.

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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

chiave unwrap aes-pkcs5-pad

Il key unwrap aes-pkcs5-pad comando scarta una chiave di payload utilizzando la chiave di wrapping AES e il meccanismo di dewrapping. `AES-PKCS5-PAD`

Le chiavi non imballate 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 di crittografia (CU)

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:
      --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
      --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
  -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'unwrapattributo `true` impostato su.

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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

scartare le chiavi aes-zero-pad

Il key unwrap aes-zero-pad comando scarta una chiave di payload nel cluster utilizzando la chiave di wrapping AES e il meccanismo di unwrapping. `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-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 di crittografia (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:
      --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
      --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
  -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: scartare una chiave di payload dai dati chiave avvolti 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": [],
        "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-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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

scartare i tasti cloudhsm-aes-gcm

Il key unwrap cloudhsm-aes-gcm comando decomprime una chiave di payload 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 di crittografia (CU)

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:
      --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
      --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
  -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 `true` impostato su.

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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<AAD>

Come valore AAD (Additional Authenticated Data) di GCM, in esadecimale.

Campo obbligatorio: no

<TAG_LENGTH_BITS>

Come lunghezza del tag GCM in bit.

Campo obbligatorio: sì

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-private`generic-secret,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

chiave unwrap rsa-aes

Il key unwrap rsa-aes comando decomprime una chiave di payload utilizzando una chiave privata RSA e il meccanismo di decompressione. `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`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
      --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
  -h, --help
          Print help
```

Esempio

Questi esempi mostrano come utilizzare il key unwrap rsa-aes comando utilizzando la chiave privata RSA con il unwrap valore dell'attributo impostato su. true

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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Campo 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

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

chiave unwrap rsa-oaep

Il key unwrap rsa-oaep comando rimuove una chiave di payload utilizzando la chiave privata RSA e il meccanismo di decompressione. `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 di crittografia (CU)

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:
      --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
      --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
```

```
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap rsa-oaep comando utilizzando la chiave privata RSA con il unwrap valore dell'attributo impostato su. true

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": [],
        "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": "0x0000000000001808e9",
      "key-info": {
        "key-owners": [
          {
            "username": "cu1",
            "key-coverage": "full"
          }
        ],
        "shared-users": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-privategeneric-secret`,`rsa-private`].

Campo obbligatorio: sì

<HASH_FUNCTION>

Specifica la funzione hash.

Valori validi:

- sha1
- sha224
- sha256
- sha384
- sha512

Campo 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

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

chiave unwrap rsa-pkcs

Il key unwrap rsa-pkcs comando decomprime una chiave di payload utilizzando la chiave privata RSA e il meccanismo di decompressione. `RSA-PKCS`

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 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 di crittografia (CU)

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:
      --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
      --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
  -h, --help
          Print help
```

Esempi

Questi esempi mostrano come utilizzare il key unwrap rsa-oaep comando utilizzando una chiave AES con il valore dell'unwrapattributo 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": [],
        "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": [],
        "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
      }
    }
  }
}
```

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 con cui scartare.

Campo 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.

Campo obbligatorio: no

<KEY_TYPE_CLASS>

Tipo di chiave e classe di chiave racchiusa [valori possibili:`aes`,,`des3`, `ec-private`generic-secret,`rsa-private`].

Campo obbligatorio: sì

<ETICHETTA>

Etichetta per la chiave aperta.

Campo obbligatorio: sì

<SESSIONE>

Crea una chiave di sessione che esiste solo nella sessione corrente. La chiave non può essere recuperata dopo la fine della sessione.

Campo obbligatorio: no

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi

Il key wrap comando nella CLI di CloudHSM esporta una copia crittografata di una chiave privata simmetrica o asimmetrica dall'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. scartare i tasti 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:

- portachiavi aes-gcm
- portachiavi aes-no-pad
- portachiavi aes-pkcs5-pad
- portachiavi aes-zero-pad
- involucro per chiavi cloudhsm-aes-gcm

- portachiavi rsa-aes
- portachiavi rsa-oaep
- portachiavi rsa-pkcs

portachiavi aes-gcm

Il key wrap aes-gcm comando esegue il wrapping di una chiave di payload utilizzando una chiave AES sull'HSM e il 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 Generazione chiavi-simmetriche aes comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
      --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"
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

<AAD>

Valore AES GCM Additional Authenticated Data (AAD), in esadecimale.

Campo obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag AES GCM in bit.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi aes-no-pad

Il key wrap aes-no-pad comando esegue il wrapping di una chiave di payload utilizzando una chiave AES sull'HSM e il 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 Generazione chiavi-simmetriche aes comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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 `true` impostato su.

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"
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi aes-pkcs5-pad

Il key wrap aes-pkcs5-pad comando esegue il wrapping di una chiave di payload utilizzando una chiave AES sull'HSM e il 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 Generazione chiavi-simmetriche aes comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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 `true` impostato su.

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="
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi aes-zero-pad

Il key wrap aes-zero-pad comando esegue il wrapping di una chiave di payload utilizzando una chiave AES sull'HSM e il 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 Generazione chiavi-simmetriche aes comando con l'`wrap`attributo `true` impostato su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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 `true` impostato su.

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"
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

involucro per chiavi cloudhsm-aes-gcm

Il key wrap cloudhsm-aes-gcm comando esegue il wrapping di una chiave di payload utilizzando una chiave AES sull'HSM e il 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 Generazione chiavi-simmetriche aes comando e l'`wrap`attributo `true` impostati su.

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
      --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"
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

<AAD>

Valore AES GCM Additional Authenticated Data (AAD), in esadecimale.

Campo obbligatorio: no

<TAG_LENGTH_BITS>

Lunghezza del tag AES GCM in bit.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi rsa-aes

Il key wrap rsa-aes comando esegue il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sull'HSM e il meccanismo di wrapping RSA-AES. 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-aes comando, è necessario innanzitutto disporre di una chiave RSA nel clusterAWS CloudHSM. È possibile generare una coppia di chiavi RSA utilizzando il Generazione chiavi-asimmetriche-coppia comando e l'`wrap`attributo impostati su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
```

```
      --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 impostato su. `true`

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=="
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo 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

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi rsa-oaep

Il key wrap rsa-oaep comando esegue il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sull'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 clusterAWS CloudHSM. È possibile generare una coppia di chiavi RSA utilizzando il Generazione chiavi-asimmetriche-coppia comando e l'`wrap`attributo impostati su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
      --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 impostato su. `true`

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=="
  }
}
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo 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

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

portachiavi rsa-pkcs

Il key wrap rsa-pkcs comando esegue il wrapping di una chiave di payload utilizzando una chiave pubblica RSA sull'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 clusterAWS CloudHSM. È possibile generare una coppia di chiavi RSA utilizzando il Generazione chiavi-asimmetriche-coppia comando e l'`wrap`attributo impostati su. `true`

Tipo di utente

I seguenti tipi di utenti possono eseguire questo comando.

- Utenti di crittografia (CU)

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:
      --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
  -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=="
  }
```

Argomenti

<PAYLOAD_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 payload.

Campo obbligatorio: sì

<PERCORSO>

Percorso del file binario in cui verranno salvati i dati chiave racchiusi.

Campo obbligatorio: sì

<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.

Campo obbligatorio: sì

Argomenti correlati

- portachiavi
- scartare i tasti

Login

È possibile utilizzare il comando login nella CLI di CloudHSM per effettuare l'accesso e disconnettersi da ogni modulo HSM in un cluster.

> **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 modifica utente-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 su più HSM. Pertanto, il tentativo di accesso potrebbe non iniziare ogni volta nello stesso HSM. 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 utente

Gli utenti seguenti possono eseguire questi comandi.

- Admin non attivato
- Admin
- Utente di crittografia (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:
        --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
```

Esempio

Example

Questo comando ti consente di accedere a tutti i moduli HSM in un cluster con le credenziali di un utente admin denominato `admin1`.

```
aws-cloudhsm >login --username admin1 --role admin
Enter password:
{
  "error_code": 0,
  "data": {
    "username": "admin1",
    "role": "admin"
  }
}
```

Argomenti

<NOME UTENTE>

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.

Campo obbligatorio: sì

<RUOLO>

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>

Specifica la password dell'utente che sta effettuando l'accesso ai moduli HSM.

Argomenti correlati

- Guida introduttiva alla CLI di CloudHSM
- Attivazione del cluster

Accedi mfa-token-firma

Utilizza il comando login mfa-token-sign nella CLI di CloudHSM dell'AWS CloudHSM per accedere utilizzando l'autenticazione a più fattori. Per utilizzare questo comando, devi prima configurare MFA per la CLI de CloudHSM.

Tipo utente

Gli utenti seguenti possono eseguire questi comandi.

- Admin
- Utente di crittografia (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:
        --token <TOKEN>    Filepath where the unsigned token file will be written
```

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"
  }
}
```

Argomenti

<TOKEN>

Percorso del file in cui verrà scritto il file del 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

Logout

È possibile utilizzare il comando logout nella CLI di CloudHSM per effettuare l'accesso e uscire da ogni modulo HSM in un cluster.

Tipo utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Utente di crittografia (CU)

Sintassi

```
aws-cloudhsm > help logout
Logout of your cluster

USAGE:
    logout

OPTIONS:
    -h, --help       Print help information
    -V, --version    Print version information
```

Esempio

Example

Questo comando ti disconnette da tutti i moduli HSM di un cluster.

```
aws-cloudhsm > logout
{
  "error_code": 0,
  "data": "Logout successful"
}
```

Argomenti correlati

- Guida introduttiva alla CLI di CloudHSM
- Attivazione del cluster

Utente

user 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 utenti è composta dai seguenti comandi:

- Modifica utente-mfa
- Modifica utente-password
- Crea utente
- Elimina utente
- Elenco utenti

Modifica utente-mfa

Attualmente, questa categoria è composta dal seguente sottocomando:

- Modifica utente-mfa token-firma

Modifica utente-mfa token-firma

Utilizza il comando user change-mfa 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 utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Utente di crittografia

Sintassi

Attualmente, per gli utenti è disponibile un'unica strategia multifattoriale: Token Firma.

```
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:
      --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
```

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 : scrivi 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"
  }
}
```

Argomenti

<RUOLO>

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.
- Utenti di crittografia: gli utenti di crittografia possono creare e gestire le chiavi e utilizzarle nelle operazioni di crittografia.

<NOME UTENTE>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il 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

<MODIFICA_PASSWORD>

Specifica in testo semplice la nuova password dell'utente la cui MFA viene registrata/annullata.

Campo obbligatorio: sì

<TOKEN>

Percorso del file in cui verrà scritto il file token non firmato.

Campo obbligatorio: sì

<APPROVAZIONE>

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.

<ANNULLA REGISTRAZIONE>

Annulla la registrazione della chiave pubblica MFA, se presente.

<MODIFICA-QUORUM>

Modifica la chiave pubblica del quorum insieme alla chiave MFA.

Argomenti correlati

- Capire la 2FA per gli utenti dell'HSM

Modifica utente-password

Utilizza il comando user change-password nella CLI di CloudHSM per modificare la password di un utente esistente nel cluster dell'AWS CloudHSM. Per abilitare l'MFA per un utente, utilizzare il comando `user change-mfa`.

Qualsiasi utente può modificare la sua 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 utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Utente di crittografia (CU)

Sintassi

> **Note**
>
> Per abilitare l'autenticazione a più fattori (MFA) per un utente, usa il comando user change-mfa.

```
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:

      --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
```

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 utente di crittografia (CU).

```
aws-cloudhsm > user change-password --username bob --role crypto-user
Enter password:
Confirm password:
{
  "error_code": 0,
  "data": {
    "username": "bob",
    "role": "crypto-user"
  }
}
```

Argomenti

<APPROVAZIONE

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.

<ANNULLA REGISTRAZIONE-MFA>

Annulla la registrazione della chiave pubblica MFA, se presente.

<ANNULLA REGISTRAZIONE-QUORUM>

Annulla la registrazione della chiave pubblica del quorum, se presente.

<PASSWORD>

Specifica la nuova password dell'utente in testo semplice.

Campo obbligatorio: sì

<RUOLO>

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.
- Utente di crittografia: gli utenti di crittografia possono creare e gestire le chiavi e utilizzarle nelle operazioni di crittografia.

<NOME UTENTE>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il 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

- Elenco di utenti
- Crea utente
- Elimina utente

Modifica utente-quorum

user change-quorum è una categoria principale per un gruppo di comandi che, se combinati con la categoria principale, creano un comando specifico per modificare il 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

Modifica utente-quorum token-firma

user change-quorum token-sign è una categoria principale per i comandi che, se combinati con questa categoria principale, creano un comando specifico per le operazioni quorum token-firma.

Attualmente, questa categoria comprende i seguenti comandi:

- Registra

Modifica utente-quorum token-firma registra

Utilizza il comando user change-quorum token-sign register nella CLI di CloudHSM per registrare la strategia del quorum di token-firma per un utente admin.

Tipo 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:
      --public-key <PUBLIC_KEY>      Filepath to public key PEM file
      --signed-token <SIGNED_TOKEN>  Filepath with token signed by user private key
```

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"
      }
    ]
  }
}
```

Argomenti

<CHIAVI-PUBBLICHE>

Percorso del file PEM della chiave pubblica.

Campo obbligatorio: sì

<TOKEN-FIRMATI>

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 il valore minimo del quorum per gli admin

- Nomi e tipi di servizi che supportano l'autenticazione del quorum

Crea utente

Il comando user create nella CLI di CloudHSM crea un utente nel cluster dell'AWS CloudHSM. Solo gli account utente con ruolo di admin possono eseguire questo comando.

Tipo 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:
      --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
```

Esempio

Questi esempi mostrano come utilizzare user create per creare nuovi utenti nei moduli HSM.

Example : creare un utente di crittografia

Questo esempio crea un account nel cluster dell'AWS CloudHSM 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"
  }
}
```

Argomenti

<NOME UTENTE>

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.

Campo obbligatorio: sì

<RUOLO>

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>

Specifica la password dell'utente che sta effettuando l'accesso ai moduli HSM.

Campo obbligatorio: sì

<APPROVAZIONE>

Specifica il percorso del file di un token firmato dal quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

Argomenti correlati

- Elenco utenti
- Elimina utente
- Modifica utente-password

Elimina utente

Il comando user delete nella CLI di CloudHSM elimina un utente dal cluster dell'AWS CloudHSM. Solo gli account utente con ruolo di admin possono eseguire questo comando. Non è possibile eliminare un utente 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:
```

```
    --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"
  }
}
```

Argomenti

<NOME UTENTE>

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.

Campo obbligatorio: sì

<RUOLO>

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.

Campo obbligatorio: sì

<APPROVAZIONE>

Specifica il percorso del file di un token firmato dal quorum per approvare l'operazione. Richiesto solo se il valore del quorum del servizio utenti è maggiore di 1.

Campo obbligatorio: sì

Argomenti correlati

- Elenco utenti
- Crea utente
- Modifica utente-password

elenco utenti

Il comando user list nella CLI di CloudHSM elenca gli account utente presenti nel cluster CloudHSM. Non è necessario che tu abbia eseguito l'accesso alla CLI di CloudHSM per eseguire questo comando.

> **Note**
>
> Se aggiungi o elimini moduli HSM, aggiorna i file di configurazione utilizzati dal client AWS CloudHSM e dagli strumenti a riga di comando. In caso contrario, le modifiche apportate potrebbero non risultare effettive su tutti i moduli HSM nel cluster.

Tipo 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
```

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
- Crea utente
- Elimina utente
- Modifica utente-password

Quorum

quorum è una categoria principale per un gruppo di comandi che, se combinata con quorum, crea un comando specifico per l'autenticazione del quorum o le 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-firma

Servizi admin: l'autenticazione del quorum viene utilizzata per servizi che necessitano dei privilegi dell'admon 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.

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 servizio	Tipo servizio	Operazioni di servizio
Utente	Admin	• Crea utente • Elimina utente • Modifica utente-password • Modifica utente-mfa

Nome servizio	Tipo servizio	Operazioni di servizio
Quorum	Admin	• segno del token del quorum set-quorum-value

Argomenti correlati

- Utilizzo dell'autenticazione del quorum per gli amministratori: prima configurazione
- Utilizzo della CLI di CloudHSM per gestire l'autenticazione del quorum (controllo dell'accesso "M of N")

Quorum token-firma

quorum token-sign è una categoria per un gruppo di comandi che, se combinati con quorum token-sign, creano un comando specifico per l'autenticazione del quorum o le operazioni M of N.

Attualmente, questa categoria comprende i seguenti comandi:

- Elimina
- Generazione
- elenco
- elenco-valori-quorum
- Elenco-timeout
- Impostazione-valore-quorum
- Impostazione-timeout

Quorum token-firma elimina

Utilizza il comando quorum token-sign delete nella CLI di CloudHSM per eliminare uno o più token per un servizio autorizzato dal quorum.

Tipo 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:
      --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
```

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"
}
```

Argomenti

<AMBITO>

L'ambito in cui i token verranno eliminati nel cluster dell'AWS CloudHSM.

Valori validi

- Utente: utilizzato per eliminare solo i token di proprietà dell'utente connesso.
- Tutti: utilizzato per eliminare tutti i token nel cluster dell'AWS CloudHSM.

Argomenti correlati

- Elenco utenti

- Crea utente
- Elimina utente

quorum token- generazione firma

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.

> **Note**
>
> Solo gli admin possono generare un token di servizio.

Servizi admin: l'autenticazione quorum viene utilizzata per servizi che necessitano i privilegi di 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.

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 servizio	Tipo servizio	Operazioni di servizio
Utente	Admin	• Crea utente • Elimina utente • Modifica utente-password • Modifica utente-mfa
Quorum	Admin	• segno del token del quorum set-quorum-value

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:
      --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
          - registration:
            Registration service is used for registering a public key for quorum
 authentication

      --token <TOKEN>
          Filepath where the unsigned token file will be written
```

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"
  }
}
```

Argomenti

<SERVIZIO>

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.
- Registra: genera un token non firmato da utilizzare per la registrazione di una chiave pubblica per l'autorizzazione del quorum.

Campo obbligatorio: sì

<TOKEN>

Percorso del file in cui verrà scritto il file del token non firmato.

Campo obbligatorio: sì

Argomenti correlati

- Nomi e tipi di servizi che supportano l'autenticazione quorum

quorum token-elenco firme

Usa il comando quorum token-sign list nella CLI di CloudHSM per elencare tutti i token-firma token quorum presenti nel cluster dell'AWS CloudHSM.

Tipo utente

Gli utenti seguenti possono eseguire questo comando.

- Admin
- Utente di crittografia (CU)

Sintassi

```
aws-cloudhsm > help quorum token-sign list
List the token-sign tokens in your cluster

Usage: quorum token-sign list
```

Esempio

Questo comando elencherà tutti i token-firma presenti nel cluster dell'AWS CloudHSM.

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": 2
        "token-timeout-seconds": 588
        "cluster-coverage": "full"
      }
    ]
  }
}
```

Argomenti correlati

- token quorum-generazione firma

Quorum token-firma elenco-valori-quorum

Utilizza il comando quorum token-sign list-quorum-values nella CLI di CloudHSM per elencare i valori del quorum impostati nel cluster dell'AWS CloudHSM.

Tipo 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
```

Esempio

Questo comando elenca i valori di quorum impostati nel cluster dell'AWS CloudHSM per ogni servizio.

Example

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 1
  }
}
```

Argomenti correlati

- Nomi e tipi di servizi che supportano l'autenticazione quorum

Quorum token-firme elenco-timeout

Utilizza il comando quorum token-sign list-timeouts nella CLI di CloudHSM per ottenere il periodo di timeout del token in secondi per tutti i tipi di token.

Tipo 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-timeouts
List timeout durations in seconds for token validity

Usage: quorum token-sign list-timeouts
```

Esempio

Example

```
aws-cloudhsm > quorum token-sign list-timeouts
{
  "error_code": 0,
  "data": {
    "generated": 600,
    "approved": 600
  }
}
```

L'output include la seguente riga:

- generato: periodo di timeout in secondi per l'approvazione di un token generato.
- approvato: periodo di timeout in secondi per l'utilizzo di un token approvato per eseguire un'operazione autorizzata dal quorum.

Argomenti

Il comando non ha argomenti.

Argomenti correlati

- quorum, token-firma, impostazione-timeout

Quorum token-firma Impostazione-valore-quorum

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 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:
      --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

      --value <VALUE>
          Value to set for service

      --approval <APPROVAL>
          Filepath of signed quorum token file to approve operation
```

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 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:

```
aws-cloudhsm > quorum token-sign list-quorum-values
{
  "error_code": 0,
  "data": {
    "user": 1,
    "quorum": 2
  }
}
```

Argomenti

<APPROVAZIONE>

Il percorso del file token firmato da approvare sull'HSM.

<SERVIZIO>

Specifica il servizio autorizzato dal quorum per il quale 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.

- Registra: genera un token non firmato da utilizzare per registrare una chiave pubblica per l'autorizzazione del quorum.

Campo obbligatorio: sì

<VALORE>

Specifica il valore del quorum da impostare. Il valore del quorum massimo è otto (8).

Campo obbligatorio: sì

Argomenti correlati

- Quorum token-firma Elenco-valori-quorum
- Nomi e tipi di servizi che supportano l'autenticazione del quorum

Quorum token-firma impostazione-timeout

Utilizza il comando quorum token-sign set-timeout nella CLI di CloudHSM per impostare il periodo di timeout del token in secondi per ogni tipo di token.

Tipo utente

Gli utenti seguenti possono eseguire questo comando.

- Admin

Sintassi

```
aws-cloudhsm > help quorum token-sign set-timeout
Set timeout duration in seconds for token validity

Usage: quorum token-sign set-timeout <--generated <GENERATED> |--approved <APPROVED>>

Options:
      --generated <GENERATED>    Timeout period in seconds for a generated (non-
approved) token to be approved
      --approved <APPROVED>    Timeout period in seconds for an approved token to be
 used to execute a quorum operation
```

Esempio

Gli esempi seguenti mostrano come utilizzare il comando quorum token-sign set-timeout per impostare il periodo di timeout del token.

```
aws-cloudhsm > quorum token-sign set-timeout --generated 900
{
  "error_code": 0,
  "data": "Set token timeout successful"
}
```

Argomenti

Il comando non ha argomenti.

Argomenti correlati

- Quorum token-firma elenco-timeout

Utility di gestione CloudHSM (CMU)

Lo strumento a riga di comando cloudhsm_mgmt_util permette ai crypto officer di gestire gli utenti nei moduli HSM. Include strumenti che consentono di creare, eliminare ed elencare utenti e modificare le password degli utenti.

KMU e CMU fanno parte della suite Client SDK 3.

Inoltre, include i comandi che permettono ai crypto user (CU) di condividere le chiavi e ottenere e impostare i relativi attributi. Questi comandi completano i comandi di gestione delle chiavi nello strumento di gestione delle chiavi primarie, key_mgmt_util.

Per una guida rapida, vedi Gestione dei cluster clonati. Per informazioni dettagliate sui comandi cloudhsm_mgmt_util ed esempi di utilizzo dei comandi, vedi riferimento al comando cloudhsm_mgmt_util.

Argomenti

- Piattaforme supportate per l'utilità di gestione AWS CloudHSM
- Nozioni di base sulla CloudHSM Management Utility (CMU)
- Installazione e configurazione del client AWS CloudHSM (Linux)

- Installare e configurare il client AWS CloudHSM (Windows)
- riferimento al comando cloudhsm_mgmt_util

Piattaforme supportate per l'utilità di gestione AWS CloudHSM

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

Nozioni di base sulla CloudHSM Management Utility (CMU)

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, devi innanzitutto utilizzare lo strumento di configurazione per aggiornare la configurazione CMU locale con il parametro `--cmu` e l'indirizzo IP di uno degli HSM del 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

```
C:\Program Files\Amazon\CloudHSM\bin\ 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

```
C:\Program Files\Amazon\CloudHSM> .\cloudhsm_mgmt_util.exe C:\ProgramData\Amazon
\CloudHSM\data\cloudhsm_mgmt_util.cfg
```

L'output deve essere simile al seguente in base al numero di moduli HSM di cui si dispone.

```
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 password. Immetti la password, il sistema la nasconde e l'output mostra che il comando ha avuto successo e che la connessione a tutti i moduli HSM del cluster è stata effettuata .

```
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. Utilizza il comando logoutHSM per disconnetterti dai moduli HSM.

```
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...
```

Installazione e configurazione del client AWS CloudHSM (Linux)

Per interagire con il modulo HSM nel cluster AWS CloudHSM, è necessario il software client AWS CloudHSM per Linux. Ti consigliamo di installarlo nell'istanza del client EC2 Linux creata precedentemente. Puoi installare un client anche se utilizzi Windows. Per ulteriori informazioni, vedi Installare e configurare il client AWS CloudHSM (Windows).

Attività

- Installa il client AWS CloudHSM e gli strumenti a riga di comando
- Modifica la configurazione del client

Installa il client AWS CloudHSM e gli strumenti a riga di comando

Connettiti all'istanza del client ed esegui i seguenti comandi per scaricare e installare gli strumenti client e a riga di comando AWS CloudHSM.

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
```

Modifica la configurazione del client

Prima di poter utilizzare il client AWS CloudHSM per connetterti al tuo cluster, devi 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 config -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. Utilizzare il seguente comando Configura per aggiornare i file di configurazione del client AWS CloudHSM e gli strumenti a riga di comando, specificando l'indirizzo IP del modulo HSM nel cluster. Per ottenere l'indirizzo IP del modulo HSM, visualizza il cluster nella console AWS

CloudHSM oppure esegui il comando AWS CLI describe-clusters. Nell'output del comando, l'indirizzo IP del modulo HSM è il valore del campo `EniIp`. Se disponi di più moduli HSM, scegli l'indirizzo IP di uno dei moduli; non è importante 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 Attivazione del cluster.

Installare e configurare il client AWS CloudHSM (Windows)

Per interagire con un modulo HSM nel cluster AWS CloudHSMsu Windows, è necessario il software client AWS CloudHSM per Windows. È consigliabile installarlo nell'istanza di Windows Server creata in precedenza.

Per installare (o aggiornare) il client e gli strumenti a riga di comando di Windows più recenti

1. Connettersi all'istanza di Windows Server.
2. Scaricare il programma di installazione AWSCloudHSMClient-latest.msi.
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 -a ***<IP of one of the HSMs>***.
   b. Riavvia il servizio del client.
   c. Esegui config -m.
4. Apri il percorso di download ed esegui il programma di installazione AWSCloudHSMClient-latest.msi.
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:

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

8. Vai a Attivazione del cluster.

Note:

- Se aggiorni il client, i file di configurazione esistenti di installazioni precedenti non verranno sovrascritti.
- Il programma di installazione del client AWS CloudHSM registra automaticamente il Cryptography API: Next Generation (CNG) e il provider di archiviazione chiavi (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 Installazione e configurazione del client AWS CloudHSM (Linux).

riferimento al comando cloudhsm_mgmt_util

Lo strumento a riga di comando cloudhsm_mgmt_util permette ai crypto officer di gestire gli utenti nei moduli HSM. Inoltre, include i comandi che permettono ai crypto user (CU) di condividere le chiavi e ottenere e impostare i relativi attributi. Questi comandi completano i comandi di gestione delle chiavi primarie nello strumento a riga di comando key_mgmt_util.

Per una guida rapida, vedi Gestione dei cluster clonati.

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 nei moduli HSM, digita listUsers o listU.

```
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 Riferimento per l'attributo della chiave.

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	Descrizione	Tipo di utente
changePswd	Modifica la password degli utenti sui moduli HSM. Qualsiasi utente può modificare la propria password. I CO possono modificare la password di chiunque.	CO

Comando	Descrizione	Tipo di utente
createUser	Crea utenti di tutti i tipi sui moduli HSM.	CO
deleteUser	Elimina utenti di tutti i tipi dai moduli HSM.	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
getAttribute	Ottiene il valore di un attributo per una chiave AWS CloudHSM e lo scrive in un file o in stdout (output standard).	CU
getCert	Ottiene il certificato di un determinato modulo HSM e lo salva nel formato di certificato desiderato.	Tutti.
getHSMInfo	Ottiene informazioni sull'hard ware su cui è in esecuzione un modulo HSM.	Tutti. L'accesso non è obbligatorio.
getKeyInfo	Ottiene i proprietari, gli utenti condivisi e lo stato di autentica zione del quorum di una chiave.	Tutti. L'accesso non è obbligatorio.

Comando	Descrizione	Tipo di utente
info	Ottiene informazioni su un modulo HSM, inclusi indirizzo IP, nome host, porta e utente corrente.	Tutti. L'accesso non è obbligatorio.
ElencaUtenti	Ottiene gli utenti in ciascuno dei moduli HSM, il tipo e l'ID utente nonché altri attributi.	Tutti. L'accesso non è obbligatorio.
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 obbligatorio.
server	Permette di entrare e uscire dalla modalità server in un modulo HSM.	Tutti.
registerQuorumPubKey	Associa un utente HSM a una coppia di chiavi RSA-2048 asimmetriche.	CO
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 nei cluster clonati AWS CloudHSM.	CU, CO

Comando	Descrizione	Tipo di utente
syncUser	Sincronizza un utente sui cluster clonati AWS CloudHSM.	CO

changePswd

Il comando changePswd di cloudhsm_mgmt_util modifica la password di un utente esistente nei moduli HSM del cluster.

Qualsiasi utente può modificare la propria password. Inoltre, i crypto officer (CO e PCO) sono in grado di modificare la password di un altro CO o di un crypto user (CU). Non è necessario immettere la password corrente per effettuare la modifica.

> **Note**
>
> Tuttavia, non potrai modificare la password di un utente che ha eseguito l'accesso al client AWS CloudHSM o key_mgmt_util.

Per risolvere i problemi relativi a changePswd

Prima di eseguire qualsiasi comando CMU è necessario 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.

Se aggiungi o elimini un modulo HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 “Argomenti”.

```
changePswd <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

Esempi

I seguenti esempi mostrano come utilizzare changePassword per reimpostare la password per l'utente corrente o qualsiasi altro utente nei moduli HSM.

Example : modifica la tua password

Qualsiasi utente nei moduli HSM può utilizzare il comando changePswd per modificare la propria password. Prima di modificare la password, utilizza info per ottenere informazioni su ciascuno dei moduli HSM hardware nel cluster, inclusi il nome utente e il tipo di utente connesso.

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 CO o PCO per modificare la password di un altro utente CO o CU nei moduli HSM. 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
```

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 Utilizzo della CMU per gestire la 2FA

```
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 Informazioni sugli utenti HSM.

Campo 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.

Campo 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.

Campo obbligatorio: sì

[-2fa </percorso/per/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 Utilizzo della CMU per gestire la 2FA..

Campo obbligatorio: no

Argomenti correlati

- Info
- Elenco Utenti
- createUser
- deleteUser

createUser

Il comando createUser in cloudhsm_mgmt_util ti consente di creare un utente sui moduli HSM. Solo i crypto officer (CO e PRECO) possono eseguire questo comando. Una volta completato correttamente, il comando crea l'utente in tutti i moduli HSM nel cluster.

Per risolvere i problemi relativi a createUser

Tuttavia, se la tua configurazione HSM non è precisa, è possibile che l'utente non venga creato su tutti i moduli. Per aggiungere l'utente a tutti i moduli HSM in cui manca, utilizza i comandi syncUser o createUser solo sui moduli HSM in cui l'utente non è presente. Per evitare errori di configurazione, esegui lo strumento di configurazione con l'opzione `-m`.

Prima di eseguire qualsiasi comando CMU devi avviare key_mgmt_util e accedere al modulo HSM. Assicurati di eseguire l'accesso con il tipo di account autorizzato a eseguire i comandi che prevedi di utilizzare.

Se aggiungi o elimini moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO, PRECO)

Sintassi

Immettete gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizzate 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 “Argomenti”.

```
createUser <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

Esempi

Questi esempi mostrano come utilizzare createUser per creare nuovi utenti nei moduli HSM.

Example : creare un crypto officer

In questo esempio viene creato un crypto officer (CO) sui moduli HSM 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 avviso spiega che il comando crea utenti su tutti i moduli HSM nel cluster. Tuttavia, se il comando ha esito negativo su un qualsiasi modulo HSM, l'utente non sarà presente su tali moduli. Per continuare, digita `y`.

L'output indica che il nuovo utente è stato creato su tutti e tre i moduli HSM nel 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
```

Quando il comando viene completato, `alice` dispone delle stesse autorizzazioni dell'utente CO `admin` sul modulo HSM, inclusa la possibilità di modificare la password di qualsiasi utente nei moduli HSM.

Il comando finale utilizza listUsers per verificare che l'utente `alice` è presente su tutti e tre i moduli HSM nel cluster. L'output mostra anche che ad `alice` è assegnato un ID utente `3`. Puoi utilizzare l'ID utente per individuare `alice` in altri comandi, come 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 avere digitato y per rispondere al messaggio di avviso, l'output indica che bob è stato creato su tutti tre i moduli HSM nel 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
```

Argomenti

Immettete gli argomenti nell'ordine specificato nel diagramma di sintassi. Utilizzate 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 Utilizzo della CMU per gestire la 2FA.

```
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 Informazioni sugli utenti HSM.

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.

Campo 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

<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) sui moduli HSM.

Campo obbligatorio: sì

[-2fa </percorso/per/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 Utilizzo della CMU per gestire la 2FA.

Campo obbligatorio: no

Argomenti correlati

- ElencaUtenti
- deleteUser
- syncUser
- changePswd

deleteUser

Il comando deleteUser in cloudhsm_mgmt_util elimina un utente dai moduli di sicurezza hardware (HSM). Solo i crypto officer (CO) possono eseguire questo comando. Non è possibile eliminare un utente 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

In questo esempio viene eliminato un crypto officer (CO) dai moduli HSM in un cluster. Il primo comando utilizza listUsers per elencare tutti gli utenti dei moduli HSM.

L'output indica che l'utente `3`, `alice`, è un CO nei moduli HSM.

```
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 comando deleteUser per eliminare `alice` dai moduli HSM.

L'output indica che il comando ha avuto esito positivo su tutti i tre i moduli HSM nel 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)
```

L'ultimo comando utilizza il comando listUsers per verificare che l'utente `alice` è stato eliminato da tutti i tre i moduli HSM nel 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
```

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 Informazioni sugli utenti HSM.

Campo obbligatorio: sì

<nome-utente>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il 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 dell'utente no.

Campo obbligatorio: sì

Argomenti correlati

- ElencaUtenti
- createUser
- syncUser
- changePswd

findAllKeys

Il comando findAllKeys in cloudhsm_mgmt_util consente di ottenere le chiavi che un determinato crypto user (CU) possiede o condivide. Inoltre, restituisce un hash dei dati dell'utente su ciascun modulo HSM. È possibile utilizzare l'hash per determinare immediatamente se gli utenti, la proprietà delle chiavi e i dati sulla condivisione delle chiavi sono gli stessi su tutti i moduli HSM nel cluster. Nell'output, le chiavi di proprietà dell'utente vengono annotate da `(o)` e le chiavi condivise vengono annotate da `(s)`.

findAllKeys restituisce chiavi pubbliche solo quando il CU specificato possiede la chiave, anche se tutti i CU sull'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 crypto officer (CO e PCO) e gli utenti dell'appliance (AU) possono eseguire questo comando. I crypto user (CU) possono eseguire i comandi seguenti:

- listUsers per trovare tutti gli utenti
- findKey in key_mgmt_util per trovare le chiavi che è possibile utilizzare
- getKeyInfo in key_mgmt_util per trovare il proprietario e gli utenti in condivisione di una determinata chiave di loro proprietà o condivisa

Prima di eseguire qualsiasi comando SMU 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 utilizzare `findAllKeys` per trovare tutte le chiavi per un utente e ottenere un hash delle informazioni dell'utente della chiave in ciascun modulo HSM.

Example : trovare le chiavi per un CU

Questo esempio utilizza findAllKeys per trovare le chiavi nei moduli HSM 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 utilizza findAllKeys per verificare che tutti gli HSM nel cluster contengano gli stessi utenti, proprietà delle chiavi e 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'utente 6, ma il valore hash sarà lo stesso per qualsiasi utente che possiede o condivide qualsiasi chiave nei moduli HSM. Se l'utente specificato non possiede o condivide alcuna chiave, come accade per un CO, il comando non restituisce un valore hash.

L'output indica che l'hash della chiave è identico per entrambi i moduli HSM nel cluster. 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)
```

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. Inserisci l'ID utente di un utente dei moduli HSM. Per trovare gli ID utente di tutti gli utenti, utilizza listUsers.

Tutti gli ID utente sono validi, ma `findAllKeys` restituisce le chiavi solo per i crypto user (CU).

Campo 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.

Campo obbligatorio: sì

<file output>

Scrive l'output nel file specificato.

Campo obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- changePswd
- deleteUser
- ElencaUtenti
- syncUser
- findKey in key_mgmt_util
- getKeyInfo in key_mgmt_util

getAttribute

Il comando getAttribute in cloudhsm_mgmt_util ottiene un valore di attributo per una chiave da tutti i moduli HSM nel cluster e lo scrive in stdout (output standard) o in un file. Solo i crypto user (CU) possono eseguire il 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. Riferimento per l'attributo 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 HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 estraibile per una chiave nei moduli HSM. Puoi utilizzare un comando come questo per determinare se puoi esportare una chiave dai moduli HSM.

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 Riferimento per l'attributo della chiave.

```
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 estraibile per la chiave con l'handle della chiave `262170` nei moduli HSM. 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), l'attributo non può essere esportato dai moduli HSM. 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
```

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. È possibile trovare gli utenti di una chiave utilizzando getKeyInfo in key_mgmt_util.

Campo obbligatorio: sì

<id attributo>

Identifica l'attributo. Inserire una costante che rappresenti un attributo oppure immettere `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 delle chiavi, vedi Riferimento per l'attributo della chiave.

Campo 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.

Campo obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- getAttribute in key_mgmt_util
- listAttributes

- setAttribute in cloudhsm_mgmt_util
- setAttribute in key_mgmt_util
- Riferimento attributo chiave

getCert

Con il comando getCert in cloudhsm_mgmt_util, è possibile recuperare i certificati di un determinato modulo HSM in un cluster. Quando esegui il comando, specifichi il tipo di certificato da recuperare. A tale scopo, puoi utilizzare uno dei valori interi corrispondenti come descritto nella sezione Argomenti più avanti. Per ulteriori informazioni sul ruolo di ognuno di questi certificati, vedi verifica dell'identità del modulo HSM.

Prima di eseguire qualsiasi comando CMU vedi 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo utente

Gli utenti seguenti possono eseguire questo comando.

- Tutti gli utenti.

Prerequisiti

Prima di iniziare, devi passare alla modalità server nel modulo HSM di destinazione. Per ulteriori informazioni, vedi server.

Sintassi

Per utilizzare il comando getCert una volta in modalità server:

```
server> getCert <file-name> <certificate-type>
```

Esempio

Prima di tutto, passa alla modalità server. 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...
```

Utilizza quindi il comando getCert. In questo esempio utilizziamo `/tmp/P0.crt` come nome del file in cui verrà salvato il certificato e 4 (certificato root del cliente) come tipo di certificato desiderato:

```
server0> getCert /tmp/P0.crt 4
getCert Success
```

Argomenti

```
getCert <file-name> <certificate-type>
```

<nome-file>

Permette di specificare il nome del file in cui viene salvato il certificato.

Campo obbligatorio: sì

<tipo-certificato>

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

- server

getHSMInfo

Il comando getHSMInfo in cloudhsm_mgmt_util consente di ottenere informazioni sull'hardware su cui viene eseguito ciascun modulo HSM, come ad esempio il modello, il numero di serie, lo stato FIPS, la memoria, la temperatura e la 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 devi 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 sui moduli HSM 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          :xxxxxxxxxxxxxxx

...
```

Argomenti correlati

- info

getKeyInfo

Il comando getKeyInfo in the key_mgmt_util restituisce gli ID utente HSM degli utenti che possono utilizzare la chiave, inclusi il proprietario e gli crypto user (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 gli ID utente HSM di utenti nei tuoi HSM, utilizza listUsers. Per trovare le chiavi di un utente specifico, utilizza findKey key_mgmt_util. I crypto officer possono utilizzare findAllKeys in 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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), il comando restituisce il proprietario della chiave (3) e i crypto user (CU) 4, con i quali 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 : ottieni 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 utilizza genSymKey per 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
```

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.

Campo obbligatorio: sì

<file output>

Scrive l'output nel file specificato, invece che in stdout. Se il file esiste, il comando lo sovrascrive senza preavviso.

Campo obbligatorio: no

Impostazione predefinita: stdout

Argomenti correlati

- getKeyInfo in key_mgmt_util
- findKey in key_mgmt_util
- findAllKeys in cloudhsm_mgmt_util
- ElencaUtenti
- shareKey

info

Il comando info in cloudhsm_mgmt_util permette di ottenere informazioni su ciascuno dei moduli HSM del cluster, tra cui nome dell'host, porta, indirizzo IP, nonché nome e tipo di utente che ha effettuato l'accesso a cloudhsm_mgmt_util nel modulo HSM.

Prima di eseguire qualsiasi comando CMU devi 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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)'
```

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. Ai moduli HSM 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, utilizza getHSMInfo.

Campo obbligatorio: sì

Argomenti correlati

- getHSMInfo
- loginHSM e logoutHSM

listAttributes

Il comando listAttributes in cloudhsm_mgmt_util elenca gli attributi di una chiave AWS CloudHSM e le costanti che li rappresentano. Puoi utilizzare queste costanti per identificare gli attributi nei comandi getAttribute e setAttribute.

Per informazioni sull'interpretazione degli attributi chiave, vedi Riferimento per l'attributo della chiave.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Tipo 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 Riferimento per l'attributo della chiave. 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
```

Parametri

-h

Visualizza l'assistenza per il comando.

Campo obbligatorio: sì

Argomenti correlati

- getAttribute
- setAttribute
- Riferimento attributo chiave

ElencaUtenti

Il comando listUsers in cloudhsm_mgmt_util restituisce gli utenti in ognuno dei moduli 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 devi 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 ciascun modulo HSM del cluster e ne visualizza i relativi 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 dei comandi di 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.
- ErroreLoginCnt: indica il numero di volte in cui l'utente non è riuscito a eseguire l'accesso.
- 2FA: indica che l'utente ha abilitato l'autenticazione a più fattori.

Argomenti correlati

- listUsers in key_mgmt_util
- createUser
- deleteUser
- changePswd

loginHSM e logoutHSM

Puoi utilizzare i comandi loginHSM e logoutHSM in cloudhsm_mgmt_util per effettuare l'accesso e disconnetterti da tutti i moduli 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 moduli HSM, aggiorna i file di configurazione utilizzati dal client AWS CloudHSM e dagli strumenti a riga di comando. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Se disponi di più HSM nel cluster, potresti avere un maggior numero di tentativi di accesso errati prima che l'account venga bloccato. Questo perché il client CloudHSM bilancia il carico su più HSM. 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 utente

Gli utenti seguenti possono eseguire questi comandi.

- Precrypto officer (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 “Argomenti”.

```
loginHSM <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

```
logoutHSM
```

Esempi

Questi esempi mostrano come utilizzare loginHSM e logoutHSM per eseguire l'accesso e uscire da tutti moduli HSM in un cluster.

Example : Accedi ai moduli HSM in un cluster

Questo comando consente di accedere a tutti i moduli HSM in un cluster con le credenziali di un utente CO denominato `admin` e una password `co12345`. L'output indica che il comando è riuscito e che l'utente è connesso ai moduli HSM (che, in questo caso, sono `server 0` e `server 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 è stato eseguito correttamente e che l'utente si è connesso ai moduli HSM.

```
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 ti disconnette dai moduli HSM ai quali avevi effettuato l'accesso (che, in questo caso, sono `server 0` e `server 1`). L'output indica che il comando è riuscito e che l'utente si è disconnesso dai moduli HSM.

```
aws-cloudhsm>logoutHSM

logoutHSM success on server 0(10.0.2.9)
logoutHSM success on server 1(10.0.3.11)
```

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 Utilizzo della CMU per gestire la 2FA

```
loginHSM <user-type> <user-name> <password |-hpswd> [-2fa </path/to/authdata>]
```

<tipo utente>

Specifica il tipo di utente che sta effettuando l'accesso al modulo HSM. Per ulteriori informazioni, vedi Tipo di utente sopra.

Campo obbligatorio: sì

<nome utente>

Specifica il nome utente dell'utente che sta effettuando l'accesso al modulo HSM.

Campo obbligatorio: sì

<password | -hpswd >

Specifica la password dell'utente che sta effettuando l'accesso ai moduli HSM. Per nascondere la password, utilizza il parametro `-hpswd` al posto della password e segui le istruzioni.

Campo obbligatorio: sì

[-2fa </percorso/per/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 Utilizzo della CMU per gestire la 2FA.

Campo obbligatorio: no

Argomenti correlati

- Nozioni di base su cloudhsm_mgmt_util
- Attivazione del cluster

registerQuorumPubChiave

Il comando registerQuorumPubKey in cloudhsm_mgmt_util associa 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 documentazione AWS CloudHSM, l'autenticazione del quorum viene talvolta definita M of N (MoFN), il che significa un minimo M di approvatori su un numero totale N di 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 “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
```

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 Informazioni sugli utenti HSM.

Valori validi:

- CO: i crypto officer possono gestire gli utenti ma non le chiavi.

Campo obbligatorio: sì

<nome-utente>

Specifica un nome intuitivo per l'utente. La lunghezza massima è 31 caratteri. L'unico carattere speciale consentito è il 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 dell'utente no.

Campo 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 Creare e firmare un token di registrazione.

Campo obbligatorio: sì

<signed-registration-token>

Specifica il percorso di un file che contiene l'hash firmato dal meccanismo SHA256_PKCS del token di registrazione. Per ulteriori informazioni, vedi Creare e firmare un token di registrazione.

Campo 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 Creare una coppia di chiavi RSA.

Campo 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 della 2FA, vedi Autenticazione del quorum e 2FA.

Argomenti correlati

- Crea una coppia di chiavi RSA

- Crea e firma un token di registrazione
- Registra una chiave pubblica con HSM
- Gestisci Autenticazione del Quorum (controllo accessi M of N)
- Autenticazione quorum e 2FA
- ElencaUtenti

server

In genere, quando esegui un comando in cloudhsm_mgmt_util, il comando ha effetto su tutti i moduli HSM nel cluster designato (modalità globale). 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. Puoi utilizzare il comando server in cloudhsm_mgmt_util per passare alla modalità server e interagire direttamente con una determinata istanza di un modulo HSM.

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 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 di server vengono assegnati nello stesso ordine in cui i moduli HSM sono presenti nel file di configurazione. 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
```

Argomenti

```
server <server-number>
```

<numero-server>

Specifica il numero di server del modulo HSM di destinazione.

Campo obbligatorio: sì

Il comando `exit` non ha argomenti.

Argomenti correlati

- syncKey
- createUser
- deleteUser

setAttribute

Il comando setAttribute in cloudhsm_mgmt_util modifica il valore dell'etichetta, crittografa e decodifica, esegue e annulla il wrapping degli attributi di una chiave nei moduli HSM. 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 genSymKey in key_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 Riferimento per l'attributo della chiave.

```
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 indica che il valore dell'attributo di decodifica è vero (1) su entrambi i moduli HSM nel cluster.

```
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 indica che il comando ha avuto esito positivo su entrambi i moduli HSM nel cluster.

```
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 indica che il valore dell'attributo di decodifica è falso (0) su entrambi i moduli HSM nel cluster.

```
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
```

Argomenti

```
setAttribute <key handle> <attribute id>
```

<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. È possibile trovare gli utenti di una chiave utilizzando getKeyInfo.

Campo 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. Riferimento per l'attributo della chiave

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 attributo chiave

Esci

Il quit comando in cloudhsm_mgmt_util esce 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 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

shareKey

Il comando shareKey in cloudhsm_mgmt_util condivide e annulla la condivisione delle chiavi che possiedi insieme ad ai crypto user. 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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 per condividere una chiave privata ECC che l'utente corrente possiede con un altro crypto user sui moduli HSM. Le chiavi pubbliche sono disponibili per tutti gli utenti dell'HSM, perciò non è possibile condividerle o annullarne la condivisione.

Il primo comando utilizza getKeyInfo per ottenere le informazioni degli utenti per la chiave 262177, una chiave privata ECC nei moduli HSM.

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 esempio utilizza shareKey per condividere la chiave 262177 con l'utente 4, un altro crypto user nei moduli HSM. L'argomento finale utilizza un valore di 1 per indicare un'operazione di condivisione.

L'output indica che l'operazione ha avuto esito positivo su entrambi i moduli HSM nel 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 indica che il comando ha avuto esito positivo su entrambi i moduli HSM. 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)
```

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 disporre di una chiave, utilizza getKeyInfo.

Campo 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.

Campo 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

syncKey

È possibile utilizzare il comando syncKey in cloudhsm_mgmt_util per sincronizzare manualmente le chiavi nelle istanze dell'HSM all'interno di un cluster o in tutti i 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 sono in genere creati in diverse regioni AWS per semplificare i processi globali di dimensionamento e di ripristino di emergenza.

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 utilizzare syncKey, è necessario creare un file di configurazione AWS CloudHSM che specifica 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 utente

I seguenti tipi di utenti possono eseguire questo comando.

- Crypto officer (CO)
- Crypto user (CU)

> **Note**
>
> I CO possono utilizzare syncKey su qualsiasi chiave, mentre i CU possono solo utilizzare questo comando sulle chiavi che possiedono. Per ulteriori informazioni, vedi the section called “Informazioni sugli utenti HSM”.

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, utilizza il comando findAllKeys per trovare tutte le chiavi che appartengono a un determinato utente.

È inoltre necessario conoscere i `server IDs` assegnati ai moduli HSM di origine e destinazione, che sono riportati nell'output di traccia restituito da cloudhsm_mgmt_util all'avvio. Questi sono assegnati nello stesso ordine nel quale i moduli HSM appaiono nel file di configurazione.

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
```

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, utilizzare findAllKeys mentre sei collegato a un server HSM.

Campo 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-cluster in AWS CLI
- server

syncUser

È possibile utilizzare il comando syncUser in cloudhsm_mgmt_util per sincronizzare manualmente i crypto user (CU) o i crypto officer (CO) su più istanze dell'HSM all'interno di un cluster o su più cluster clonati. AWS CloudHSM non sincronizza automaticamente gli utenti. Di solito, puoi gestire gli utenti in modalità globale, in modo che tutti i moduli HSM in un cluster siano 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 sono in genere creati in diverse regioni AWS per semplificare i processi globali di dimensionamento e di ripristino di emergenza.

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 moduli HSM, aggiorna i file di configurazione per CMU. In caso contrario, le modifiche apportate potrebbero non essere effettive su tutti i moduli HSM nel cluster.

Tipo 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 individuare il `user ID`, utilizzare il comando listUsers per elencare tutti gli utenti negli HSM di un cluster.

È inoltre necessario conoscere i `server ID` assegnati ai moduli HSM di origine e destinazione, che sono riportati nell'output di traccia restituito da cloudhsm_mgmt_util all'avvio. Questi sono assegnati nello stesso ordine nel quale i moduli HSM appaiono nel file di configurazione.

Se stai sincronizzando un HSM su tutti i cluster clonati, segui le istruzioni in Utilizzo di 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
```

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.

Campo 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-clusters in AWS CLI
- server

Utility di gestione delle chiavi (KMU)

L'utility di gestione delle chiavi (KMU) è uno strumento a riga di comando che aiuta i crypto user (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 Nozioni di base su key_mgmt_util. Per informazioni dettagliate sui comandi, vedi Riferimento al comando key_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi. Riferimento per l'attributo della chiave

Per utilizzare key_mgmt_util con Linux, effettua la connessione all'istanza del client e vedi Installazione e configurazione del client AWS CloudHSM (Linux). Se utilizzi Windows, vedi Installa e configura il client AWS CloudHSM (Windows).

Argomenti

Nozioni di base su key_mgmt_util

AWS CloudHSM include due strumenti a riga di comando con il AWS CloudHSM software client . 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

In caso di messaggio di errore o risultato imprevisto per un comando, vedi gli argomenti Risoluzione dei problemi relativi a AWS CloudHSM per ricevere assistenza. Per dettagli sui comandi key_mgmt_util, vedi Riferimento al comando key_mgmt_util.

Configurazione di key_mgmt_util

Completa la configurazione seguente prima di utilizzare key_mgmt_util.

Avvio del client AWS CloudHSM

Prima di utilizzare key_mgmt_util, devi avviare il client AWS CloudHSM. Il client è un daemon che stabilisce una comunicazione end-to-end crittografata con gli HSM nel cluster. Lo strumento key_mgmt_util utilizza la connessione del client per comunicare con i moduli HSM nel cluster. Senza ciò, key_mgmt_util non funziona.

Per avviare il client AWS CloudHSM

Utilizzare il seguente comando per avviare il client AWS CloudHSM.

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 Windows client 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
```

Avvio di key_mgmt_util

Una volta avviato il client AWS CloudHSM, utilizzare il comando seguente 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

```
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.

Utilizzo di base di key_mgmt_util

Consulta i seguenti argomenti per l'utilizzo di base dello strumento key_mgmt_util.

Argomenti

- Accedere ai moduli HSM.
- Disconnessione dai moduli HSM
- Arresto di key_mgmt_util

Accedere ai moduli HSM.

Utilizza il comando loginHSM per effettuare la connessione ai moduli HSM. Il comando seguente effettua l'accesso come crypto user (CU) nominato `example_user`. L'output indica un accesso riuscito per tutti i tre i moduli HSM nel cluster.

```
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>
```

Disconnessione dai moduli HSM

Utilizza il comando logoutHSM per disconnetterti dai moduli HSM.

```
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
```

Arresto di key_mgmt_util

Utilizza il comando exit per arrestare key_mgmt_util.

```
Command:  exit
```

Installazione e configurazione del client AWS CloudHSM (Linux)

Per interagire con il modulo HSM nel cluster AWS CloudHSM, è necessario il software client AWS CloudHSM per Linux. 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 client AWS CloudHSM (Windows).

Attività

- Installazione del client AWS CloudHSM e degli strumenti a riga di comando
- Modifica della configurazione del client

Installazione del client AWS CloudHSM e degli strumenti a riga di comando

Connettiti all'istanza del client ed esegui i seguenti comandi per scaricare e installare gli il client AWS CloudHSM e gli strumenti a 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
```

Modifica della configurazione del client

Prima di poter utilizzare il client AWS CloudHSM per connetterti al tuo cluster, devi 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. Utilizzare il seguente comando configurazione per aggiornare i file di configurazione del client AWS CloudHSM e gli strumenti a riga di comando, specificando l'indirizzo IP del modulo HSM nel cluster. Per ottenere l'indirizzo IP del modulo HSM, visualizza il cluster nella console AWS CloudHSM oppure esegui il comando AWS CLI describe-clusters. Nell'output del comando, l'indirizzo IP del modulo HSM è il valore del campo `EniIp`. Se si dispone di più moduli HSM, scegliere l'indirizzo IP di uno dei moduli; non è importante 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 Attivazione del cluster.

Installa e configura il client AWS CloudHSM (Windows)

Per interagire con un modulo HSM nel cluster AWS CloudHSM, è necessario il software client AWS CloudHSM per Windows. 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. Scarica la versione più recente (AWSCloudHSMClient-latest.msi) dalla pagina dei download.
3. Apri la posizione di download ed esegui il programma di installazione (AWSCloudHSMClient-latest.msi) con privilegi di amministratore.
4. Segui le istruzioni del programma di installazione, quindi scegli Chiudi al termine dell'installazione.
5. Copia il certificato di emissione autofirmato - quello che hai utilizzato per firmare il certificato del cluster- nella cartella `C:\ProgramData\Amazon\CloudHSM`.
6. Esegui seguente comando per aggiornare i file di configurazione. Assicurati di arrestare e avviare il client durante la riconfigurazione se decidi di aggiornarla:

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

7. Passare a Attivazione del cluster.

Note:

- Se aggiorni il client, i file di configurazione esistenti di installazioni precedenti non verranno sovrascritti.
- Il programma di installazione AWS CloudHSM per Windows registra automaticamente gli 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, installa il client Linux Per ulteriori informazioni, vedi Installazione e configurazione del client AWS CloudHSM (Linux).

Riferimento al comando key_mgmt_util

Lo strumento a riga di comando key_mgmt_util ti consente di gestire le chiavi nei moduli HSM del cluster. Potrai quindi creare, eliminare e trovare le chiavi e i relativi attributi. Include vari comandi, ognuno dei quali è descritto in dettaglio in questo argomento.

Per una guida rapida, vedi Nozioni di base su key_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi. Riferimento per l'attributo della chiave 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 .

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	Descrizione
WrapAnnullaWrapChiave	Consente di crittografare e decodificare i contenuti di una chiave in un file.
EliminaChiave	Consente di eliminare una chiave dai moduli HSM.
StringaErrore2	Consente di recuperare l'errore che corrisponde a un codice di errore esadecimale .
Esci	Esce da key_mgmt_util.

Comando	Descrizione
EsportaChiavePrivata	Consente di esportare su un file su disco una copia di una chiave privata.
EsportaChiavePubblica	Consente di esportare da un HSM su un file una copia di una chiave pubblica.
EsportaChiaveSimmetrica	Consente di esportare su file una copia in testo normale di una chiave simmetrica dagli HSM.
EstraiOggettoMascherato	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.
TrovaSingolaChiave	Consente di verificare che una chiave sia presente in tutti i moduli HSM del cluster.
GeneraCoppiaChiaviDSA	Consente di creare una coppia di chiavi Digital Signing Algorithm (DSA) nei moduli HSM.
GeneraCoppiaChiaviECC	Consente di creare una coppia di chiavi Elliptic Curve Cryptography (ECC) nei moduli HSM.
GeneraCoppiaChiaviRSA	Consente di generare una coppia di chiavi asimmetriche RSA nei moduli HSM.
GeneraChiaviSimmetriche	Consente di generare una coppia di chiavi simmetriche negli HSM.
OttieniAttributo	Consente di recuperare i valori degli attributi di una chiave di AWS CloudHSM e li scrive in un file.
OttieniChiavePrivataCavium	Crea una versione falsa in formato PEM di una chiave privata e la esporta in un file.

Comando	Descrizione
OttieniCertificato	Consente di recuperare certificati di partizioni HSM e li salva in un file.
OttieniInfoChiavi	Consente di recuperare l'ID utente HSM degli utenti che possono utilizzare la chiave. Se la chiave è controllata dal quorum, consente di recuperare il numero di utenti del quorum.
help	Visualizza informazioni di aiuto sui comandi disponibili in key_mgmt_util.
ImportaChiavePrivata	Importa una chiave privata in un HSM.
ImportaChiavePubblica	Importa una chiave pubblica in un HSM.
ImportaChiaveSimmetrica	Consente di importare nell'HSM una copia in testo normale di una chiave simmetrica da un file.
InserisciOggettoMascherato	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 una chiave privata reale o una chiave PEM falsa.
ElencaAttributi	Consente di elencare gli attributi di una chiave AWS CloudHSM e le costanti che li rappresen tano.
ElencaUtenti	Consente di recuperare gli utenti negli HSM, il tipo e l'ID utente nonché altri attributi.
loginHSM e logoutHSM	Accedere e uscire dagli HSM in un cluster.

Comando	Descrizione
ImpostaAttributo	Consente di convertire una chiave di sessione in una chiave persistente.
Firma	Generare una firma per un file utilizzando una chiave privata scelta.
AnnullaWrapChiave	Consente di importare nei moduli HSM una chiave di wrapping (crittografata) da un file.
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.

aesWrapUnwrap

Il comando aesWrapUnwrap esegue la crittografia o la decodifica dei contenuti 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.

aesWrapUnwrap utilizza Wrapping Chiave AES. 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 comando imSymKey per importare una chiave crittografata, il comando restituisce un errore poiché la chiave crittografata non presenta il formato richiesto per una chiave di testo normale di questo 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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo 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`.

Campo 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.

Campo 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 nuova chiave di wrapping, utilizza genSymKey per creare una chiave AES (tipo 31).

Campo 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.

Campo 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`.

Campo obbligatorio: no

Argomenti correlati

- exSymKey
- imSymKey
- unWrapKey
- wrapKey

deleteKey

Il comando deleteKey in key_mgmt_util consente di eliminare una chiave dall'HSM. 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 deleteKey per eliminare le chiavi dai moduli HSM.

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.
```

Parametri

-h

Visualizza il testo di aiuto della riga di comando per il comando.

Campo 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

StringaErrore2

Il comando assistente Error2String in key_mgmt_util restituisce l'errore che corrisponde a un codice di errore esadecimale 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, getKeyInfo indica che l'utente corrente (utente 4) può utilizzare la chiave nelle operazioni di crittografia. 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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

-r

Specifica un codice di errore esadecimale. L'indicatore esadecimale `0x` è obbligatorio.

Campo obbligatorio: sì

Esci

Il comando exit in key_mgmt_util ti fa 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, è necessario avviare key_mgmt_util.

Sintassi

```
exit
```

Parametri

Questo comando non ha parametri.

Argomenti correlati

- Avvio di key_mgmt_util

EsportaChiavePrivata

Il comando exportPrivateKey in key_mgmt_util esporta una chiave privata asimmetrica in un HSM su 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 che abbia `OBJ_ATTR_WRAP` e un valore attributi `OBJ_ATTR_DECRYPT` di 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

```
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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-k`

Specifica l'handle della chiave privata da esportare.

Campo 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.

Campo obbligatorio: sì

`-out`

Consente di specificare il nome del file in cui verrà scritta la chiave privata esportata.

Campo 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`)

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- ImportaChiavePrivata
- wrapChiave
- AnnullaWrapChiave
- GeneraChiaveSimmetrica

exportPubKey

Il comando exportPubKey in key_mgmt_util esporta una chiave pubblica di 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, utilizzare 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto della riga di comando per il comando.

Campo obbligatorio: sì

`-k`

Specifica l'handle della chiave pubblica da esportare.

Campo obbligatorio: sì

`-out`

Specifica il nome del file in cui verrà scritta la chiave pubblica esportata.

Campo obbligatorio: sì

Argomenti correlati

- importPubKey
- Genera Chiavi

exSymKey

Il comando exSymKey nello strumento key_mgmt_util esporta una copia non crittografata di una chiave simmetrica dall'HSM e la memorizza 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 di testo non crittografata, come quelle esportate da `exSymKey`, utilizzare 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, utilizza exportPubKey. Per esportare le chiavi private, utilizza exportPrivateKey.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'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 exSymKey per esportare le chiavi simmetriche di tua proprietà dai moduli HSM.

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 utilizza genSymKey per creare 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 per l'attributo della 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 di (wordcount) wc, 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"
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo 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, utilizza il comando getKeyInfo.

Campo 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 nuova chiave di wrapping, utilizza genSymKey per creare una chiave AES (digita 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.

Campo 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.

Campo obbligatorio: sì

-m

Specifica il meccanismo di wrapping. L'unico valore valido è 4, che rappresenta il meccanismo NIST_AES_WRAP.

Campo 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 di testo non crittografato.

Campo obbligatorio: no

Impostazione predefinita: utilizzo della chiave di wrapping sull'HSM per annullare il wrapping.

Argomenti correlati

- genSymKey
- imSymKey
- wrapKey

extractMaskedObject

Il comando extractMaskedObject in key_mgmt_util estrae una chiave da un modulo HSM e la salva su un file come oggetto nascosto. 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-o`

Specifica l'handle della chiave da estrarre come oggetto mascherato.

Campo 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 cluster AWS CloudHSM da un backup precedente

findKey

Utilizzare il comando findKey in key_mgmt_util per cercare le chiavi attraverso i valori degli attributi delle chiavi. 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 findKey per trovare e identificare le chiavi negli HSM.

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 negli HSM.

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 determinata chiave, utilizzare getKeyInfo o findAllKeys in 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
```

Parametri

-h

Visualizza l'assistenza per il comando.

Campo 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

Campo 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.

Campo 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`.

Campo 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`.

Campo obbligatorio: no

-sess

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`.

Campo obbligatorio: no

-u

Trova le chiavi che gli utenti specificati e l'utente corrente condividono. Digitare un elenco separato da virgole degli ID utente HSM, come `-u 3` o `-u 4,7`. Per trovare gli ID degli utenti su un HSM, utilizzare listUsers.

Quando si specifica un ID utente, findKey restituisce le chiavi per quell'utente. Quando si specificano più ID utente, findKey restituisce le chiavi che possono utilizzare tutti gli utenti specificati.

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 che qualsiasi utente possiede o condivide, i crypto officer (CO) possono utilizzare findAllKeys in cloudhsm_mgmt_util.

Campo 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).

Campo 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. È anche possibile calcolare un KCV al di fuori dell'HSM, ad esempio dopo aver esportato una chiave. Si possono poi 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 di un blocco 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.

Campo 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
- getAttribute
- findAllKeys in cloudhsm_mgmt_util
- Documentazione di riferimento per l'attributo della chiave

findSingleKey

Il comando findSingleKey nello strumento key_mgmt_util verifica che una chiave sia presente in tutti i moduli HSM del cluster.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'HSM come crypto user (CU).

Sintassi

```
findSingleKey -h

findSingleKey -k <key-handle>
```

Esempio

Example

Questo comando verifica che la chiave 252136 sia presente su tutti e tre moduli HSM nel cluster.

```
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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo 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

genDSAKeyPair

Il comando genDSAKeyPair nello strumento key_mgmt_util genera una coppia di chiavi Digital Signing Algorithm (DSA) nei tuoi moduli 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 di un utente specifico, utilizza 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 getKeyInfo sulla chiave privata (handle chiave 17). 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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

-m

Specifica la lunghezza del modulo in bit. L'unico valore valido è `2048`.

Campo 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.

Campo 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 dell'ID.

Campo obbligatorio: no

-min_srv

Specifica il numero minimo di moduli HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un valore inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave importata. 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 `m_value` di una chiave, utilizzare 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

Campo obbligatorio: no

-successivo

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.

Campo 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 i dati che potrebbe essere necessario decodificare dopo la fine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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.

Campo obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri crypto user (CU) dell'HSM l'autorizzazione ad utilizzare questa chiave nelle operazioni di crittografia. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digita un elenco separato da virgole degli ID utente dell'HSM, come `-u 5,6`. Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare gli ID utente dell'HSM dei CU sull'HSM, utilizza ElencaUtenti. 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 importata.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- genRSAKeyPair
- genSymKey
- genECCKeyPair

genECCKeyPair

Il comando `genECCKeyPair` nello strumento key_mgmt_util genera una coppia di chiavi con Crittografia a curva ellittica (ECC) nei moduli 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 di un utente specifico, utilizza 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

Questi esempi mostrano come utilizzare genECCKeyPair per creare coppie di chiavi ECC negli HSM.

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 HSM con modalità FIPS, il comando ha esito negativo. Il messaggio riporta che un server del cluster non è disponibile, ma questo in genere non indica un problema con i moduli HSM nel cluster.

```
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
```

Parametri

-h

Visualizza l'assistenza per il comando.

Campo 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

Campo obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la chiave privata. 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.

Campo 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.

Campo obbligatorio: no

-min_srv

Specifica il numero minimo di moduli HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su una cifra inferiore ai moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

-m_valore

Specifica il numero di utenti che devono approvare le operazioni di crittografia che utilizzano la chiave pubblica e privata della coppia. Digitare 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 `m_value` di una chiave, utilizzare 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

Campo obbligatorio: no

-successivo

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.

Campo obbligatorio: no

-sessione

Crea una chiave che esiste solo nella sessione corrente. La chiave non può essere recuperata al termine 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 potrebbe aver bisogno di decodificare dopo la fine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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 indeterminato e viene restituito solo quando la chiave è sincronizzata con il numero minimo di server.

Campo obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri crypto user (CU) HSM l'autorizzazione per utilizzare questa chiave nelle operazioni di crittografia. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digitare un elenco separato da virgole di ID utenti HSM, come -u `5,6`. Non includere l'ID utente HSM dell'utente attuale. Per trovare gli ID utente HSM dei CU su HSM, utilizza ElencaUtenti. 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 importata.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- genSymKey
- genRSAKeyPair
- genDSAKeyPair

genRSAKeyPair

Il comando genRSAKeyPair nello strumento key_mgmt_util genera una coppia di chiavi asimmetriche 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 di un utente specifico, utilizza 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 genRSAKeyPair per creare coppie di chiavi asimmetriche nei moduli HSM.

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 Riferimento per l'attributo della chiave.

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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

-m

Specifica la lunghezza del modulo in bit. Il valore minimo è 2048.

Campo obbligatorio: sì

-e

Specifica l'esponente pubblico. Il valore deve essere un numero dispari maggiore o uguale a 65537.

Campo obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la chiave privata. 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.

Campo 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 dell'ID.

Campo obbligatorio: no

-min_srv

Specifica il numero minimo di moduli HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un valore inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo 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 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 `m_value` di una chiave, utilizzare 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

Campo obbligatorio: no

-successivo

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.

Campo 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 potresti aver bisogno di decodificare dopo la fine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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.

Campo obbligatorio: no

-u

Condivide la chiave privata della coppia con gli utenti specificati. Questo parametro fornisce agli altri crypto user (CU) HSM l'autorizzazione per utilizzare questa chiave nelle operazioni di crittografia. Le chiavi pubbliche possono essere utilizzate da qualsiasi utente senza condividerle.

Digitare un elenco separato da virgole degli ID utente dell'HSM, come `-u 5,6`. Non includere l'ID utente HSM dell'utente attuale. Per trovare gli ID utente dell'HSM dei CU su HSM, utilizza ElencaUtenti. 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.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- genSymKey
- genDSAKeyPair
- genECCKeyPair

genSymKey

Il comando genSymKey nello strumento key_mgmt_util consente di generare una chiave simmetrica nei moduli 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 genSymKey per creare chiavi simmetriche nei moduli HSM.

> **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 la sincronizzazione della chiave con tutti i moduli HSM nel 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 viene sincronizzata con il numero minimo di moduli HSM specificato 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 di un utente specifico, utilizza getKeyInfo. Per trovare le chiavi in base ai valori degli attributi, usa findKey.

Per informazioni sull'interpretazione degli attributi chiave, vedi Riferimento per l'attributo della chiave.

```
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 (Parametri)

-h

Visualizza l'aiuto per il comando.

Campo 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: SEGRETA_GENERICA. Una chiave segreta generica è una matrice di byte non conforme a un determinato standard, come i requisiti per una chiave AES.
- 18: RC4. Le chiavi RC4 non sono valide sui moduli HSM con modalità FIPS.
- 21: Triple DES (3DES). Non consentito dopo il 2023 per la conformità FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.
- 31: AES

Campo 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)

Campo obbligatorio: sì

-l

Specifica un'etichetta definita dall'utente per la chiave privata. 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.

Campo 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.

Campo 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.

Campo obbligatorio: no

-min_srv

Specifica il numero minimo di HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un cifra inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo 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 `m_value` di una chiave, utilizzare 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

Campo obbligatorio: no

-successivo

Rende la chiave non estraibile. La chiave generata non può essere esportata dall'HSM.

Impostazione predefinita: la chiave è estraibile.

Campo 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 dopo la fine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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.

Campo obbligatorio: no

-u

Condivide la chiave con gli utenti specificati. Questo parametro fornisce agli altri crypto user (CU) HSM l'autorizzazione ad utilizzare questa chiave nelle operazioni di cifratura.

Digita un elenco separato da virgole degli ID utente HSM, come `-u 5,6`. Non includere l'ID utente dell'HSM dell'utente attuale. Per trovare gli ID utente HSM dei CU su HSM, utilizzare ElencaUtenti. 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.

Campo obbligatorio: no

Argomenti correlati

- exSymKey
- genRSAKeyPair
- genDSAKeyPair
- genECCKeyPair
- setAttribute

getAttribute

Il comando getAttribute in key_mgmt_util scrive uno o tutti i valori degli attributi per una chiave AWS CloudHSM su un file. 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 attributo di una chiave da tutti i moduli HSM in 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 and setAttribute in cloudhsm_mgmt_util. Per informazioni sull'interpretazione degli attributi chiave, vedi Riferimento per l'attributo della chiave.

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 getAttribute per ottenere gli attributi delle chiavi nei tuoi HSM.

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 Riferimento per l'attributo della chiave.

```
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 Riferimento per l'attributo della chiave.

```
$  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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo 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. È possibile trovare gli utenti di una chiave utilizzando getKeyInfo.

Campo 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 Riferimento per l'attributo della chiave.

Campo obbligatorio: sì

-out

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

getCaviumPrivKey

Il comando getCaviumPrivKey in key_mgmt_util esporta una chiave privata da un HSM in formato PEM falso. Il file PEM falso, che non contiene il materiale della chiave privata attuale ma riferimenti alla chiave privata nell'HSM, può essere utilizzato per stabilire l'offload SSL/TLS dal server Web a AWS CloudHSM. Per ulteriori informazioni, vedi Offload SSL/TLS su Linux.

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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-k`

Specifica l'handle della chiave privata da esportare in formato PEM falso.

Campo obbligatorio: sì

`-out`

Consente di specificare il nome del file in cui la chiave PEM falsa verrà scritta.

Campo obbligatorio: sì

Argomenti correlati

- importPrivateKey
- Offload SSL/TLS su Linux

getCert

Il comando getCert in key_mgmt_util recupera una partizione di certificati HSM e li salva 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-f`

Consente di specificare il nome del file in cui il certificato recuperato verrà salvato.

Campo 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
- getCert (in cloudhsm_mgmt_util)

getKeyInfo

Il comando getKeyInfo in key_mgmt_util restituisce gli ID degli utenti HSM che possono utilizzare la chiave, inclusi il proprietario e dei crypto user (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 gli ID degli utenti HSM nei tuoi moduli HSM, utilizza ElencaUtenti. 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 il comando 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 presenta l'handle 21. La chiave privata presenta l'handle 20.

Quando esegui getKeyInfo sulla chiave privata (20), il comando restituisce il proprietario della chiave (3) e i crytpo user (CU) 4 e 5, con i quali 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 il comando genRSAKeyPair per creare una coppia di chiavi RSA che viene 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
```

Parametri

-h

Visualizza il testo di aiuto per il comando.

Campo 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

- getKeyInfo in cloudhsm_mgmt_util
- ElencaUtenti
- findKey

- findAllKeys in cloudhsm_mgmt_util

Aiuto

Il comando help nella key_mgmt_util mostra le informazioni su tutti i comandi key_mgmt_util disponibili.

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
```

Parametri

Questo comando non ha parametri.

Argomenti correlati

- loginHSM e logoutHSM

importPrivateKey

Il comando importPrivateKey in key_mgmt_util importa una chiave privata asimmetrica da un file in un 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 stai cercando di associare una chiave AWS CloudHSM con un certificato, fai riferimento a questo argomento.

> **Note**
>
> Non è possibile importare una chiave PEM protetta da password utilizzando una chiave simmetrica o privata.

Bisogna 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, utilizzare 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 a 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 della chiave 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto della riga di comando per il comando.

Campo obbligatorio: sì

`-l`

Specifica un'etichetta definita dall'utente per la chiave privata.

Campo obbligatorio: sì

`-f`

Specifica il nome del file della chiave da importare.

Campo obbligatorio: sì

`-w`

Specifica l'handle di chiave di una chiave di wrapping. Questo parametro è obbligatorio. Per trovare le handle della chiave, utilizzare 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, utilizzare genSymKey per creare una chiave AES (digitare 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.

Campo obbligatorio: sì

`-sess`

Specifica la chiave importata come una chiave di sessione.

Impostazione predefinita: la chiave importata è detenuta come persistente (token) nel cluster.

Campo obbligatorio: no

`-id`

Specifica l'ID della chiave da importare.

Impostazione predefinita: nessun valore dell'ID.

Campo 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`.

Di default: 0

Campo obbligatorio: no

`-min_srv`

Specifica il numero minimo di HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creato.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un numero inferiore di HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

`-timout`

Specifica il numero di secondi di attesa per la sincronizzazione della chiave tra HSM quando viene incluso il parametro `min-serv`. Se non viene specificato un numero, il polling prosegue in eterno.

Impostazione predefinita: nessun limite

Campo obbligatorio: no

`-u`

Specifica l'elenco degli utenti con cui condividere la chiave privata importata. Questo parametro fornisce agli altri utenti crittografici HSM (crypto user, CU) l'autorizzazione per utilizzare la chiave importata nelle operazioni di cifratura.

Inserire un elenco separato da virgole degli ID utente HSM, come `-u 5,6`. Non includere l'ID utente HSM dell'utente attuale. Per trovare gli ID utente HSM dei CU su HSM, utilizzare listUsers.

Impostazione predefinita: soltanto l'utente attuale può utilizzare la chiave importata.

Campo 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.

Default: Utilizza il codice di wrapping specificato nel parametro `-w` per eseguire il wrapping e per annullarlo.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- wrapKey
- unWrapKey
- genSymKey
- exportPrivateKey

importPubKey

Il comando importPubKey in key_mgmt_util importa una chiave pubblica in un formato PEM in un 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-l`

Specifica un'etichetta definita dall'utente per la chiave pubblica.

Campo obbligatorio: sì

`-f`

Specifica il nome del file della chiave da importare.

Campo obbligatorio: sì

`-sess`

Designa la chiave importata come una chiave di sessione.

Impostazione predefinita: la chiave importata è detenuta come persistente (token) nel cluster.

Campo obbligatorio: no

`-id`

Specifica l'ID della chiave da importare.

Impostazione predefinita: nessun valore dell'ID.

Campo obbligatorio: no

`-min_srv`

Specifica il numero minimo di moduli HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creato.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un valore inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

`-timeout`

Specifica il numero di secondi di attesa per la sincronizzazione della chiave tra moduli HSM quando viene incluso il parametro `min-serv`. Se non viene specificato un numero, il processo prosegue a tempo indefinito.

Impostazione predefinita: nessun limite

Campo obbligatorio: no

Argomenti correlati

- exportPubKey
- Genera Chiavi

imSymKey

Il comando imSymKey nello strumento key_mgmt_util importa una copia non crittografata di una chiave simmetrica da un file nell'HSM. È possibile utilizzarla per importare le chiavi generate con qualsiasi metodo al di fuori dell'HSM e le chiavi esportate da un HSM, come ad esempio le chiavi che il comando exSymKey scrive su un file.

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 le chiavi crittografate, utilizzare i comandi wrapKey e unWrapKey.

Inoltre, il comando imSymKey importa solo le chiavi simmetriche. Per importare le chiavi pubbliche, utilizzare importPubKey. Per importare le chiavi private, utilizzare importPrivateKey o 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 utilizzare imSymKey per importare le chiavi simmetriche nei moduli HSM.

Example : importazione di una chiave simmetrica AES

In questo esempio viene utilizzato imSymKey per importare una chiave simmetrica AES nei moduli HSM.

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-forImport.key 32
```

Il secondo comando utilizza imSymKey per importare la chiave AES dal file `aes256.key` nei moduli HSM. 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 che esiste sia nei moduli HSM che nei cluster. Una volta che la chiave di wrapping condivisa è stata posizionata, è possibile utilizzare wrapKey e unWrapKey per spostare le chiavi crittografate tra i cluster.

L'utente CU che effettua questa operazione deve avere il permesso di accedere ai moduli HSM su entrambi i cluster.

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 sui moduli HSM 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 key_mgmt_util nel cluster 2 e viene eseguito un comando imSymKey per importare la chiave nel file `aes.key` nei moduli HSM del cluster 2. Questo comando utilizza la chiave 252152, una chiave AES nei moduli HSM nel cluster 2, come chiave di wrapping.

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 negli HSM di entrambi i cluster, è ora possibile condividere le chiavi crittografate tra i cluster senza esporre la chiave non crittografata. 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
```

Parametri

-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.

Campo obbligatorio: no

-f

Specifica il file che contiene la chiave da importare.

Il file deve contenere una copia non crittografata di una chiave AES o Triple DES della lunghezza specificata. Le chiavi RC4 e DES non sono valide nei moduli HSM con modalità FIPS.

- AES: 16, 24 o 32 byte
- Triple DES (3DES): 24 byte

Campo obbligatorio: sì

-h

Visualizza l'aiuto per il comando.

Campo 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 dell'ID.

Campo obbligatorio: no

-l

Specifica un'etichetta definita dall'utente per la chiave. Digita una stringa

È possibile utilizzare qualsiasi frase che consenta di identificare la chiave. Poiché l'etichetta non deve essere necessariamente univoca, è possibile utilizzarla per raggruppare e classificare le chiavi.

Campo obbligatorio: sì

-min_srv

Specifica il numero minimo di HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su una cifra inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

-sessione

Crea una chiave che esiste solo per la 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.

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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.

Campo 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

Campo obbligatorio: sì

-u

Condivide la chiave importata con utenti specificati. Questo parametro fornisce agli altri utenti crittografici HSM (CU) l'autorizzazione per utilizzare questa chiave nelle operazioni di cifratura.

Digita un ID o un elenco separato da virgole degli ID utente HSM, come -u `5,6`. Non includere l'ID utente 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.

Campo 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 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 nuova chiave di wrapping, utilizza genSymKey per creare una chiave AES (digita 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.

Campo 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.

Campo obbligatorio: no

Argomenti correlati

- genSymKey
- exSymKey
- wrapKey
- unWrapKey

- exportPrivateKey
- exportPubKey

insertMaskedObject

Il comando insertMaskedObject in key_mgmt_util inserisce un oggetto nascosto da un file in un determinato HSM. 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, tra cui le chiavi nonextractable (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 AWS CloudHSM file di configurazione.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto della riga di comando per il comando.

Campo obbligatorio: sì

`-f`

Specifica il nome del file dell'oggetto nascosto da inserire.

Campo 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

Campo 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 polling prosegue in eterno.

Impostazione predefinita: nessun limite

Campo obbligatorio: no

Argomenti correlati

- extractMaskedObject
- syncKey
- Copiare un backup tra regioni
- Creazione di un cluster AWS CloudHSM da un backup precedente

IsValidKeyHandlefile

Il IsValidKeyHandlefile comando in key_mgmt_util viene utilizzato per scoprire se un file 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. Questo file può essere utilizzato per stabilire l'offload SSL/TLS dal server Web a AWS CloudHSM. Per ulteriori informazioni, vedi Offload SSL/TLS su Linux.

> **Note**
>
> IsValidKeyHandlefilefunziona solo per le chiavi RSA.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'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
```

Parametri

Questo comando accetta i parametri seguenti.

`-h`

Visualizza il testo di aiuto per il comando.

Campo obbligatorio: sì

`-f`

Specifica il file di chiave privata RSA da verificare per verificare la presenza di materiale chiave valido.

Campo obbligatorio: sì

Argomenti correlati

- getCaviumPrivChiave
- Offload SSL/TLS su Linux

listAttributes

Il comando listAttributes in key_mgmt_util elenca gli attributi di una chiave AWS CloudHSM e le costanti che li rappresentano. Puoi utilizzare queste costanti per identificare gli attributi nei comandi getAttribute e setAttribute. Per informazioni sull'interpretazione degli attributi delle chiavi, vedi Riferimento per l'attributo della chiave.

Prima di eseguire un comando key_mgmt_util, devi avviare key_mgmt_util e accedere all'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 Riferimento per l'attributo della chiave.

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

- getAttribute
- setAttribute
- Riferimento attributo chiave

ElencaUtenti

Il comando listUsers in key_mgmt_util restituisce gli utenti nei moduli HSM, insieme al tipo di utente e ad altri attributi.

In key_mgmt_util, il comando listUsers restituisce un output che rappresenta tutti i moduli HSM nel cluster, anche se non sono coerenti. 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 i crypto user (CU) sono autorizzati a eseguire. 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 dei moduli HSM nel cluster e i relativi attributi. È possibile utilizzare l'attributo `User ID` 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.
- ErroreLoginCnt: indica il numero di volte in cui l'utente non è riuscito a eseguire l'accesso.
- 2FA: indica che l'utente ha abilitato l'autenticazione a più fattori.

Parametri

-h

Visualizza l'assistenza per il comando.

Campo obbligatorio: sì

Argomenti correlati

- listUsers su cloudhsm_mgmt_util
- findKey
- getAttribute
- getKeyInfo

loginHSM e logoutHSM

È possibile utilizzare i comandi loginHSM e logoutHSM in key_mgmt_util per effettuare l'accesso e la disconnessione da ogni modulo HSM in un cluster. Una volta eseguito l'accesso ai moduli HSM, è possibile usare key_mgmt_util per eseguire un'ampia gamma di operazioni di gestione delle chiavi, tra cui la generazione di chiavi pubbliche e private, la sincronizzazione e il wrapping.

Prima di eseguire qualsiasi comando key_mgmt_util, devi avviare key_mgmt_util. Per gestire le chiavi con key_mgmt_util, è necessario accedere all'HSM come crypto user (CU).

> **Note**
>
> Se superi cinque tentativi di accesso errati, il tuo account viene bloccato. Se il cluster è stato creato prima del 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, puoi consentire ulteriori tentativi di accesso errati prima che l'account venga bloccato. Questo perché il client CloudHSM bilancia il carico su più moduli HSM. 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 illustra come accedere e disconnettersi dai moduli HSM in un cluster con i comandi `logoutHSM` e `loginHSM`.

Example : Accedi ai moduli HSM

Questo comando accede ai moduli HSM come crypto user (`CU`) con il nome utente `example_user` e la password `aws`. L'output mostra che hai effettuato l'accesso a tutti i moduli HSM 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 password. Si immette la password, il sistema la nasconde e l'output mostra che il comando è stato eseguito correttamente e che l'utente si è connesso ai moduli HSM.

```
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 : Disconnetterti dai moduli HSM

Questo comando consente di eseguire la disconnessione dai moduli HSM. L'output mostra che ti sei disconnesso da tutti i moduli HSM del 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
```

Parametri

-h

Mostra aiuto per questo comando.

-u

Specifica il tipo di utente di accesso. Per utilizzare key_mgmt_util, è necessario accedere come CU.

Campo obbligatorio: sì

-s

Specifica il nome utente di accesso.

Campo 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

setAttribute

Il comando setAttribute in key_mgmt_util converte una chiave valida solo per la sessione corrente in una chiave persistente che esiste fino a quando non viene eliminata. 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 parametro `-sess` di genSymKey per 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 Riferimento per l'attributo della chiave.

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
```

Parametri

-h

Visualizza l'aiuto per il comando.

Campo 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.

Campo 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

Firma

Il comando sign in key_mgmt_util utilizza 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-f`

Nome del file da firmare.

Campo obbligatorio: sì

`-k`

L'handle della chiave privata da utilizzare per la firma.

Campo 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
`SHA512_RSA_PKCS_PSS`	8

Meccanismo di firma	Numero intero corrispondente
SHA224_RSA_PKCS_PSS	9
ECDSA_SHA1	15
ECDSA_SHA224	16
ECDSA_SHA256	17
ECDSA_SHA384	18
ECDSA_SHA512	19

Campo obbligatorio: sì

-out

Il nome del file in cui viene salvato il file firmato.

Campo obbligatorio: sì

Argomenti correlati

- Verifica
- importPrivateKey
- genRSAKeyPair
- genECCKeyPair
- genSymKey
- Genera Chiavi

unWrapKey

Il comando unWrapKey dello strumento key_mgmt_util consente di importare una chiave di wrapping (crittografata) simmetrica o privata 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 importate funzionano come chiavi generate da AWS CloudHSM. 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 utilizzare unWrapKey per importare una chiave con wrapping da un file dei moduli HSM. 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 (Parametri)

-h

Visualizza l'aiuto per il comando.

Campo obbligatorio: sì

-f

Il percorso e il nome del file contenente la chiave su cui è stato eseguito il wrapping.

Campo 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, usa genSymKey per generare una chiave AES (tipo 31) o genRSAKeyPair per generare una coppia di chiavi RSA (tipo 0). Se utilizzi una coppia di chiavi RSA, assicurati di avvolgere la chiave con una delle chiavi e di scartarla con l'altra. Per accertarsi 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`.

Campo 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 dopo la fine della sessione.

Per cambiare una chiave di sessione in una chiave persistente (token), usa setAttribute.

Impostazione Predefinita: la chiave è persistente.

Campo obbligatorio: no

-min_srv

Specifica il numero minimo di HSM su cui la chiave importata è sincronizzata prima che il valore del parametro `-timeout` scada. Se la chiave non è sincronizzata sul numero di server specificato nel tempo allocato, non viene creata.

AWS CloudHSM sincronizza automaticamente ogni chiave su ogni modulo HSM nel cluster. Per velocizzare il processo, imposta il valore di `min_srv` su un cifra inferiore al numero di moduli HSM nel cluster e imposta un valore di timeout basso. Tuttavia, alcune richieste potrebbero non generare una chiave.

Impostazione predefinita: 1

Campo obbligatorio: no

timeout

Specifica per quanto tempo (in secondi) il comando attende la sincronizzazione di una chiave con il numero di HSM 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.

Campo 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.

Campo obbligatorio: no

-successivo

Rende la chiave non estraibile. La chiave generata non può essere esportata dall'HSM.

Impostazione predefinita: la chiave è estraibile.

Campo 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, vedere la nota più avanti in questa sezione). Vedi la nota 1 di seguito per una modifica imminente.	12

Campo obbligatorio: sì

> **Note**
>
> Quando utilizzi il meccanismo di wrapping `RSA_OAEP`, la dimensione massima della chiave per cui è possibile eseguire il wrapping viene determinata dalla chiave RSA

e il periodo di hash specificato come segue: dimensione massima della chiave = (modulusLengthInBytes-2* hashLengthInBytes-2).
Quando utilizzi il meccanismo di wrapping RSA_PKCS, la dimensione massima della chiave per cui è possibile eseguire il wrapping viene determinata dalla chiave RSA come segue: la dimensione massima della chiave = (modulusLengthInBytes -11).

-t

Algoritmo hash	Valore
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (valido per i meccanismi RSA_AES e RSA_OAEP)	6

Campo 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.

Campo 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.

Campo 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)

  Campo 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`

  Campo obbligatorio: sì

È inoltre possibile specificare i parametri `-noheader` seguenti:

- -id

  L'ID da aggiungere alla chiave su cui è stato annullato il wrapping.

  Campo obbligatorio: no

- -i

  Il vettore di inizializzazione (IV) da utilizzare per annullare il wrapping.

  Campo obbligatorio: no

[1] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Argomenti correlati

- wrapKey
- exSymKey
- imSymKey

Verifica

Il comando verify in key_mgmt_util conferma se un file è stato firmato da 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 firmati in AWS CloudHSM con l'operazione 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
```

Parametri

Questo comando accetta i parametri seguenti.

`-f`

Il nome del file di messaggio originale.

Campo obbligatorio: sì

`-s`

Il nome del file firmato.

Richiede: sì

`-k`

L'handle della chiave pubblica che si pensa sia stata utilizzata per firmare il file.

Campo 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

wrapKey

Il comando wrapKey in key_mgmt_util esporta una copia criptata di una chiave simmetrica o privata da un HSM a 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 importare la chiave crittografata nell'HSM, utilizzare unWrapKey. Per esportare una chiave di testo non crittografato da un HSM, utilizzare exSymKey o exportPrivateKey come appropriato. Il comando aesWrapUnwrap non è in grado di decrittografare le chiavi (annullarne il wrapping) crittografate con 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 (Parametri)

-h

Visualizza l'aiuto per il comando.

Campo 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 Riferimento per l'attributo della chiave.

Puoi esportare solo le chiavi di tua proprietà. Per trovare il proprietario di una chiave, utilizza il comando getKeyInfo.

Campo 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, usa genSymKey per generare una chiave AES (tipo 31) o genRSAKeyPair per generare una coppia di chiavi RSA (tipo 0). Se utilizzi una coppia di chiavi RSA, assicurati di avvolgere la chiave con una delle chiavi e di scartarla con l'altra. Per accertarsi 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`.

Campo 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.

Campo 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
`RSA_OAEP` (per la dimensione massima dei dati, vedi la nota più avanti in questa sezione)	8
`AES_GCM`	10
`CLOUDHSM_AES_GCM`	11

Meccanismo	Valore
RSA_PKCS (per la dimensione massima dei dati, vedere la nota più avanti in questa sezione). Vedi la nota 1 di seguito per una modifica imminente.	12

Campo obbligatorio: sì

> **Note**
>
> Quando utilizzi il meccanismo di wrapping RSA_0AEP, la dimensione massima della chiave per cui è possibile eseguire il wrapping viene determinata dalla chiave RSA e il periodo di hash specificato come segue: la dimensione massima della chiave = (modulusLengthInBytes-2* hashLengthInBytes-2).
> Quando utilizzi il meccanismo di wrapping RSA_PKCS, la dimensione massima della chiave per cui è possibile eseguire il wrapping viene determinata dal modulo della chiave RSA come segue: la dimensione massima della chiave = (modulusLengthInBytes -11).

-t

Il valore che rappresenta l'algoritmo hash. CloudHSM supporta i seguenti algoritmi:

Algoritmo hash	Valore
SHA1	2
SHA256	3
SHA384	4
SHA512	5
SHA224 (valido per i meccanismi RSA_AES e RSA_0AEP)	6

Campo obbligatorio: no

-aad

Il nome del file contenente AAD.

> **Note**
>
> Valido solo per i meccanismi `AES_GCM` e `CLOUDHSM_AES_GCM`.

Campo 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.

Campo 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`.

Campo 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`.

Campo 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.

Campo obbligatorio: no

[1] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Argomenti correlati

- exSymKey
- imSymKey
- unWrapKey

Riferimento per l'attributo della chiave

I comandi key_mgmt_util utilizzano costanti per rappresentare gli attributi delle chiavi in un 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: True. La chiave può essere utilizzata per decodificare i dati.
OBJ_ATTR_DERIVE	268	0: Falso. 1: True. La funzione ricava la chiave.
OBJ_ATTR_DESTROYABLE	370	0: Falso. 1: True.
OBJ_ATTR_ENCRYPT	260	0: Falso. 1: True. La chiave può essere utilizzata per crittografare i dati.
OBJ_ATTR_EXTRACTABLE	354	0: Falso.

Attributo	Costante	Valori
		1: True. La chiave può essere esportata dai moduli HSM.
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 nei moduli HSM. 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 ECPoint "Q" in formato 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. La chiave non può essere esportata dai moduli HSM.
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: True. 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 al 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 chiave:

- Chiavi simmetriche: primi 3 byte del risultato della crittografia di un blocco 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.

SDK del client AWS CloudHSM

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 presenta due versioni principali e Client SDK 5 è la più recente. 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 Piattaforme supportate da Client SDK 5.

Per informazioni sull'utilizzo di Client SDK 3, consulta la pagina Client SDK precedente (Client SDK 3).

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”

OpenSSL Dynamic Engine AWS CloudHSM consente di eseguire l'offload delle operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

the section called “Provider JCE”

Il provider JCE AWS CloudHSM è conforme alla Java Cryptographic Architecture (JCA). Il provider consente di eseguire operazioni crittografiche sull'HSM.

the section called “Provider KSP e CNG”

Il client AWS CloudHSM per Windows include i provider CNG e KSP. Attualmente, solo Client SDK 3 supporta i provider CNG e KSP.

Piattaforme supportate da Client SDK 5

Il supporto di base è diverso per ogni versione di Client SDK dell'AWS CloudHSM. Il supporto della piattaforma per i componenti di un SDK in genere corrisponde al supporto di base, ma non sempre. Per stabilire il supporto della piattaforma per un determinato componente, assicurati innanzitutto

che la piattaforma desiderata compaia nella sezione di base dell'SDK, quindi verifica la presenza di esclusioni o qualsiasi altra informazione pertinente nella sezione del componente.

L'AWS CloudHSM supporta solo sistemi operativi a 64 bit.

Il supporto della piattaforma cambia nel tempo. Le versioni precedenti del CloudHSM Client SDK potrebbero non supportare tutti i sistemi operativi elencati qui. Utilizza le note sulla versione per determinare il supporto del sistema operativo per le versioni precedenti di CloudHSM Client SDK. Per ulteriori informazioni, vedi Download per Client SDK dell'AWS CloudHSM.

Per le piattaforme supportate per il precedente Client SDK, vedi Piattaforme supportate da Client SDK 3

Client SDK 5 non richiede un client daemon.

Supporto Linux per Client SDK 5

Piattaforme supportate	Architettura x86_64	Architettura ARM
Amazon Linux 2	Sì	Sì
Amazon Linux 2023	Sì	No
CentOS 7 (7.8+)	Sì	No
Red Hat Enterprise Linux 7 (7.8+)	Sì	No
Red Hat Enterprise Linux 8 (8.3+)	Sì	No
Red Hat Enterprise Linux 9 (9.2+)	Sì	No
Ubuntu 20.04 LTS	Sì	No
Ubuntu 22.04 LTS	Sì	No

Nota: 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 Client SDK 5

- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Supporto serverless per Client SDK 5

- AWS Lambda
- Docker/ECS

Supporto per componenti

Libreria PKCS #11

La libreria PKCS #11 è un componente multipiattaforma che corrisponde al supporto di base di Linux e Windows per 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”.

OpenSSL Dynamic Engine

OpenSSL Dynamic Engine è un componente solo per Linux che richiede OpenSSL 1.0.2, 1.1.1 o 3.x.

Provider JCE

Il provider JCE è un Java SDK compatibile con OpenJDK 8, OpenJDK 11, OpenJDK 17 e OpenJDK 21 su tutte le piattaforme supportate.

Vantaggi del Client SDK 5

Rispetto al Client SDK 3, il 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.

Funzionalità e meccanismi aggiuntivi

Il Client SDK 5 include funzionalità e meccanismi aggiuntivi che non sono disponibili nel Client SDK 3 e il Client SDK5 continuerà ad aggiungere altri meccanismi in futuro.

Migrazione da Client SDK 3 a Client SDK 5

Per istruzioni dettagliate sulla migrazione da Client SDK 3 a Client SDK 5, consulta le istruzioni di migrazione per ogni singolo Client SDK:

- Migra la tua libreria PKCS #11 da Client SDK 3 a Client SDK 5
- Migra il tuo OpenSSL Dynamic Engine da Client SDK 3 a Client SDK 5
- Esegui la migrazione del tuo provider JCE da Client SDK 3 a Client SDK 5
- Migrazione da Client SDK 3 CMU e KMU a Client SDK 5 CloudHSM CLI

Per funzionalità o casi d'uso non supportati dalla CLI di CloudHSM, contatta l'assistenza.

> **Note**
>
> La libreria Client SDK 5 PKCS #11 è ora supportata su piattaforme Windows. È in grado di gestire la maggior parte dei casi d'uso in cui i provider CNG e KSP possono e devono essere considerati sostitutivi. Attualmente KSP è disponibile solo in Client SDK 3.

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.

Per informazioni sul processo di bootstrap, consulta la pagina Connessione al cluster. Per la risoluzione dei problemi, vedereProblemi noti per la libreria PKCS #11.

Per informazioni sull'utilizzo di Client SDK 3, consulta la pagina Client SDK precedente (Client SDK 3).

Argomenti

- Installa Client SDK 5 per la libreria PKCS #11
- Libreria PKCS #11
- Tipi di chiave supportati
- Meccanismi supportati
- Operazioni API supportate
- Attributi chiave supportati
- Codici di esempio per la libreria PKCS #11
- Migra la tua libreria PKCS #11 da Client SDK 3 a Client SDK 5
- Configurazioni avanzate per PKCS #11

Installa Client SDK 5 per la libreria PKCS #11

Questo argomento fornisce istruzioni per l'installazione della versione più recente della libreria PKCS #11 delle diverse versioni di 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.

Installazione

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 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
```

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
```

CentOS 7 (7.8+)

Installa la libreria PKCS #11 per CentOS 7.8+ 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
```

RHEL7 (7.8+)

Installa la libreria PKCS #11 per RHEL 7.9+ 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
```

RHEL8 (8.3+)

Installa la libreria PKCS #11 per RHEL 8.3+ sull'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
```

RHEL9 (9.2+)

Installa la libreria PKCS #11 per RHEL9 (9.2+) sull'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
```

Ubuntu 20.04 LTS

Installa la libreria PKCS #11 per Ubuntu 20.04 LTS sull'architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-pkcs11_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-pkcs11_latest_u20.04_amd64.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
```

Windows Server 2016

Installa la libreria PKCS #11 per Windows Server 2016 sull'architettura X86_64:

1. Scarica la libreria PKCS #11 per Client SDK 5.

2. Esegui il programma di installazione della libreria PKCS #11 (11-latest.msi) con privilegi amministrativi di Windows. AWSCloudHSMPKCS

Windows Server 2019

Installa la libreria PKCS #11 per Windows Server 2019 sull'architettura X86_64:

1. Scarica la libreria PKCS #11 per Client SDK 5.
2. Eseguite il programma di installazione della libreria PKCS #11 (AWSCloudHSMPKCS11-latest.msi) con privilegi amministrativi di Windows.

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:\ProgramFiles\Amazon\CloudHSM
     ```

     Script di configurazione e file di log Windows:

     ```
     C:\ProgramData\Amazon\CloudHSM
     ```

Libreria PKCS #11

Quando si utilizza la libreria PKCS #11, l'applicazione viene eseguita come crypto user (CU) specifico negli HSM. 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 formato seguente:

```
<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

La libreria PKCS #11 supporta i seguenti tipi di chiave.

Tipo di chiavi	Descrizione
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 a seguire per una modifica imminente.
EC	Genera chiavi con le curve secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384) e secp521r1 (P-521).
GENERIC_SECRET	Genera segreti generici da 1 a 800 byte.
RSA	Genera chiavi RSA da 2048 bit a 4096 bit, con incrementi di 256 bit.

[1] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Meccanismi supportati

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.

La libreria PKCS #11 supporta i seguenti algoritmi:

- Crittografia e decrittografia: AES-CBC, AES-CTR, AES-ECB, AES-GCM, DES3-CBC, DES3-ECB, RSA-OAEP e RSA-PKCS
- Firma e verifica: RSA, HMAC e ECDSA; con e senza hashing
- Hash/digest: SHA1, SHA224, SHA256, SHA384 e SHA512
- Wrapping della chiave: AES Key Wrap[1], AES-GCM, RSA-AES e RSA-OAEP

Argomenti

Funzioni di generazione di chiavi e coppie di chiavi

La libreria software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni di generazione di chiavi e coppie di chiavi.

- `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_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 software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni di firma e verifica. Con Client SDK 5, i dati sono sottoposti a hash in locale 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, invece, vi 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

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 File digest

La libreria software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni File digest. Con Client SDK 5, i dati sono sottoposti a hash in locale 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 software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni di crittografia e decrittografia.

- `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 software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni di derivazione.

- `CKM_SP800_108_COUNTER_KDF`

Funzioni di wrapping e annullamento del wrapping

La libreria software AWS CloudHSM per la libreria PKCS #11 consente di utilizzare i seguenti meccanismi per le funzioni di wrapping e annullamento del wrapping.

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

Annotazioni sui meccanismi

- [1] 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.
- [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 aDimensione massima dei dati per ogni meccanismo.
- [3] Meccanismo definito dal fornitore. Per utilizzare i meccanismi definiti dal fornitore CloudHSM, le applicazioni PKCS #11 devono includere `/opt/cloudhsm/include/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.
- [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] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Operazioni API supportate

La libreria PKCS #11 supporta le seguenti operazioni API PKCS #11.

- `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 supportati

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 utilizzi la libreria PKCS #11, 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.

Interpretazione della tabella degli attributi relativi alla libreria PKCS #11

La tabella della libreria PKCS #11 contiene un elenco di attributi che differiscono per i tipi di chiave. Indica se un determinato attributo è supportato da un particolare tipo di chiave quando si usa una determinata funzione di crittografia 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`	✓	✓	✓	✓	

Attributo	Tipo di chiavi				Valore predefinito
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
CKA_DERIVE	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	False
CKA_SIGN_RECOVER	✖	✖	✖	✖	
CKA_VERIFY	✖	✓	✖	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	

Attributo	Tipo di chiavi				Valore predefinito
CKA_WRAP	✖	✓	✖	✓	False
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	
CKA_TRUSTED	✖	✓	✖	✓	False
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	False
CKA_UNWRAP	✓	✖	✓	✖	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	

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
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False

Attributo	Tipo di chiavi			Valore predefinito
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	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
CKA_WRAP_WITH_TRUSTED	✓	✓	✓	False
CKA_UNWRAP	✓	✓	✖	False

Attributo	Tipo di chiavi			Valore predefinito
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	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	

Attributo	Tipo di chiavi							Valore predefinito
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

Attributo	Tipo di chiavi							Valore predefinito
		EC privato	EC pubblico	RSA privato	RSA pubblico	AES	DES3	Segreto generico	

Attributo	Tipo di chiavi							Valore predefinito
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
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓	False

Attributo	Tipo di chiavi							Valore predefinito
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	✖	✖	False
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓	False
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	R	✖	R	✖	R	R	R	

Attributo	Tipo di chiavi							Valore predefinito
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	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✓	✖	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✓	✖	✖	✖	✖	
CKA_PRIVATE_EXPONENT	✖	✖	✓[2]	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✓[2]	✓[2]	✖	✖	✖	

Attributo	Tipo di chiavi							Valore predefinito
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	✓	✓	✓	✓	✓	

Attributo	Tipo di chiavi					Valore predefinito
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
CKA_DESTROYABLE	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✖	✖	✖	✖	False
CKA_VERIFY	✖	✖	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi					Valore predefinito
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	
CKA_MODULUS	✖	✖	✖	✖	✖	
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	✖	✖	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	✖	✖	
CKA_EC_PARAMS	✖	✖	✖	✖	✖	
CKA_EC_POINT	✖	✖	✖	✖	✖	
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
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	True
CKA_DESTROYABLE	✓[1]	✓[1]	✓[1]	True
CKA_SIGN	✓	✓	✓	False

Attributo	Tipo di chiavi			Valore predefinito
CKA_SIGN_RECOVER	✖	✖	✖	
CKA_VERIFY	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_UNWRAP	✓	✓	✖	False
CKA_SENSITIVE	R	R	R	True
CKA_EXTRACTABLE	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
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	✖	✖	✖	
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	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓

Attributo	Tipo di chiavi						
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓
CKA_SIGN_RECOVER	✖	✖	✓	✖	✖	✖	✖
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✖	✖	✖	✓	✖	✖	✖
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓

Attributo	Tipo di chiavi						
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	R	R
CKA_MODULUS	✖	✖	✓	✓	✖	✖	✖
CKA_MODULUS_BITS	✖	✖	✖	✓	✖	✖	✖
CKA_PRIME_1	✖	✖	S	✖	✖	✖	✖
CKA_PRIME_2	✖	✖	S	✖	✖	✖	✖
CKA_COEFFICIENT	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_1	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖

Attributo	Tipo di chiavi						
CKA_PUBLIC_EXPONENT	✖	✖	✓	✓	✖	✖	✖
CKA_EC_PARAMS	✓	✓	✖	✖	✖	✖	✖
CKA_EC_POINT	✖	✓	✖	✖	✖	✖	✖
CKA_VALUE	S	✖	✖	✖	✓	✓	✓
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.

Modifica degli attributi

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

La specifica nel modello di un attributo non supportato da una chiave specifica 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.

Codici di esempio per la libreria PKCS #11

Gli esempi di codice riportati GitHub mostrano come eseguire attività di base utilizzando la libreria PKCS #11.

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

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

Migra la tua libreria PKCS #11 da Client SDK 3 a Client SDK 5

Utilizzate questo argomento per migrare la libreria PKCS #11 da Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi del Client SDK 5

NelAWS 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 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`
`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-pkcs11
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-pkcs11
   ```

2. 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
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

3. Installa la libreria Client SDK PKCS #11 seguendo la procedura riportata di seguito. Installa Client SDK 5 per la libreria PKCS #11
4. 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
5. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Best practice per AWS CloudHSM

Configurazioni avanzate per PKCS #11

Il provider PKCS #11 AWS CloudHSM include le seguenti configurazioni avanzate che non fanno 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

Connessione a più slot con PKCS #11

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

Prerequisiti per slot multipli

- Due o più cluster AWS CloudHSM a cui desideri connetterti, insieme ai relativi certificati di cluster (ovvero il file CustomerCA.crt per ciascun 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 la pagina Getting started with 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 ???.

Configura la libreria PKCS #11 per la funzionalità multislot

Per configurare la libreria PKCS #11 per la funzionalità multislot, segui questa procedura:

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.

configure-pkcs11 add-cluster

Quando ti connetti a più slot con PKCS #11, utilizza il comando configure-pkcs11 add-cluster 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>]
        [--server-client-cert-file <CLIENT CERTIFICATE FILE>]
        [--server-client-key-file <CLIENT KEY FILE>]
        [-h, --help]
```

Esempi

Aggiungere 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

```
C:\Program Files\Amazon\CloudHSM\> .\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, hai la possibilità di specificare l'endpoint API AWS CloudHSM da utilizzare per la chiamata, che potrebbe essere necessario per varie configurazioni di rete, ad esempio per 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-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

```
C:\Program Files\Amazon\CloudHSM\> .\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 “Parametri”.

Parametri

--cluster-id ***<Cluster ID>***

Effettua una chiamata `DescribeClusters` per trovare tutti gli indirizzi IP a interfaccia di rete elastica (ENI) HSM del cluster associati all'ID del cluster. Il sistema aggiunge gli indirizzi IP ENI ai file di configurazione AWS CloudHSM.

> **Note**
>
> Se utilizzi il parametro `--cluster-id` da un'istanza EC2 all'interno di un VPC che non ha accesso alla rete Internet pubblica, devi creare un endpoint VPC di interfaccia da collegare a AWS CloudHSM. Per ulteriori informazioni sugli endpoint VPC, consulta la pagina ???.

Campo obbligatorio: sì

--endpoint ***<Endpoint>***

Specifica l'endpoint API AWS CloudHSM utilizzato per effettuare la chiamata `DescribeClusters`. È necessario impostare questa opzione insieme a `--cluster-id`.

Campo obbligatorio: no

-- hsm-ca-cert <HsmCA/Certificate/file>

Specifica il file del certificato CA HSM

Campo 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.

Campo obbligatorio: no

-- server-client-cert-file <Client/certificate/file>

Percorso al certificato client utilizzato per l'autenticazione reciproca client-server TLS.

Utilizza questa opzione solo se non desideri utilizzare la chiave predefinita e il certificato SSL/TLS inclusi in Client SDK 5. È necessario impostare questa opzione insieme a `--server-client-key-file`.

Campo obbligatorio: no

-- server-client-key-file <Client/key/file>

Percorso alla chiave del client utilizzato per l'autenticazione reciproca client-server TLS

Utilizza questa opzione solo se non desideri utilizzare la chiave predefinita e il certificato SSL/TLS inclusi in Client SDK 5. È necessario impostare questa opzione insieme a `--server-client-cert-file`.

Campo obbligatorio: no

configure-pkcs11 remove-cluster

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

Rimuovere 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

```
C:\Program Files\Amazon\CloudHSM\> .\configure-pkcs11.exe remove-cluster --cluster-
id cluster-1234567
```

Per ulteriori informazioni sul parametro `--cluster-id`, vedi the section called “Parametri”.

Parametro

--cluster-id ***<Cluster ID>***

L'ID del cluster da rimuovere dalla configurazione

Campo obbligatorio: sì

Comandi di ripetizione dei tentativi per PKCS #11

Client SDK 5.8.0 e le versioni successive dispongono di una strategia di ripetizione dei tentativi automatica integrata che tenterà di eseguire nuovamente le operazioni limitate da HSM dal lato client. Quando un HSM limita le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, gli SDK del client ripeteranno i tentativi per le operazioni limitate fino a 3 volte, con backoff esponenziale. Questa strategia di tentativi automatica può essere impostata su una delle due modalità: off e standard.

- off: l'SDK del client non eseguirà alcuna strategia di ripetizione dei tentativi per le operazioni limitate dall'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e le versioni successive. In questa modalità, gli SDK del client ripeteranno automaticamente i tentativi per le operazioni limitate con un backoff esponenziale.

Per ulteriori informazioni, vedi Limitazione (della larghezza di banda della rete) HSM.

Imposta i comandi di ripetizione dei tentativi sulla modalità off

Linux

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Linux

- È possibile utilizzare i seguenti comandi per impostare la configurazione di ripetizione dei tentativi sulla modalità off:

```
$ sudo /opt/cloudhsm/bin/configure-pkcs11 --default-retry-mode off
```

Windows

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Windows

- È possibile utilizzare i seguenti comandi per impostare la configurazione di ripetizione dei tentativi sulla modalità off:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-pkcs11.exe --default-retry-
mode off
```

OpenSSL Dynamic Engine

OpenSSL Dynamic Engine AWS CloudHSM consente di eseguire l'offload delle operazioni crittografiche sul cluster CloudHSM tramite l'API OpenSSL.

AWS CloudHSM fornisce un OpenSSL Dynamic Engine, sul quale trovi maggiori informazioni alla pagina Offload SSL/TLS su Linux. Per un esempio sull'uso di AWS CloudHSM con OpenSSL, consulta questo blog sulla sicurezza AWS. Per ulteriori informazioni sulle piattaforme supportate per gli SDK, consulta la pagina the section called “Piattaforme supportate”. Per la risoluzione dei problemi, vedereProblemi noti per OpenSSL Dynamic Engine.

Per informazioni sull'utilizzo di Client SDK 3, consulta la pagina Client SDK precedente (Client SDK 3).

Per ulteriori informazioni, consulta gli argomenti seguenti.

Argomenti

- Installazione di OpenSSL Dynamic Engine
- Tipi di chiave di OpenSSL Dynamic Engine
- Meccanismi di OpenSSL Dynamic Engine
- Migra il tuo OpenSSL Dynamic Engine da Client SDK 3 a Client SDK 5
- Configurazioni avanzate per OpenSSL

Installazione di OpenSSL Dynamic Engine

> **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 2

   Installare 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
   ```

   Installare 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
   ```

   Amazon Linux 2023

   Installa OpenSSL Dynamic Engine per Amazon Linux 2023 sull'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
```

CentOS 7 (7.8+)

Installare OpenSSL Dynamic Engine per CentOS 7 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
```

RHEL7 (7.8+)

Installare OpenSSL Dynamic Engine per RHEL 7 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
```

RHEL8 (8.3+)

Installare OpenSSL Dynamic Engine per RHEL 8 sull'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
```

RHEL9 (9.2+)

Installa OpenSSL Dynamic Engine per RHEL9 (9.2+) 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
```

Ubuntu 20.04 LTS

Installare OpenSSL Dynamic Engine per Ubuntu 20.04 LTS sull'architettura X86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/
cloudhsm-dyn_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-dyn_latest_u20.04_amd64.deb
```

Ubuntu 22.04 LTS

Installa OpenSSL Dynamic Engine per Ubuntu 22.04 LTS 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
```

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 Uso della CMU per gestire gli utenti.

```
$ 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`.

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 di OpenSSL Dynamic Engine

AWS CloudHSMOpenSSL Dynamic Engine supporta i seguenti tipi di chiavi.

Tipo di chiavi	Descrizione
EC	Firma/verifica ECDSA per i tipi di chiavi P-256, P-384 e secp256k1. Per generare chiavi EC interoperabili con il motore OpenSSL, consulta la pagina Generazione chiavi-file.
RSA	Generazione di chiavi RSA per chiavi a 2048, 3072 e 4096 bit.Firma/verifica RSA. Viene eseguito l'offload della verifica sul software OpenSSL.

Meccanismi di OpenSSL Dynamic Engine

Scopri come usare i meccanismi di AWS CloudHSM OpenSSL Dynamic Engine.

Funzioni di firma e verifica

AWS CloudHSMOpenSSL Dynamic Engine consente di utilizzare i seguenti meccanismi per le funzioni di firma e verifica.

Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non vi è alcun limite alla dimensione dei dati che possono essere sottoposti a hash.

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

Migra il tuo OpenSSL Dynamic Engine da Client SDK 3 a Client SDK 5

Usa questo argomento per migrare il tuo OpenSSL Dynamic Engine da Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi del Client SDK 5

NelAWS 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 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-dyn
   ```

   CentOS 7

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

   RHEL 7

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

   RHEL 8

   ```
   $ sudo yum remove cloudhsm-dyn
   ```

2. 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
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

3. Installa il Client SDK OpenSSL Dynamic Engine seguendo i passaggi riportati di seguito. Installazione di OpenSSL Dynamic Engine
4. 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
5. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Best practice per AWS CloudHSM

Configurazioni avanzate per OpenSSL

Il provider OpenSSL AWS CloudHSM include le seguenti configurazioni avanzate che non fanno parte delle configurazioni generali utilizzate dalla maggior parte dei clienti. Queste configurazioni offrono funzionalità aggiuntive.

- Comandi di ripetizione dei tentativi per OpenSSL

Comandi di ripetizione dei tentativi per OpenSSL

Client SDK 5.8.0 e le versioni successive dispongono di una strategia di ripetizione dei tentativi automatica integrata che tenterà di eseguire nuovamente le operazioni limitate da HSM dal lato client. Quando un HSM limita le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, gli SDK del client ripeteranno i tentativi per le operazioni limitate fino a 3 volte, con backoff esponenziale. Questa strategia di tentativi automatica può essere impostata su una delle due modalità: off e standard.

- off: l'SDK del client non eseguirà alcuna strategia di ripetizione dei tentativi per le operazioni limitate dall'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e le versioni successive. In questa modalità, gli SDK del client ripeteranno automaticamente i tentativi per le operazioni limitate con un backoff esponenziale.

Per ulteriori informazioni, vedi Limitazione (della larghezza di banda della rete) HSM.

Imposta i comandi di ripetizione dei tentativi sulla modalità off

Linux

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Linux

- 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
```

Windows

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Windows

- È possibile utilizzare i seguenti comandi per impostare i comandi di ripetizione dei tentativi sulla modalità off:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-dyn.exe --default-retry-mode
 off
```

Provider JCE

Il provider JCE AWS CloudHSM 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 JCE AWS CloudHSM viene talvolta definito provider JCE. Utilizza il provider JCE e il JDK per trasferire le operazioni crittografiche sull'HSM. Per la risoluzione dei problemi, vedereProblemi noti per l'SDK JCE.

Per informazioni sull'utilizzo di Client SDK 3, vedi Client SDK precedente (Client SDK 3).

Argomenti

Installare e utilizzare il provider JCE AWS CloudHSM per Client SDK 5

Il provider JCE è compatibile con OpenJDK 8, OpenJDK 11, OpenJDK 17 e OpenJDK 21. Puoi scaricarli entrambi dal sito Web di OpenJDK.

> **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

- Installa il provider JCE
- Fornire le credenziali al provider JCE
- Nozioni di base sulla gestione delle chiavi nel provider JCE

Installa il provider JCE

1. Utilizzare i seguenti comandi per scaricare e installare il provider JCE.

   Amazon Linux 2

   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
   ```

   Amazon Linux 2 su 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
   ```

   Amazon Linux 2023

   Amazon Linux 2023 su 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
```

CentOS 7 (7.8+)

CentOS 7 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
```

RHEL7 (7.8+)

RHEL 7 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
```

RHEL8 (8.3+)

RHEL 8 su 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
```

RHEL9 (9.2+)

RHEL9 (9.2+) su 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
```

Ubuntu 20.04 LTS

Ubuntu 20.04 LTS su architettura x86_64:

```
$ wget https://s3.amazonaws.com/cloudhsmv2-software/CloudHsmClient/Focal/cloudhsm-jce_latest_u20.04_amd64.deb
```

```
$ sudo apt install ./cloudhsm-jce_latest_u20.04_amd64.deb
```

Ubuntu 22.04 LTS

Ubuntu 22.04 LTS su 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
```

Windows Server 2016

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

```
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
```

Windows Server 2019

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

```
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 sul bootstrap, vedi 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`

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.

  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

Nozioni di base sulla gestione delle chiavi nel provider JCE

Le nozioni di base sulla gestione delle chiavi nel provider JCE implicano l'importazione, l'esportazione di chiavi, il caricamento di chiavi 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.

Tipi di chiavi supportati

La libreria software AWS CloudHSM per Java consente di generare i seguenti tipi di chiavi.

Tipo di chiavi	Descrizione
AES	Genera chiavi AES a 128, 192 e 256 bit.
Triple DES (3DES, DESede)	Genera una chiave DES tripla a 192 bit [1] Vedi nota a piè di pagina per una modifica imminente.
EC	Genera coppie di chiavi EC - curve NIST secp224r1 (P-224), secp256r1 (P-256), secp256k1 (Blockchain), secp384r1 (P-384), e secp521r1 (P-521).
SEGRETO_GENERICO	Genera segreti generici da 1 a 800 byte.
HMAC	Supporto hash per SHA1, SHA224, SHA256, SHA384, SHA512.
RSA	Genera chiavi RSA da 2048 bit a 4096 bit, con incrementi di 256 bit.

[1] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Meccanismi supportati

Per maggiori informazioni sulle interfacce e sulle classi di motore Java Cryptography Architecture (JCA) supportate da AWS CloudHSM, vedi i seguenti argomenti.

Argomenti

Funzioni di generazione chiavi e coppie di chiavi

La libreria software AWS CloudHSM per Java consente di utilizzare le seguenti operazioni per le funzioni di generazione di chiavi e coppie di chiavi.

- `RSA`
- `EC`
- `AES`
- `DESede (Triple DES)`[vedi nota1]
- `GenericSecret`

Funzioni di cifratura

La libreria software AWS CloudHSM per Java supporta le seguenti combinazioni di algoritmo, modalità e padding.

Algoritmo	Modalità	Padding	Note
AES	CBC	`AES/CBC/NoPadding` `AES/CBC/PKCS5Padding`	Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE` .

Algoritmo	Modalità	Padding	Note
			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`.

Algoritmo	Modalità	Padding	Note
AES	GCM	`AES/GCM/NoPadding`	Implementa `Cipher.WRAP_MODE`, `Cipher.UNWRAP_MODE`, `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_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/NoPadding`
		
		`AESWrap/ECB/PKCS5Padding`
		
		`AESWrap/ECB/ZeroPadding`
			Implementa `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`.

Algoritmo	Modalità	Padding	Note

DESede (Triple DES)	CBC	`DESede/CBC/PKCS5Padding`
		
		`DESede/CBC/NoPadding`
			Implementa `Cipher.ENCRYPT_MODE` e `Cipher.DECRYPT_MODE`. Vedi 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 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 software AWS CloudHSM 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 hash localmente nel software prima di essere inviati all'HSM per la firma/verifica. 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`

Funzioni set digest

La libreria software AWS CloudHSM per Java supporta i seguenti messaggi di digest. Con Client SDK 5, l'hashing dei dati viene eseguito localmente nel software. Ciò significa che non vi è alcun limite alla dimensione dei dati che possono essere sottoposti a hash dall'SDK.

- `SHA-1`
- `SHA-224`
- `SHA-256`
- `SHA-384`
- `SHA-512`

Funzioni di codice di autenticazione dei messaggi basato su hash (HMAC) (HMAC).

La libreria software AWS CloudHSM 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)

Funzioni di codice di autenticazione dei messaggi basato su crittografia (CMAC)

I CMAC (codici di autenticazione dei messaggi basati su crittografia) 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 software AWS CloudHSM per Java supporta i seguenti algoritmi HMAC.

- `AESCMAC`

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 CloudHSMdispone di due tipi di fabbriche chiave per JCE:

SecretKeyFactory: utilizzato per importare o derivare chiavi simmetriche. Utilizzando SecretKeyFactory, è possibile passare una chiave supportata o una chiave supportata KeySpec per importare o derivare chiavi simmetriche. AWS CloudHSM Di seguito sono riportate le specifiche supportate per: KeyFactory

- Sono supportate le seguenti KeySpecclassi SecretKeyFactory del `generateSecret` metodo For:
  - KeyAttributesMappuò essere usato per importare byte chiave con attributi aggiuntivi come chiave CloudHSM. Un esempio può essere trovato qui.
  - SecretKeySpecpuò essere usato per importare una specifica chiave simmetrica come chiave CloudHSM.
  - AesCmacKdfParameterSpecpuò essere usato per derivare chiavi simmetriche utilizzando un'altra chiave AES CloudHSM.

> **Note**
>
> SecretKeyFactoryil `translateKey` metodo accetta qualsiasi chiave che implementa l'interfaccia chiave.

KeyFactory: utilizzato per importare chiavi asimmetriche. Utilizzando KeyFactory, è possibile passare una chiave supportata o supportata KeySpec per importare una chiave asimmetrica. AWS CloudHSM Per ulteriori informazioni, fai riferimento a queste risorse:

- KeyFactoryIl `generatePublic` metodo di For, sono supportate le seguenti KeySpecclassi:
- KeyAttributesMap CloudHSM per RSA ed EC, tra cui: KeyTypes
  - KeyAttributesMap CloudhSM per RSA ed EC public. KeyTypes Un esempio può essere trovato qui
  - X509 sia EncodedKeySpec per RSA che per EC Public Key
  - RSA per chiave pubblica RSA PublicKeySpec
  - EC PublicKeySpec per EC Public Key
- KeyFactoryIl `generatePrivate` metodo di For, sono supportate KeySpecle seguenti classi:

- KeyAttributesMap CloudHSM per RSA ed EC, tra cui: KeyTypes
  - KeyAttributesMap CloudhSM per RSA ed EC public. KeyTypes Un esempio può essere trovato qui
  - PKCS8 EncodedKeySpec per EC e RSA Private Key
  - RSA per chiave privata RSA PrivateCrtKeySpec
  - EC PrivateKeySpec per chiave privata EC

KeyFactoryIl `translateKey` metodo di For, accetta qualsiasi chiave che implementa l'interfaccia chiave.

Annotazioni sui meccanismi

[1] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Attributi chiave Java supportati

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

- Comprendere degli attributi
- Attributi supportati
- Impostazione attributi per una chiave

Comprendere 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. Partendo dal provider JCE versione 3.1, AWS CloudHSM fornisce un'estensione proprietaria che 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, AWS 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	
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 WRAP è impostato su vero.
DERIVE				Consente di utilizzare una chiave per

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
				derivare altre chiavi.
ENCRYPT	TRUE	TRUE		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.
KEY_TYPE				Utilizzato per identificare il tipo di chiave (AES, DESede, generica segreta, EC o RSA).

Attributo	Valore predefinito			Note
	Chiave simmetrica	Chiave pubblica in una coppia di chiavi	Chiave privata in una coppia di chiavi	
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.
OBJECT_CLASS				Utilizzato per identificare la classe dell'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 chiarezza , gli utenti non possono accedere alle chiavi in AWS CloudHSM fino a quando non vengono autenticati, anche se questo attributo è impostato su FALSO.

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	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		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.

> **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 oggetto simile a Java Map che puoi 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:

Operazione	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	put(keyAttribute, value)

Operazione	Valore restituito	Metodo **KeyAttributesMap**
	non esiste alcuna mappatura per un attributo chiave	
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”.

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 la libreria software AWS CloudHSM per Java

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 JCEAWS CloudHSM per eseguire attività di base. Altri esempi sono disponibili su GitHub.

- Esegui l'accesso a un modulo HSM
- Gestisci chiavi
- Genera di chiavi simmetriche
- Genera chiavi asimmetriche
- Crittografa e decodifica con AES-GCM
- Crittografa e decodifica con AES-CTR
- Crittografa e decodifica con Desede-ECB vedi nota 1
- Firma e verifica con chiavi RSA
- Firma e verifica con Chiavi EC
- Usa attributi chiave supportati
- Usa store chiavi di CloudHSM

[1] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Provider JCE Javadocs AWS CloudHSM

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 Javadocs più recente per AWS CloudHSM, vedi la sezione Versione più recente nella pagina Download.

È possibile importare Javadocs in un ambiente di sviluppo integrato (IDE) o visualizzarli in un Web browser.

Utilizzo della classe KeyStore Java AWS CloudHSM

La classe AWS CloudHSM `KeyStore` fornisce un archivio di chiavi PKCS12 per scopi speciali. Questo archivio chiavi può archiviare i certificati insieme ai dati della chiave e correlarli ai dati della chiave memorizzati su AWS CloudHSM. La classe AWS CloudHSM `KeyStore` implementa l'interfaccia del provider di servizi (SPI) `KeyStore` della Java Cryptography Extension (JCE). Per ulteriori informazioni sull'utilizzo di `KeyStore`, vedi Classe KeyStore.

> **Note**
>
> Poiché i certificati sono informazioni pubbliche e, per massimizzare la capacità di archiviazione per le chiavi di crittografia, AWS CloudHSM non supporta l'archiviazione dei certificati sui moduli HSM.

Scegliere l'archivio chiavi appropriato

Il provider Java Cryptographic Extension (JCE) AWS CloudHSM offre un KeyStore AWS CloudHSM per scopi speciali. La classe AWS CloudHSM `KeyStore` supporta l'offload delle operazioni della chiave sull'HSM, l'archiviazione locale dei certificati e le operazioni basate sui certificati.

Carica il KeyStore CloudHSM per scopi speciali come segue:

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inizializzazione di KeyStore AWS CloudHSM

Accedi al KeyStore AWS CloudHSM nello stesso modo in cui accedi al provider JCE. È possibile utilizzare le variabili di ambiente o il file delle proprietà di sistema ed è necessario accedere prima di iniziare a utilizzare KeyStore CloudHSM. Per un esempio di accesso a un HSM utilizzando JCE, vedi Accedi a un HSM.

Se lo si desidera, è possibile specificare una password per crittografare il file PKCS12 locale che contiene i dati dell'archivio chiavi. Quando crei l'archivio chiavi AWS CloudHSM, imposta la password e forniscila quando usi i metodi di caricamento, impostazioni e ottenimento.

Istanziare un nuovo oggetto KeyStore CloudHSM come segue:

```
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);
```

Utilizzo di KeyStore AWS CloudHSM

AWS CloudHSM KeyStore è conforme alle specifiche JCE Classe KeyStore 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. Imposta 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 gli oggetti archiviati localmente nel file PKCS12 e gli oggetti residenti in 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 chiavi controlla gli oggetti archiviati localmente nel file PKCS12 e gli oggetti residenti nell'HSM.

- `DeleteEntry`

  Elimina una voce di certificato dal file PKCS12 locale. L'eliminazione dei dati chiave archiviati in un HSM non è supportata utilizzando il KeyStore AWS CloudHSM. 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 chiave 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`

  OttieniChiave viene passato all'HSM e restituisce un oggetto chiave corrispondente all'etichetta specificata. Poiché `getKey` interroga direttamente l'HSM, può essere utilizzato per qualsiasi chiave sull'HSM indipendentemente dal fatto che sia stata generata dall'archivio chiavi.

  ```
  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 sia il file PKCS12 che l'HSM per l'alias.

- `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 metodo KeyStore `SetKeyEntry` 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, utilizzare 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 file PKCS12 e lo protegge con la password indicata. Inoltre, persiste tutte le chiavi caricate (che sono impostate usando le chiamate `setKey`).

Esegui la migrazione del tuo provider JCE da Client SDK 3 a Client SDK 5

Utilizzate questo argomento per migrare il vostro provider JCE da Client SDK 3 a Client SDK 5. Per i vantaggi della migrazione, consulta. Vantaggi del Client SDK 5

NelAWS 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 e API personalizzate 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 di successo.

Per consultare le istruzioni di migrazione per tutti i provider, consultaMigrazione da 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`	In Client SDK 5, la classe Provider viene chiamata `CloudHsmP`	Un esempio di come inizializ zare l'`CloudHsmP`

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
	e ha il nome Provider. `Cavium`	`rovider` e ha il nome Provider. `CloudHSM`	`rovider` 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		Per un esempio su come utilizzare

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
	variabili di ambiente continueranno a funzionar e per l'accesso implicito durante la migrazione da Client SDK 3 a Client SDK 5.		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]Al`	In Client SDK 5, `KeyAttributesMap` viene utilizzato per specifica	Per un esempio su come utilizzare `KeyAttributesMap` per

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 chiavi. Per un frammento di codice, consulta la nota a piè di pagina. 1	re gli attributi di generazione delle chiavi. Per un frammento di codice, consulta la nota a piè di pagina. 2	generare una chiave simmetrica, consulta l'esempio nel repository di SymmetricKeys esempio Github di AWS CloudHSM.
Generazione di coppie di chiavi	In Client SDK 3, `Cavium[Ke y-type]Al 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	In Client SDK 5, `KeyPairAt tributesMap` viene utilizzato per specificare questi parametri. Per un frammento di codice, consulta la nota a piè di pagina. 4	Per un esempio su come `KeyAttrib utesMap` generare una chiave asimmetri ca, 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.	Poiché gli handle dei tasti non fanno parte delle specifich e JCE, Client SDK 5 utilizza etichette chiave (alias) per fare riferimento alle chiavi. Come prerequisito per l'utilizzo delle etichette chiave, si prevede che le chiavi nell'HSM abbiano	Per ulteriori informazi oni, consulta la nota 1 a piè di pagina.

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
		etichette univoche. A tale scopo è possibile utilizzare il Set di chiavi-attributi comando nella CLI di CloudHSM.	
Trova le chiavi	Quando si cerca una chiave per etichetta (alias) in scenari in cui sono presenti più chiavi con la stessa etichetta (alias) `Cavium` KeyStore, verrà restituita solo la prima chiave trovata.	Con `CloudHSM` KeyStore, questo stesso scenario comporterà la generazione di un'eccezione. Per risolvere questo problema, si consiglia di impostare etichette univoche per le chiavi utilizzando il Set di chiavi-attributi comando nella CLI di CloudHSM.	

Cosa è cambiato	Cosa c'era in Client SDK 3	Che cos'è in Client SDK 5	Esempio
Trova tutte le chiavi	La ricerca delle chiavi in Client SDK 3 richiede l'utilizzo `Util.findAllKeys()` e il successivo filtraggio in base a tipo/attributo di chiave specifici	Sebbene Client SDK 5 non supporti la ricerca di tutte le chiavi, Client SDK 5 semplifica la ricerca delle chiavi 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 nella tua applicazione.	Un esempio che utilizza la `KeyStoreWithAttributes` classe per trovare una chiave è disponibile nel repository di esempio di AWS CloudHSM Github e un frammento di codice è mostrato in. 2
Eliminazione della chiave	Client SDK 3 utilizza `Util.deleteKey()` per eliminare una chiave.	L'oggetto Key in Client SDK 5 implementa l'`Destroyable` interfaccia che consente di eliminare le chiavi utilizzando il metodo destroy di questa interfaccia.	Un codice di esempio che mostra la funzionalità di eliminazione delle chiavi è disponibile nel repository di esempio Github di CloudHSM. Un frammento di esempio per ogni SDK è mostrato in. 3

- [1] Set di chiavi-attributi Per informazioni sull'utilizzo della CLI di CloudhSM, vedere per impostare un'etichetta univoca per un determinato keyhandle.

Una volta impostate le etichette univoche, puoi usare il KeyStore per trovare quella chiave nell'applicazione:

```
keyStore.getKey("uniquelabel1234", null)
```

- [2] di seguito viene mostrato uno snippet:

```
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);
```

- [3] 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 crittografia/decrittografia/ avvolgimento di Cipher.

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-jce
```

CentOS 7

```
$ sudo yum remove cloudhsm-jce
```

RHEL 7

```
$ sudo yum remove cloudhsm-jce
```

RHEL 8

```
$ sudo yum remove cloudhsm-jce
```

2. 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
```

> **Note**
>
> Le configurazioni personalizzate devono essere nuovamente abilitate.

3. Installa il provider Client SDK JCE seguendo la procedura riportata di seguito. Installare e utilizzare il provider JCE AWS CloudHSM per Client SDK 5
4. 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
5. Nel tuo ambiente di sviluppo, prova la tua applicazione. Aggiorna il codice esistente per risolvere le modifiche sostanziali prima della migrazione finale.

Argomenti correlati

- Best practice per AWS CloudHSM

Configurazioni avanzate per JCE

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 cluster con il provider JCE

Questa configurazione consente a una singola istanza de, client di comunicare con più 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 classe `CloudHsmProvider` l'l'implementazione di AWS CloudHSM della classe del Provider di Java Security. Ogni istanza di questa classe rappresenta una connessione all'intero cluster AWS CloudHSM. 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());
}
```

Configurazione **CloudHsmProvider**

`CloudHsmProvider` può essere configurato in due modi:

1. Configura con file (configurazione predefinita)
2. Configura usando il codice

Configura con file (configurazione predefinita)

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 usando il codice

A partire dalla versione 5.8.0 di Client SDK, puoi anche configurare l'utilizzo del codice Java `CloudHsmProvider`. 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 per ottenere l'identificatore `provider.getName()`.

Connessione a più cluster

Come indicato sopra, ogni `CloudHsmProvider` rappresenta una connessione al cluster CloudHSM. 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'operazione AES/GCM/NoPadding:

```
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());
```

Comandi Nuovo tentativo per JCE

Client SDK 5.8.0 e le versioni successive dispongono di una strategia di ripetizione dei tentativi automatica integrata che tenterà di eseguire nuovamente le operazioni limitate da HSM dal lato client. Quando un HSM limita le operazioni perché è troppo occupato nell'esecuzione di operazioni precedenti e non può accettare altre richieste, gli SDK del client ripeteranno i tentativi per le operazioni limitate fino a 3 volte, con backoff esponenziale. Questa strategia di tentativi automatica può essere impostata su una delle due modalità: off e standard.

- off: l'SDK del client non eseguirà alcuna strategia di ripetizione dei tentativi per le operazioni limitate dall'HSM.
- standard: questa è la modalità predefinita per Client SDK 5.8.0 e le versioni successive. In questa modalità, gli SDK del client ripeteranno automaticamente i tentativi per le operazioni limitate con un backoff esponenziale.

Per ulteriori informazioni, vedi Limitazione (della larghezza di banda della rete) HSM.

Imposta i comandi di ripetizione dei tentativi sulla modalità off

Linux

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Linux

- È possibile utilizzare i seguenti comandi per impostare la configurazione di ripetizione dei tentativi sulla modalità off:

```
$ sudo /opt/cloudhsm/bin/configure-jce --default-retry-mode off
```

Windows

Come impostare i comandi di ripetizione dei tentativi su off per Client SDK 5 su Windows

- È possibile utilizzare i seguenti comandi per impostare la configurazione di ripetizione dei tentativi sulla modalità off:

```
C:\Program Files\Amazon\CloudHSM\bin\ .\configure-jce.exe --default-retry-mode
 off
```

Estrazione delle chiavi tramite JCE

La Java Cryptography Extension (JCE) utilizza un'architettura che consente di collegare diverse implementazioni di crittografia. AWS CloudHSM presenta uno di questi provider JCE che scarica le operazioni crittografiche sull'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 delle chiavi in chiaro.

> **Important**
>
> JCE scarica le operazioni su 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 JCE AWS CloudHSM consente l'estrazione di chiavi pubbliche per funzionare con provider JCE esterni. I seguenti metodi sono sempre consentiti:

Classe	Metodo	Formato (getEncoded)
Chiave pubblica EC	getEncoded()	X.509
	getW()	N/D
RSAPublicKey	getEncoded()	X.509
	getPublicExponent()	N/D
CloudHsmRsaPrivateCrtKey	getPublicExponent()	N/D

Per impostazione predefinita, il provider JCE AWS CloudHSM 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 vero.
   - Per impostazione predefinita, l'attributo `EXTRACTABLE` per le chiavi private e segrete è impostato su vero. 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.
   - `getEncodedgetPrivateExponent`, 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 Usare chiavi affidabili per controllare l'annullamento del wrapping.

Consentire al provider JCE AWS CloudHSM di estrarre chiavi private segrete da AWS CloudHSM

> **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

   ```
   C:\Program Files\Amazon\CloudHSM\> .\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 #11
	getS()	N/D
RSAPrivateCrtKey	getEncoded()	X.509
	getPrivateExponent()	N/D
	getPrimeP()	N/D
	getPrimeQ()	N/D
	getPrimeExponentP()	N/D
	getPrimeExponentQ()	N/D
	getCrtCoefficient()	N/D

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

```
C:\Program Files\Amazon\CloudHSM\> .\configure-jce.exe --disable-clear-key-
extraction-in-software
```

Cryptography API: Next Generation (CNG) e provider di archiviazione delle chiavi (KSP) per Microsoft Windows

Il software client AWS CloudHSM per Windows Server include i provider CNG e KSP richiesti. Attualmente, solo Client SDK 3 supporta i provider CNG e KSP.

I provider di archiviazione delle chiavi (KSP) 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, vedi 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. A un livello superiore, l'implementazione di CNG AWS CloudHSM offre le seguenti funzionalità.

- Primitive crittografiche - consentono di eseguire operazioni fondamentali di crittografia.
- Importazione ed esportazione di chiavi - consente di importare ed esportare chiavi asimmetriche.
- API di protezione dei dati (CNG DPAPI) - consente di crittografare e decodificare i dati con facilità.
- Archiviazione e recupero delle chiavi - consente di archiviare e isolare in modo sicuro la chiave privata di una coppia di chiavi asimmetriche.

Argomenti

Installazione dei provider CNG e KSP per Windows

I provider CNG e KSP vengono installati quando si installa il client di Windows AWS CloudHSM. È possibile installare il client seguendo i passaggi descritti in Installa il client (Windows).

Configurare ed eseguire il client Windows AWS CloudHSM

Per avviare il client Windows CloudHSM, è necessario soddisfare per prima cosa i Prerequisiti. Quindi, aggiornare i file di configurazione che i provider utilizzano e avviare il client completando la procedura seguente. È necessario eseguire questi passaggi la prima volta che si utilizza KSP e i provider CNG e dopo aver aggiunto o rimosso moduli HSM nel cluster. In questo modo, AWS CloudHSM è in grado di sincronizzare i dati e mantenere la coerenza su tutti i moduli HSM nel cluster.

Fase 1: interruzione del client AWS CloudHSM

Prima di aggiornare i file di configurazione utilizzati dal provider, arresta il client AWS CloudHSM. 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 Windows client 1.1.1 e versioni precedenti:

  Usa Ctrl + C nella finestra di comando in cui hai avviato il client AWS CloudHSM.

Fase 2: aggiornamento dei file di configurazione AWS CloudHSM

Questa fase usa il parametro `-a` di Strumento di configurazione per aggiungere l'indirizzo IP dell'interfaccia di rete elastica (ENI) di uno degi moduli HSM nel cluster per il file di configurazione.

```
C:\Program Files\Amazon\CloudHSM configure.exe -a <HSM ENI IP>
```

Per ottenere l'indirizzo IP ENI di un HSM nel cluster, accedi alla console AWS CloudHSM, scegli cluster e seleziona il cluster desiderato. È inoltre possibile utilizzare l'operazione DescribeClusters, il comando describe-clusters o il cmdlet PowerShell Get-HSM2Cluster. Digitare solo un indirizzo IP ENI. Non importa l'indirizzo IP ENI utilizzato.

Fase 3: avvio del client AWS CloudHSM

Quindi, avvia o riavvia il client AWS CloudHSM. Quando viene avviato, il client AWS CloudHSM utilizza l'indirizzo IP ENI nel proprio file di configurazione per eseguire query sul cluster. Quindi aggiunge gli indirizzi IP ENI di tutti i moduli HSM del cluster sul file di informazione del cluster.

- Per client Windows dalla versione 1.1.2 in poi:

```
C:\Program Files\Amazon\CloudHSM>net.exe start AWSCloudHSMClient
```

- Per Windows client 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.

```
C:\Program Files\Amazon\CloudHSM>ksp_config.exe -enum
```

```
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.

```
C:\Program Files\Amazon\CloudHSM>cng_config.exe -register
```

Se il provider CNG manca, esegui il comando seguente.

```
C:\Program Files\Amazon\CloudHSM>ksp_config.exe -register
```

Prerequisiti Windows AWS CloudHSM

Prima di avviare il client Windows AWS CloudHSM e utilizzare i provider KSP e CNG, devi 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 2.0.4 del client AWS CloudHSM 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 i CU esistenti, utilizza ElencaUtenti.

Ad esempio:

```
setx /m n3fips_password test_user:password123
```

Associare una chiave AWS CloudHSM a un certificato

Prima di poter utilizzare le chiavi AWS CloudHSM con strumenti di terze parti, ad esempio SignTool di Microsoft, devi importare i metadati della chiave nell'archivio certificati locale e associare i metadati 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. Ad 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. Utilizza un'utility, ad esempio certutil, per visualizzare le informazioni necessarie sul certificato. Il seguente esempio di output da certutil mostra il nome del container 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
```

Passaggio 3: associare la chiave privata AWS CloudHSM al certificato

Per associare la chiave al certificato, assicurati innanzitutto di avviare il client daemon AWS CloudHSM. 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 client daemon AWS CloudHSM sia ancora in esecuzione. Quindi, utilizzare il comando di certutil, -repairstore, per aggiornare il numero di serie del certificato. L'esempio seguente mostra il comando e l'output. Vedi la documentazione di Microsoft per informazioni sul comando -repairstore.

```
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 ProviderPrivate key is NOT exportableEncryption test
 passedCertUtil: -repairstore command completed successfully.
```

Dopo aver aggiornato il numero di serie del certificato è possibile utilizzare questo certificato e la chiave privata AWS CloudHSM corrispondente con qualsiasi strumento di firma di terze parti su Windows.

Esempio di codice per il provider CNG

⚠ ** Codice di solo esempio: Non per uso produttivo **
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 Windows AWS CloudHSM.

```
// 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;
}
```

Client SDK precedente (Client SDK 3)

L'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 vantaggi e i vantaggi che offre, vedi Vantaggi del 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 Client SDK 3 a Client SDK 5

La documentazione di Client SDK 3 è elencata in questo argomento.

Per scaricarla, vedi Download.

Controlla la versione dell'SDK del tuo client

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
```

Confronto dei componenti del 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 ha lo stesso valore di 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

Componente	Client SDK 5	Client SDK 3
OpenSSL Dynamic Engine	Sì	Sì
Provider KSP e CNG		Sì
Programma di gestione CloudHSM (CMU) [1]	Sì	Sì
Programma 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.

Argomenti

Piattaforme supportate da Client SDK 3

Client SDK 3 richiede un daemon del client e offre strumenti a riga di comando, tra cui CloudHSM Management Utility (CMU), key management utility (KMU) e lo strumento di configurazione.

Il supporto di base è diverso per ciascuna versione dell'SDK del client di AWS CloudHSM. Generalmente le piattaforme supportate per i componenti di un SDK corrispondono al supporto di base, ma non è sempre così. Per determinare le piattaforme supportate per un determinato componente, assicurati innanzitutto che la piattaforma desiderata sia presente nella sezione di base dell'SDK, quindi verifica le esclusioni o qualsiasi altra informazione pertinente nella sezione del componente.

Le piattaforme supportate cambiano nel tempo. Le versioni precedenti dell'SDK del client di CloudHSM potrebbero non supportare tutti i sistemi operativi elencati qui. Verifica se il sistema

operativo supporta le versioni precedenti dell'SDK del client di CloudHSM consultando le note di rilascio. Per ulteriori informazioni, consulta Download per Client SDK dell'AWS CloudHSM.

AWS CloudHSM supporta solo sistemi operativi a 64 bit.

Indice

Supporto di 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

- Microsoft Windows Server 2012
- Microsoft Windows Server 2012 R2
- Microsoft Windows Server 2016
- Microsoft Windows Server 2019

Componenti supportati

Libreria PKCS #11

La libreria PKCS #11 è un componente esclusivamente per Linux che corrisponde al supporto di base di Linux. Per ulteriori informazioni, consulta the section called “Supporto di Linux”.

Provider JCE

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 di Linux”.

- Richiede OpenJDK 1.8

OpenSSL Dynamic Engine

OpenSSL Dynamic Engine è un componente esclusivamente 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.

Provider KSP e CNG

I provider CNG e KSP sono componenti esclusivamente per Windows che corrispondono al supporto di base di Windows. Per ulteriori informazioni, consulta Supporto Windows.

Aggiornamento del Client SDK 3 su Linux

Con il Client SDK 3.1 dell'AWS CloudHSM e versioni successive, la versione del client daemon e tutti i componenti che installi devono corrispondere per poter effettuare 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

Scarica la versione più recente del client daemon dell'AWS CloudHSM e scegli i 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 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
```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

Supporto per OpenSSL Dynamic Engine non ancora disponibile.

Libreria PKCS #11 per Client SDK 3

PKCS #11 è uno standard per l'esecuzione di operazioni di crittografia su moduli di sicurezza hardware (HSM).

Per informazioni sul processo di bootstrap, consulta la pagina Connessione al cluster.

Argomenti

Installazione di Client SDK 3 per la libreria PKCS #11

Prerequisiti per Client SDK 3

La libreria PKCS #11 richiede il client AWS CloudHSM.

Se non hai installato e configurato il client AWS CloudHSM, procedi nel modo descritto in Installazione del client (Linux). Dopo aver installato e configurato il client, eseguire questo comando per avviarlo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl cloudhsm-client start
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

CentOS 8

```
$ sudo systemctl cloudhsm-client start
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 8

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 20.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

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
  ```

Autenticazione nella libreria PKCS #11 (Client SDK 3)

Quando si utilizza la libreria PKCS #11, l'applicazione viene eseguita come crypto user (CU) specifico negli HSM. L'applicazione è in grado di visualizzare e gestire solo le chiavi di proprietà e condivise dall'utente di crittografia. È possibile usare un CU esistente negli HSM o crearne uno nuovo. 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 formato seguente:

```
<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 (Client SDK 3)

La libreria PKCS #11 supporta i seguenti tipi di chiave.

Tipo di chiavi	Descrizione
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 (Triple DES)	Genera chiavi DES3 a 192 bit. Vedi la nota 1 a seguire per una modifica imminente.
GENERIC_SECRET	Genera segreti generici da 1 a 64 byte.

- [1] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Meccanismi supportati (Client SDK 3)

La libreria PKCS #11 supporta i seguenti algoritmi:

- Crittografia e decrittografia: AES-CBC, AES-CTR, AES-ECB, AES-GCM, DES3-CBC, DES3-ECB, RSA-OAEP e RSA-PKCS
- Firma e verifica: RSA, HMAC e ECDSA; con e senza hashing
- Hash/digest: SHA1, SHA224, SHA256, SHA384 e SHA512
- Wrapping della chiave: AES Key Wrap,[4] AES-GCM, RSA-AES e RSA-OAEP
- Derivazione chiave: ECDH,[5] SP800-108 CTR KDF

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 ✓ indica che AWS CloudHSM supporta il meccanismo per la funzione. Non supportiamo tutte le funzioni elencate nella specifica PKCS #11. Una ✖ 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)	Avvolgere e UnWrap
CKM_RSA_PKCS_KEY_PAIR_GEN	✓						
CKM_RSA_X9_31_KEY_PAIR_GEN	✓[2]						
CKM_RSA_X_509		✓			✓		
CKM_RSA_PKCS see note 8		✓[1]	✖		✓[1]		✓[1]
CKM_RSA_PKCS_OAEP					✓[1]		✓[6]

Meccanismo	Funzioni						
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]						
CKM_SHA224_RSA_PKCS_PSS	✓[3.2]						
CKM_SHA256_RSA_PKCS_PSS	✓[3.2]						

Meccanismo	Funzioni						
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]					
CKM_ECDSA_SHA512		✓[3.2]					
CKM_ECDH1_DERIVE						✓[5]	
CKM_SP800_108_COUNTER_KDF						✓	

Meccanismo	Funzioni						
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	✓						
CKM_DES3_CBC see note 8					✓[3.3]		✖
CKM_DES3_CBC_PAD see note 8					✓		✖
CKM_DES3_ECB see note 8					✓		✖

Meccanismo	Funzioni						
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]			
CKM_SHA256_HMAC		✓[3.3]					
CKM_SHA384				✓[3.1]			
CKM_SHA384_HMAC		✓[3.3]					
CKM_SHA512				✓[3.1]			
CKM_SHA512_HMAC		✓[3.3]					

Meccanismo	Funzioni						
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]
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 utilizza un approccio diverso per l'hashing 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 utilizza un approccio diverso per l'hashing 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
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
`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 per la libreria PKCS #11. `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/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

  Per ulteriori informazioni sul wrapping delle chiavi AES, consulta la pagina sul wrapping delle chiavi AES.
- [8] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Operazioni API supportate (Client SDK 3)

La libreria PKCS #11 supporta le seguenti operazioni API PKCS #11.

- 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 supportati (Client SDK 3)

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 utilizzi la libreria PKCS #11, 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 l'esempio su come generare chiavi attraverso la libreria PKCS #11.

Interpretazione della tabella degli attributi relativi alla libreria PKCS #11

La tabella della libreria PKCS #11 contiene un elenco di attributi che differiscono per i tipi di chiave. Indica se un determinato attributo è supportato da un particolare tipo di chiave quando si usa una determinata funzione di crittografia 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`	✓	✓	✓	✓	

Attributo	Tipo di chiavi				Valore predefinito
CKA_LOCAL	R	R	R	R	True
CKA_TOKEN	✓	✓	✓	✓	False
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	True
CKA_ENCRYPT	✖	✓	✖	✓	False
CKA_DECRYPT	✓	✖	✓	✖	False
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

Attributo	Tipo di chiavi				Valore predefinito
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	
CKA_TRUSTED	✖	✓	✖	✓	False
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	False
CKA_UNWRAP	✓	✖	✓	✖	False
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]	

Attributo	Tipo di chiavi				Valore predefinito
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	✖	✖	✖	✖	
CKA_VALUE_LEN	✖	✖	✖	✖	

Attributo	Tipo di chiavi				Valore predefinito
CKA_CHECK_VALUE		R	R	R	R	

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
CKA_ENCRYPT	✓	✓	✖	False
CKA_DECRYPT	✓	✓	✖	False
CKA_DERIVE	✓	✓	✓	False

Attributo	Tipo di chiavi			Valore predefinito
CKA_MODIFIABLE	✓[1]	✓[1]	✓[1]	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
CKA_WRAP_WITH_TRUSTED	✓	✓	✓	False
CKA_UNWRAP	✓	✓	✖	False
CKA_UNWRAP_TEMPLATE	✓	✓	✖	

Attributo	Tipo di chiavi			Valore predefinito
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	✖	✖	✖	
CKA_COEFFICIENT	✖	✖	✖	
CKA_EXPONENT_1	✖	✖	✖	
CKA_EXPONENT_2	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
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

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]	

Attributo	Tipo di chiavi							Valore predefinito
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
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

Attributo	Tipo di chiavi							Valore predefinito
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✓[4]	✖	✖	✖	
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	R	✖	R	✖	R	R	R	
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓	True

Attributo	Tipo di chiavi							Valore predefinito
CKA_NEVER_EXTRACTABLE	R	✖	R	✖	R	R	R	
CKA_MODULUS	✖	✖	✓[2]	✓[2]	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	✖	✖	
CKA_PRIME_1	✖	✖	✓	✖	✖	✖	✖	
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]	✖	✖	✖	✖	✖	

Attributo	Tipo di chiavi							Valore predefinito
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

Attributo	Tipo di chiavi					Valore predefinito
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
CKA_DESTROYABLE	✓	✓	✓	✓	✓	True
CKA_SIGN	✓	✓	✓	✓	✓	False
CKA_SIGN_RECOVER	✖	✓[3]	✖	✖	✖	False
CKA_VERIFY	✖	✖	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	✖	✖	
CKA_WRAP	✖	✖	✓	✓	✖	False

Attributo	Tipo di chiavi					Valore predefinito
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	
CKA_MODULUS	✖	✖	✖	✖	✖	
CKA_MODULUS_BITS	✖	✖	✖	✖	✖	
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		✖	✖	✖	✖	✖	
CKA_EC_PARAMS		✖	✖	✖	✖	✖	
CKA_EC_POINT		✖	✖	✖	✖	✖	
CKA_VALUE		✖	✖	✖	✖	✖	
CKA_VALUE_LEN		✖	✖	✖	✖	✖	
CKA_CHECK_VALUE		R	R	R	R	R	

DeriveKey

Attributo	Tipo di chiavi				Valore predefinito
		AES	DES3	Segreto generico	

Attributo	Tipo di chiavi			Valore predefinito
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
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	✖	✖	✖	

Attributo	Tipo di chiavi			Valore predefinito
CKA_VERIFY	✓	✓	✓	False
CKA_VERIFY_RECOVER	✖	✖	✖	
CKA_WRAP	✓	✓	✖	False
CKA_UNWRAP	✓	✓	✖	False
CKA_SENSITIVE	✓	✓	✓	True
CKA_EXTRACTABLE	✓	✓	✓	True
CKA_NEVER_EXTRACTABLE	R	R	R	
CKA_ALWAYS_SENSITIVE	R	R	R	
CKA_MODULUS	✖	✖	✖	
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	✖	✖	✖	
CKA_PUBLIC_EXPONENT	✖	✖	✖	
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	✓	✓	✓	✓	✓	✓	✓
CKA_TOKEN	✓	✓	✓	✓	✓	✓	✓
CKA_PRIVATE	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]	✓[1]
CKA_ENCRYPT	✖	✖	✖	✓	✓	✓	✖
CKA_DECRYPT	✖	✖	✓	✖	✓	✓	✖
CKA_DERIVE	✓	✓	✓	✓	✓	✓	✓
CKA_MODIFIABLE	✓	✓	✓	✓	✓	✓	✓

Attributo	Tipo di chiavi						
CKA_DESTROYABLE	✓	✓	✓	✓	✓	✓	✓
CKA_SIGN	✓	✖	✓	✖	✓	✓	✓
CKA_SIGN_RECOVER	✖	✖	✓	✖	✖	✖	✖
CKA_VERIFY	✖	✓	✖	✓	✓	✓	✓
CKA_VERIFY_RECOVER	✖	✖	✖	✓	✖	✖	✖
CKA_WRAP	✖	✖	✖	✓	✓	✓	✖
CKA_WRAP_TEMPLATE	✖	✓	✖	✓	✓	✓	✖
CKA_TRUSTED	✖	✓	✖	✓	✓	✓	✓
CKA_WRAP_WITH_TRUSTED	✓	✖	✓	✖	✓	✓	✓
CKA_UNWRAP	✖	✖	✓	✖	✓	✓	✖
CKA_UNWRAP_TEMPLATE	✓	✖	✓	✖	✓	✓	✖
CKA_SENSITIVE	✓	✖	✓	✖	✓	✓	✓

Attributo	Tipo di chiavi						
CKA_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_NEVER_EXTRACTABLE	✓	✖	✓	✖	✓	✓	✓
CKA_ALWAYS_SENSITIVE	R	R	R	R	R	R	R
CKA_MODULUS	✖	✖	✓	✓	✖	✖	✖
CKA_MODULUS_BITS	✖	✖	✖	✓	✖	✖	✖
CKA_PRIME_1	✖	✖	S	✖	✖	✖	✖
CKA_PRIME_2	✖	✖	S	✖	✖	✖	✖
CKA_COEFFICIENT	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_1	✖	✖	S	✖	✖	✖	✖
CKA_EXPONENT_2	✖	✖	S	✖	✖	✖	✖
CKA_PRIVATE_EXPONENT	✖	✖	S	✖	✖	✖	✖

Attributo	Tipo di chiavi						
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

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

La specifica nel modello di un attributo non supportato da una chiave specifica 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.

Codici di esempio per la libreria PKCS #11 (Client SDK 3)

Gli esempi di codice riportati GitHub mostrano come eseguire attività di base utilizzando la libreria PKCS #11.

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

Installazione di Client SDK 3 per OpenSSL Dynamic Engine

Client SDK 3 richiede un daemon del 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.

Argomenti

Prerequisiti per OpenSSL Dynamic Engine con Client SDK 3

Per ulteriori informazioni sulle piattaforme supportate, consulta la pagina Piattaforme supportate da Client SDK 3.

Per poter utilizzare il motore dinamico AWS CloudHSM per OpenSSL, è necessario disporre del client AWS CloudHSM.

Il client è un demone che stabilisce una comunicazione end-to-end crittografata con gli HSM del cluster e il motore OpenSSL comunica localmente con il client. Per installare e configurare il client AWS CloudHSM, consulta la pagina Installazione del client (Linux). Poi utilizza il seguente comando per avviarlo.

Amazon Linux

```
$ sudo start cloudhsm-client
```

Amazon Linux 2

```
$ sudo systemctl cloudhsm-client start
```

CentOS 6

```
$ sudo systemctl start cloudhsm-client
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 6

```
$ sudo systemctl start cloudhsm-client
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Installazione di OpenSSL Dynamic Engine per Client SDK 3

La procedura seguente descrive come installare e configurare il motore dinamico AWS CloudHSM per OpenSSL. Per ulteriori informazioni sull'upgrade, consulta la pagina Aggiornamento del 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>
```

Utilizzo di OpenSSL Dynamic Engine per Client SDK 3

Per usare il motore dinamico di AWS CloudHSM per OpenSSL da un'applicazione integrata OpenSSL, assicurarsi 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 dinamico di AWS CloudHSM per OpenSSL dalla riga di comando di OpenSSL, usare l'opzione `-engine` per specificare il motore dinamico OpenSSL denominato `cloudhsm`. Ad esempio:

```
$ openssl s_server -cert server.crt -key server.key -engine cloudhsm
```

Client SDK 3 per provider JCE

Il provider JCE AWS CloudHSM 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 JCE AWS CloudHSM viene talvolta definito provider JCE. Utilizza il provider JCE e il JDK per trasferire le operazioni crittografiche sull'HSM.

Argomenti

- Installare e utilizzare il provider JCE AWS CloudHSM per Client SDK 3
- Meccanismi supportati per Client SDK 3
- Attributi chiave Java per Client SDK 3 supportati
- Esempi di codice per la libreria software AWS CloudHSM per Java per Client SDK 3
- Utilizzo della classe Java KeyStore AWS CloudHSM per Client SDK 3

Installare e utilizzare il provider JCE AWS CloudHSM per Client SDK 3

È necessario disporre del client AWS CloudHSM prima di poter utilizzare il provider JCE.

Il client è un daemon che stabilisce una comunicazione end-to-end crittografata con i moduli HSM nel cluster. Il provider JCE comunica localmente con il client. Se non hai installato e configurato il pacchetto client AWS CloudHSM, fallo seguendo i passaggi descritti in 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 cloudhsm-client start
```

CentOS 7

```
$ sudo systemctl cloudhsm-client start
```

CentOS 8

```
$ sudo systemctl cloudhsm-client start
```

RHEL 7

```
$ sudo systemctl cloudhsm-client start
```

RHEL 8

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 16.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 18.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Ubuntu 20.04 LTS

```
$ sudo systemctl cloudhsm-client start
```

Argomenti

Installazione del 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 Aggiornamento del 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`

Convalida dell'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 *<nome utenteHSM>* e *<password>* con le credenziali di un crypto user (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)
```

Fornire le credenziali al provider JCE

I moduli HSM necessitano di autenticare l'applicazione Java, prima che l'applicazione sia in grado di utilizzarli. Ogni applicazione può utilizzare una sessione. 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 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 a quello 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 login implicito complica la gestione degli errori quando le credenziali non sono valide o si sono verificati problemi con i moduli HSM durante la sessione di autenticazione.

Nozioni di base sulla gestione delle chiavi nel provider JCE

Le nozioni di base sulla gestione delle chiavi nel provider JCE implicano l'importazione, l'esportazione di chiavi, il caricamento di chiavi 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 maggiori informazioni sulle interfacce e sulle classi di motore Java Cryptography Architecture (JCA) supportate da AWS CloudHSM, vedi i seguenti argomenti.

Argomenti

- Chiavi supportate
- Cifrature supportate
- Digest supportati
- Algoritmi codice di autenticazione dei messaggi basato su hash (HMAC) supportati
- Meccanismi di firma/verifica supportati
- Annotazioni sui meccanismi

Chiavi supportate

La libreria software AWS CloudHSM 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 software AWS CloudHSM per Java supporta le seguenti combinazioni di algoritmo, 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`	Implementa `Cipher.EN`

Algoritmo	Modalità	Padding	Note
		`AES/ECB/PKCS5Padding`	`CRYPT_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` 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.

Algoritmo	Modalità	Padding	Note

AESWrap	ECB	`AESWrap/ECB/ZeroPadding`
		
		`AESWrap/ECB/NoPadding`
		
		`AESWrap/ECB/PKCS5Padding`
			Implementa `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE`, Usare AES di trasformazione.

DESede (Triple 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 chiavi DESede hanno una dimensione di 192 bit.
			
			Vedi la nota 1 di seguito per una modifica imminente.

Algoritmo	Modalità	Padding	Note

DESede (Triple 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 chiavi DESede hanno una dimensione di 192 bit.
			
			Vedi 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 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 software AWS CloudHSM per Java supporta i seguenti messaggi di digest.

- `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 software AWS CloudHSM per Java supporta i seguenti algoritmi HMAC.

- `HmacSHA1`
- `HmacSHA224`
- `HmacSHA256`
- `HmacSHA384`
- `HmacSHA512`

Meccanismi di firma/verifica supportati

La libreria software AWS CloudHSM 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] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Attributi chiave Java per Client SDK 3 supportati

In questo argomento viene descritto come utilizzare un'estensione proprietaria per la libreria Java versione 3.1 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
- 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

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 è replicata in tutti i moduli HSM nel cluster e inclusa 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:

Operazione	Valore restituito	Metodo **`CloudHSMKeyAttributesMap`**
Ottenere il valore di un attributo chiave per una chiave esistente	Oggetto (contenente il valore) o nulla	get(keyAttribute)
Compilare 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 il relativo argomento.

Esempio di modello di generatore

Gli sviluppatori troveranno generalmente più comodo impostare gli attributi chiave tramite il modello di generatore. Ad 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, vedi l'archivio Javadoc e l'esempio in 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 software AWS CloudHSM per Java per 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 JCEAWS CloudHSM per eseguire attività di base. Altri esempi sono disponibili su GitHub.

- Esegui l'accesso a un modulo HSM
- Gestisci chiavi
- Genera 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] Non consentito dopo il 2023 per la conformità al FIPS secondo le linee guida del NIST. Per informazioni dettagliate, vedi Conformità FIPS 140: meccanismo di deprecazione 2024.

Utilizzo della classe Java KeyStore AWS CloudHSM per Client SDK 3

La classe AWS CloudHSM `KeyStore` fornisce un archivio di chiavi PKCS12 per uso speciale che consente l'accesso alle chiavi AWS CloudHSM tramite applicazioni come keytool e jarsigner. Questo archivio chiavi può archiviare i certificati insieme ai dati chiave e correlarli ai dati chiave memorizzati su AWS CloudHSM.

> **Note**
>
> Poiché i certificati sono informazioni pubbliche e, per massimizzare la capacità di archiviazione per le chiavi di crittografia, AWS CloudHSM non supporta l'archiviazione dei certificati sui moduli HSM.

La classe AWS CloudHSM `KeyStore` implementa l'Interfaccia del Provider del Servizio (SPI) `KeyStore` della Java Cryptography Extension (JCE). Per ulteriori informazioni sull'utilizzo di `KeyStore`, vedi Classe KeyStore.

Scegliere l'archivio chiavi appropriato

Il provider Java Cryptographic Extension (JCE) AWS CloudHSM viene fornito con un pass-through predefinito e di sola lettura che passa tutte le transazioni all'HSM. Questo archivio di chiavi predefinito è diverso da quello AWS CloudHSM per uso speciale. Nella maggior parte delle situazioni, è possibile ottenere prestazioni di runtime e velocità effettiva migliori utilizzando l'impostazione predefinita. È consigliabile utilizzare l'archivio di chiavi AWS CloudHSM solo per le applicazioni in cui è necessario il supporto per certificati e operazioni basate su certificati, oltre a scaricare le operazioni chiave nell'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");
```

Carica il KeyStore CloudHSM per scopi speciali come segue:

```
KeyStore ks = KeyStore.getInstance("CloudHSM")
```

Inizializzazione di KeyStore AWS CloudHSM

Accedi all'archivio chiavi AWS CloudHSM nello stesso modo in cui accedi al provider JCE per . È possibile utilizzare le variabili di ambiente o il file delle proprietà di sistema ed è necessario accedere prima di iniziare a utilizzare l'archivio chiavi di CloudHSM. Per un esempio di accesso a un HSM utilizzando JCE, vedi Accedi a un HSM.

Se lo desideri, è possibile specificare una password per crittografare il file PKCS12 locale che contiene i dati dell'archivio chiavi. Quando crei l'archivio chiavi AWS CloudHSM, imposti la password e la fornisci quando usi i metodi di caricamento, impostazione e ottenimento.

Istanziare un nuovo oggetto KeyStore CloudHSM come segue:

```
ks.load(null, null);
```

Scrivere i dati del keystore 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);
```

Usare KeyStore AWS CloudHSM

Un oggetto KeyStore CloudHSM viene generalmente utilizzato attraverso un'applicazione di terze parti come jarsigner o keytool. Puoi anche accedere direttamente all'oggetto con il codice.

KeyStore AWS CloudHSM è conforme alle specifiche JCE Classe KeyStore 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 gli oggetti archiviati localmente nel file PKCS12 e gli oggetti residenti in 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 chiavi controlla gli oggetti archiviati localmente nel file PKCS12 e gli oggetti residenti nell'HSM.

- `DeleteEntry`

Elimina una voce di certificato dal file PKCS12 locale. L'eliminazione dei dati chiave archiviati in un HSM non è supportata utilizzando il KeyStore AWS CloudHSM. 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 `getKey` interroga direttamente l'HSM, può essere utilizzato per qualsiasi chiave sull'HSM indipendentemente dal fatto che sia stata generata dall'archivio chiavi.

```
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 file PKCS12 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 metodo KeyStore `SetKeyEntry` 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 file PKCS12 e lo protegge con la password indicata. Inoltre, persiste tutte le chiavi caricate (che sono impostate usando le chiamate `setKey`).

Integrazione di applicazioni di terze parti con AWS CloudHSM

Alcuni dei casi d'uso per AWS CloudHSM prevedono l'integrazione di applicazioni software di terze parti con gli HSM nel tuo cluster AWS CloudHSM. Integrando software di terze parti con AWS CloudHSM, puoi raggiungere una serie di obiettivi relativi alla sicurezza. Gli argomenti seguenti illustrano come raggiungerne alcuni.

Argomenti

Migliorare la sicurezza del server Web con l'offload SSL/TLS 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 pubblica SSL/TLS per stabilire una sessione HTTPS con ciascun client. Questo processo implica l'uso di molte risorse di calcolo dei server Web, ma è possibile eseguire l'offload di una parte dell'attività nel cluster AWS CloudHSM, un'operazione definita come accelerazione SSL. L'offload consente di ridurre l'onere computazionale sui server Web e di fornire sicurezza aggiuntiva tramite l'archiviazione delle chiavi private dei server negli HSM.

I seguenti argomenti forniscono una panoramica sul funzionamento dell'offload SSL/TLS con AWS CloudHSM e tutorial per la configurazione dell'offload SSL/TLS con AWS CloudHSM per le seguenti piattaforme.

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

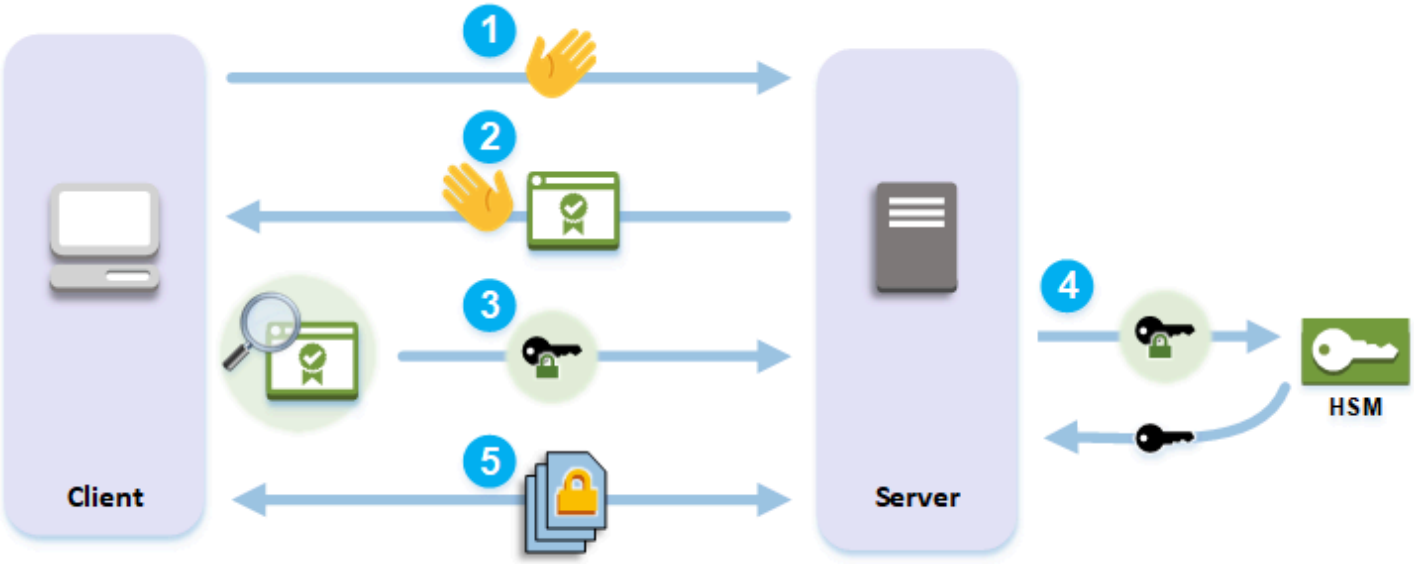
Argomenti

- Funzionamento dell'offload SSL/TLS con AWS CloudHSM
- Offload SSL/TLS su Linux
- Utilizzo di IIS con CNG per l'offload SSL/TLS su Windows
- Aggiunta di un sistema di bilanciamento del carico con Elastic Load Balancing (facoltativo)

Funzionamento dell'offload SSL/TLS con AWS CloudHSM

Per stabilire una connessione HTTPS, il server Web esegue un processo di handshake con i client. Nell'ambito di questo processo, il server esegue l'offload di parte dell'elaborazione crittografica nei moduli HSM, 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 server SSL/TLS sia firmato da un certificato radice attendibile per il client.
   b. Estrae la chiave pubblica dal certificato del server.
   c. Genera un segreto premaster e lo crittografa con la chiave pubblica del server.
   d. Invia il segreto premaster crittografato al server.
4. Per decrittografare il segreto premaster del client, il server lo invia all'HSM. Il modulo HSM utilizza la chiave privata nell'HSM per decrittografare il segreto premaster e quindi invia il segreto premaster al server. Indipendentemente, il client e il server utilizzano il segreto premaster e alcune informazioni dal messaggio di saluto per calcolare un segreto master.
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 ulteriori informazioni su come configurare l'offload SSL/TLS con AWS CloudHSM, vedi uno dei seguenti argomenti:

- Offload SSL/TLS su Linux
- Utilizzo di IIS con CNG per l'offload SSL/TLS su Windows

Offload SSL/TLS su Linux

Con AWS CloudHSM, puoi eseguire l'offload SSL/TLS su Linux con NGINX, Apache e Tomcat. Per ulteriori informazioni, consulta gli argomenti correlati riportati di seguito.

Argomenti

Utilizzo di NGINX o Apache con OpenSSL per l'offload SSL/TLS su Linux

Questo argomento fornisce istruzioni dettagliate per la configurazione dell'offload SSL/TLS con AWS CloudHSM in un server Web Linux.

Argomenti

Panoramica

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.

Argomenti

- Prerequisiti per Client SDK 5
- Prerequisiti per Client SDK 3

Prerequisiti per Client SDK 5

Per configurare l'offload SSL/TLS per il server Web con Client SDK 5 è necessario quanto segue:

- Un cluster AWS CloudHSM 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

   - NGINX

     ```
     $ sudo yum install nginx
     ```

   - Apache

     ```
     $ sudo yum install httpd24 mod24_ssl
     ```

   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
     ```

   CentOS 7

   - Per le informazioni su come scaricare l'ultima versione di NGINX su CentOS 7, consulta il sito Web di NGINX.

L'ultima versione di NGINX disponibile per CentOS 7 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
```

Red Hat 7

- Per le informazioni su come scaricare l'ultima versione di NGINX su Red Hat 7, consulta il sito Web di NGINX.

  L'ultima versione di NGINX disponibile per Red Hat 7 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
```

CentOS 8

- 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
```

Ubuntu 18.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 20.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 22.04

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

3. Utilizza la CLI di CloudHSM per creare un 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.

Dopo aver completato queste operazioni, andare su Fase 2: generazione o importazione di una chiave privata e certificato SSL/TLS.

Note

- Per utilizzare SELinux (Security-Enhanced Linux) e i server Web, è necessario consentire le connessioni TCP in uscita sulla porta 2223, ovvero 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 Aggiunta di un modulo HSM.
- 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.

Prerequisiti per Client SDK 3

Per configurare l'offload SSL/TLS per il server Web con Client SDK 3 è necessario quanto segue:

- Un cluster AWS CloudHSM attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Linux con il seguente software installato:
  - Il client e gli strumenti a riga di comando AWS CloudHSM.
  - L'applicazione del server Web NGINX o Apache.
  - Il motore dinamico AWS CloudHSM per OpenSSL.

- 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. Completa le fasi descritte in Nozioni di base. Sarà quindi disponibile un cluster attivo con un HSM e un'istanza client Amazon EC2. L'istanza EC2 sarà configurata con gli strumenti a riga di comando. Utilizzare questa istanza client come server Web.
2. Effettuare la connessione 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.
3. Su un'istanza Linux EC2 che ha accesso al tuo cluster, installa il server Web NGINX o Apache:

   Amazon Linux

   - NGINX

     ```
     $ sudo yum install nginx
     ```

   - Apache

     ```
     $ sudo yum install httpd24 mod24_ssl
     ```

   Amazon Linux 2

   - La versione 1.19 di NGINX è l'ultima versione di NGINX compatibile con il motore di Client SDK 3 su Amazon Linux 2.

     Per ulteriori informazioni e per scaricare la versione 1.19 di NGINX, consulta il sito Web di NGINX.

   - Apache

     ```
     $ sudo yum install httpd mod_ssl
     ```

CentOS 7

- La versione 1.19 di NGINX è l'ultima versione di NGINX compatibile con il motore di Client SDK 3 su CentOS 7.

  Per ulteriori informazioni e per scaricare la versione 1.19 di NGINX, consulta il sito Web di NGINX.

- Apache

```
$ sudo yum install httpd mod_ssl
```

Red Hat 7

- La versione 1.19 di NGINX è l'ultima versione di NGINX compatibile con il motore di Client SDK 3 su Red Hat 7.

  Per ulteriori informazioni e per scaricare la versione 1.19 di NGINX, consulta il sito Web di NGINX.

- Apache

```
$ sudo yum install httpd mod_ssl
```

Ubuntu 16.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

Ubuntu 18.04

- NGINX

```
$ sudo apt install nginx
```

- Apache

```
$ sudo apt install apache2
```

4. (Facoltativo) Aggiungi altri HSM sul cluster. Per ulteriori informazioni, consulta Aggiunta di un modulo HSM.
5. Utilizzare cloudhsm_mgmt_util per creare un CU. Per ulteriori informazioni, consulta Gestione degli utenti HSM. 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: generazione o importazione di una chiave privata e certificato SSL/TLS.

Fase 2: generazione o importazione di una chiave privata e certificato SSL/TLS

Per abilitare il protocollo HTTPS, l'applicazione del tuo server Web (NGINX o Apache) necessita di una chiave privata e di un corrispondente certificato SSL/TLS. Per utilizzare l'offload SSL/TLS del server Web con AWS CloudHSM devi archiviare la chiave privata in un HSM nel tuo cluster AWS CloudHSM. Questa operazione può essere eseguita in uno dei seguenti modi:

- In mancanza di una chiave privata e di un certificato corrispondente, genera una chiave privata in un HSM, che serve a creare una richiesta di firma del certificato (CSR), che si utilizza per creare il certificato SSL/TLS.
- Se sono già disponibili una chiave privata e un certificato corrispondente, importa la chiave privata in un HSM.

Indipendentemente dal metodo scelto tra i precedenti, si esporta una chiave privata PEM falsa dall'HSM, che è un file di chiave privata in formato PEM contenente un riferimento alla chiave privata archiviata nell'HSM (non è la chiave privata effettiva). Il server Web utilizza la chiave privata PEM falsa per identificare la chiave privata nell'HSM durante l'offload SSL/TLS.

Completa una delle seguenti operazioni:

- Generazione di una chiave privata e di un certificato
- Importa una chiave privata e un certificato esistenti

Generazione di una chiave privata e di un certificato

Generazione di una chiave privata

Questa sezione mostra come generare una coppia di chiavi utilizzando Key Management Utility (KMU) di Client SDK 3. Una volta generata una coppia di chiavi all'interno dell'HSM, è possibile esportarla come file PEM falso e generare il certificato corrispondente.

Le chiavi private generate con Key Management Utility (KMU) possono essere utilizzate sia con Client SDK 3 che con Client SDK 5.

Installazione e configurazione di Key Management Utility (KMU)

1. Effettuare la connessione all'istanza del client.
2. Installa e configura Client SDK 3.
3. Eseguire il comando seguente per avviare il client AWS CloudHSM.

   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
```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

4. Eseguire il comando seguente per avviare lo strumento a riga di comando key_mgmt_util.

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

5. Eseguire il comando seguente per accedere all'HSM. Sostituire il *<nome utente>* e la *<password>* con i valori relativi all'utente di crittografia (CU).

```
Command: loginHSM -u CU -s <user name> -p <password>>
```

Generazione di una chiave privata

A seconda del caso d'uso, è possibile generare una coppia di chiavi RSA o EC. Completa una delle seguenti operazioni:

- Come generare una chiave privata RSA su un HSM

  Utilizza il comando `genRSAKeyPair` per generare una coppia di chiavi RSA. Questo esempio illustra la generazione di una coppia di chiavi RSA con un modulo di 2048, un esponente pubblico di 65537 e un'etichetta di *`tls_rsa_keypair`*.

```
Command: genRSAKeyPair -m 2048 -e 65537 -l tls_rsa_keypair
```

Se il comando ha dato esito positivo, dovresti vedere il seguente output che indica che hai generato una coppia di chiavi RSA.

```
Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

                Cfm3GenerateKeyPair:    public key handle: 7    private key handle: 8

                Cluster Status:
                Node id 1 status: 0x00000000 : HSM Return: SUCCESS
```

- Come generare una chiave privata EC su un HSM

  Utilizza il comando `genECCKeyPair` per generare una coppia di chiavi EC. Questo esempio illustra la generazione di una coppia di chiavi EC con un ID curva pari a 2 (corrispondente alla curva `NID_X9_62_prime256v1`) e un'etichetta di *`tls_ec_keypair`*.

  ```
  Command: genECCKeyPair -i 2 -l tls_ec_keypair
  ```

  Se il comando ha dato esito positivo, dovresti vedere il seguente output che indica che hai generato una coppia di chiavi EC.

  ```
  Cfm3GenerateKeyPair returned: 0x00 : HSM Return: SUCCESS

                  Cfm3GenerateKeyPair:    public key handle: 7    private key handle: 8

                  Cluster Status:
                  Node id 1 status: 0x00000000 : HSM Return: SUCCESS
  ```

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.

> **Note**
>
> I file PEM falsi generati con Key Management Utility (KMU) possono essere utilizzati sia con Client SDK 3 che con Client SDK 5.

Identifica l'handle della chiave corrispondente alla chiave che desideri esportare come PEM falsa, quindi esegui il comando seguente per esportare la chiave privata in formato PEM falso e salvarla in un file. Sostituire i valori seguenti con i propri valori.

- *<private_key_handle>*: handle della chiave privata generata. Questo handle è stato generato da uno dei comandi di generazione della chiave nella fase precedente. Nell'esempio sopra riportato, l'handle della chiave privata è 8.
- *<web_server_fake_PEM.key>*: il nome del file su cui verrà scritta la chiave PEM falsa.

```
Command: getCaviumPrivKey -k <private_key_handle> -out <web_server_fake_PEM.key>
```

Exit (Esci)

Esegui il comando seguente per arrestare key_mgmt_util.

```
Command: exit
```

Ora dovresti avere un nuovo file sul sistema, situato nel percorso specificato da *<web_server_fake_PEM.key>* nel 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 ti affidi a un'autorità di certificazione, invia il file della CSR a tale autorità e utilizza il certificato SSL/TLS firmato che ti è stato fornito nel server Web per HTTPS.

In alternativa all'utilizzo di un'autorità di certificazione, è possibile utilizzare AWS CloudHSM OpenSSL Dynamic Engine per creare un certificato auto-firmato. 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.

Installazione e configurazione di OpenSSL Dynamic Engine

1. Effettuare la connessione all'istanza del client.
2. Per l'installazione e la configurazione, esegui una delle seguenti operazioni:
   - the section called “Installazione di OpenSSL Dynamic Engine”
   - the section called “OpenSSL Dynamic Engine”

Generazione di un certificato

1. Ottieni una copia del file KeyStore 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). Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa. Sostituire *<web_server.csr>* con il nome del file che contiene la CSR.

   Il comando `req` è interattivo. Ogni campo deve essere compilato e le informazioni 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 AWS CloudHSM OpenSSL Dynamic Engine per firmare la CSR con la chiave privata nell'HSM. In questo modo viene creato un certificato autofirmato. Sostituire i valori seguenti nel comando con i propri valori.

- *<web_server.csr>*: il nome del file che contiene la CSR.
- *<web_server_fake_PEM.key>*: il nome del file che contiene la chiave privata PEM falsa.
- *<web_server.crt>*: il 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.

Importa una chiave privata e un certificato esistenti

Se disponi già di una chiave privata e di un certificato SSL/TLS corrispondente che usi per HTTPS sul server Web, puoi importare la chiave in un HSM seguendo la procedura descritta in questa sezione.

> **Note**
>
> Alcune note sull'importazione di chiavi private e sulla compatibilità con Client SDK:
>
> - L'importazione di una chiave privata esistente richiede Client SDK 3.
> - È possibile utilizzare le chiavi private di Client SDK 3 con Client SDK 5.
> - OpenSSL Dynamic Engine per Client SDK 3 non supporta le piattaforme Linux più recenti, che sono invece supportate con l'implementazione di OpenSSL Dynamic Engine per Client SDK 5. È possibile importare una chiave privata esistente utilizzando Key Management Utility (KMU) fornita con Client SDK 3, quindi utilizzare quella chiave privata e l'implementazione di OpenSSL Dynamic Engine con Client SDK 5 per supportare l'offload SSL/TLS sulle piattaforme Linux più recenti.

Per importare una chiave privata esistente in un HSM con Client SDK 3

1. Esegui la connessione all'istanza del client Amazon EC2. Se necessario, copiare la chiave privata esistente e il certificato nell'istanza.
2. Installa e configura Client SDK 3
3. Eseguire il comando seguente per avviare il client AWS CloudHSM.

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
```

Ubuntu 20.04 LTS

```
$ sudo service cloudhsm-client start
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

4. Eseguire il comando seguente per avviare lo strumento a riga di comando key_mgmt_util.

```
$ /opt/cloudhsm/bin/key_mgmt_util
```

5. Eseguire il comando seguente per accedere all'HSM. Sostituire il *<nome utente>* e la *<password>* con i valori relativi all'utente di crittografia (CU).

```
Command: loginHSM -u CU -s <user name> -p <password>
```

6. Eseguire i comandi seguenti per importare la chiave privata in un HSM.

   a. Eseguire il comando seguente per creare una chiave di wrapping simmetrica valida solo per la sessione corrente. Di seguito sono riportati il comando e l'output.

```
Command: genSymKey -t 31 -s 16 -sess -l wrapping_key_for_import

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
```

   b. Eseguire il comando seguente per importare la chiave privata esistente in un HSM. Di seguito sono riportati il comando e l'output. Sostituire i valori seguenti con i propri valori:

   - *<web_server_existing.key>*: il nome del file che contiene la chiave privata.
   - *<web_server_imported_key>*: l'etichetta per la chiave privata importata.
   - *<wrapping_key_handle>*: handle della chiave di wrapping generato dal comando precedente. Nell'esempio precedente, l'handle della chiave di wrapping è 6.

```
Command: importPrivateKey -f <web_server_existing.key> -
l <web_server_imported_key> -w <wrapping_key_handle>

BER encoded key length is 1219
Cfm3WrapHostKey returned: 0x00 : HSM Return: SUCCESS
Cfm3CreateUnwrapTemplate returned: 0x00 : HSM Return: SUCCESS
Cfm3UnWrapKey returned: 0x00 : HSM Return: SUCCESS
```

```
Private Key Unwrapped.  Key Handle: 8
Cluster Error Status
Node id 0 and err state 0x00000000 : HSM Return: SUCCESS
```

7. Eseguire il comando seguente per esportare la chiave privata nel falso formato PEM e salvarla in un file. Sostituire i valori seguenti con i propri valori.

   - *<private_key_handle>*: handle della chiave privata importata. Questo handle è stato generato dal secondo comando nella fase precedente. Nell'esempio sopra riportato, l'handle della chiave privata è 8.
   - *<web_server_fake_PEM.key>*: il nome del file che contiene la chiave privata PEM falsa esportata.

   ```
   Command: getCaviumPrivKey -k <private_key_handle> -out <web_server_fake_PEM.key>
   ```

8. Eseguire il comando seguente per arrestare key_mgmt_util.

   ```
   Command: exit
   ```

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. In questo modo viene completata la configurazione del software del server Web Linux per l'offload SSL/TLS 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. Sostituire *<web_server.crt>* con il nome del certificato del 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. Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

```
$ 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

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:

- Se si utilizza Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se si utilizza Client SDK 5

```
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 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:

- Se si utilizza Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se si utilizza Client SDK 5

```
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 {
    }
}
```

CentOS 7

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:

- Se si utilizza Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se si utilizza Client SDK 5

```
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 {
    }
}
```

CentOS 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 7

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:

- Se si utilizza Client SDK 3

```
ssl_engine cloudhsm;
env n3fips_password;
```

- Se si utilizza Client SDK 5

```
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 {
    }
}
```

Ubuntu 16.04 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 n3fips_password;
```

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 {
        }
    }
```

Ubuntu 18.04 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 20.04 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 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

Salva il file.

8. Eseguire il backup del file di configurazione `systemd`, quindi impostare il percorso `EnvironmentFile`.

   Amazon Linux

   Nessuna operazione necessaria.

   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
      ```

   CentOS 7

   Nessuna operazione necessaria.

   CentOS 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 7

Nessuna operazione necessaria.

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
```

Ubuntu 16.04

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 18.04

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 20.04 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 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

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

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):

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Salva il file.

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):

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Salva il file.

CentOS 7

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):

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Salva il file.

CentOS 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 del CU.

Salva il file.

Red Hat 7

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):

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Salva 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 del CU.

Salva il file.

Ubuntu 16.04 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):

```
n3fips_password=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Salva il file.

Ubuntu 18.04 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 del CU.

Salva il file.

Ubuntu 20.04 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 del CU.

Salva il file.

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

11. Avviare il server Web NGINX.

Amazon Linux

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):

```
$ sudo service nginx start
```

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
```

CentOS 7

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
```

CentOS 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 7

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 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
```

Ubuntu 16.04 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 18.04 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 20.04 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 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

12. (Facoltativo) Configura la piattaforma per avviare NGINX all'avvio.

Amazon Linux

```
$ sudo chkconfig nginx on
```

Amazon Linux 2

```
$ sudo systemctl enable nginx
```

CentOS 7

Nessuna operazione necessaria.

CentOS 8

```
$ sudo systemctl enable nginx
```

Red Hat 7

Nessuna operazione necessaria.

Red Hat 8

```
$ sudo systemctl enable nginx
```

Ubuntu 16.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 18.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 20.04 LTS

```
$ sudo systemctl enable nginx
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

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 Amazon EC2.
2. Definisci le posizioni predefinite per i certificati e le chiavi private per la piattaforma.

Amazon Linux

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 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
```

CentOS 7

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
```

CentOS 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 7

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 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
```

Ubuntu 16.04 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 18.04 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 20.04 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 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

3. Copia il certificato del server Web nella posizione richiesta per la piattaforma.

Amazon Linux

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Amazon Linux 2

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

CentOS 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

CentOS 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Red Hat 7

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Red Hat 8

```
$ sudo cp <web_server.crt> /etc/pki/tls/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server.crt> /etc/ssl/certs/localhost.crt
```

Sostituire *<web_server.crt>* con il nome del certificato del server Web.

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

4. Copia la tua chiave privata PEM falsa nella posizione richiesta per la piattaforma.

Amazon Linux

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Amazon Linux 2

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

CentOS 7

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

CentOS 8

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Red Hat 7

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Red Hat 8

```
$ sudo cp <web_server_fake_PEM.key> /etc/pki/tls/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Ubuntu 16.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Ubuntu 18.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Ubuntu 20.04 LTS

```
$ sudo cp <web_server_fake_PEM.key> /etc/ssl/private/localhost.key
```

Sostituire *<web_server_fake_PEM.key>* con il nome del file che contiene la chiave privata PEM falsa.

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

5. Cambia la proprietà di questi file se richiesto dalla piattaforma.

Amazon Linux

```
$ 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 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.

CentOS 7

```
$ sudo chown apache /etc/pki/tls/certs/localhost.crt /etc/pki/tls/private/
localhost.key
```

Fornisce il permesso di lettura all'utente denominato apache.

CentOS 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 7

```
$ 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.

Ubuntu 16.04 LTS

Nessuna operazione necessaria.

Ubuntu 18.04 LTS

Nessuna operazione necessaria.

Ubuntu 20.04 LTS

Nessuna operazione necessaria.

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

6. Configura le direttive Apache per la piattaforma.

Amazon Linux

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
```

Salva il file.

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
```

Salva il file.

CentOS 7

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
```

Salva il file.

CentOS 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
```

Salva il file.

Red Hat 7

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
```

Salva 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
```

Salva il file.

Ubuntu 16.04 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
```

Salva il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 18.04 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
```

Salva il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 20.04 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
```

Salva il file.

Abilita il modulo SSL e la configurazione predefinita del sito SSL:

```
$ sudo a2enmod ssl
$ sudo a2ensite default-ssl
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

7. Configura un file environment-values per la piattaforma.

Amazon Linux

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Amazon Linux 2

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

CentOS 7

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

CentOS 8

Apri il file di servizio httpd:

```
/lib/systemd/system/httpd.service
```

Aggiungi quanto segue alla sezione `[Service]`:

```
EnvironmentFile=/etc/sysconfig/httpd
```

Red Hat 7

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
```

Ubuntu 16.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Ubuntu 18.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Ubuntu 20.04 LTS

Nessuna operazione necessaria. I valori dell'ambiente vanno in `/etc/sysconfig/httpd`

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

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

   Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

   - Se si utilizza Client SDK 3

     ```
     n3fips_password=<CU user name>:<password>
     ```

   - Se si utilizza Client SDK 5

     ```
     CLOUDHSM_PIN=<CU user name>:<password>
     ```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Amazon Linux 2

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

CentOS 7

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

CentOS 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 del CU.

Red Hat 7

Utilizza un editor di testo per modificare `/etc/sysconfig/httpd`.

- Se si utilizza Client SDK 3

```
n3fips_password=<CU user name>:<password>
```

- Se si utilizza Client SDK 5

```
CLOUDHSM_PIN=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del 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 del CU.

> **Note**
>
> Client SDK 5 introduce la variabile di ambiente `CLOUDHSM_PIN` per l'archiviazione delle credenziali del CU.

Ubuntu 16.04 LTS

Utilizza un editor di testo per modificare `/etc/apache2/envvars`.

```
export n3fips_password=<CU user name>:<password>
```

Sostituisci *<CU user name>* e *<password>* con le credenziali del CU.

Ubuntu 18.04 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 del 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 20.04 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 del 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 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

9. Avviare il server Web Apache.

Amazon Linux

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

Amazon Linux 2

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

CentOS 7

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

CentOS 8

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

Red Hat 7

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

Red Hat 8

```
$ sudo systemctl daemon-reload
$ sudo service httpd start
```

Ubuntu 16.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 18.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 20.04 LTS

```
$ sudo service apache2 start
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

10. (Facoltativo) Configura la tua piattaforma per avviare Apache all'avvio.

Amazon Linux

```
$ sudo chkconfig httpd on
```

Amazon Linux 2

```
$ sudo chkconfig httpd on
```

CentOS 7

```
$ sudo chkconfig httpd on
```

CentOS 8

```
$ systemctl enable httpd
```

Red Hat 7

```
$ sudo chkconfig httpd on
```

Red Hat 8

```
$ systemctl enable httpd
```

Ubuntu 16.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 18.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 20.04 LTS

```
$ sudo systemctl enable apache2
```

Ubuntu 22.04 LTS

Il supporto per OpenSSL Dynamic Engine non è ancora disponibile.

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'offload SSL/TLS con AWS CloudHSM, aggiungi 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. In seguito, crea una connessione HTTPS al tuo server Web e verifica che stia utilizzando il certificato configurato per l'offload SSL/TLS con 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. Selezionare Add Rule (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, seleziona 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 Save (Salva).

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo avere aggiunto il server Web a un gruppo di sicurezza, puoi verificare che l'offload SSL/TLS stia utilizzando il certificato auto-firmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 certificato SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata del server Web è archiviata in un modulo HSM nel tuo cluster AWS CloudHSM.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiunta di un sistema di bilanciamento del carico con Elastic Load Balancing (facoltativo).

Utilizzo di Tomcat con JSSE per l'offload SSL/TLS su Linux

Questo argomento fornisce istruzioni dettagliate per la configurazione dell'offload SSL/TLS utilizzando Java Secure Socket Extension (JSSE) con l'SDK JCE di AWS CloudHSM.

Argomenti

Panoramica

In AWS CloudHSM, i server Web Tomcat funzionano su Linux per supportare HTTPS. L'SDK JCE di AWS CloudHSM fornisce un'interfaccia che può essere utilizzata con JSSE (Java Secure Socket Extension) per consentire l'uso degli HSM per tali server Web. AWS CloudHSM JCE è il bridge che consente la connessione di 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

Attieniti a questi prerequisiti per utilizzare un server Web Tomcat con AWS CloudHSM per l'offload SSL/TLS su Linux. È necessario soddisfare tali prerequisiti per configurare l'offload SSL/TLS 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 cluster AWS CloudHSM attivo con almeno due moduli di sicurezza hardware (HSM) su cui è installato e configurato JCE per 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 per AWS CloudHSM su un cluster AWS CloudHSM attivo con almeno due moduli di sicurezza hardware (HSM). Per ulteriori informazioni sull'installazione, consulta la pagina su JCE per Client SDK 5.
2. Su un'istanza Linux EC2 che ha accesso al cluster AWS CloudHSM, 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 Utilizzo di Client SDK 5 per l'integrazione con Java Keytool e Jarsigner.

Dopo aver completato queste operazioni, andare su Fase 2: generazione o importazione di una chiave privata e certificato SSL/TLS.

Note

- Per utilizzare SELinux (Security-Enhanced Linux) e i server Web, è necessario consentire le connessioni TCP in uscita sulla porta 2223, ovvero 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 Aggiunta di un modulo HSM.
- 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: generazione o importazione di una chiave privata e certificato SSL/TLS

Per abilitare il protocollo HTTPS, l'applicazione del server Web Tomcat necessita di una chiave privata e di un certificato SSL/TLS corrispondente. Per utilizzare l'offload SSL/TLS del server Web con AWS CloudHSM devi archiviare la chiave privata in un HSM nel tuo cluster AWS CloudHSM.

> **Note**
>
> In mancanza di una chiave privata e di un certificato corrispondente, genera una chiave privata in un HSM, che serve a creare una richiesta di firma del certificato (CSR), che si utilizza per creare il certificato SSL/TLS.

Si crea un file KeyStore AWS CloudHSM locale che contiene un riferimento alla chiave privata sull'HSM e al certificato associato. Il server Web utilizza il file KeyStore AWS CloudHSM per identificare la chiave privata sull'HSM durante l'offload SSL/TLS.

Argomenti

- Generazione di una chiave privata
- Generazione di un certificato auto-firmato

Generazione di una chiave privata

Questa sezione illustra come generare una coppia di chiavi utilizzando KeyTool di JDK. Una volta generata una coppia di chiavi all'interno dell'HSM, è possibile esportarla come file KeyStore 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.

Utilizza il comando **`genkeypair`** in KeyTool per generare una coppia di chiavi RSA

1. Dopo aver sostituito le *<VARIABLES>* indicate sotto con i dati specifici, utilizza il seguente comando per generare un file KeyStore denominato `jsse_keystore.keystore`, che conterrà un riferimento alla 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 desiderato in cui generare il file KeyStore.
   - ***<UNIQUE ALIAS FOR KEYS>***: serve a identificare in modo univoco la chiave sull'HSM. Questo alias verrà impostato come attributo ETICHETTA della chiave.
   - ***<KEY PASSWORD>***: password che protegge il riferimento alla chiave, memorizzato nel file KeyStore locale.
   - ***<KEYSTORE PASSWORD>***: password del file KeyStore locale.
   - ***<JSSE KEYSTORE NAME>***: nome del file KeyStore.
   - ***<CERT DOMAIN NAME>***: nome distinto X.500.
   - ***<KEY ALGORITHM>***: algoritmo della 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, SHA1WithRSA, SHA224WithRSA, SHA256WithRSA, SHA384WithRSA e SHA512WithRSA).

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 ti affidi a un'autorità di certificazione, invia il file della CSR a tale autorità e utilizza il certificato SSL/TLS firmato che ti è stato fornito nel server Web per HTTPS.

In alternativa all'utilizzo di un'autorità di certificazione, è possibile utilizzare KeyTool per creare un certificato auto-firmato. 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 KeyTool e 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 le *<VARIABLES>* indicate sotto con i dati specifici, esegui il seguente comando per firmare la CSR con la 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 le *<VARIABLES>* indicate sotto con i dati specifici, esegui il seguente comando per importare un certificato firmato come certificato attendibile. Questo passaggio archivierà il certificato nella voce KeyStore identificata dall'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 file 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. In questo modo viene completata la configurazione del software del server Web Linux per l'offload SSL/TLS con AWS CloudHSM. Per ulteriori informazioni, consulta la documentazione di riferimento relativa alla configurazione di Apache Tomcat 9.

Arresta il server

- Dopo aver sostituito le *<VARIABLES>* indicate sotto con i dati specifici, esegui il seguente comando per arrestare il server Tomcat prima di aggiornare la configurazione

  ```
  $ /<TOMCAT DIRECTORY>/bin/shutdown.sh
  ```

  - ***<TOMCAT DIRECTORY>***: la directory di installazione di Tomcat.

Aggiorna il classpath di Tomcat

1. Effettuare la connessione all'istanza del client.
2. Individua la cartella di installazione di Tomcat.
3. Dopo aver sostituito le *<VARIABLES>* con i dati specifici, usa il seguente comando per aggiungere la libreria Java e il percorso Java CloudHSM nel classpath di Tomcat, situato nel file Tomcat/bin/catalina.sh.

   ```
   $ 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 le *<VARIABLES>* con i 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>***: directory in cui si trova il file KeyStore.
   - ***<JSSE KEYSTORE NAME>***: nome del file KeyStore.
   - ***<KEYSTORE PASSWORD>***: password del file KeyStore locale.
   - ***<KEY PASSWORD>***: password che protegge il riferimento alla chiave, memorizzato nel file KeyStore locale.
   - ***<UNIQUE ALIAS FOR KEYS>***: serve a identificare in modo univoco la chiave sull'HSM. Questo alias verrà impostato come attributo ETICHETTA della chiave.
   - ***<TOMCAT PATH>***: il percorso alla cartella Tomcat.

Avvio del server

- Dopo aver sostituito le *<VARIABLES>* indicate sotto con i dati specifici, utilizza il seguente comando per avviare il server Tomcat:

```
$ /<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'offload SSL/TLS con AWS CloudHSM, aggiungi 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. In seguito, crea una connessione HTTPS al tuo server Web e verifica che stia utilizzando il certificato configurato per l'offload SSL/TLS con 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 Crea gruppo di sicurezza.
4. Per 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. Selezionare Add Rule (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, seleziona 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 Save (Salva).

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo avere aggiunto il server Web a un gruppo di sicurezza, puoi verificare che l'offload SSL/TLS stia utilizzando il certificato auto-firmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 certificato SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata del server Web è archiviata in un modulo HSM nel tuo cluster AWS CloudHSM.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiunta di un sistema di bilanciamento del carico con Elastic Load Balancing (facoltativo).

Utilizzo di IIS con CNG per l'offload SSL/TLS su Windows

Questo tutorial fornisce istruzioni dettagliate per la configurazione dell'offload SSL/TLS con AWS CloudHSM in un server Web Windows.

Argomenti

- Fase 4: abilitazione del traffico HTTPS e verifica del certificato

Panoramica

In Windows Internet Information Services (IIS) per l'applicazione del server Web Windows Server supporta HTTPS in modo nativo. Il provider di archiviazione delle chiavi (KSP) AWS CloudHSM per Cryptography API: Next Generation (CNG) Microsoft fornisce l'interfaccia che consente a IIS di utilizzare gli HSM nel cluster per l'offload e lo storage delle chiavi di crittografia. Il KSP AWS CloudHSM è il bridge che consente la connessione di 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

Per configurare l'offload SSL/TLS per il server Web con AWS CloudHSM è necessario quanto segue:

- Un cluster AWS CloudHSM attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue un sistema operativo Windows su cui sia installato il seguente software:
  - Il software client AWS CloudHSM 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. È inoltre presente un'istanza del client di Amazon EC2 con Windows Server con il software client AWS CloudHSM per Windows installato.
2. (Facoltativo) Aggiungi altri HSM sul cluster. Per ulteriori informazioni, consulta Aggiunta di un modulo HSM.
3. Connettersi al server Windows. Per ulteriori informazioni, consulta la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
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. Nella fase 5, se hai utilizzato Gestione credenziali di Windows 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 la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
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 Next (Successivo).
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Next (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 Next (Successivo).
9. Leggere le informazioni su Web Server Role (IIS) (Ruolo Web Server). Quindi scegli Next (Successivo).
10. Per Select server roles (Seleziona ruoli server), accettare le impostazioni predefinite o modificarle in base alle esigenze. Quindi scegli Next (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 il protocollo HTTPS, il server Web necessita di un certificato SSL/TLS e di una chiave privata corrispondente. Per utilizzare l'offload SSL/TLS con AWS CloudHSM, è necessario archiviare

la chiave privata nell'HSM nel 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 un CSR
- Come ottenere un certificato firmato e importarlo

Crea un CSR

Utilizza un KSP AWS CloudHSM sul tuo Windows Server per creare una CSR.

Per creare un CSR

1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
2. Utilizza il seguente comando per avviare il daemon del client AWS CloudHSM.

   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. Eseguire il comando certreq Windows per creare una CSR dal file `IISCertRequest.inf` creato nella fase precedente. L'esempio seguente salva la CSR in un file denominato `IISCertRequest.csr`. Se si utilizza un nome di file diverso per il file della richiesta di firma di certificato, è necessario sostituire *`IISCertRequest.inf`* con il nome file appropriato. È anche possibile sostituire *`IISCertRequest.csr`* con un nome di file diverso per il 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 viene utilizzata una CA, è necessario inviarle il file CSR (`IISCertRequest.csr`) e creare tramite essa un certificato SSL/TLS firmato.

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. È anche possibile sostituire *SelfSignedCA.key* con il nome di file che contiene la 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. Sostituire *SelfSignedCA.key* con il nome de file che contiene la chiave privata (se diverso). È anche possibile sostituire *SelfSignedCA.crt* con il nome di file che contiene il 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 auto-firmato
  - *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. In questo modo viene completata la configurazione del software del server Web Windows per l'offload SSL/TLS 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 la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
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 fare doppio clic su Autorità di certificazione radice attendibili.
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`. Scegliere Open (Apri).
9. Seleziona 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 la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
2. Avvia il daemon del client AWS CloudHSM.
3. Copia il certificato firmato del server Web (quello creato al termine della fase precedente di questo tutorial) nel server Windows.
4. Nel server Windows eseguire il comando certreq Windows per accettare il certificato firmato, come nell'esempio seguente. Sostituire *`IISCert.crt`* con il nome del file che contiene il certificato firmato del 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'offload SSL/TLS con AWS CloudHSM, aggiungi 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. In seguito, crea una connessione HTTPS al tuo server Web e verifica che stia utilizzando il certificato configurato per l'offload SSL/TLS con 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. Selezionare Add Rule (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, seleziona 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 Save (Salva).

Verifica dell'utilizzo da parte di HTTPS del certificato configurato

Dopo avere aggiunto il server Web a un gruppo di sicurezza, puoi verificare che l'offload SSL/TLS stia utilizzando il certificato auto-firmato. Per farlo, puoi utilizzare un browser Web o uno strumento come OpenSSL s_client.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 certificato SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

Per verificare l'offload SSL/TLS 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 instradare il nome del 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 SSL/TLS corrisponda a quello configurato per l'uso da parte del server Web.

A questo punto disponi di un sito Web protetto con HTTPS. La chiave privata del server Web è archiviata in un modulo HSM nel tuo cluster AWS CloudHSM.

Per aggiungere un sistema di bilanciamento del carico, consulta la pagina Aggiunta di un sistema di bilanciamento del carico con Elastic Load Balancing (facoltativo).

Aggiunta di un sistema di bilanciamento del carico con Elastic Load Balancing (facoltativo)

Dopo aver configurato l'offload SSL/TLS con un server Web, puoi creare ulteriori server Web e un sistema di bilanciamento del carico Elastic Load Balancing che instrada 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

- Creazione di una sottorete per un secondo server Web
- Creazione del secondo server Web
- Creazione del sistema di bilanciamento del carico

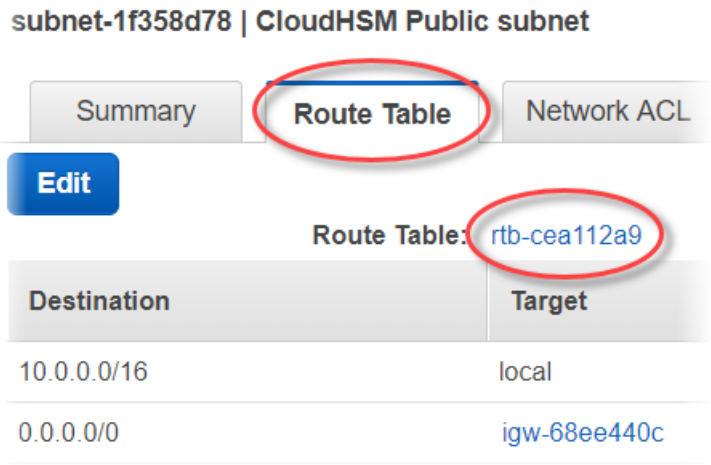
Creazione di una sottorete per un secondo server Web

Prima di creare un altro server Web, devi creare una nuova sottorete nello stesso VPC che contiene il tuo server Web esistente e il cluster AWS CloudHSM.

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, scegliere il VPC AWS CloudHSM che contiene il server Web esistente e il cluster AWS CloudHSM.
   c. Per Availability Zone (Zona di disponibilità), scegliere una zona di disponibilità diversa da quella che contiene il server Web esistente.
   d. Per Blocco CIDR IPv4, digita 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).

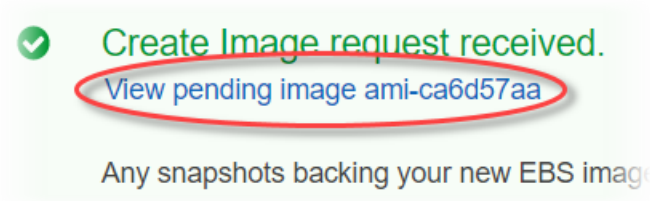
Creazione del secondo server Web

Completa le fasi seguenti per creare un secondo server Web con la stessa configurazione del tuo server Web esistente.

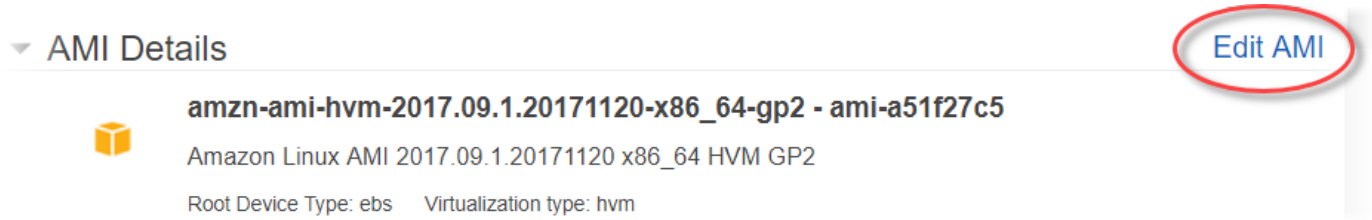
Per creare un secondo server Web

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. Seleziona il link di visualizzazione dell'immagine ami-***<AMI ID>*** in sospeso.



   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, seleziona 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, seleziona Le mie AMI. Quindi cancellare il testo nella casella di ricerca.
10. Accanto all'immagine del server Web, scegliere Select (Seleziona).
11. Seleziona Sì, voglio proseguire con questa AMI (***<image name>*** - ami-***<AMI ID>***).
12. Seleziona 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 conto tuo quando hai creato il cluster. È necessario scegliere il gruppo di sicurezza per consentire all'istanza del server Web di connettersi ai moduli HSM 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.

    Scegliere Review and Launch (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 Prendo atto di avere accesso al file della chiave privata selezionata (***<private key file name>***.pem) e che senza questo file non potrei 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).

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.

Per creare un sistema di bilanciamento del carico

1. Apri la sezioneSistemi 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. Seleziona Next: Revisione.
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. Ciò significa che puoi verificare se sta inviando il traffico HTTPS al tuo server Web con l'offload SSL/TLS con 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 instradare il nome del 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. Sostituisci *<DNS name>* con il nome DNS del sistema di bilanciamento del carico in uso.

```
openssl s_client -connect <DNS name>:443
```

> **Tip**
>
> Puoi utilizzare un servizio DNS come Amazon Route 53 per instradare il nome del 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.

A questo punto disponi di un sito Web protetto tramite HTTPS, con la chiave privata del server Web archiviata in un modulo HSM nel 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.

Configurazione di Windows Server come autorità di certificazione (CA) con AWS CloudHSM

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, sarà configurata una CA tramite Windows Server e AWS CloudHSM. 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 configuri questo ruolo, puoi utilizzare un provider di archiviazione delle chiavi (KSP) AWS CloudHSM per creare e archiviare la chiave privata della CA nel tuo cluster AWS CloudHSM. Il KSP è il bridge che consente la connessione dal tuo server Windows al tuo cluster AWS CloudHSM. 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

Windows Server come autorità di certificazione, fase 1: configurazione dei prerequisiti

Per configurare Windows Server come autorità di certificazione (CA) con AWS CloudHSM, ti serve quanto segue:

- Un cluster AWS CloudHSM attivo con almeno un HSM.
- Un'istanza Amazon EC2 con un sistema operativo Windows Server e software client AWS CloudHSM 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. È inoltre presente un'istanza del client di Amazon EC2 con Windows Server con il software client AWS CloudHSM per Windows installato.
2. (Facoltativo) Aggiungi altri HSM sul cluster. Per ulteriori informazioni, consulta Aggiunta di un modulo HSM.
3. Effettuare la connessione all'istanza del client. Per ulteriori informazioni, consulta la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
4. Crea un crypto user (CU) consultando la pagina sulla gestione degli utenti HSM con la CLI di CloudHSM o la pagina sulla gestione degli utenti HSM con 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. Nella fase 5, se hai utilizzato Gestione credenziali di Windows 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 creare una CA Windows Server con AWS CloudHSM, consulta la sezione Creare un'autorità di certificazione (CA) Windows Server.

Windows Server come autorità di certificazione, fase 2: creazione di un'autorità di certificazione (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, puoi utilizzare un provider di archiviazione delle chiavi (KSP) AWS CloudHSM per creare e archiviare la chiave privata della CA nel tuo cluster AWS CloudHSM.

> **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 la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
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 Next (Successivo).
6. Per Server Selection (Selezione server) scegliere Select a server from the server pool (Seleziona un server dal gruppo di server). Quindi scegli Next (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. Seleziona 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 Next (Successivo).
13. Per Role Services (Servizi ruolo), selezionare Certification Authority (Autorità di certificazione). Quindi scegli Next (Successivo).
14. Per Setup Type (Tipo di installazione) selezionare Standalone CA (CA indipendente). Quindi scegli Next (Successivo).
15. Per CA Type (Tipo CA) selezionare Root CA (CA radice). Quindi scegli Next (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 Next (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.

    Seleziona 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.

    Seleziona Successivo.
19. Per Validity Period (Periodo di validità) specificare un periodo di tempo in anni, mesi, settimane o giorni. Quindi scegli Next (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 Next (Successivo).
21. Per Confirmation (Conferma) rivedere le informazioni relative alla CA, quindi scegliere Configure (Configura).
22. Selezionare Close (Chiudi) e poi ancora Close (Chiudi).

È adesso disponibile 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.

Windows Server come autorità di certificazione, fase 3: firma di una richiesta di firma del certificato (CSR) con la CA Windows Server con AWS CloudHSM

È possibile utilizzare la CA 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

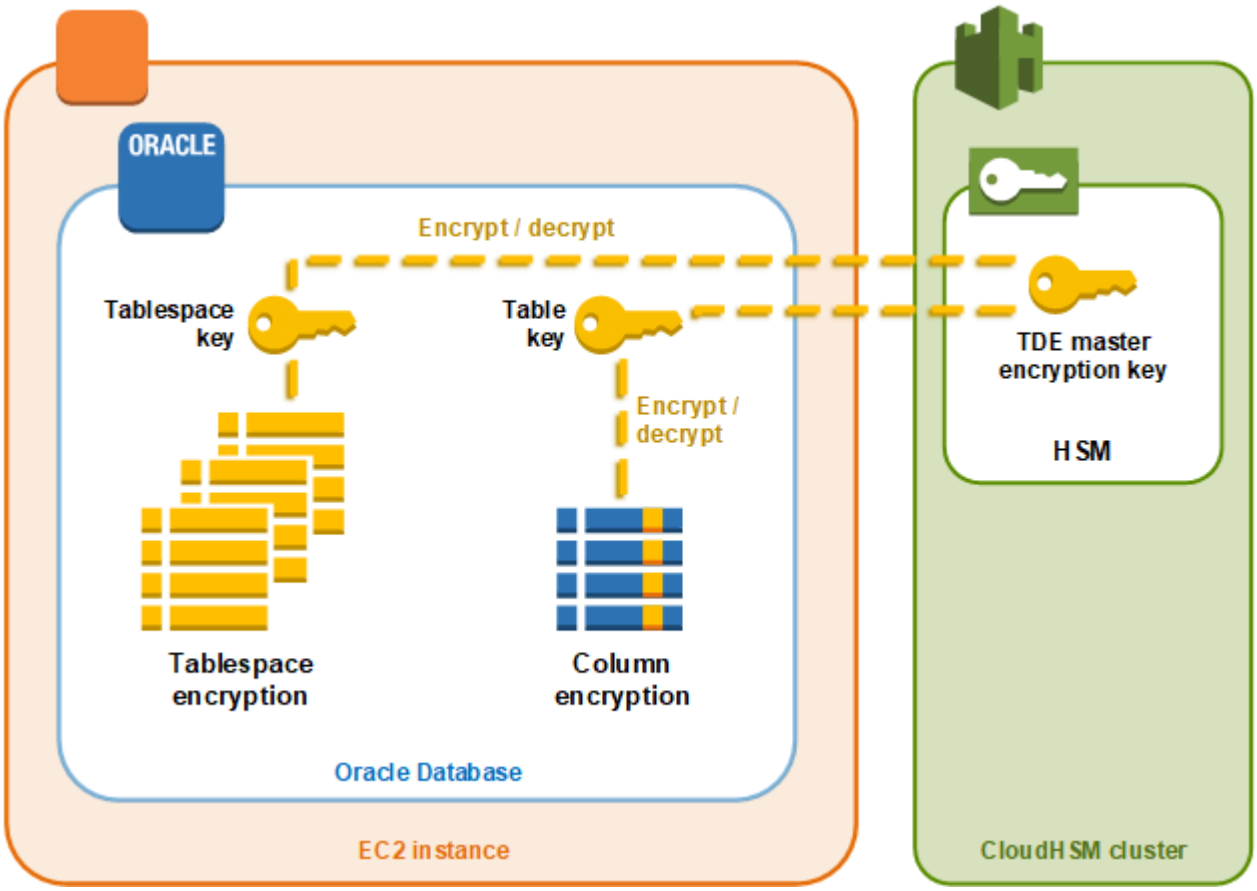
1. Effettuare la connessione al server Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
2. Su Windows Server avviare Server Manager.
3. Nel pannello di controllo Server Manager, 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 Windows Server con AWS CloudHSM e di un certificato valido firmato dalla CA Windows Server.

Oracle Database Transparent Data Encryption (TDE) 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.

Per configurare l'integrazione di Oracle TDE con AWS CloudHSM

1. Seguire la procedura in Configurazione dei prerequisiti per preparare l'ambiente.
2. Seguire la procedura in Configurazione del database per configurare Oracle Database per l'integrazione con il cluster AWS CloudHSM.

Oracle TDE con AWS CloudHSM: configurazione dei prerequisiti

Per effettuare l'integrazione di Oracle TDE con AWS CloudHSM, hai bisogno dei seguenti elementi:

- Un cluster AWS CloudHSM attivo con almeno un HSM.
- Un'istanza Amazon EC2 che esegue il sistema operativo Amazon Linux con il seguente software installato:
  - Il client e gli strumenti a riga di comando AWS CloudHSM.
  - La libreria software AWS CloudHSM per PKCS #11.
  - Oracle Database. AWS CloudHSM supporta l'integrazione di 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 impostare 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. Anche il client e gli strumenti a riga di comando AWS CloudHSM saranno installati e configurati.
2. (Facoltativo) Aggiungi altri HSM sul cluster. Per ulteriori informazioni, consulta Aggiunta di un modulo HSM.
3. Connettiti alla tua istanza del client Amazon EC2 ed esegui le operazioni descritte di seguito:
   a. Installa la libreria software AWS CloudHSM 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 Gestione degli utenti HSM.

Una volta completate queste fasi, puoi Configurazione del database.

Oracle TDE con AWS CloudHSM: configurazione del database e creazione della chiave di crittografia principale

Per integrare Oracle TDE con il cluster AWS CloudHSM, consulta i seguenti argomenti:

1. Aggiornamento della configurazione di Oracle Database per utilizzare gli HSM nel cluster come modulo di sicurezza esterno. Per informazioni sui moduli di sicurezza esterni, vedi la sezione Introduction to Transparent Data Encryption nella Oracle Database Advanced Security Guide.
2. Creazione della chiave di crittografia principale di Oracle TDE negli HSM del cluster.

Aggiornamento della configurazione di Oracle Database

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))
   ```

   Salva il file.
4. Esegui il seguente comando per creare la directory in cui Oracle Database presume si trovi il file della libreria software PKCS #11 di AWS CloudHSM.

   ```
   sudo mkdir -p /opt/oracle/extapi/64/hsm
   ```

5. Esegui uno dei seguenti comandi per copiare la libreria software AWS CloudHSM per il file PKCS #11 nella directory creata nella fase 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.

Creazione della chiave di crittografia principale di 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, consulta "Problema: Oracle imposta l'attributo PCKS #11 `CKA_MODIFIABLE` durante la generazione della chiave principale, ma HSM non lo supporta" in Problemi noti per l'integrazione di applicazioni di terze parti.

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. Sostituisci il *<Nome utente CU>* con quello dell'utente di crittografia (CU). Sostituire la *<password>* con quella del 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. Sostituisci il *<Nome utente CU>* con il nome dell'utente di crittografia (CU). Sostituire la *<password>* con quella del 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>";
```

Uso di Microsoft SignTool con AWS CloudHSM 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 ricevitore è in grado di verificare l'integrità dei dati decrittografando 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 attendibile. AWS CloudHSM fornisce hardware convalidato di protezione di livello 3 FIPS 140-2 hardware per te per proteggere queste chiavi con accesso single-tenant esclusivo.

Molte organizzazioni utilizzano Microsoft SignTool, uno strumento a riga di comando che firma, verifica e applica i timestamp sui file per semplificare il processo di firma del codice. È possibile utilizzare AWS CloudHSM per archiviare in modo sicuro coppie di chiavi finché non vengono richieste da SignToo, creando un flusso di lavoro facilmente automatizzabile per la firma dei dati.

I seguenti argomenti forniscono una panoramica di come utilizzare SignTool con AWS CloudHSM:

Argomenti

- Microsoft SignTool con AWS CloudHSM Fase 1: impostazione dei prerequisiti
- Microsoft SignTool con AWS CloudHSM Fase 2: crea un certificato di firma

- Microsoft SignTool con il AWS CloudHSM passaggio 3: firmare un file

Microsoft SignTool con AWS CloudHSM Fase 1: impostazione dei prerequisiti

Per usare Microsoft SignTool con AWS CloudHSM, è necessario:

- 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 cluster AWS CloudHSM attivo nello stesso cloud pubblico virtuale (VPC) come istanza di EC2. Il cluster deve contenere almeno un HSM.
- Un utente di crittografia (CU) che possiede e gestisce le chiavi del cluster AWS CloudHSM.
- Un file non firmato o eseguibile.
- Microsoft Windows Software Development Kit (SDK).

Per configurare i prerequisiti per l'utilizzo di 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 si desidera ospitare il proprio Windows Server CA, seguire i passaggi 1 e 2 in Configuring Windows Server as a Certificate Authority with AWS CloudHSM (Configurare Windows Server come autorità di certificazione). In caso contrario, continuare a utilizzare i CA di terze parti pubblicamente affidabili.
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 cluster AWS CloudHSM e il tuo CA per creare un certificato di firma.

Microsoft SignTool con AWS CloudHSM Fase 2: crea 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 un CSR utilizzando l'eseguibile **`certreq`**

1. Effettua la connessione all'istanza EC2 Windows, se non è stato ancora fatto. Per ulteriori informazioni, consulta la pagina Connessione a un'istanza nella Guida per l'utente di Amazon EC2 per le istanze Windows.
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, una nuova coppia di chiavi viene generata nel tuo cluster AWS CloudHSM, e la chiave privata della coppia viene utilizzata per creare il 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. Passare alla cartella Pending Requests (Richieste in sospeso) espandendo il menu CA server. Fai clic con il pulsante destro del mouse sulla richiesta creata e in All Tasks (Tutte le attività) scegliere Issue (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 file codificato DER X.509 in un percorso sicuro 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 .

Microsoft SignTool con il AWS CloudHSM passaggio 3: firmare un file

Ora sei pronto per l'uso SignTool e il certificato importato per firmare il tuo file di esempio. Per farlo, è necessario conoscere l'hash SHA-1 del certificato o l'identificazione personale. L'impronta

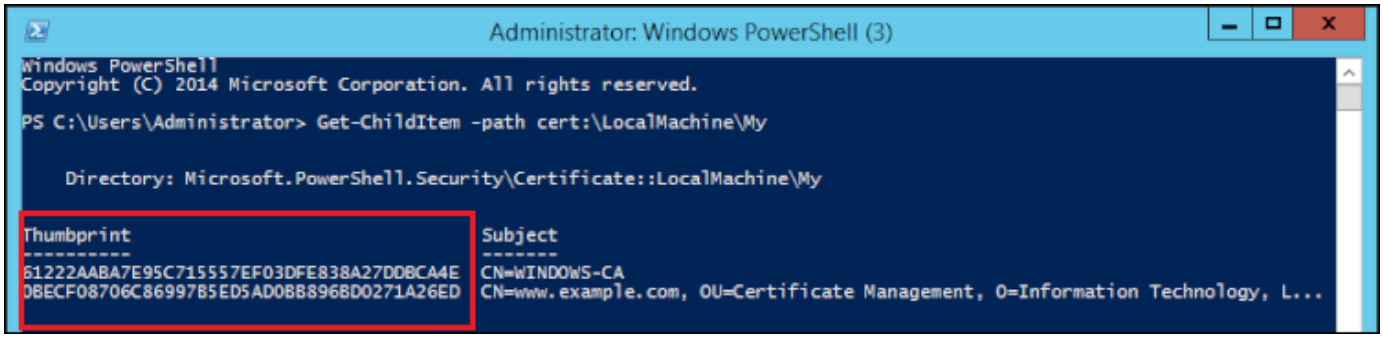
digitale viene utilizzata per garantire che vengano utilizzati SignTool solo certificati verificati da. AWS CloudHSM In questo esempio, lo usiamo PowerShell per ottenere l'hash del certificato. È inoltre possibile utilizzare l'interfaccia utente grafica del CA o l'SDK di Windows `certutil` eseguibile.

Per ottenere l'identificazione personale del certificato e utilizzarlo per firmare un file

1. Apri PowerShell come amministratore ed esegui il seguente comando:

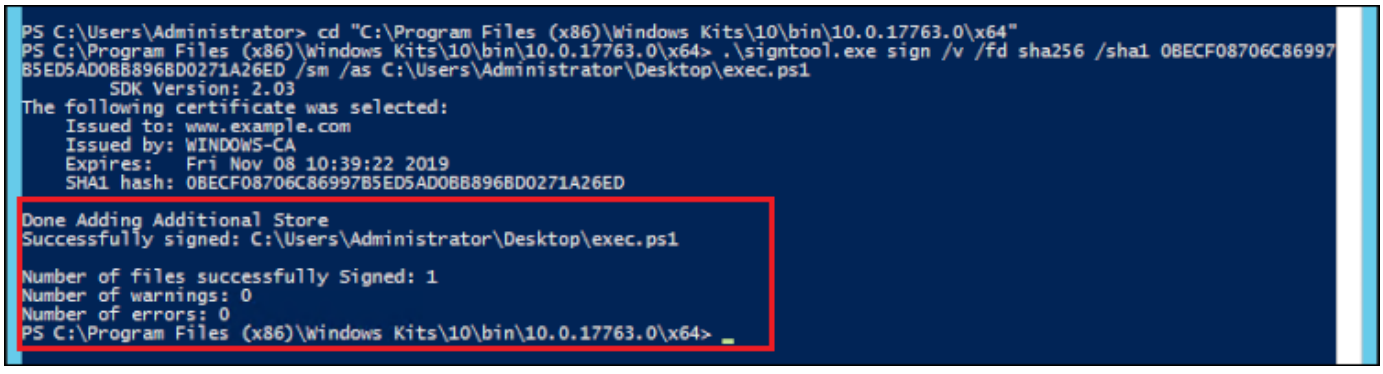
   ```
   Get-ChildItem -path cert:\LocalMachine\My
   ```

   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
```

Java Keytool e Jarsigner

AWS CloudHSM offre l'integrazione con le utilità Java Keytool e Jarsigner tramite Client SDK 3 e Client SDK 5. La procedura per utilizzare questi strumenti varia a seconda della versione dell'SDK del client del download attuale:

- Utilizzo di Client SDK 5 per l'integrazione con Java Keytool e Jarsigner
- Utilizzo di Client SDK 3 per l'integrazione con Java Keytool e Jarsigner

Utilizzo di Client SDK 5 per l'integrazione con Java Keytool e Jarsigner

AWS CloudHSM è un archivio di chiavi JCE per usi speciali che utilizza certificati associati alle chiavi dell'HSM tramite strumenti di terze parti quali `keytool` e `jarsigner`. AWS CloudHSM non memorizza i certificati in HSM, poiché i certificati sono dati pubblici e non riservati. L'archivio chiavi AWS CloudHSM archivia i certificati in un file locale e associa i certificati alle chiavi corrispondenti dell'HSM.

Quando si utilizza l'archivio chiavi AWS CloudHSM per generare nuove chiavi, non vengono generate voci nel file dell'archivio delle chiavi locale: le chiavi vengono create nell'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 certificati nell'archivio chiavi AWS CloudHSM, il provider verifica che una coppia di chiavi con l'alias corrispondente esista sull'HSM e quindi associa il certificato fornito alla coppia di chiavi corrispondente.

Argomenti

- Prerequisiti
- Utilizzo dell'archivio chiavi AWS CloudHSM con Keytool
- Utilizzo di Key Store AWS CloudHSM con Jarsigner
- Problemi noti

Prerequisiti

Per utilizzare l'archivio delle chiavi AWS CloudHSM, è necessario innanzitutto inizializzare e configurare l'SDK JCE AWS CloudHSM.

Fase 1: Installare JCE

Per installare JCE, inclusi i prerequisiti del client AWS CloudHSM, segui alla procedura per installare la 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**
>
> JCE AWS CloudHSM offre varie opzioni di accesso. Per utilizzare l'archivio chiavi AWS CloudHSM con applicazioni di terze parti, devi 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 chiavi AWS CloudHSM. Per ulteriori informazioni, vedi l'articolo sull'utilizzo dell'archivio chiavi AWS CloudHSM.

Passaggio 3: Registrazione del provider JCE

Per registrare il provider JCE nella CloudProvider configurazione Java, procedi nel seguente modo:

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 provider AWS CloudHSM come priorità più elevata può influire negativamente sulle prestazioni del sistema, poiché al provider AWS CloudHSM verrà assegnata la priorità per le operazioni che possono essere trasferite in sicurezza sul software. Come procedura ottimale, specifica sempre il provider che desideri utilizzare per un'operazione, indipendentemente dal fatto che si tratti del provider AWS CloudHSM o di un provider basato su software.

> **Note**
>
> L'indicazione delle opzioni `-providerName`, `-providerclass`, e `-providerpath` della riga di comando quando si generano le chiavi con keytool con l'archivio chiavi AWS CloudHSM può causare errori.

Utilizzo dell'archivio chiavi AWS CloudHSM con Keytool

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. In questo articolo vengono illustrati i parametri specifici da utilizzare con varie funzioni keytool quando si utilizza AWS CloudHSM come root per l'attendibilità attraverso l'archivio chiavi AWS CloudHSM.

Quando utilizzi keytool con l'archivio chiavi AWS CloudHSM, specifica i seguenti argomenti per qualsiasi comando keytool:

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 dell'archivio chiavi utilizzando l'archivio chiavi AWS CloudHSM, vedi Usare KeyStore AWS CloudHSM. Per utilizzare un archivio di chiavi esistente, specificane il nome (incluso il percorso) utilizzando l'argomento -keystore per keytool. Se specifichi un file di archivio chiavi inesistente in un comando keytool, l'archivio chiavi AWS CloudHSM crea un nuovo file di archivio chiavi.

Crea nuove chiavi con Keytool

È possibile utilizzare keytool per generare qualsiasi tipo di chiave supportato dall'SDK JCE di AWS CloudHSM.

> **Important**
>
> Una chiave generata tramite keytool viene generata nel software e quindi importata in AWS CloudHSM come chiave permanente estraibile.

Ti consigliamo vivamente di generare chiavi non esportabili al di fuori del keytool e quindi di importare i certificati corrispondenti nell'archivio chiavi. Se utilizzi chiavi RSA o EC estraibili tramite keytool e jarsigner, i provider esportano chiavi dal AWS CloudHSM e quindi utilizzano la chiave localmente per le operazioni di firma.

Se disponi di più istanze client connesse al cluster AWS 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 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: Per generare una chiave AES-256 simmetrica e salvarla in un file di archivio chiavi denominato "my_keystore.store", nella directory di lavoro. Sostituisci *<etichetta segreta>* con un'etichetta univoca.

Linux

```
$ keytool -genseckey -alias <secret label> -keyalg aes \
 -keysize 256 -keystore my_keystore.store \
 -storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
```

Windows

```
PS C:\> keytool -genseckey -alias <secret label> -keyalg aes `
 -keysize 256 -keystore my_keystore.store `
 -storetype CloudHSM -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Esempio 2: Per generare una coppia di chiavi RSA 2048 e salvarla in un file di archivio chiavi denominato "my_keystore.store" nella directory di lavoro. Sostituisci*<Etichetta coppia chiavi RSA>* con un'etichetta unica.

Linux

```
$ keytool -genkeypair -alias <RSA key pair label> \
 -keyalg rsa -keysize 2048 \
 -sigalg sha512withrsa \
 -keystore my_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 my_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.

Elimina una chiave usando Keytool

L'archivio chiavi AWS CloudHSM non supporta l'eliminazione delle chiavi. È possibile eliminare le chiavi utilizzando il metodo distruggi dell'interfaccia Distruggibile.

```
((Destroyable) key).destroy();
```

Generare una CSR usando Keytool

Ottieni la massima flessibilità nella generazione di una richiesta di firma del certificato (CSR) se utilizzi OpenSSL Dynamic Engine. Il comando seguente utilizza keytool per generare un CSR per una coppia di chiavi con l'alias, `my-key-pair`.

Linux

```
$ keytool -certreq -alias <key pair label> \
 -file my_csr.csr \
 -keystore my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -certreq -alias <key pair label> `
 -file my_csr.csr `
 -keystore my_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 “Utilizzo di Keytool per importare certificati intermedi e radice nell'archivio chiavi AWS CloudHSM”.

Utilizzo di Keytool per importare certificati intermedi e radice nell'archivio chiavi AWS CloudHSM

Per importare un certificato CA devi abilitare la verifica di una catena di certificati completa su un certificato appena importato. Il comando seguente mostra un esempio.

Linux

```
$ keytool -import -trustcacerts -alias rootCAcert \
 -file rootCAcert.cert -keystore my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -import -trustcacerts -alias rootCAcert `
 -file rootCAcert.cert -keystore my_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se si connettono più istanze di client al cluster AWS CloudHSM, l'importazione di un certificato nell'archivio chiavi di un'istanza del client non renderà automaticamente disponibile il certificato in altre istanze del client. È necessario importare il certificato in ogni istanza del client.

Utilizzo di Keytool per eliminare i certificati dall'archivio chiavi AWS CloudHSM

Il comando seguente mostra un esempio di come eliminare un certificato da un archivio chiavi keytool Java.

Linux

```
$ keytool -delete -alias mydomain \
 -keystore my_keystore.store \
 -storetype CLOUDHSM \
 -J-classpath '-J/opt/cloudhsm/java/*'
```

Windows

```
PS C:\> keytool -delete -alias mydomain `
 -keystore my_keystore.store `
 -storetype CLOUDHSM `
```

```
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Se si connettono più istanze del client al cluster AWS CloudHSM, l'eliminazione di un certificato nell'archivio chiavi di un'istanza del client non rimuoverà automaticamente il certificato da altre istanze del client. È necessario eliminare il certificato su ogni istanza del client.

Importazione di un certificato di lavoro nell'archivio chiavi AWS CloudHSM 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 my_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 my_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.

Esportazione di un certificato utilizzando Keytool

Nell'esempio seguente viene generato un certificato in formato binario X.509. Per esportare un certificato leggibile dall'uomo, aggiungere `-rfc` al comando `-exportcert`.

Linux

```
$ keytool -exportcert -alias <key pair label> \
 -file my_exported_certificate.crt \
 -keystore my_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 my_keystore.store `
 -storetype CLOUDHSM `
 -J-classpath '-J"C:\Program Files\Amazon\CloudHSM\java\*"'
```

Utilizzo di Key Store AWS CloudHSM con Jarsigner

Jarsigner è una popolare utilità a riga di comando per firmare file JAR utilizzando una chiave archiviata in modo sicuro su un HSM. Nella documentazione AWS CloudHSM non è previsto un tutorial completo su Jarsigner In questa sezione vengono illustrati i parametri di Jarsigner da utilizzare per firmare e verificare le firme con AWS CloudHSM come radice di fiducia attraverso l'archivio chiavi AWS CloudHSM.

Impostazione di chiavi e certificati

Prima di poter firmare i file 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. Imposta le chiavi di firma e i certificati associati e la catena di certificati che devono essere archiviati nell'archivio chiavi AWS CloudHSM dell'istanza del server o del client corrente. Crea le chiavi su AWS CloudHSM e quindi importa i metadati associati nell'archivio chiavi AWS CloudHSM. 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 all'archivio chiavi AWS CloudHSM. Vedere lo snippet di codice in the section called “Firmare un file JAR usando AWS CloudHSM e Jarsigner” 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 “Utilizzo di Keytool per importare certificati intermedi e radice nell'archivio chiavi AWS CloudHSM”.

Firmare un file JAR usando AWS CloudHSM e Jarsigner

Utilizza il seguente comando per firmare un file JAR:

Linux;

Per OpenJDK 8

```
jarsigner -keystore my_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>
```

Per OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -keystore my_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 OpenJDK8

```
jarsigner -keystore my_keystore.store `
```

```
 -signedjar signthisclass_signed.jar `
 -sigalg sha512withrsa `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*;C:\Program Files\Java
\jdk1.8.0_331\lib\tools.jar' `
  "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass.jar <key pair label>
```

Per OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -keystore my_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 OpenJDK8

```
jarsigner -verify \
 -keystore my_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>
```

Per OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -verify \
 -keystore my_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 8

```
jarsigner -verify `
 -keystore my_keystore.store `
 -sigalg sha512withrsa `
 -storetype CloudHSM `
 -J-classpath '-JC:\Program Files\Amazon\CloudHSM\java\*;C:\Program Files\Java
\jdk1.8.0_331\lib\tools.jar' `
 "-J-Djava.library.path='C:\Program Files\Amazon\CloudHSM\lib\'" `
 signthisclass_signed.jar <key pair label>
```

Per OpenJDK 11, OpenJDK 17 e OpenJDK 21

```
jarsigner -verify `
 -keystore my_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>
```

Problemi noti

1. Non supportiamo le chiavi EC con Keytool e Jarsigner.

Utilizzo di Client SDK 3 per l'integrazione con Java Keytool e Jarsigner

AWS CloudHSM è un archivio di chiavi JCE per uso speciale che utilizza certificati associati alle chiavi dell'HSM tramite strumenti di terze parti quali `keytool` e `jarsigner`. AWS CloudHSM non memorizza i certificati nell'HSM, poiché i certificati sono dati pubblici e non riservati. L'archivio chiavi

AWS CloudHSM archivia i certificati in un file locale e associa i certificati alle chiavi corrispondenti dell'HSM.

Quando si utilizza l'archivio chiavi AWS CloudHSM per generare nuove chiavi, non vengono generate voci nel file dell'archivio delle chiavi locale: le chiavi vengono create nell'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 certificati nell'archivio chiavi AWS CloudHSM, il provider verifica che una coppia di chiavi con l'alias corrispondente esista sull'HSM e quindi associa il certificato fornito alla coppia di chiavi corrispondente.

Argomenti

Prerequisiti

Per utilizzare l'archivio delle chiavi AWS CloudHSM, è necessario innanzitutto inizializzare e configurare l'SDK JCE AWS CloudHSM.

Fase 1: Installare JCE

Per installare JCE, inclusi i prerequisiti del client AWS CloudHSM, segui alla procedura per installare la 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 chiavi AWS CloudHSM con applicazioni di terze parti, devi 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 chiavi AWS CloudHSM. Per ulteriori informazioni, vedi l'articolo sull'utilizzo dell'archivio chiavi AWS CloudHSM.

Passaggio 3: Registrazione del provider JCE

Per registrare il provider JCE, nella CloudProvider configurazione Java.

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 quando si generano chiavi con l'archivio chiavi AWS CloudHSM, ciò causerà errori.

Utilizzo dell'archivio chiavi AWS CloudHSM con Keytool

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. In questo articolo vengono illustrati i parametri specifici da utilizzare con varie funzioni keytool quando si utilizza AWS CloudHSM come root per l'attendibilità attraverso l'archivio chiavi AWS CloudHSM.

Quando utilizzi keytool con l'archivio chiavi AWS CloudHSM, 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 dell'archivio chiavi utilizzando l'archivio chiavi AWS CloudHSM, vedi Usare KeyStore AWS CloudHSM. Per utilizzare un archivio di chiavi esistente, specificane il nome (incluso il percorso) utilizzando l'argomento -keystore per keytool. Se specifichi un file di archivio chiavi inesistente in un comando keytool, l'archivio chiavi AWS CloudHSM crea un nuovo file di archivio chiavi.

Crea nuove chiavi con Keytool

È possibile utilizzare keytool per generare qualsiasi tipo di chiave supportato da SDK JCE di AWS CloudHSM. 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 in AWS CloudHSM come chiave permanente estraibile.

Le istruzioni per creare chiavi non estraibili direttamente sull'HSM e quindi utilizzarle con keytool o Jarsigner, sono mostrate nel codice di esempio in Registrazione delle chiavi preesistenti con l'archivio chiavi 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 chiavi da AWS CloudHSM e quindi utilizzano la chiave localmente per le operazioni di firma.

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 un 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: Per generare una chiave AES-256 simmetrica e salvarla in un file di archivio chiavi denominato "my_keystore.store", nella directory di lavoro. Sostituisci *<etichetta segreta>* con un'etichetta univoca.

```
keytool -genseckey -alias <secret label> -keyalg aes \
  -keysize 256 -keystore my_keystore.store \
  -storetype CloudHSM -J-classpath '-J/opt/cloudhsm/java/*' \
  -J-Djava.library.path=/opt/cloudhsm/lib/
```

Esempio 2: Per generare una coppia di chiavi RSA 2048 e salvarla in un file di archivio chiavi denominato "my_keystore.store" nella directory di lavoro. Sostituisci*<etichetta della coppia di chiavi RSA>* con un'etichetta unica.

```
keytool -genkeypair -alias <RSA key pair label> \
        -keyalg rsa -keysize 2048 \
        -sigalg sha512withrsa \
        -keystore my_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Esempio 3: Per generare una chiave ED p256 e salvarla in un file di archivio chiavi denominato "my_keystore.store" nella directory di lavoro. Sostituisci*<Etichetta coppia chiavi ec>* con un'etichetta unica.

```
keytool -genkeypair -alias <ec key pair label> \
        -keyalg ec -keysize 256 \
        -sigalg SHA512withECDSA \
        -keystore my_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 chiave usando Keytool

L'archivio chiavi AWS CloudHSM non supporta l'eliminazione delle chiavi. Per eliminare la chiave, è necessario utilizzare la funzione `deleteKey` dello strumento a riga di comando di AWS CloudHSM, deleteKey.

Generare un CSR usando Keytool

Ottieni la massima flessibilità nella generazione di una richiesta di firma del certificato (CSR) se utilizzi OpenSSL Dynamic Engine. Il comando seguente utilizza keytool per generare un CSR per una coppia di chiavi con l'alias, `my-key-pair`.

```
keytool -certreq -alias <key pair label> \
        -file my_csr.csr \
        -keystore my_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, vedi Importazione dei certificati intermedi e radice nell'archivio chiavi AWS CloudHSM utilizzando Keytool.

Utilizzo di Keytool per importare certificati intermedi e radice nell'archivio chiavi AWS CloudHSM

Per importare un certificato CA devi abilitare la verifica di una catena di certificati completa su un certificato appena importato. Il comando seguente mostra un esempio.

```
keytool -import -trustcacerts -alias rootCAcert \
        -file rootCAcert.cert -keystore my_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Se si connettono più istanze di client al cluster AWS CloudHSM, l'importazione di un certificato nell'archivio chiavi di un'istanza del client non renderà automaticamente disponibile il certificato in altre istanze del client. È necessario importare il certificato in ogni istanza del client.

Utilizzo di Keytool per eliminare i certificati dall'archivio chiavi AWS CloudHSM

Il comando seguente mostra un esempio di come eliminare un certificato da un archivio chiavi keytool Java.

```
keytool -delete -alias mydomain -keystore \
        -keystore my_keystore.store \
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Se si connettono più istanze del client al cluster AWS CloudHSM, l'eliminazione di un certificato nell'archivio chiavi di un'istanza del client non rimuoverà automaticamente il certificato da altre istanze del client. È necessario eliminare il certificato su ogni istanza del client.

Importazione di un certificato di lavoro nell'archivio chiavi AWS CloudHSM 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 my_certificate.crt \
        -keystore my_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, vedo il codice di esempi in[Registrazione delle chiavi preesistenti nell'archivio chiavi 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.

Esportazione di un certificato utilizzando Keytool

Nell'esempio seguente viene generato un certificato in formato binario X.509. Per esportare un certificato leggibile dall'uomo, aggiungere `-rfc` al comando `-exportcert`.

```
keytool -exportcert -alias <key pair label> \
        -file my_exported_certificate.crt \
        -keystore my_keystore.store \
```

```
        -storetype CLOUDHSM \
        -J-classpath '-J/opt/cloudhsm/java/*' \
        -J-Djava.library.path=/opt/cloudhsm/lib/
```

Utilizzo di Key Store AWS CloudHSM con Jarsigner

Jarsigner è una popolare utilità a riga di comando per firmare file JAR utilizzando una chiave archiviata in modo sicuro su un HSM. Nella documentazione AWS CloudHSM non è previsto un tutorial completo su Jarsigner In questa sezione vengono illustrati i parametri di Jarsigner da utilizzare per firmare e verificare le firme con AWS CloudHSM come radice di fiducia attraverso l'archivio chiavi AWS CloudHSM.

Impostazione di chiavi e certificati

Prima di poter firmare i file JAR con Jarsigner, assicurati di aver impostato o completato i seguenti passaggi:

1. Seguire le indicazioni contenute nei prerequisiti dell'archivio chiavi AWS CloudHSM.
2. Imposta le chiavi di firma e i certificati associati e la catena di certificati che devono essere archiviati nell'archivio chiavi AWS CloudHSM dell'istanza del server o del client corrente. Crea le chiavi su AWS CloudHSM e quindi importa i metadati associati nell'archivio chiavi AWS CloudHSM. Utilizzare il codice di esempio nella registrazione delle chiavi preesistenti con l'archivio chiavi AWS CloudHSM per importare i metadati nell'archivio chiavi. Se desideri utilizzare keytool per impostare le chiavi e i certificati, vedi 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 all'archivio chiavi AWS CloudHSM. Vedere lo snippet di codice in Firmare un file JAR utilizzando AWS CloudHSM e Jarsigner per istruzioni sull'utilizzo del codice Java per verificare la catena di certificati. Se preferisci, puoi utilizzare keytool per importare i certificati. Per istruzioni sull'utilizzo di keytool, vedere Utilizzo di Keytool per importare certificati intermedi e radice nell'archivio chiavi AWS CloudHSM.

Firmare un file JAR usando AWS CloudHSM e Jarsigner

Utilizza il seguente comando per firmare un file JAR:

```
jarsigner -keystore my_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 my_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

Nell'elenco seguente viene fornito l'elenco corrente dei problemi noti.

- 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'HSM AWS CloudHSM come chiave persistente (token) durante il processo di aggiunta della chiave.
- Quando si utilizza keytool con archivio chiavi AWS CloudHSM, non specificare le opzioni `-providerName`, `-providerclass` o `-providerpath` sulla riga di comando. 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 essere rimosso/disabilitato dall'elenco dei provider nel file java.security. Se si utilizzano chiavi EC estraibili tramite keytool e Jarsigner, i provider esportano i bit chiave dall'HSM AWS CloudHSM e utilizzano la chiave localmente per le operazioni di firma. Si sconsiglia di utilizzare chiavi esportabili con keytool o Jarsigner.

Registrazione di chiavi preesistenti con Key Store AWS CloudHSM

Per la massima sicurezza e flessibilità negli attributi e nell'etichettatura, ti consigliamo di generare le chiavi di firma utilizzando key_mgmt_util. È inoltre possibile utilizzare un'applicazione Java per generare la chiave in AWS CloudHSM.

Nella sezione seguente viene fornito un codice di esempio che illustra come generare una nuova coppia di chiavi in HSM e registrarla utilizzando chiavi esistenti importate nell'archivio chiavi AWS CloudHSM. 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 cercare una chiave per etichetta è disponibile nell'KeyUtilitiesRunneresempio.java on. GitHub

> **Important**
>
> La registrazione di una chiave AWS CloudHSM memorizzata in un archivio chiavi locale non esporta la 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 alises 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());
```

```
        }
    }

}
```

Altre integrazioni di fornitori di terze parti

Diversi fornitori di terze parti supportano AWS CloudHSM come radice di attendibilità. 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 tuo carico di lavoro in AWS può fare affidamento sui benefici di latenza, disponibilità, affidabilità ed elasticità di CloudHSM. L'elenco seguente include i fornitori di terze parti che supportano CloudHSM.

> **Note**
>
> AWS non promuove né garantisce fornitori di terze parti.

- 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 come radici di attendibilità per ulteriore protezione.
- Thycotic Secrets Server consente ai clienti di gestire credenziali sensibili su account privilegiati. Supporta AWS CloudHSM come radice di attendibilità.
- L'adattatore KMIP di P6R consente di utilizzare le proprie istanze AWS CloudHSM tramite un'interfaccia KMIP standard.
- PrimeKey EJBCA è una soluzione open source comune per PKI. Consente di creare e archiviare coppie di chiavi in modo sicuro con AWS CloudHSM.
- Casella KeySafe fornisce la gestione delle chiavi di crittografia per i contenuti del cloud a molte organizzazioni con requisiti di sicurezza, privacy e di conformità alle normative rigidi. I clienti possono proteggere ulteriormente le chiavi KeySafe direttamente in AWS Key Management Service o indirettamente in AWS CloudHSM tramite AWS KMS Custom Key Store.
- Insyde Software supporta AWS CloudHSM come radice di attendibilità per la firma del firmware.
- F5 BIG-IP LTM supporta AWS CloudHSM come una radice di attendibilità.

- 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 di AWS CloudHSM

Oltre alle funzionalità di log integrate nel Client SDK, puoi anche utilizzare AWS CloudTrail, File di log Amazon CloudWatch e Amazon CloudWatch per il monitoraggio dell'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

Usa CloudTrail per monitorare tutte le chiamate API nel tuo account AWS, incluse le chiamate effettuate per creare ed eliminare cluster, moduli di sicurezza hardware (HSM) e tag di risorse.

CloudWatch Logs

Usa CloudWatch Logs per monitorare i log delle tue istanze HSM, che includono eventi per creare ed eliminare utenti HSM, modificare le password degli utenti, creare ed eliminare chiavi e altro ancora.

CloudWatch

Utilizza CloudWatch per monitorare lo stato del cluster in tempo reale.

Argomenti

Utilizzo dei log SDK del client

È possibile recuperare i log generati dall'SDK del client. AWS CloudHSM offre un'implementazione del logging con Client SDK 3 e Client SDK 5.

Argomenti

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:

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
```

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 daemon del client AWS CloudHSM. La posizione dei log dipende dal sistema operativo dell'istanza del client Amazon EC2 su cui viene eseguito il daemon del client.

Amazon Linux

In Amazon Linux, i log del client AWS CloudHSM sono 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 del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

CentOS 7

In CentOS 7, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

CentOS 8

In CentOS 8, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

RHEL 7

In Red Hat Enterprise Linux 7, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

RHEL 8

In Red Hat Enterprise Linux 8, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 16.04

In Ubuntu 16.04, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

Ubuntu 18.04

In Ubuntu 18.04, i log del client AWS CloudHSM sono raccolti e archiviati nel registro. È possibile utilizzare journalctl per visualizzare e gestire tali log. Ad esempio, utilizza il comando seguente per visualizzare i log del client AWS CloudHSM.

```
journalctl -f -u cloudhsm-client
```

Windows

- Per client Windows dalla versione 1.1.2:

  I log del client AWS CloudHSM vengono scritti su un file `cloudhsm.log` nella cartella dei file di programma AWS CloudHSM (`C:\Program Files\Amazon\CloudHSM\`). A ogni nome di file di log viene aggiunto un suffisso con un timestamp che indica quando il client AWS CloudHSM è stato avviato.

- Per Windows client 1.1.1 e versioni precedenti:

  Il log del client non vengono scritti su un file. I log vengono visualizzati nella finestra del prompt dei comandi o di PowerShell in cui è stato avviato il client AWS CloudHSM.

Uso di AWS CloudTrail e AWS CloudHSM

AWS CloudHSM è integrato con AWS CloudTrail, un servizio che offre un record delle operazioni eseguite da un utente, un ruolo o un servizio AWS in AWS CloudHSM. CloudTrail acquisisce tutte le chiamate API AWS CloudHSM come eventi. Le chiamate acquisite includono le chiamate dalla console di AWS CloudHSM e le chiamate di codice alle operazioni delle API AWS CloudHSM.
Se si crea un trail, è possibile abilitare la distribuzione continua di eventi CloudTrail in un bucket Amazon S3, inclusi gli eventi per AWS CloudHSM. Se non si configura un trail, è comunque possibile visualizzare gli eventi più recenti nella console di CloudTrail in Event history (Cronologia eventi).
Le informazioni raccolte da CloudTrail consentono di determinare la richiesta effettuata ad AWS CloudHSM, l'indirizzo IP da cui è partita la richiesta, l'autore della richiesta, il momento in cui è stata eseguita e altri dettagli.

Per ulteriori informazioni su CloudTrail, vedi la Guida per l'utente di AWS CloudTrail. Per un elenco delle operazioni API di AWS CloudHSM, vedi Azioni nel Riferimento API AWS CloudHSM.

Informazioni su CloudTrail AWS CloudHSM

CloudTrail è abilitato sull'account AWS al momento della sua creazione. Quando si verifica un'attività in AWS CloudHSM, tale attività viene registrata in un evento CloudTrail insieme ad altri eventi di servizio AWS nella Cronologia eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS. Per ulteriori informazioni, vedi Visualizzazione di eventi nella cronologia degli eventi di CloudTrail.

Per una registrazione continua degli eventi nell'account AWS che includa gli eventi per AWS CloudHSM, crea un trail. Un percorso abilita la distribuzione da parte di CloudTrail dei file di log in 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 nella partizione AWS e distribuisce i file di registro nel bucket Amazon S3 specificato. Inoltre, è possibile configurare altri servizi AWS per analizzare con maggiore dettaglio e usare i dati evento raccolti nei log CloudTrail. Per ulteriori informazioni, consulta gli argomenti seguenti:

- Panoramica della creazione di un percorso

- Servizi e integrazioni CloudTrail supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di log CloudTrail da più regioni e Ricezione di file di log CloudTrail da più account

CloudTrail registra tutte le operazioni AWS CloudHSM, incluse operazioni di sola lettura, ad esempio `DescribeClusters` e `ListTags`, nonché le operazioni di gestione, ad esempio `InitializeCluster`, `CreatHsm` e `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 servizio AWS.

Per ulteriori informazioni, vedi Elemento userIdentity di CloudTrail.

Comprensione delle voci dei file di log di AWS CloudHSM

Un trail è una configurazione che consente la distribuzione di eventi come i file di log in un bucket Amazon S3 specificato. I file di log di CloudTrail possono contenere una o più voci di log. Un evento rappresenta una singola richiesta da un'origine e include informazioni sull'operazione richiesta, sulla data e sull'ora dell'operazione, sui parametri richiesti e così via. I file di log CloudTrail non sono una traccia di pila ordinata delle chiamate API pubbliche e di conseguenza non devono apparire in base a un ordine specifico.

L'esempio seguente mostra una voce di log di CloudTrail che illustra l'azione AWS CloudHSM`CreateHsm`.

```
{
    "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"
}
```

Utilizzo dei file di log Amazon CloudWatch e Audit Logs AWS CloudHSM

Quando un HSM nel tuo account riceve un comando dagli strumenti a riga di comandoAWS CloudHSM o dalle librerie software, registra l'esecuzione del comando in un modulo di log dell'audit. 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 audit di HSM e li invia ai File di log Amazon CloudWatch a tuo nome. È possibile utilizzare le funzioni di CloudWatch Logs per gestire i tuoi log di audit AWS CloudHSM, tra cui le operazioni di ricerca e di filtraggio dei log e l'esportazione dei dati di log su Amazon S3. Puoi lavorare con i tuoi log di audit HSM nella Console di Amazon CloudWatch o utilizzare i comandi CloudWatch Logs nella AWS CLI e nei File di Log CloudWatch di SDK.

Argomenti

Funzionamento del log degli audit dell'HSM

Il log degli audit è automaticamente abilitato in tutti i cluster AWS CloudHSM. Non può essere disabilitato o disattivato e nessuna impostazione può impedire a AWS CloudHSM di esportare i log su CloudWatch Logs. 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. I log di audit dei diversi HSM, anche quelli appartenenti allo stesso cluster, potrebbero variare. Ad esempio, solo il primo HSM in ogni cluster registra l'inizializzazione dell'HSM. Gli eventi di inizializzazione non appaiono nei log di moduli HSM clonati da backup. Analogamente, quando si crea una chiave, l'HSM che genera la chiave registra un evento di generazione della chiave. Gli altri moduli HSM nel cluster registrano un evento quando ricevono la chiave tramite sincronizzazione.

AWS CloudHSM raccoglie i log e li pubblica su nel tuo account CloudWatch Logs. Per comunicare con il servizio CloudWatch Logs a tuo nome, AWS CloudHSM utilizza un ruolo legato al servizio. La

policy IAM associata al ruolo consente a AWS CloudHSM di eseguire solo le attività richieste per inviare i log dell'audit a CloudWatch Logs.

> **Important**
>
> Se hai creato un cluster prima del 20 gennaio 2018 ma non un ruolo collegato al servizio, devi crearne uno manualmente. Questo è necessario affinché CloudWatch possa ricevere i log di audit 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 dei log di audit dell'HSM in CloudWatch Logs

File di log Amazon CloudWatch organizza i log di audit in gruppi di log e, all'interno di un gruppo di log, in flussi di log.. Ogni voce di log è un evento. AWS CloudHSM crea un gruppo di log per ogni cluster e un flusso di log per ogni HSM nel cluster. Non è necessario creare componenti CloudWatch Logs o modificare le impostazioni.

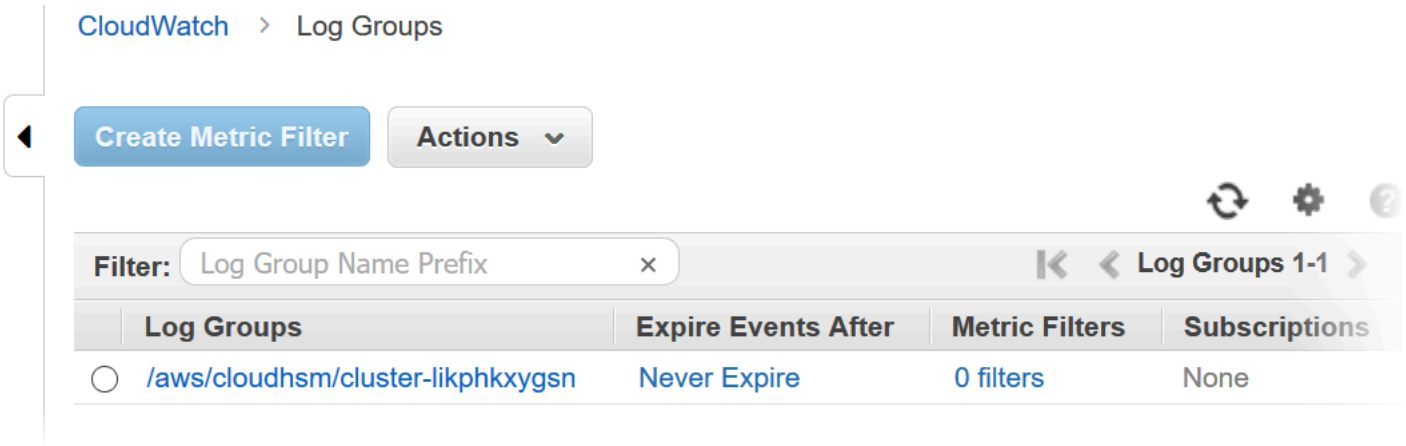
- Il nome del gruppo di log è `/aws/cloudhsm/<cluster ID>`, ad esempio `/aws/cloudhsm/cluster-likphkxygsn`. Quando utilizzi il nome del gruppo di log in un comando AWS CLI o PowerShell, assicurati di includerli tra virgolette doppie.
- Il nome del flusso di log è l'ID HSM; ad esempio, `hsm-nwbbiqbj4jk`.

  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.

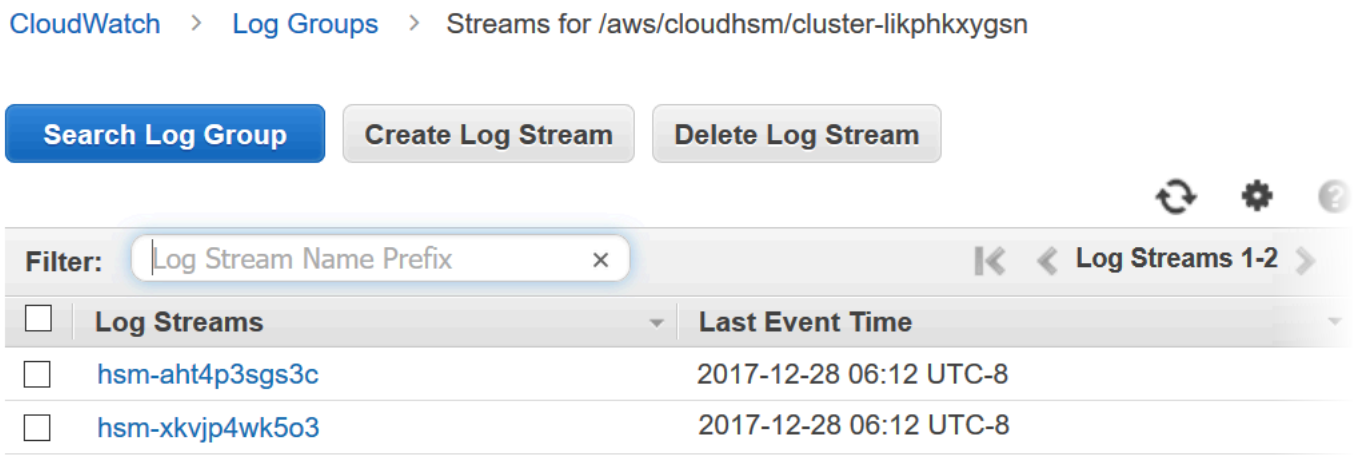
Per informazioni sui concetti relativi a CloudWatch, vedi Concetti nella Guida per l'utente di File di log Amazon CloudWatch.

È possibile visualizzare i log di audit per un modulo HSM dalla pagina CloudWatch Logs in AWS Management Console, i comandi CloudWatch Logs nella AWS CLI, i CloudWatch Logs cmdlet PowerShell o nei SDK CloudWatch Logs. Per ulteriori informazioni, vedi Visualizzazione dei dati di log nella Guida per l'utente di File di log Amazon CloudWatch.

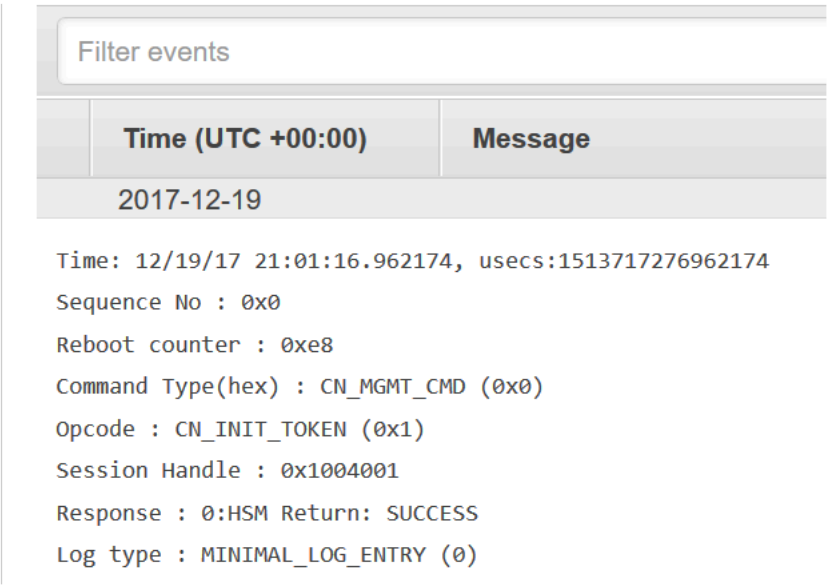
Ad esempio, di seguito è riportata l'immagine del gruppo di log per il cluster `cluster-likphkxygsn` nel AWS Management Console.



Quando si sceglie il nome del gruppo di log del cluster, è possibile visualizzare il flusso di log per ciascun modulo HSM del cluster. L'immagine seguente mostra i flussi di log per gli HSM nel cluster `cluster-likphkxygsn`.

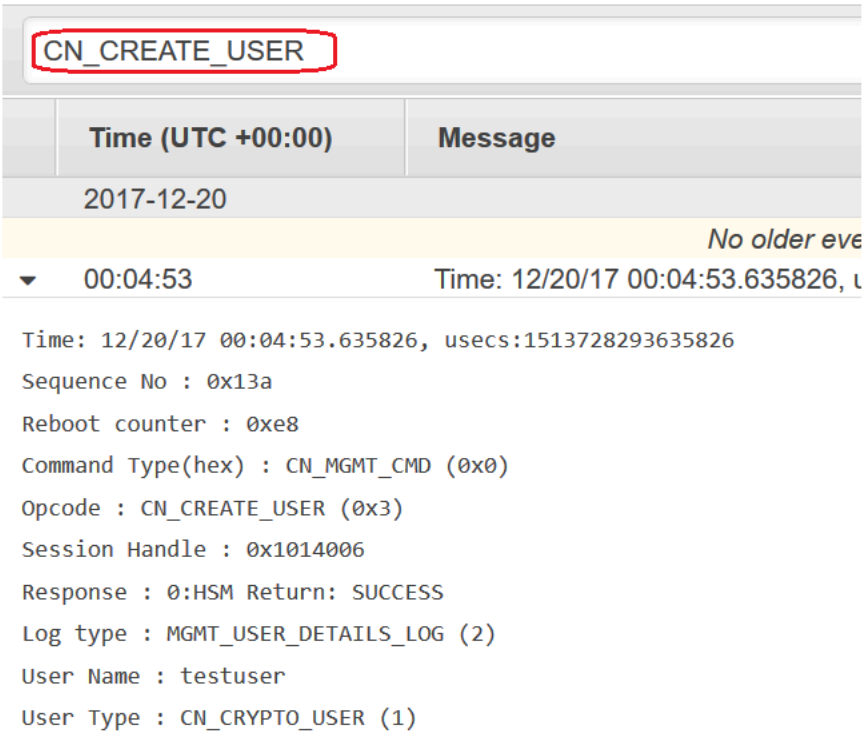


Quando si sceglie un nome per il flusso di log HSM, è possibile visualizzare gli eventi nel log di audit. Ad esempio, questo evento, che ha un numero di sequenza 0x0 e un `Opcode` di `CN_INIT_TOKEN`, è in genere il primo evento per il primo HSM in ogni cluster. Registra l'inizializzazione dell'HSM nel cluster.



Puoi utilizzare tutte le numerose funzionalità in CloudWatch Logs per gestire i log di audit. 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 log di audit HSM

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 campi seguenti sono comuni a ogni evento AWS CloudHSM nel log di audit.

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 valori `Opcode` nei log di audit AWS CloudHSM, vedi Riferimento dei log di audit.

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 del log AWS CloudHSM 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 log per esaminare il ciclo di vita dei moduli di sicurezza hardware e comprenderne meglio il 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

Il flusso di log di audit per il primo HSM in ogni cluster differisce in modo significativo dai flussi di log di altri moduli HSM nel cluster. Il log di audit per il primo HSM in ogni cluster registra la propria creazione e inizializzazione. I log di HSM aggiuntivi nel cluster, generati da backup, cominciano con un evento di accesso.

> **Important**
>
> Le seguenti voci di inizializzazione non appariranno nei log CloudWatch di cluster inizializzati prima del rilascio della funzionalità di log dell'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 gli HSM successivi nello stesso cluster sono cloni del primo, utilizzano la stessa chiave di crittografia della 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 gli altri moduli HSM nel cluster registrano un evento di accesso per l'utente. Puoi trovare gli eventi di login corrispondenti nei flussi di log di altri HSM nel cluster poco dopo l'evento di login 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 presenta 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 illustra l'effetto della creazione di una chiave in un cluster con più moduli HSM. La chiave è generata su un HSM, estratta dall'HSM come oggetto mascherato e inserita negli altri HSM come oggetto mascherato.

> **Note**
>
> Gli strumenti del client potrebbero non riuscire a sincronizzare la chiave. Oppure il comando potrebbe includere il parametro min_srv, che consente di sincronizzare la chiave solo per il numero specificato di moduli HSM. In entrambi i casi, il servizio AWS CloudHSM sincronizza la chiave per gli altri moduli HSM nel cluster. Poiché i moduli HSM registrano nei propri log solo i comandi di gestione lato client, la sincronizzazione lato server non viene registrata nel log dell'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)
```

Il CU esegue un comando exSymKey per generare una chiave simmetrica. 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 i moduli HSM 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 log di audit che vengono registrati quando un crypto user (CU) esporta le chiavi da un cluster con più moduli HSM.

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)
```

Il CU esegue un comando exSymKey per esportare la chiave 7, una chiave AES a 256 bit. Il comando utilizza una chiave 6, una chiave AES a 256 bit sui moduli HSM, come chiave di wrapping.

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 HSM nel cluster.

Esempio: Importare una chiave

Questo esempio illustra gli eventi dei log di audit che vengono registrati al momento dell'importazione di chiavi nei moduli HSM di un cluster. In questo esempio, il crypto user (CU) utilizza il comando imSymKey per importare una chiave AES nei moduli HSM. 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 creata o importata una nuova chiave, gli strumenti del client tentano automaticamente di sincronizzare la nuova chiave con altri moduli HSM nel 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 di altri moduli HSM nel 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)
```

Riferimento dei log di audit

AWS CloudHSM registra i comandi di gestione HSM negli eventi del registro di controllo. Ogni evento ha un valore di codice operativo (`0pcode`) che identifica l'operazione avvenuta e la relativa risposta. È possibile utilizzare i valori `0pcode` per cercare, ordinare e filtrare i log.

La tabella seguente definisce i `0pcode` valori in un registro di AWS CloudHSM controllo.

Codice operazione (Opcode)	Descrizione
Login utente: questi eventi includono il nome utente e il tipo di utente	
`CN_LOGIN (0xd)`	Login utente
`CN_LOGOUT (0xe)`	Disconnessione 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 crypto user (CU)
`CN_CREATE_CO`	Creazione di un crypto officer (CO)
`CN_DELETE_USER`	Eliminazione di un utente
`CN_CHANGE_PSWD`	Modifica della password di un utente
`CN_SET_M_VALUE`	Set autenticazione del quorum (M of N) for a user action
`CN_APPROVE_TOKEN`	Approve a autenticazione del quorum token for a user action
`CN_DELETE_TOKEN`	Delete one or more token del quorum
`CN_GET_TOKEN`	Request a signing token to initiate a funzionamento del quorum
Gestione chiavi: questi eventi includono l'handle della chiave.	
`CN_GENERATE_KEY`	Generazione di una chiave simmetrica

Codice operazione (Opcode)	Descrizione
`CN_GENERATE_KEY_PAIR (0x19)`	Generate an asymmetric key pair
`CN_CREATE_OBJECT`	Import a public key (without wrapping)
`CN_MODIFY_OBJECT`	Set a key attribute
`CN_DESTROY_OBJECT (0x11)`	Deletion of a chiave di sessione
`CN_TOMBSTONE_OBJECT`	Deletion of a chiave token
`CN_SHARE_OBJECT`	Condivisione e annullamento della condivisione di una chiave
`CN_WRAP_KEY`	Export an encrypted copy of a key (wrapKey)
`CN_UNWRAP_KEY`	Import an encrypted copy of a key (Annullamento wrap chiave)
`CN_DERIVE_KEY`	Derive a symmetric key from an existing key
`CN_NIST_AES_WRAP`	Crittografa o decrittografa una chiave con una chiave AES
`CN_INSERT_MASKED_OBJECT_USER`	Insert an encrypted key with attributes from another HSM in the cluster.
`CN_EXTRACT_MASKED_OBJECT_USER`	Wraps/encrypts a key with attributes from the HSM to be sent to another HSM in the cluster.
Back up HSMs	
`CN_BACKUP_BEGIN`	Begin the backup process
`CN_BACKUP_END`	Completed the backup process
`CN_RESTORE_BEGIN`	Begin restoring from a backup
`CN_RESTORE_END`	Completed the restoration process from a backup

Codice operazione (Opcode)	Descrizione
Certificate-Based Authentication	
`CN_CERT_AUTH_STORE_CERT`	Stores the cluster certificate
HSM Instance Commands	
`CN_INIT_TOKEN (0x1)`	Start the HSM initialization process
`CN_INIT_DONE`	The HSM initialization process has finished
`CN_GEN_KEY_ENC_KEY`	Generate a key encryption key (KEK)
`CN_GEN_PSWD_ENC_KEY (0x1d)`	Generate a password encryption key (PEK)
HSM crypto commands	
`CN_FIPS_RAND`	Generate a FIPS-compliant random number

Visualizzazione dei parametri di CloudWatch per AWS CloudHSM

Utilizza CloudWatch per monitorare il cluster AWS CloudHSM 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:

Parametro	Descrizione
HsmUnhealthy	L'istanza HSM non viene eseguita correttamente. AWS CloudHSM sostituis ce automaticamente le istanze non integre per conto tuo. Puoi scegliere di espandere la dimensione del cluster in modo proattivo per ridurre l’impatto delle prestazioni durante la sostituzione dell'HSM.
HsmTemper ature	Temperatura di collegamento del processore hardware. Se la temperatura raggiunge 110 gradi centigradi, il sistema si arresta.
HsmKeysSe ssionOccupied	Numero di chiavi di sessione utilizzate dall'istanza HSM.

Parametro	Descrizione
HsmKeysTokenOccupied	Numero di chiavi token utilizzate dall'istanza HSM e dal cluster.
HsmSslCtxsOccupied	Numero di canali con crittografia end-to-end 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 del client è configurato per aprire due sessioni con ciascuna istanza HSM in un canale con crittografia end-to-end. AWS CloudHSM può avere inoltre fino a 2 connessioni aperte con l'HSM per monitorare l'integrità degli HSM.
HsmUsersAvailable	Numero di utenti aggiuntivi che è possibile creare. È uguale al numero massimo di utenti (indicato in HsmUsersMax) meno il numero di utenti creati finora.
HsmUsersMax	Numero massimo di utenti che è possibile creare nell'istanza HSM. Attualmente tale numero è pari a 1024.
Interface Eth2OctetsInput	La somma cumulativa del traffico in entrata verso l'HSM finora.
Interface Eth2OctetsOutput	La somma cumulativa del traffico in uscita verso l'HSM finora.

AWS CloudHSM Prestazioni

Configurazione cluster: per i cluster di produzione, si devono avere almeno due istanze HSM distribuite su 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. La tabella seguente mostra le prestazioni generali degli algoritmi crittografici più comuni eseguiti su un'istanza EC2. 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.

Operazione	Cluster a due HSM [1]	Cluster a tre HSM [2]	Cluster a sei HSM [3]
Segno RSA a 2048 bit	2.000 operazioni/sec	3.000 operazioni/sec	5.000 operazioni/sec
Segno EC P256	500 operazioni/sec	750 operazioni/sec	1.500 operazioni/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.

HSM limitato

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 (della larghezza di banda della rete) HSM

Sicurezza in AWS CloudHSM

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. 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 nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. I revisori esterni testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito dei AWS Programmi diAWS conformità dei Programmi di conformità dei di . Per ulteriori informazioni sui programmi di conformità applicabili AWS CloudHSM, consulta AWS Services in Scope by Compliance Program .
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della tua 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. I seguenti argomenti mostrano come eseguire la configurazione AWS CloudHSM per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a usare altri servizi AWS che ti aiutano a monitorare e proteggere AWS CloudHSM le tue risorse.

Indice

Protezione dei dati in AWS CloudHSM

Il modello di responsabilità AWS condivisa modello di di 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. Inoltre, sei responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS che utilizzi. Per ulteriori informazioni sulla privacy dei dati, vedi Domande frequenti sulla privacy dei dati. Per informazioni sulla protezione dei dati in Europa, consulta il post del blog AWS Shared Responsibility Model and GDPR nel Blog sulla sicurezzaAWS .

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center or AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con le risorse. AWS È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail
- 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 140-2 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) 140-2.

Ti consigliamo vivamente 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 o Servizi AWS utilizzi la console, l'API AWS CloudHSM 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 fornisci un URL a un server esterno, ti suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la tua 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 CloudHSMI dati vengono crittografati utilizzando una chiave di crittografia unica e temporanea. Per ulteriori informazioni, consulta Backup del cluster AWS CloudHSM.

Crittografia dei dati in transito

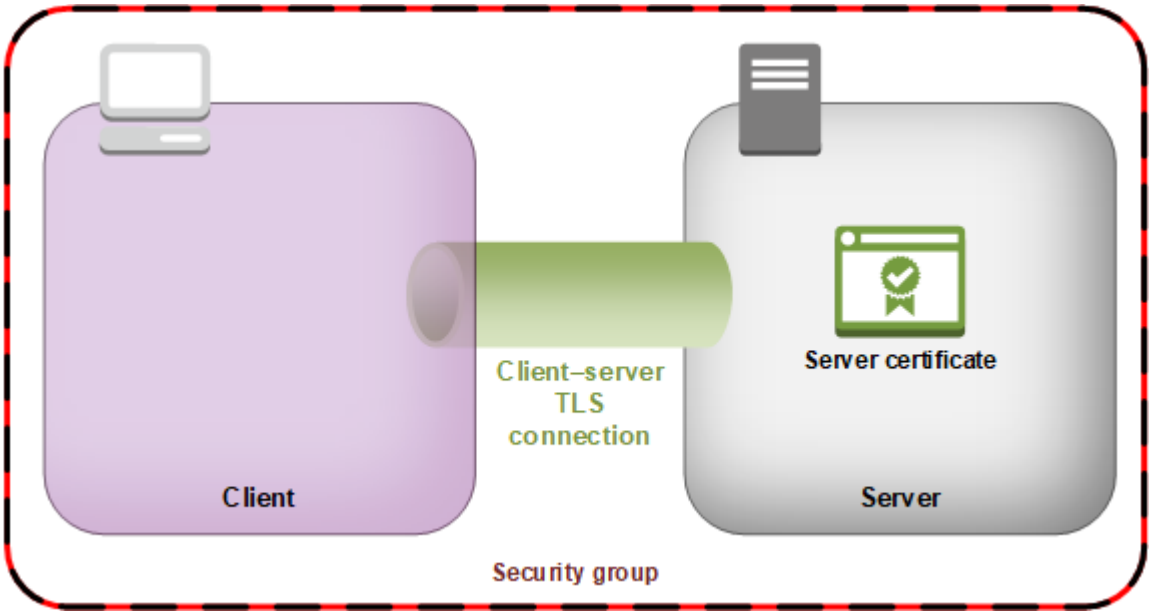
La comunicazione tra il AWS CloudHSM client e l'HSM del cluster è crittografata da un capo all'altro. Questa comunicazione può essere decifrata solo dal client e dagli HSM. Per ulteriori informazioni, consulta nd-to-end crittografia E.

AWS CloudHSM crittografia del client end-to-end

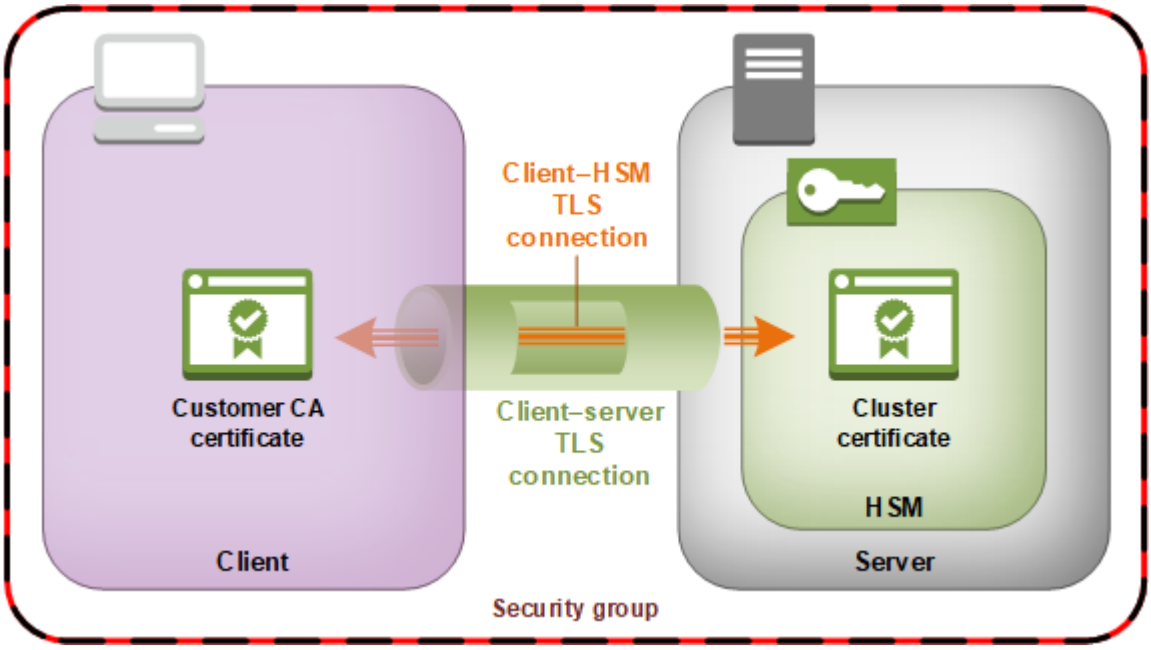
La comunicazione tra il client e l'HSM nel tuo cluster è crittografata end-to-end. Solo il tuo client e il tuo HSM possono decifrarla.

Il seguente processo spiega come il client stabilisce una comunicazione end-to-end crittografata 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, l'HSM crittografa tutti i dati prima di inviarli a. AWS CloudHSMI 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.

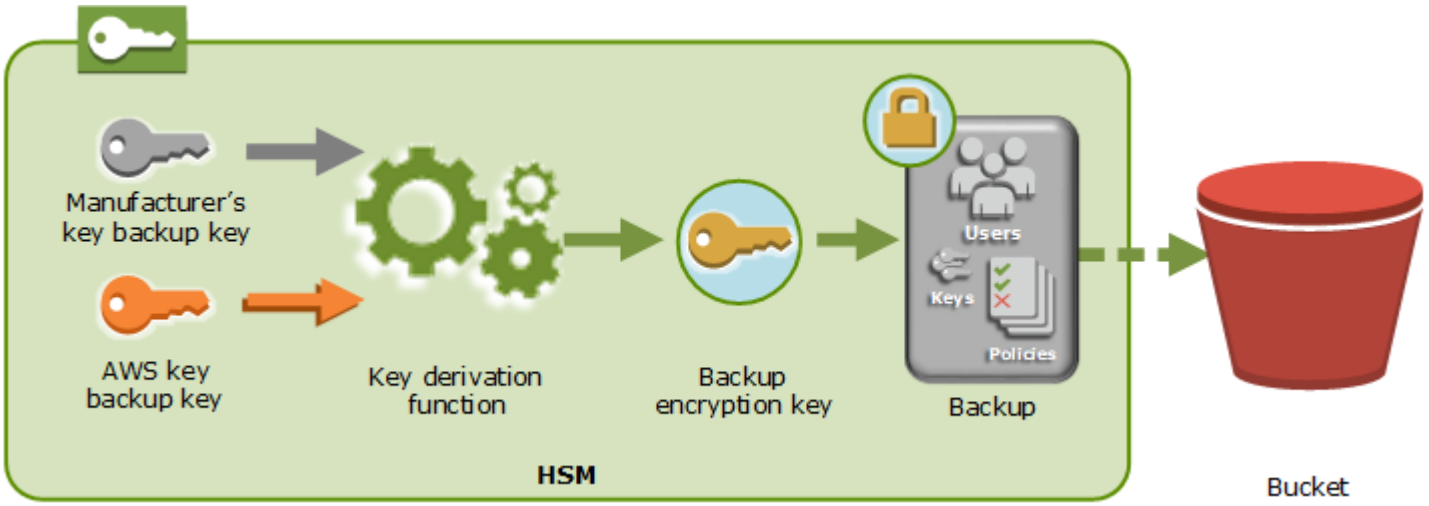
Per crittografare i dati, l'HSM utilizza una chiave di crittografia univoca e temporanea (EBK). L'EBK è una chiave di crittografia AES a 256 bit generata all'interno dell'HSM quando si esegue un backup. AWS CloudHSM L'HSM crea l'EBK e poi la utilizza per crittografare i dati dell'HSM con un metodo di wrapping della chiave AES approvato dai FIPS e conforme alla Pubblicazione Speciale 800-38F del NIST. Quindi l'HSM fornisce i dati crittografati a. AWS CloudHSMI 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 creare una PBK, l'HSM utilizza una funzione di derivazione della chiave (KDF) approvata dai FIPS in modalità counter

conforme alla Pubblicazione Speciale 800-108 del NIST. 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 (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 moduli AWSdi protezione hardware di proprietà dello 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 di archiviare un backup in Amazon S3, il servizio 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 i criteri 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. Ogni istruzione è strutturata come mostrato nell'esempio seguente.

```
{
  "Version": "2012-10-17",
  "Statement":[{
    "Effect": "effect",
    "Action": "action",
    "Resource": "resource-arn",
    "Condition": {
      "condition": {
        "key":"value"
      }
    }
  }]
}
```

- 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 sulla politica IAM, puoi specificare qualsiasi azione API che AWS CloudHSM offre. 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 di azioni API AWS CloudHSM che iniziano con`List`.

```
"Action": "cloudhsm:List*"
```

Per specificare tutte le azioni API per AWS CloudHSM, utilizzate il carattere jolly *, come mostrato nell'esempio seguente.

```
"Action": "cloudhsm:*"
```

Per l'elenco delle azioni API per AWS CloudHSM, vedi 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 ha chiavi contestuali 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 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— Garantisce l'accesso in sola lettura alle funzionalità. AWS CloudHSM

Politiche 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 CloudHSMCollegare 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 nuove funzionalità di AWS CloudHSM gestione vengono aggiunte nel tempo, 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 policy 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: Autorizzazioni di sola lettura

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.

```
{
   "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. È necessario includere l'`iam:CreateServiceLinkedRole`azione per consentire AWS CloudHSM la creazione automatica del ruolo AWSServiceRoleForCloudHSMcollegato al servizio nel proprio account. Questo ruolo

consente di registrare AWS CloudHSM gli eventi. Per ulteriori informazioni, consulta Ruoli collegati ai servizi per AWS CloudHSM.

> **Note**
>
> Per un elenco dei permessi specifici di ogni servizio, consulta Operazioni, risorse e chiavi di condizione per l' AWS CloudHSM nella Guida all'autorizzazione del servizio.

```
{
   "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 dell' AWS CloudHSM API, incluse le azioni per eliminare gli HSM e i cluster. Include anche autorizzazioni aggiuntive per operazioni Amazon EC2specifiche. È necessario includere l'`iam:CreateServiceLinkedRole`azione per consentire AWS CloudHSM la creazione automatica del ruolo AWSServiceRoleForCloudHSMcollegato al servizio nel proprio account. Questo ruolo consente di registrare AWS CloudHSM gli eventi. Per ulteriori informazioni, consulta Ruoli collegati ai servizi per AWS CloudHSM.

```
{
   "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":"*"
   }
}
```

Ruoli collegati ai servizi per AWS CloudHSM

La policy IAM che hai creato in precedenza Politiche gestite dal cliente per AWS CloudHSM include l'azione`iam:CreateServiceLinkedRole`. AWS CloudHSM definisce un ruolo collegato al servizio

denominato. AWSServiceRoleForCloudHSM Il ruolo è predefinito AWS CloudHSM e include le autorizzazioni che AWS CloudHSM richiedono la chiamata di 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 relativa AWS CloudHSM ai ruoli consente di creare gruppi di CloudWatch log e flussi di log di Amazon Logs e scrivere eventi di log per tuo conto. È possibile visualizzarli qui sotto e nella console IAM;.

```
{
    "Version": "2018-06-12",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
}
```

La politica di fiducia per il AWSServiceRoleForCloudHSMruolo consente di AWS CloudHSM assumere il ruolo.

```
{
  "Version": "2018-06-12",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudhsm.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
```

```
  ]
}
```

Creazione di un ruolo collegato ai servizi (Automatico)

AWS CloudHSM crea il AWSServiceRoleForCloudHSMruolo quando si crea un cluster se si include l'`iam:CreateServiceLinkedRole`azione nelle autorizzazioni definite al momento della creazione del gruppo di AWS CloudHSM amministratori. Per informazioni, consulta Politiche gestite dal cliente per AWS CloudHSM.

Se disponi già di uno o più cluster e desideri solo aggiungere il AWSServiceRoleForCloudHSMruolo, puoi utilizzare la console, il comando create-cluster o l'operazione CreateClusterAPI per creare un cluster. Quindi usa la console, il comando delete-cluster o l'operazione DeleteClusterAPI per eliminarlo. 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 Nozioni di base su AWS CloudHSM per creare un cluster se lo si sta creando solo per aggiungere il ruolo. AWSServiceRoleForCloudHSM

Creazione di un ruolo collegato ai servizi (Manuale)

Puoi utilizzare la console o l'API IAM per creare il AWSServiceRoleForCloudHSMruolo. AWS CLIPer ulteriori informazioni, consulta Creazione di un ruolo collegato ai servizi nella Guida per l'utente IAM.

Modifica del ruolo collegato ai servizi

AWS CloudHSM non consente di modificare il AWSServiceRoleForCloudHSMruolo. 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 di 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 CLIl'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 del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Conformità

AWS CloudHSM fornisce politiche di sicurezza approvate dalla FIPS per gli HSM. Inoltre, AWS CloudHSM soddisfa i requisiti di conformità PCI-PIN, PCI-3DS e SOC2. Affidarsi a un HSM con convalida FIPS può aiutarti a soddisfare i requisiti di conformità aziendali, contrattuali e normativi per la sicurezza dei dati nel cloud. AWS Per ulteriori informazioni, consulta le informazioni riportate di seguito.

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. Gli HSM forniti da AWS CloudHSM sono certificati FIPS 140-2 livello 3 (Certificato #4218). Per ulteriori informazioni in merito, fai riferimento alla convalida FIPS per hardware.

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 relativi al numero di identificazione personale (PIN), informazioni utilizzate per le transazioni presso gli sportelli automatici e i terminali (POS). point-of-sale Da gennaio2023 l'AWS CloudHSM è conforme al PCI PIN. 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 fornisce un ulteriore livello di sicurezza per gli acquisti online. L' AWS CloudHSM è conforme allo standard 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 Domande frequenti sulla conformità PCI-PIN

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). point-of-sale

L'attestato di conformità (AOC) e il riepilogo della responsabilità PCI-PIN 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.

Domande frequenti

D: Cos'è l'attestato di conformità e il riepilogo della responsabilità?

L'attestato di conformità (AOC) è prodotto da un Qualified PIN Assessor (QPA) che attesta AWS CloudHSM la conformità ai controlli applicabili nello standard PCI-PIN. La matrice riassuntiva delle responsabilità descrive i controlli, ossia le rispettive responsabilità dei clienti e dei relativi clienti. AWS CloudHSM

D: Come posso ottenere l' AWS CloudHSM attestato di conformità?

L'attestazione di conformità PCI-PIN (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 il "Riepilogo della responsabilità delAWS CloudHSM PCI PIN» nel pacchetto AWS PCI PIN Compliance Package, disponibile per i clienti tramite AWS Artifact, un portale self-service per l'accesso su richiesta ai report di conformità di 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'attestazione di conformità PCI-PIN (AOC)?

I clienti devono gestire la propria conformità PCI-PIN. È necessario eseguire un processo formale di attestazione PCI-PIN tramite un Qualified PIN Assessor (QPA) per verificare che il carico di lavoro dei pagamenti soddisfi tutti i controlli/requisiti PCI-PIN. 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 requisiti PCI-PIN 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 requisiti chiave del ciclo di vita della gestione delle chiavi previsti dallo standard PCI-PIN.

D: Quali AWS CloudHSM controlli sono conformi allo standard PCI-PIN?

L'AOC riassume i controlli valutati dal AWS CloudHSM QPA. Il riepilogo della responsabilità PCI-PIN è disponibile ai 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 traduzione del PIN e il DUKPT?

No, AWS CloudHSM fornisce moduli di protezione 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 servizi che utilizzano. AWS CloudHSMSe 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 obsolete le funzionalità per rimanere conformi ai requisiti di FIPS 140, PCI-DSS, PCI-PIN, PCI-3DS e SOC2. Questa pagina elenca le modifiche attualmente in vigore.

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 supporto per questi sistemi cesserà il 1° gennaio 2024 nelle nostre istanze conformi al Federal Information Processing Standard (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 con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi 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 Cluster a elevata disponibilità e sistema di bilanciamento del carico.

Sicurezza dell'infrastruttura in AWS CloudHSM

In quanto servizio gestito, AWS CloudHSM è protetto dalle procedure di sicurezza della rete AWS globale descritte nel white paper Amazon Web Services: Overview of Security Processes.

Utilizzi chiamate API AWS pubblicate per accedere AWS CloudHSM attraverso 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 AWS Security Token Service (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).

Quando crei un HSM, AWS CloudHSM inserisci un'elastic network interface (ENI) nella sottorete in modo da poter interagire con i tuoi HSM. Per ulteriori informazioni, consulta Architettura 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 di Amazon EC2.

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 “Informazioni sugli utenti HSM”.

AWS CloudHSM ed endpoint VPC

Puoi stabilire una connessione privata tra il tuo VPC e creare un AWS CloudHSM endpoint VPC di interfaccia. Gli endpoint di interfaccia sono basati su una tecnologia che consente di accedere in modo privato alle AWS CloudHSM API senza un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione AWS Direct Connect. AWS PrivateLink Le istanze nel tuo VPC

non necessitano di indirizzi IP pubblici per comunicare con AWS CloudHSM le API. 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 tue sottoreti.

Per ulteriori informazioni, consulta Interface VPC endpoints (AWS PrivateLink) nella Amazon VPC User Guide.

Considerazioni sugli endpoint AWS CloudHSM VPC

Prima di configurare un endpoint VPC di interfaccia per AWS CloudHSM, assicurati di esaminare le proprietà e le limitazioni degli endpoint dell'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 interfaccia per l' AWS CloudHSM

Puoi creare un endpoint VPC per il AWS CloudHSM servizio utilizzando la console Amazon VPC o (). AWS Command Line Interface AWS CLIPer ulteriori informazioni, consulta Creazione di un endpoint dell'interfaccia nella Guida per l'utente di Amazon VPC.

Per creare un endpoint VPC per AWS CloudHSM, usa 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, l'hostname AWS CloudHSM DNS standard (`https://cloudhsmv2.<region>.amazonaws.com`) viene risolto sul tuo endpoint VPC.

Questa opzione rende più semplice utilizzare l'endpoint VPC. Per impostazione predefinita, AWS gli SDK AWS CLI utilizzano il nome host AWS CloudHSM DNS standard, quindi non è necessario specificare l'URL dell'endpoint VPC nelle applicazioni e nei comandi.

Per ulteriori informazioni, consulta Accesso a un servizio tramite un endpoint dell'interfaccia in Guida per l'utente di Amazon VPC.

Creazione di una policy per gli 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: policy degli endpoint VPC per le azioni AWS CloudHSM

Di seguito è riportato un esempio di policy sugli endpoint per. AWS CloudHSMSe associata a un endpoint, questa politica consente l'accesso alle AWS CloudHSM azioni elencate per tutti i principali su tutte le risorse. Vedi altre Gestione delle identità e degli accessi per AWS CloudHSM azioni e AWS CloudHSM 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 viene gestito da terze parti e deve essere prima valutato dal NIST per la conformità allo standard FIPS 140-2 livello 3. È possibile installare solo firmware con crittografia firmata secondo lo standard FIPS a cui AWS non ha accesso.

Risoluzione dei problemi relativi a AWS CloudHSM

In caso di problemi con AWS CloudHSM, puoi trovare soluzioni nei seguenti argomenti.

Argomenti

Problemi noti

AWS CloudHSM è caratterizzato dai problemi noti descritti di seguito. Scegli un argomento per saperne di più.

Argomenti

Problemi noti per tutte le istanze HSM

I seguenti problemi hanno un impatto su tutti gli utenti di AWS CloudHSM a prescindere dall'uso dello strumento a riga di comando key_mgmt_util, dell'SDK PKCS #11, dell'SDK JCE o dell'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

Inoltre, il wrapping della chiave senza riempimento e riempimento a zeri non è supportato.

- Impatto: non è previsto alcun impatto in caso di wrapping e annullamento del wrapping mediante questo algoritmo all'interno di AWS CloudHSM. Tuttavia, il wrapping delle chiavi con AWS CloudHSM non può essere annullato all'interno di altri HSM o del software che necessita di conformità alla specifica di non riempimento. Questo perché otto byte di dati di riempimento

potrebbero essere aggiunti alla fine dei dati della chiave durante un wrapping conforme agli standard. Il wrapping esterno delle chiavi non può essere annullato 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 CreateHsmfunzione per impostare l'elastic network interface (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, consulta Installare e configurare il client AWS CloudHSM (Linux) o Installare e configurare il client AWS CloudHSM (Windows).

Problema: esiste un limite massimo di 16 KB sui dati che possono essere sottoposti a hashing e firma da parte di AWS CloudHSM

- 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. Il client e gli SDK restituiranno esplicitamente un errore se i dati buffer sono di dimensioni superiori a 64 KB. Per trarre vantaggio dalla correzione, è necessario aggiornare il client e gli SDK alla versione 1.1.1 o successiva.

Problema: non è stato possibile specificare le chiavi importate 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 WrapKey unWrapKey e i comandi in key_mgmt_util è stato rimosso

- Risoluzione: quando si utilizza WrapKey unWrapKey o i comandi, è necessario utilizzare l'opzione per `-m` specificare il meccanismo. Vedi gli esempi in WrapKey unWrapKeyo negli articoli per maggiori informazioni.

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 durante l'esecuzione dello 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 ad almeno due. Per le indicazioni su come eseguire questa operazione, consulta la pagina Configurazione delle opzioni dei metadati dell'istanza.

Problemi noti per la libreria PKCS #11

Argomenti

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, consulta la pagina IV personalizzati con lunghezza non conforme per il wrapping di chiavi AES.
  - 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 utilizzano sempre l'IV predefinito **`0xA6A6A6A6A6A6A6A6`** per il wrapping e l'annullamento del wrapping delle chiavi AES

Gli IV forniti dall'utente 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 è 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 è 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`.
- Workaround: (Soluzione) eseguire l'hashing dei dati all'interno dell'applicazione e utilizzare AWS CloudHSM solo per la firma dell'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 un 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: non è possibile eseguire l'hashing di più di 16 KB di dati

Per buffer di dimensioni maggiori, solo i primi 16 KB saranno oggetto di hashing e restituiti. I dati in eccesso vengono ignorati senza alcun avviso.

- 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. Il client e gli SDK restituiranno esplicitamente un errore se i dati buffer sono di dimensioni superiori a 64 KB. Per trarre vantaggio dalla correzione, è necessario aggiornare il client e gli SDK alla versione 1.1.1 o successiva.

Problema: i buffer per le operazioni API **C_Encrypt** e **C_Decrypt** non possono superare 16 KB quando si usa il meccanismo **CKM_AES_GCM**

Inoltre, AWS CloudHSM non supporta la crittografia AES-GCM in più parti.

- 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 dati in parti e crittografare ogni parte utilizzandola singolarmente. `AES_GCM` Se lo utilizzi`AES_GCM`, devi gestire la divisione dei dati e

la successiva crittografia. AWS CloudHSMnon esegue la crittografia AES-GCM multiparte per te. Si noti che FIPS richiede la generazione del vettore di inizializzazione (IV) sull'HSM. `AES-GCM` Pertanto, l'IV per ogni parte di dati crittografati AES-GCM 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 saranno annunciati nel forum AWS CloudHSM e nella pagina della cronologia delle versioni.

Problema: la derivazione della chiave Diffie-Hellman a curva ellittica (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 inizialmente disponibile sul client e quindi viene importata nell'HSM. Un handle della chiave viene quindi restituito all'applicazione.
- Workaround: (Soluzione) se si sta implementando SLL/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.
- Resolution status: (Stato di risoluzione) stiamo sviluppando la possibilità di eseguire la derivazione della chiave ECDH completamente all'interno dell'HSM. Non appena disponibile, l'aggiornamento dell'implementazione sarà annunciato nella pagina della cronologia delle versioni.

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.

  Esempi di sequenze di funzioni errate:

  - `C_EncryptInit`/`C_EncryptUpdate` seguito da `C_Encrypt`
  - `C_DecryptInit`/`C_DecryptUpdate` seguito da `C_Decrypt`
  - `C_SignInit`/`C_SignUpdate` seguito da `C_Sign`
  - `C_VerifyInit`/`C_VerifyUpdate` seguito da `C_Verify`
  - `C_FindObjectsInit` seguito da `C_FindObjectsInit`
- Soluzione alternativa: conformemente alla specifica PKCS #11, l'applicazione deve utilizzare la sequenza corretta di chiamate di funzione per operazioni a parte singola e in più parti. 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 di sola lettura con `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 solo per Windows

- Problema: con le versioni di AWS CloudHSM Client SDK 5 dalla 5.0.0 alla 5.4.0 su Linux, il file di intestazione `/opt/cloudhsm/include/pkcs11/cryptoki.h` è compatibile solo con i sistemi operativi Windows.
- Impatto: è possibile riscontrare dei problemi quando si tenta di includere il file di intestazione nell'applicazione su sistemi operativi basati su Linux.
- Stato della risoluzione: esegui l'aggiornamento alla versione 5.4.1 o successiva di AWS CloudHSM Client SDK 5, che include una versione del file di intestazione compatibile con Linux.

Problemi noti per l'SDK JCE

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

- 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 è di sola lettura KeyStore

- 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.
- Resolution status: (Stato di risoluzione) stiamo aggiungendo il supporto per l'archiviazione di certificati nel keystore. Non appena disponibile, la funzionalità sarà annunciata nella pagina della cronologia delle versioni.

Problema: i buffer per la crittografia AES-GCM non possono superare 16.000 byte

La crittografia AES-GCM in più parti non è supportata.

- Impact: (Impatto) non è possibile utilizzare AES-GCM per crittografare dati di dimensioni superiori a 16.000 byte.
- Workaround: (Soluzione) è possibile utilizzare un meccanismo alternativo, ad esempio AES-CBC, oppure 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 che il vettore di inizializzazione (IV) per AES-GCM sia generato sull'HSM, l'IV per ogni parte di dati crittografata da AES-GCM 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. 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 Diffie-Hellman a curva ellittica (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 a JCE. Quando si utilizza la `KeyAgreement` classe per derivare a SecretKey, questa è prima disponibile sul client e quindi viene importata nell'HSM. Un handle della chiave viene quindi restituito all'applicazione.
- Workaround: (Soluzione) se si sta implementando SLL/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.
- Resolution status: (Stato di risoluzione) stiamo sviluppando la possibilità di eseguire la derivazione della chiave ECDH completamente all'interno dell'HSM. Quando disponibile, annunceremo l'implementazione aggiornata nella pagina della cronologia delle versioni.

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, mentre dovrebbe invece essere il numero di bit di 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: converti la dimensione della chiave da bit a byte prima di utilizzare la KeyGenerator classe o l' KeyAttribute enum per generare chiavi utilizzando il provider AWS CloudHSM JCE se utilizzi le versioni di Client SDK da 5.4.0 a 5.4.2.
- Stato della risoluzione: aggiorna la versione di Client SDK alla versione 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
```
```

Questi avvisi non hanno alcun impatto. Siamo consapevoli del problema e stiamo lavorando per risolverlo. Non è necessaria alcuna risoluzione o soluzione alternativa.

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.

Operazione	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
SecureRandom	Nessun metodo necessario

Operazione	Metodo/i di completamento
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 Client SDK a una di queste versioni.

Problema: perdita di memoria del Client SDK 5 con le operazioni FindKey

- Impatto: l'operazione API FindKey presenta una perdita di memoria in JCE nelle versioni Client SDK 5.10.0 e precedenti. Se utilizzi l'`findKey`API più volte nell'applicazione, ciò comporterà una maggiore crescita della memoria e di conseguenza aumenterà l'ingombro di memoria nell'applicazione. Nel tempo ciò potrebbe causare errori di limitazione o richiedere il riavvio dell'applicazione.
- Soluzione alternativa: consigliamo l'aggiornamento a Client SDK 5.11.0. Se ciò non è possibile, consigliamo di non chiamare l'`findKey`API più volte nell'applicazione. Piuttosto, riutilizzate il più possibile la chiave restituita in precedenza dall'`findKey`operazione precedente.
- Stato della risoluzione: aggiorna la versione dell'SDK del client alla 5.11.0 o successiva, che include una correzione per questo problema.

Problemi noti per OpenSSL Dynamic Engine

Ecco i problemi noti per OpenSSL Dynamic Engine

Argomenti

- Problema: non è possibile installare OpenSSL Dynamic Engine per Client SDK 3 su RHEL 8, CentOS 8 o Ubuntu 18.04 LTS
- Problema: deprecazione SHA-1 Sign and Verify su RHEL9 (9.2+)
- Problema: AWS CloudHSM OpenSSL Dynamic Engine non è compatibile con il provider FIPS per OpenSSL v3.x

Problema: non è possibile installare 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

- Impact: (Impatto) per ottimizzare le prestazioni, l'SDK non è configurato per l'offload di funzioni aggiuntive come la generazione di numeri casuale o le operazioni 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

- Impatto: qualsiasi chiamata alla crittografia e alla decrittografia RSA con imbottitura OAEP non riesce e genera un errore. divide-by-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 o Provider JCE.

- 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 motore AWS CloudHSM 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 chiavi, utilizzare PKCS #11 o JCE nelle applicazioni oppure utilizzare `key_mgmt_util` nella AWS 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 per Client SDK 5. Per ulteriori informazioni, consulta le pagine relative alle piattaforme supportate e a OpenSSL Dynamic Engine.

Problema: deprecazione SHA-1 Sign and Verify su RHEL9 (9.2+)

- Impatto: l'utilizzo del digest dei messaggi SHA-1 per scopi crittografici è stato dichiarato obsoleto in RHEL9 (9.2+). Di conseguenza, le operazioni di firma e verifica con SHA-1 utilizzando OpenSSL Dynamic Engine avranno esito negativo.
- Soluzione alternativa: se lo scenario richiede l'uso di SHA-1 per firmare/verificare firme crittografiche esistenti o di terze parti, consulta Enhancing RHEL Security: Understanding SHA-1 deprecation on RHEL9 (9.2+) e RHEL9 (9.2+) 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 OpenSSL 3.x.
- Soluzione alternativa: per utilizzare AWS CloudHSM OpenSSL Dynamic Engine con le versioni 3.x di OpenSSL, assicurati che il provider «predefinito» sia configurato. Scopri di più sul provider predefinito sul sito web di OpenSSL.

Problemi noti per le istanze Amazon EC2 che eseguono Amazon Linux 2

Problema: Amazon Linux 2 versione 2018.07 utilizza un pacchetto **`ncurses`** (versione 6) aggiornato che al momento non è compatibile con gli SDK AWS CloudHSM

Viene visualizzato il seguente errore durante l'esecuzione di cloudhsm_mgmt_util o key_mgmt_util AWS CloudHSM:

```
/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 per l'integrazione di applicazioni di terze parti

Problema: Client SDK 3 non supporta l'impostazione dell'attributo PKCS #11 **`CKA_MODIFIABLE`** da parte di Oracle durante la generazione della chiave principale

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 saranno annunciati nel forum AWS CloudHSM e nella pagina della cronologia delle versioni.

Errori di sincronizzazione delle chiavi in Client SDK 3

In Client SDK 3, se la sincronizzazione lato client dà errore, AWS CloudHSM esegue una pulizia al massimo delle capacità di tutte le chiavi indesiderate create (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 in grado di rimuovere e contrassegnare il materiale di chiave indesiderato, è necessario eliminare il materiale di 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 non è riuscito a rimuovere e contrassegnare il materiale di chiave indesiderato.

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 incrementare la durabilità delle chiavi token, AWS CloudHSM non riesce a eseguire le operazioni di creazione di chiavi che non hanno esito positivo nel numero minimo di HSM specificato nelle impostazioni di sincronizzazione lato client. Per ulteriori informazioni, consulta la pagina sulla sincronizzazione delle chiavi in AWS CloudHSM.

Client SDK 3: verifica delle prestazioni HSM con lo strumento pkpspeed

Questo argomento descrive come verificare le prestazioni HSM con Client SDK 3.

Per verificare le prestazioni degli HSM nel cluster AWS CloudHSM, è possibile utilizzare lo strumento pkpspeed (Linux) o pkpspeed_blocking (Windows) incluso in Client SDK 3. Lo strumento pkpspeed viene eseguito in condizioni ideali ed esegue una chiamata diretta all'HSM per eseguire le operazioni senza utilizzare 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 Installazione e configurazione del client AWS CloudHSM (Linux). Per ulteriori informazioni sull'installazione del client in un'istanza di Windows, consulta Installare e configurare il client AWS CloudHSM (Windows).

Dopo aver installato e configurato il client AWS CloudHSM, eseguire questo comando 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 (per maggiori dettagli, consulta le domande frequenti di AWS CloudHSM). È possibile verificarlo utilizzando gli strumenti della CLI come indicato nella Guida per l'utente di AWS CloudHSM ed eseguendo il comando getHSMInfo 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. Consulta la nota 1 a seguire per una modifica imminente.
- Basic AES: testa 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] Non consentito dopo il 2023 per conformità a FIPS secondo le linee guida del NIST. 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/...
```

L'utente di Client SDK 5 contiene valori incoerenti

Il comando `user list` restituisce un elenco di tutti gli utenti e delle relative proprietà nel cluster. Se una delle proprietà di un utente presenta il valore "incoerente", tale 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",`	Questo utente risulta essere un crypto user in alcuni HSM e un	1. Effettua l'accesso come amministr atore.

Proprietà incoerente	Output esemplificativo dell'elenco di utenti	Implicazione	Metodo di ripristino
	"role": "**inconsistent** ", "locked": "false", "mfa": [], "cluster-coverage": "full" }	amministratore in altri. Questo può accadere se due SDK tentano di creare contempor aneamente lo stesso utente con ruoli diversi. È necessario rimuovere l'utente e ricrearlo con il ruolo desiderato.	2. Elimina l'utente su tutti gli HSM: user delete --username **<user's name>** --role admin 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'utente di effettuare l'accesso e riattivare l'autenticazione a più fattori (per questa operazione l'utente dovrà firmare i token e fornire la propria chiave pubblica in un file PEM): user change-mfa token-sign --username **<user's name>** --role **<desired role>** —token <File>

Errore rilevato durante il controllo della disponibilità delle chiavi

Problema: un 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 generalmente nei cluster con un solo HSM o nei cluster con due HSM nel 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 Generazione chiavi-simmetriche oGenerazione chiavi-asimmetriche-coppia .
- È stata avviata un'operazione Elenco delle chiavi.
- 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 Parametri dello Strumento di configurazione.
- Se utilizzi un cluster con due HSM, evita di utilizzare le operazioni che hanno generato l'errore, salvo durante il codice di inizializzazione.
- Aumenta il numero di HSM nel cluster ad almeno tre.

Estrazione di chiavi tramite 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.

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.

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
- Protezione ed estrazione delle chiavi da un HSM

Limitazione (della larghezza di banda della rete) HSM

Quando il carico di lavoro supera la capacità HSM del cluster, riceverai messaggi di errore che indicano che gli HSM sono congestionati 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 di ripetizione dei tentativi 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 Aggiunta o rimozione di moduli HSM in un cluster AWS CloudHSM.

> **Important**
>
> Si consiglia di testare il carico di carico del cluster per determinare il carico massimo da prevedere e quindi di aggiungere un altro HSM per garantire un'elevata disponibilità.

Sincronizzazione degli utenti HSM tra gli HSM nel cluster

Per gestire gli utenti HSM, utilizza lo strumento a riga di comando AWS CloudHSM noto come cloudhsm_mgmt_util. Questo comunica solo con gli HSM che sono nel file di configurazione dello strumento e non rileva gli altri HSM nel cluster che non sono nel file di configurazione.

AWS CloudHSM sincronizza le chiavi sui tuoi HSMs con tutti gli altri HSMs nel cluster, ma non sincronizza le policy o gli utenti HSM. Quando utilizzi cloudhsm_mgmt_util per gestire gli utenti HSM, queste modifiche degli utenti potrebbero interessare solo alcuni HSM del cluster, quelli nel file di configurazione cloudhsm_mgmt_util. Questo può causare dei problemi quando AWS CloudHSM sincronizza le chiavi degli HSM nel cluster, perché gli utenti proprietari delle chiavi potrebbero non essere presenti su tutti gli HSM nel cluster.

Per evitare questi problemi, modifica il file di configurazione cloudhsm_mgmt_util prima di gestire gli utenti. Per ulteriori informazioni, consulta ???.

Connessione al cluster persa

Quando hai configurato il AWS CloudHSM client, hai fornito l'indirizzo IP del primo HSM nel tuo cluster. Questo indirizzo IP viene salvato nel file di configurazione per il client AWS CloudHSM. 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 client AWS CloudHSM

1. Utilizzare uno dei seguenti modi per trovare l'indirizzo IP di un HSM attivo nel cluster.

   - Visualizzare la scheda HSMs (HSM) nella pagina dei dettagli del cluster nella console AWS CloudHSM.

- Utilizza AWS Command Line Interface (AWS CLI) per eseguire il comando describe-clusters.

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 client AWS CloudHSM.

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 di audit AWS CloudHSM mancanti in CloudWatch

Se è stato creato un cluster prima del 20 gennaio 2018, è necessario configurare manualmente un ruolo collegato al servizio in modo da consentire la distribuzione di dei log di audit di tale 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 il wrapping di chiavi AES

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

Il problema ha delle ripercussioni solo se riscontri tutte le condizioni seguenti:

Condition	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 indiretta mente 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'e-mail dal team AWS CloudHSM, probabilmente stai utilizzando la versione 3.0.0 della libreria PKCS #11. Per verificare la versione del software sulle istanze dell'applicazione, utilizza questo comando:

Condition	Come saperlo
	`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 su GitHub.
3. Identifica e recupera le chiavi con wrapping esistenti

   È necessario identificare le chiavi di cui è stato eseguito il wrapping utilizzando la versione 3.0.0 della libreria PKCS #11 e in seguito contattare l'assistenza (https://aws.amazon.com/support) per recuperare le chiavi.

> **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 di problemi di creazione dei cluster

Quando crei un cluster, AWS CloudHSM; crea un ruolo legato al servizio AWSServiceRoleForCloudHSM, se tale ruolo non esiste già. Se AWS CloudHSM non è in grado di creare un ruolo legato al servizio, il tuo tentativo di creazione del cluster potrebbe non andare a buon fine.

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 AWS Support. Per ulteriori informazioni sul ruolo legato al servizio AWSServiceRoleForCloudHSM, consulta Ruoli collegati ai servizi per AWS CloudHSM.

Argomenti

Aggiungere l'autorizzazione mancante

Per creare un ruolo legato al servizio, l'utente deve disporre dell'autorizzazione `iam:CreateServiceLinkedRole`. Se l'utente IAM che desidera creare il cluster non dispone di questa autorizzazione, la procedura di creazione del cluster non va a buon fine quando tenta di creare il ruolo legato 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

È possibile utilizzare la console IAM, la CLI o l'API per creare il ruolo legato al servizio AWSServiceRoleForCloudHSM. Per ulteriori informazioni, consulta Creazione di un ruolo collegato ai servizi nella Guida per l'utente IAM.

Utilizzare un utente non federato

Gli utenti federati, le cui credenziali sono originate al di fuori di 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.

Dopo la creazione del ruolo legato al servizio, se preferisci, puoi eliminare il cluster creato dall'utente non federato. L'eliminazione del cluster non ha effetti sul ruolo. Pertanto qualsiasi utente con le autorizzazioni richieste, inclusi gli utenti federati, può creare cluster AWS CloudHSM nel tuo account.

Per verificare che il ruolo sia stato creato, apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/ e seleziona Ruoli. Altrimenti, utilizza il comando get-role in AWS CLI.

```
$  aws iam get-role --role-name AWSServiceRoleForCloudHSM
{
    "Role": {
        "Description": "Role for CloudHSM service operations",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "cloudhsm.amazonaws.com"
                    }
                }
            ]
        },
        "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 del client

AWS CloudHSM offre strumenti per Client SDK 3 e Client SDK 5 per raccogliere informazioni sull'ambiente in uso per consentire al supporto AWS di risolvere i problemi.

Argomenti

- Strumento di supporto per Client SDK 5
- Strumento di supporto per Client SDK 3

Strumento di supporto per Client SDK 5

Lo script 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 ]_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`.

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.

  ```
  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.

```
C:\Program Files\Amazon\CloudHSM\bin\jce_info.exe
```

Recupero dei log da un ambiente serverless

Per la configurazione di ambienti serverless, come Fargate o Lambda, si consiglia di configurare il tipo di log AWS CloudHSM su `term`. Una volta configurato su `term`, l'ambiente serverless sarà in grado di generare l'output su CloudWatch.

Per ottenere i log del client da CloudWatch, consulta la pagina Utilizzo di gruppi di log e flussi di log nella Guida per l'utente per i file di log Amazon CloudWatch.

Strumento di supporto per Client SDK 3

Lo script 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.

Quote AWS CloudHSM

Le quote, precedentemente note come limiti, sono i valori assegnati per le risorse AWS. I seguenti limiti si applicano alle risorse AWS CloudHSM per ciascuna regione AWS e ciascun 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.

Quote del servizio

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
Numero massimo di chiavi per cluster	3.300
Numero massimo di utenti per cluster	1,024
Lunghezza massima del nome utente	31 caratteri
Lunghezza richiesta della password	Da 7 a 32 caratteri
Numero massimo di connessioni client simultanee per cluster [1]	900

Risorsa	Quota
Numero massimo di sessioni PKCS #11 per applicazione	1,024

[1] Una connessione client per Client SDK 3 è un daemon del client. Per Client SDK 5, una connessione client è un'applicazione.

Per ulteriori informazioni, consulta Risorse di sistema.

Risorse di sistema

Le quote delle risorse di sistema sono quote su ciò che il client AWS CloudHSM può utilizzare quando è in esecuzione.

I descrittori di file sono un meccanismo del sistema operativo per identificare e gestire i file aperti in base al processo.

Il daemon del client CloudHSM utilizza i descrittori di file per gestire le connessioni tra le applicazioni e il client, nonché tra il client e il server.

Per impostazione predefinita, la configurazione del client CloudHSM allocherà 3000 descrittori di file. Questo valore predefinito è progettato per ottenere una sessione ottimale e una capacità di threading tra il daemon del client e i moduli HSM.

In rari casi, se esegui il client in un ambiente con risorse limitate, potrebbe essere necessario modificare questi valori predefiniti.

> **Note**
>
> Modificando questi valori, le prestazioni del client CloudHSM potrebbero presentare problemi e/o l'applicazione potrebbe diventare inutilizzabile.

1. Modificare il file `/etc/security/limits.d/cloudhsm.conf`.

```
#
```

```
# DO NOT EDIT THIS FILE
#
hsmuser soft nofile 3000
hsmuser hard nofile 3000
```

2. Modificare i valori numerici, in base alle esigenze.

> **Note**
>
> La quota `soft` deve essere inferiore o uguale alla quota `hard`.

3. Riavvia il processo del daemon del client CloudHSM.

> **Note**
>
> Questa opzione di configurazione non è disponibile sulle piattaforme Microsoft Windows.

Download per Client SDK dell'AWS CloudHSM

Download

A marzo 2021, AWS CloudHSM ha rilasciato la versione 5.0.0 del Client SDK, che introduce un Client SDK completamente nuovo con requisiti, funzionalità e supporto della 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, ad eccezione del supporto per i provider CNG e KSP. Per ulteriori informazioni, consulta Confronto dei componenti del Client SDK.

> **Note**
>
> Per informazioni sulle piattaforme supportate da ciascun Client SDK, consulta e. Piattaforme supportate da Client SDK 5 Piattaforme supportate da Client SDK 3

Versione più recente

Questa sezione include la versione più recente di Client SDK.

Versione Client SDK 5: versione 5.11.0

Amazon Linux 2

Scarica il software versione 5.11.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- OpenSSL Dynamic Engine (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- JCE provider (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

Scarica il software versione 5.11.0 per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 5ac16449ec149c9b5e7776865803245ab17d0f1ad56df80173840c5e8d257b19)
- OpenSSL Dynamic Engine (SHA256 checksum 28c2eb7f3f60172b0186e5c25f71bb7341537058a71f288673936766048083c1)
- JCE provider (SHA256 checksum 06c9d9d281c12b1d2bd9a7b601d6317e46cedf175706bbfa3e4dcaed6ba05448)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 218982bb17aa751969a7866b0a9ff27e7aa5007a07817627d9cc1f7d60a78160)

Amazon Linux 2023

Scarica il software versione 5.11.0 per Amazon Linux 2023 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 55310ab333d18bcfabdc4b74115b040386b4508934bdff93e1d054c4c4a6f9ea)
- OpenSSL Dynamic Engine (SHA256 checksum f3d4934dc872a9b5212a180b9814ca2af3eca01ee228a8725563f1770add0dce)
- JCE provider (SHA256 checksum 757d3abb515aeb08f4b1c83970ee0979399efee00ee78c9a9dbec05f4ed9768d)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 22af8f0501ff9a45a9e0683a408a63771c2c06c66abf5478d310d6d32e013555)

CentOS 7 (7.8+)

Scarica il software versione 5.11.0 per CentOS 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- OpenSSL Dynamic Engine (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- JCE provider (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

RHEL7 (7.8+)

Scarica la versione 5.11.0 del software per RHEL 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 9fc0cd7cf003a7cb7e42dbd19671d58a97fc3b3d871d284dc6ae7fd226598772)
- OpenSSL Dynamic Engine (SHA256 checksum 1df6669c971440d446890b0fbeb74125a423df7b14e7ac4577347be7ef176572)
- JCE provider (SHA256 checksum 148a3f1de55a68e3bb525fb2994645333a52c2e9e46946dd8d90fcbc90ab64fd)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum a68f4a56d4c539cfcc8a1e56e19b5ff385bb24936ea5f349255b4e9bfbee9aab)

RHEL8 (8.3+)

Scarica la versione 5.11.0 del software per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b95b9f588656fb14fd08bb66ce0e0da807b96daa38348dec07a508c9bef7403a)
- OpenSSL Dynamic Engine (SHA256 checksum 7bb437b91a52e863b2b00ff7f427ce22522026daf757be873ee031ec6ffffd88)
- JCE provider (SHA256 checksum e0db887e05eb535314f4d99f21da12d87d35ebb8baf9726f4ce8f01d9df0ea01)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 8485b5a6d679767ca9b4f611718159a643cf3e85090a8e4d20fe53c3707e25c3)

RHEL9 (9.2+)

Scarica la versione 5.11.0 del software per RHEL9 (9.2+) su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 87b56a20accf67df53a203b7f115655b2acfaec4516682d4976d9475b10bec8e)
- OpenSSL Dynamic Engine (SHA256 checksum 83a6b58572e985df937beede4b10e867b0ac6050ace8010dc8d535be365d2747)
- JCE provider (SHA256 checksum ee95213d02d913250478d0793d6dd578e5c54d765e635c7468a49bdff4c2a6f3)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 7e168ed3bef8e9c5110645e9960680e9a57f7b94e16aec71422e3c67ebc58fb5)

Ubuntu 20.04 LTS

Scarica il software versione 5.11.0 per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum abc3a339d1fe5850db65620804e9a910f8b4f913624ef9b7189f2f0df1825c01)
- OpenSSL Dynamic Engine (SHA256 checksum 075fc3f9974d552f27ad67fa92c8abff31b756b9add875b8cd4957e6801583a4)
- JCE provider (SHA256 checksum 5de45c519133a0dae8da3ac01809db7974be25c14c15eb773fc5c972c0178c13)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 83e0e4505a063792c19feb3d4cfd032b9089091916168d92b0f51a967a007734)

Ubuntu 22.04 LTS

Scarica il software versione 5.11.0 per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum b8f20be125c8530b2a7bd945956e9c04296fba5634af408b40be4e03bdbad72a)
- OpenSSL Dynamic Engine (SHA256 checksum d728c156eb4ee5c67159e57d6b092785800baa5fb61c14d64f460a8b8f53a778)
- JCE provider (SHA256 checksum 44e943b8cd1176ad666e249342687744a280c6222df58b5a9f084c932f628284)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 8ccf5389d459611be813e42d7f9d040090f94f3fe88f9d110bcfb25e9619e4a7)

Windows Server 2016

Scarica la versione 5.11.0 del software per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum aa4bce5be15bbe0978b7205c619bb91c55a8e0f1f4636be311f24878f7709e07)
- JCE provider (SHA256 checksum 004cdb9ecb4a4d72458084997de7f562fb76a4e2f0567009f1dfafa7b2bded47)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 679795db759fda4823232142297a281e21a7d6f32cb5ddd6ac4c479866fa33b7)

Windows Server 2019

Scarica la versione 5.11.0 del software per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum aa4bce5be15bbe0978b7205c619bb91c55a8e0f1f4636be311f24878f7709e07)

- JCE provider (SHA256 checksum 004cdb9ecb4a4d72458084997de7f562fb76a4e2f0567009f1dfafa7b2bded47)
  - Javadocs per AWS CloudHSM (SHA256 checksum fb469ae53b516338f3326b402b15b7b84912801a8c25a28cd31a5da0631cd3c5)
- CLI CloudHSM (SHA256 checksum 679795db759fda4823232142297a281e21a7d6f32cb5ddd6ac4c479866fa33b7)

Client SDK 5.11.0 aggiunge nuove funzionalità, migliora la stabilità e include correzioni di bug per tutti gli SDK.

Supporto per piattaforma

- È stato aggiunto il supporto per Amazon Linux 2023 e RHEL9 (9.2+) per tutti gli SDK.
- È stato rimosso il supporto per Ubuntu 18.04 LTS a causa della recente fine del suo ciclo di vita.
- È stato rimosso il supporto per Amazon Linux a causa della recente fine del suo ciclo di vita.

CLI CloudhSM

- Sono stati aggiunti i seguenti comandi:
  - segno crittografico
  - verifica crittografica
  - chiave di importazione pem
  - scartare i tasti
  - portachiavi
- Generazione chiavi-fileora supporta l'esportazione di chiavi pubbliche.

OpenSSL Dynamic Engine

- L'AWS CloudHSMOpenSSL Dynamic Engine è ora supportato su piattaforme installate con una versione 3.x della libreria OpenSSL. Ciò include Amazon Linux 2023, RHEL9 (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.
- Aggiunto il nuovo attributo `ID` chiave.

- Introdotta una nuova `DataExceptionCause` variante per l'esaurimento dei tasti:`DataExceptionCause.KEY_EXHAUSTED`.

Correzione di bug e miglioramenti

- È stata aumentata la lunghezza massima dell'`label`attributo da 126 a 127 caratteri.
- Risolto un bug che impediva l'apertura delle chiavi EC con il meccanismo. RsaOaep
- È stato risolto un problema noto relativo all'operazione FindKey nel provider JCE. Fare riferimento a Problema: perdita di memoria del Client SDK 5 con le operazioni FindKey per ulteriori dettagli.
- Registrazione migliorata in tutti gli SDK per le chiavi Triple DES che hanno raggiunto il limite massimo di blocchi di crittografia, secondo FIPS 140-2.
- Sono stati aggiunti problemi noti per OpenSSL Dynamic Engine. Per informazioni dettagliate, vedi Problemi noti per OpenSSL Dynamic Engine.

Versioni precedenti di Client SDK

Questa sezione elenca le versioni precedenti di Client SDK.

Versione 5.10.0

Amazon Linux

Scarica la versione 5.10.0 del software per Amazon Linux su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum d63adf3e96c19c2d894b2defcbadd916dbb0398993050b1358bd93a36aa5acab)
- OpenSSL Dynamic Engine (SHA256 checksum 4daa3e591ffd5f7ce8ef3759c41deaa38867f5e5d21f15927aea83afb1678ac5)
- JCE provider (SHA256 checksum 6c1ac94d3080f1c609d9dafbcb14480911beef3a488c4ed6f2b11b377da9b477)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum c12617fcd7990ba53e96f477979b410e3a5f17842ca7a912861b8b820809b5b5)

Amazon Linux 2

Scarica la versione 5.10.0 per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)
- OpenSSL Dynamic Engine (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- JCE provider (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

Scarica la versione 5.10.0 del software per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 5d8dfd835f1ed5a7f5a4fcc8ecf81cfa29883aca7e2985de69b5db723ab663db)
- OpenSSL Dynamic Engine (SHA256 checksum 91fb8efe2646bf0dbd9087554baa09554714e9d56e9bfd5c0dc3023a9f485574)
- JCE provider (SHA256 checksum 99f6e55c37fdf00085a816d46835aeff54470797b3b71f4d28a70dc79c9caf44)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum 4a88ba9b4cf0dd5573f3dd88ab9dc257e4c486069cb529c5d554979ee2dd83af)

CentOS 7 (7.8+)

Scarica la versione 5.10.0 del software per CentOS 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)
- OpenSSL Dynamic Engine (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- JCE provider (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

RHEL7 (7.8+)

Scarica la versione 5.10.0 del software per RHEL 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum fc47e705e57a0bfd433f7b46c9477a70df5c442a8ad9c2969bcef38e328e4933)

- OpenSSL Dynamic Engine (SHA256 checksum 0aca262df6780995c9b884fcb8765bbd64acaf21b2286ec4d05a9a90edb3d4cb)
- JCE provider (SHA256 checksum b5be7f73c4bcffc5da6f89f324e6b3db5b091610464c8bd38dbddfff0484b2c2)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum e8cf09966890b88a61e695dc034874a445093300359d5d6a86b5a546803920bb)

RHEL8 (8.3+)

Scarica la versione 5.10.0 del software per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 96afb7042a148ddc7a60ab6235b49e176d0460d1c2957bd76ca3d8406ac1cb03)
- OpenSSL Dynamic Engine (SHA256 checksum 2caad2bffea8aef73c91ad422d09772ef830fe7f80a7be19020e6a107eadfbe8)
- JCE provider (SHA256 checksum 3543551f08fbe3900821ea2d4ea148b4e86e2334bc94d7ffef6f3b831457cd71)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum 812eccaadfc490f13bcd0b0a835ef58f3a3d4344ad7e0a237de476dd24509525)

Ubuntu 18.04 LTS

Scarica la versione 5.10.0 del software per Ubuntu 18.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum be4c61766b8b46e1f6c14c3dcf90aaab9f38240fcd9c68b4009704276c5f6f4a)
- OpenSSL Dynamic Engine (SHA256 checksum 64bd8af827b6dc3786e8ad28858cbc4ef6a0fd42164a0945f427eddcf5f02858)
- JCE provider (SHA256 checksum 9fcbdf08e93641468588b608173f26f18781bbc029ed95b2e086da29a968cc00)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum 13808bddddb7eedeb2b8486d23a9976c7fa8d9220149a6b9400626bcaff3b513)

> **Note**
>
> A causa della recente fine del ciclo di vita di Ubuntu 18.04 LTS, non AWS CloudHSM sarà più in grado di supportare questa piattaforma con la prossima versione.

Ubuntu 20.04 LTS

Scarica la versione 5.10.0 del software per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 99ae96504580ff85ed4958a582903a847f666bdaafafbe887a5a76db58f24500)
- OpenSSL Dynamic Engine (SHA256 checksum 13e3f6fe086acf9617b163f66e3941f973daa583fb9322d16c396aa29fc3611d)
- JCE provider (SHA256 checksum 44562cebd9af1aa965840cd9bcb237e518d24c715b3c8bca1405c9c1871835e2)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum ab71b4ec531c5e6d05c91539c7edc1c07e6c748052ebf6200f148cb6812538c5)

Ubuntu 22.04 LTS

Scarica la versione 5.10.0 del software per Ubuntu 22.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ee331a44fbe4936ec98a3ae55d58e67ed38e8bbff0a4f4ce8b1bd8239b75877b)
- Supporto per Support for OpenSSL Dynamic Engine non ancora disponibile per questa piattaforma.
- JCE provider (SHA256 checksum 9e44d14dd33624f6fe36711633013e47e4a93f4d4635e08900546113ded56e3d)
  - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum 2df361546848cd3f8965b1007dca42a0c959eb10d9e3f4995e8e1c852406751d)

Windows Server 2016

Scarica la versione 5.10.0 del software per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7aae9bfd99a6dd0f4d376c227c206c01847f83a9efd774d1063d76cc6fdaa89f)
- JCE provider (SHA256 checksum 1c58fd651e51be2ba59051a87aceca0452990b29837b8a7efabcd510ccbf8c1f)

    - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudhSM (SHA256 checksum f745a2236c9eb9f6f128313eddc35795bd5e47fdf67332bedeb2554201b61a24)

Windows Server 2019

Scarica la versione 5.10.0 del software per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 7aae9bfd99a6dd0f4d376c227c206c01847f83a9efd774d1063d76cc6fdaa89f)
- JCE provider (SHA256 checksum 1c58fd651e51be2ba59051a87aceca0452990b29837b8a7efabcd510ccbf8c1f)
    - Javadocs per AWS CloudHSM (SHA256 checksum dcbb870c6bd58c6770ba7a2b616c6103a5efb3bdeab831ce8f9c82cc09a9870f)
- CLI CloudHSM (SHA256 checksum f745a2236c9eb9f6f128313eddc35795bd5e47fdf67332bedeb2554201b61a24)

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.

Correzione di bug e miglioramenti

- 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

Amazon Linux

Scarica la versione 5.9.0 del software per Amazon Linux su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 4f368be41f006b751ac41b14e1435c27841f60bbde0f032ec02a359fea637dcf)
- OpenSSL Dynamic Engine (SHA256 checksum 81af0d34683825cd6ff844ccacf9c8f4842a4ba76e3875a89121d09a286b4490)
- JCE provider (SHA256 checksum e8e5bc09d8e0b3cb24f30ab420fe08902a19073012335ac94382ec55fcc45abd)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 17284144b45043204ce012fe8b62b1973f10068950abedbd9c2c6172ed0979c6)

Amazon Linux 2

Scarica la versione 5.9.0 del software per Amazon Linux 2 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- OpenSSL Dynamic Engine (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- JCE provider (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

Scarica la versione 5.9.0 del software per Amazon Linux 2 su architettura ARM64:

- Libreria PKCS #11 (SHA256 checksum 4337dca5a08c5194b1118fa197bb4a4f7988df4e1b961e6f2e367295ba99d61d)
- OpenSSL Dynamic Engine (SHA256 checksum 4f08689934e877662a7ce64554fb04eb4b2c213b936018609ff187d100e34a85)
- JCE provider (SHA256 checksum b337b80271a2d308949d5911971fe6ad35df4e34876a481fcac347f1d897fe39)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum a4d466e6b5f74dcd283ba32c9dd87441941d5e5a05936b7c2b4cc7ef85eb1071)

CentOS 7 (7.8+)

Scarica la versione 5.9.0 del software per CentOS 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- OpenSSL Dynamic Engine (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- JCE provider (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

RHEL7 (7.8+)

Scarica la versione 5.9.0 del software per RHEL 7 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum e5affca37abc4ff76369237649830feb32fccd3fa05199cc2021230137093c56)
- OpenSSL Dynamic Engine (SHA256 checksum 848a2e31550bbc2b0223468877baa2a8cda3131ef8537856b31db226d55c4170)
- JCE provider (SHA256 checksum 884f483ef3e9c7def92e3ff01b226e5cbf276d96dcb2f6f56009516f19d41dc0)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 2e62d5a27cff46d9fb47d656afeccd9dbfb5413bfd2267dd3c8fb7960fef7f26)

RHEL8 (8.3+)

Scarica la versione 5.9.0 del software per RHEL 8 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 081887f6ea1d9df9d1e409b2b5bde83e965c42229acbeb1f950c8fe478361edc)
- OpenSSL Dynamic Engine (SHA256 checksum 6b0500a42fd57c39f076f14e5079f80145b6ebd2c441395761eb04600c07bda5)
- JCE provider (SHA256 checksum 2bc7ac26b259af92a65fbd5a30d5eb2a92ce0e70efe41feb53bf82f168aa90bb)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 79ecbe9b4c5316ccf447d8c59b76b5ac2cc854bd79cd50c1f29197aa8cb080db)

Ubuntu 18.04 LTS

Scarica la versione 5.9.0 del software per Ubuntu 18.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum bc6d2227edd7b5a83fed32741fbacbb1756d5df89ebb3435d96f0609a180db65)
- OpenSSL Dynamic Engine (SHA256 checksum 2d6a26434fa6faf337f1dfb42de033220fa405a82d4540e279639a03b3ee6e9d)
- JCE provider (SHA256 checksum e12aef122f490e9026452ce31c25625b1accb9a5866b3d470488f10f047f1873)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum f0bcabe594db3e8ff86cc0f65c2a10858d34452eb6b9fc33d7aac05c0f5f4f30)

Ubuntu 20.04 LTS

Scarica la versione 5.9.0 del software per Ubuntu 20.04 LTS su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum 15dde8182f432de9e7d369b05e384e1f2d80dcca85db3b16ecc26cdef1a34bb9)
- OpenSSL Dynamic Engine (SHA256 checksum c8ba94a999038af87d4905b7c1feb4cc87e20d1776a32ef6f6d11ee000b5a896)
- JCE provider (SHA256 checksum de33cd3e8130a06d9da5207079533aac8276a1319ac435a3737b4f65bd8fb972)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum cfa31535ad9a99a5113496c06fbace38e9593491aca9bb031a18b51075973e68)

Windows Server 2016

Scarica la versione 5.9.0 del software per Windows Server 2016 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ab5380805b0e17dd89dbbefd3fbda8b54da3c140f82e9f3d021850c31837bbe3)
- JCE provider (SHA256 checksum f0941d7a20193818133de8a742d3b848ea19abaf25f5a71ac65949ce5a37c533)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 131530ffe5caff963d483f440d06dcfb41dc11b0f8d78f1dd07bb07f76aeb6d2)

Windows Server 2019

Scarica la versione 5.9.0 del software per Windows Server 2019 su architettura x86_64:

- Libreria PKCS #11 (SHA256 checksum ab5380805b0e17dd89dbbefd3fbda8b54da3c140f82e9f3d021850c31837bbe3)
- JCE provider (SHA256 checksum f0941d7a20193818133de8a742d3b848ea19abaf25f5a71ac65949ce5a37c533)
  - Javadocs per AWS CloudHSM (SHA256 checksum 6343427177180c8f61eec0341e827fbba29420ed2033c0e4b4803d49a3df7763)
- CLI CloudHSM (SHA256 checksum 131530ffe5caff963d483f440d06dcfb41dc11b0f8d78f1dd07bb07f76aeb6d2)

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
- Risolto un problema noto relativo all'esaurimento del pool di sessioni

Versione 3.4.4

Per aggiornare il Client SDK 3 sulle piattaforme Linux, è necessario utilizzare un comando batch che aggiorna contemporaneamente il client daemon e tutte le librerie. Per ulteriori informazioni sull'aggiornamento, vedi Aggiornamento di Client SDK 3.

Per scaricare il software, scegli la scheda del tuo sistema operativo preferito, quindi seleziona il link di ogni pacchetto software.

Amazon Linux

Scarica la versione 3.4.4 del software per Amazon Linux:

- AWS CloudHSM Client (SHA256 checksum 900de424d70f41e661aa636f256a6a79cc43bea6b0fe6eb95c2aaa63e5289505)
- Libreria PKCS #11 (SHA256 checksum a3f93f084d59fee5d7c859292bc02cb7e7f15fb06e971171ebf9b52bbd229c30)
- OpenSSL Dynamic Engine (SHA256 checksum 8db07b9843d49016b0b6fec46d39881d94e426fcaae1cee2747be14af9313bb0)
- JCE provider (SHA256 checksum 360617c55bf4caa8e6e78ede079ca68cf9ef11473e7918154c22ba908a219843)
- AWS CloudHSMProgramma di gestione (SHA256 checksum c9961ffe38921131bd6f3702e10d73588e68b8ab10fbb241723e676f4fa8c4fa)

Amazon Linux 2

Scarica la versione 3.4.4 del software per Amazon Linux 2:

- AWS CloudHSM Client (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Libreria PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- OpenSSL Dynamic Engine (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- JCE provider (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

CentOS 6

L'AWS CloudHSM non supporta CentOS 6 con la versione 3.4.4 di Client SDK.

Utilizza the section called “Versione 3.2.1” con CentOS 6 o scegli una piattaforma supportata.

CentOS 7 (7.8+)

Scarica la versione 3.4.4 del software per CentOS 7:

- AWS CloudHSM Client (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Libreria PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- OpenSSL Dynamic Engine (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- JCE provider (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

CentOS 8

Scarica la versione 3.4.4 del software per CentOS 8:

- AWS CloudHSM Client (SHA256 checksum 81639c9ec83e501709c4117ba9d98b23dea7838a206ed244c9c6cc0d65130f8c)
- Libreria PKCS #11 (SHA256 checksum 9a15daa87b8616cf03a6bf6b375f53451ef448dbc54bf2c27fbc2be7823fc633)
- JCE provider (SHA256 checksum 2b1c4208992903cf7bcc669c1392c59a64fbfc82e010c626ffa58d0cb8e9126b)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 3adbecc802e0854c23aa4b8d80540d1748903c8dba93b6c8042fb7885051c360)

> **Note**
>
> A causa della recente fine del ciclo di vita di CentOS 8, dalla prossima versione non potremo più supportare questa piattaforma.

RHEL 6

AWS CloudHSMnon supporta RedHat Enterprise Linux 6 con Client SDK versione 3.4.4.

Utilizzalo the section called “Versione 3.2.1” per RedHat Enterprise Linux 6 o scegli una piattaforma supportata.

RHEL7 (7.8+)

Scarica la versione 3.4.4 del software per RedHat Enterprise Linux 7:

- AWS CloudHSM Client (SHA256 checksum 7d61d835ae38c6ce121d102b516527f342a76ac31733768097d5cab8bc482610)
- Libreria PKCS #11 (SHA256 checksum 2099f324ff625e1a46d96c1d5084263ca1d650424d7465ead43fe767d6687f36)
- OpenSSL Dynamic Engine (SHA256 checksum 6d8e81ad1208652904fe4b6abc4f174e866303f2302a6551c3fbef617337e663)
- JCE provider (SHA256 checksum 70e3cdce143c45a76e155ffb5969841e0153e011f59eb9f2c6e6be0707030abf)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 5a702fe5e50dc6055daa723df71a0874317c9ff5844eea30104587a61097ecf4)

RHEL8 (8.3+)

Scarica la versione 3.4.4 del software per RedHat Enterprise Linux 8:

- AWS CloudHSM Client (SHA256 checksum 81639c9ec83e501709c4117ba9d98b23dea7838a206ed244c9c6cc0d65130f8c)
- Libreria PKCS #11 (SHA256 checksum 9a15daa87b8616cf03a6bf6b375f53451ef448dbc54bf2c27fbc2be7823fc633)
- JCE provider (SHA256 checksum 2b1c4208992903cf7bcc669c1392c59a64fbfc82e010c626ffa58d0cb8e9126b)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 3adbecc802e0854c23aa4b8d80540d1748903c8dba93b6c8042fb7885051c360)

Ubuntu 16.04 LTS

Scarica la versione 3.4.4 del software per Ubuntu 16.04 LTS:

- AWS CloudHSM Client (SHA256 checksum 317c92c2e0b5d60afab1beb947f053d13ddaacb994cccc2c2b898e997ece29b9)
- Libreria PKCS #11 (SHA256 checksum 91451c420c51488a022569fd32f052a3b988a2883ea4c2ac952acb61a2fea37c)
- OpenSSL Dynamic Engine (SHA256 checksum 4098771ad0e38df9bf14d50520ca49b9395f819f0387e2bc3b0e61abb5888e66)
- JCE provider (SHA256 checksum e136ff183271c2f9590a9fccb8261a7eb809506686b070e3854df1b8686c6641)
- AWS CloudHSMProgramma di gestione (SHA256 checksum cbf24a4032f393a913a9898b1b27036392104e8e05d911cab84049b2bcca2541)

> **Note**
>
> A causa dell'imminente fine del ciclo di vita di Ubuntu 16.04, dalla prossima versione intendiamo interrompere il supporto per questa piattaforma.

Ubuntu 18.04 LTS

Scarica la versione 3.4.4 del software per Ubuntu 18.04 LTS:

- AWS CloudHSM Client (SHA256 checksum cf57d5e0e95efbf032aac8887aebd59ac8cc80e97c69e7c39fdad40873374fe8)
- Libreria PKCS #11 (SHA256 checksum 428f8bdad7925db5401112f707942ee8f3ca554f4ab53fa92237996e69144d2f)
- JCE provider (SHA256 checksum 1ff17b8f7688e84f7f0bfc96383564dca598a1cab2f2c52c888d0361682f2b9e)
- AWS CloudHSMProgramma di gestione (SHA256 checksum afe253046146ed6177c520b681efc680dac1048c4a95b3d8ad0f305e79bbe93e)

Windows Server

L'AWS CloudHSM supporta versioni a 64 bit di Windows Server 2012, Windows Server 2012 R2, Windows Server 2016 e Windows Server 2019. Il software client 3.4.4 dell'AWS CloudHSM per Windows Server include i provider CNG e KSP richiesti. Per informazioni dettagliate, vedi Installazione e configurazione del client AWS CloudHSM (Windows). Scarica l'ultima versione (3.4.4) del software per Windows Server:

- AWS CloudHSM for Windows Server (SHA256 checksum d51a7db588e9121d8f0b0351606bd986e1c4de6547f2c8235200dc8a5ffbe53e)
- AWS CloudHSMProgramma di gestione (SHA256 checksum 0c12d7da9086735cdf189535937a8e036163009c5018dcaf2ee9cdbb6bd4c06f)

La versione 3.4.4 aggiunge aggiornamenti al provider JCE.

Software client dell'AWS CloudHSM

- 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.

Versioni deprecate

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 del quorum per CloudHSM CLI, l'offload SSL/TLS con JSSE, il supporto multi-slot per PKCS #11, il supporto multi-cluster/multiutente per JCE, l'estrazione delle chiavi con JCE, KeyFactory per JCE supportato, nuove configurazioni di riprova per i codici di ritorno non terminali e include una migliore stabilità e correzioni dei 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: e. SecretKeyFactory KeyFactory

CL 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 per piattaforma

- 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 della 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
  - Modalità NoPadding AES/CBC/ Encrypt e Decrypt
  - Modalità di crittografia e di decodifica AES/ECB/PKCS5Padding
  - Modalità NoPadding AES/CTR/Encrypt e Decrypt
  - Modalità NoPadding AES/GCM/Wrap and Unwrap
  - Modalità di crittografia e di decodifica DESede/ECB/PKCS5Padding
  - Modalità Desede/CBC/ Encrypt and Decrypt NoPadding
  - Modalità NoPadding AESwrap/ECB/Wrap and Unwrap
  - Modalità Wrap e Unwrap AESwrap/ECB/PKCS5Padding
  - Modalità ZeroPadding AESwrap/ECB/Wrap and Unwrap
  - Modalità Wrap e Unwrap RSA/ECB/PKCS1Padding
  - Modalità Wrap e Unwrap RSA/ECB/OAEPPadding
  - RSA/ECB/OAEP con modalità Wrap e Unwrap SHA-1 E MGF1 Padding
  - RSA/ECB/OAEP con modalità Wrap e Unwrap SHA-224 E MGF1 Padding
  - RSA/ECB/OAEP con modalità Wrap e Unwrap SHA-256 E MGF1 Padding
  - RSA/ECB/OAEP con modalità Wrap e Unwrap SHA-384 E MGF1 Padding
  - RSA/ECB/OAEP con modalità Wrap e Unwrap SHA-512 E MGF1 Padding
  - Modalità Wrap e Unwrap RSAAESWrap/ECB/OAEP Padding
  - RSAAESWrap/ECB/OAEP con modalità Wrap e Unwrap SHA-1 E MGF1 Padding
  - RSAAESWrap/ECB/OAEP con modalità Wrap e Unwrap SHA-224 E MGF1 Padding

  - RSAAESwrap/ECB/OAEP con modalità Wrap e Unwrap SHA-256 E MGF1 Padding
  - RSAAESwrap/ECB/OAEP con modalità Wrap e Unwrap SHA-384 E DMGF1 Padding
  - RSAAESwrap/ECB/OAEP con modalità Wrap e Unwrap SHA-512 E MGF1 Padding
- 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 ECC per curve NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1
  - DESede (3DES)
  - GenericSecret
  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512
- Firma/verifica
  - RASSA-PSS
  - SHA1 con RSA/PSS
  - SHA224 con RSA/PSS
  - SHA256 con RSA/PSS
  - SHA384 con RSA/PSS
  - SHA512 con RSA/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

- Tipo di chiavi
  - 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 curve NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1.
  - DESede (3DES)
  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifre (solo crittografia e decodifica)
  - AES/GCM/ NoPadding
  - AES/BCE/ NoPadding
  - AES/CBC/PKCS5/NoPadding
  - Desede/BCE/ NoPadding
  - DESede/CBC/PKCS5Padding
  - AES/CTR/ NoPadding
  - RSA/ECB/PKCS1Padding
  - RSA/ECB/OAEPPadding
  - RSA/ECB/OAEP con Padding SHA-1 e MGF1
  - RSA/ECB/OAEP con Padding SHA-224 e MGF1
  - RSA/ECB/OAEP con Padding SHA-256 e MGF1
  - RSA/ECB/OAEP con Padding SHA-384 e MGF1
  - RSA/ECB/OAEP con Padding SHA-512 e MGF1
- File digest
  - SHA-1
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- Firma/verifica
  - 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

- Aggiunge il supporto per la firma/verifica ECDSA con le curve P-256, P-384 e secp256k1.
- Aggiunto supporto per le piattaforme: Amazon Linux, Amazon Linux 2, Centos 7.8+, RHEL 7.9+.
- Aggiunto supporto per la versione OpenSSL 1.0.2.
- Maggiore stabilità e correzione dei bug.

Provider JCE

- Tipo di chiavi
  - 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 curve NIST secp256r1 (P-256), secp384r1 (P-384) e secp256k1.
  - DESede (3DES)

  - HMAC - con supporto hash SHA1, SHA224, SHA256, SHA384, SHA512.
- Cifre (solo crittografia e decodifica)
  - AES/GCM/ NoPadding
  - AES/BCE/ NoPadding
  - AES/CBC/PKCS5/NoPadding
  - Desede/BCE/ NoPadding
  - DESede/CBC/PKCS5Padding
  - AES/CTR/ NoPadding
  - RSA/ECB/PKCS1Padding
  - RSA/ECB/OAEPPadding
  - RSA/ECB/OAEP con Padding SHA-1 e MGF1
  - RSA/ECB/OAEP con Padding SHA-224 e MGF1
  - RSA/ECB/OAEP con Padding SHA-256 e MGF1
  - RSA/ECB/OAEP con Padding SHA-384 e MGF1
  - RSA/ECB/OAEP con Padding SHA-512 e MGF1
- File digest
  - SHA-1
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- Firma/verifica
  - 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

- ECDSA - curve 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 crittografare/decodificare

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 CKM_RSA_PKCS_OAEP Wrap/Unwrap
- 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:
    - Firma RSA utilizza RSA PKCS con SHA1/224/256/384/512 e RSA PSS
    - 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 Offload SSL/TLS su CentOS 8.3+, Red Hat Enterprise Linux (RHEL) 8.3 e Ubuntu 18.04 LTS, incluso NGINX 1.19 (per suite di crittografia selezionate).

  Per ulteriori informazioni, vedi Usare Offload SSL/TLS su Linux.

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.

Tipo di chiavi:

- 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 3.4.3

La versione 3.4.3 aggiunge aggiornamenti al provider JCE.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- Aggiunta l'autenticazione 2FA per operatori crittografici (CO). Per ulteriori informazioni, vedi Gestione dell'autenticazione a due fattori per operatori di crittografia.
- Supporto della piattaforma rimosso 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.
- Supporto della piattaforma rimosso per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto per Linux.

OpenSSL Dynamic Engine

- Versione aggiornata per coerenza
- Supporto della piattaforma rimosso per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto per Linux.

Provider JCE

- Maggiore stabilità e correzione dei bug.
- Supporto della piattaforma rimosso per RedHat Enterprise Linux 6 e CentOS 6. Per ulteriori informazioni, vedi Supporto per 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'implementazione della libreria PKCS #11 e lo standard PKCS #11 nell'AWS CloudHSM, nuove piattaforme e altri miglioramenti.

Software client dell'AWS CloudHSM

- Aggiunta supporto della piattaforma per CentOS 8, RHEL 8 e Ubuntu 18.04 LTS. Per ulteriori informazioni, vedi ???.

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, vedi ???.

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, vedi ???.

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.

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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

Software client dell'AWS CloudHSM

- 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.

Software client dell'AWS CloudHSM

- 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 wrap delle chiavi e unwrap delle chiavi.
  - 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 con zero padding e wrap delle chiavi AES senza padding. Per ulteriori informazioni, vedi wrapping delle chiavi AES.
- È possibile configurare la lunghezza del salt per le firme RSA-PSS. Per informazioni su come utilizzare questa funzionalità, consulta Configurable salt length for RSA-PSS signatures on. 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. Ciò significa che `Cipher.WRAP_MODE` e `Cipher.UNWRAP_MODE` non funzionano più con i meccanismi AES/ECB e 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 eseguito il wrapping delle chiavi con la versione client 3.0.0 o precedente, allora in 3.1.1 devi usare AESWrap/ECB/PKCS5Padding per annullare il wrapping delle 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 eseguire iterazioni su tutte le chiavi disponibili, vedi Find all keys on. 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.

Rilasci E. nd-of-life

L'AWS CloudHSM notifica la fine del ciclo di vita delle versioni non più compatibili con il servizio. Per preservare la sicurezza della tua applicazione, ci riserviamo il diritto di rifiutare attivamente le connessioni delle end-of-life versioni.

- Al momento non sono end-of-life disponibili versioni dell'SDK del client.

Cronologia dei documenti

Questo argomento descrive gli aggiornamenti importanti alla Guida per l'utente per AWS CloudHSM.

Argomenti

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 Client SDK dell'AWS CloudHSM

Modifica	Descrizione	Data
Aggiunta nuova versione	Rilasciata la versione AWS CloudHSM client 5.11.0.	17 gennaio 2024
Aggiunta nuova versione	Lancio versione 5.10.0 del client AWS CloudHSM.	28 luglio 2023
Aggiunta nuova versione	Lancio versione 5.9.0 del client AWS CloudHSM.	23 maggio 2023
Aggiunta nuova versione	Lancio versione 5.8.0 del client AWS CloudHSM.	16 marzo 2023
Aggiunta nuova versione	Lancio versione 5.7.0 del client AWS CloudHSM.	16 novembre 2022
Aggiunta nuova versione	AWS CloudHSMRilascio 5.6.0 del client.	1 settembre 2022

Aggiunta nuova versione	Lancio versione 5.5.0 del client AWS CloudHSM.	13 maggio 2022
Aggiunta nuova versione	Lancio versione 5.4.2 del client AWS CloudHSM.	18 marzo 2022
Aggiunta nuova versione	Lancio versione 5.4.1 del client AWS CloudHSM.	10 febbraio 2022
Aggiunta nuova versione	Lancio versione 5.4.0 del provider JCE AWS CloudHSM per piattaforme Windows.	1 febbraio 2022
Aggiunta nuova versione	Lancio versione 5.4.0 del client AWS CloudHSM, che aggiunge il supporto iniziale per il provider JCE per tutte le piattaforme Linux.	28 gennaio 2022
Aggiunta nuova versione	Lancio versione 5.3.0 del client AWS CloudHSM.	3 gennaio 2022
Aggiunta nuova versione	Lancio versione 3.4.4 del client AWS CloudHSM.	3 gennaio 2022
Aggiunta nuova versione	Lancio 3.4.3 del client AWS CloudHSM.	20 dicembre 2021
Aggiunta nuova versione	Lancio versione 3.4.2 del client AWS CloudHSM.	15 dicembre 2021
Aggiunta nuova versione	Lancio versione 3.4.1 del client AWS CloudHSM.	10 dicembre 2021
Aggiunta nuova versione	Lancio versione 5.2.1 del client AWS CloudHSM.	4 ottobre 2021
Aggiunta nuova versione	Lancio versione 3.4.0 del client AWS CloudHSM.	25 agosto 2021

Aggiunta nuova versione	Lancio versione 5.2.0 del client AWS CloudHSM.	3 agosto 2021
Aggiunta nuova versione	Lancio versione 3.3.2 del client AWS CloudHSM.	2 luglio 2021
Aggiunta nuova versione	Lancio versione 5.1.0 del client AWS CloudHSM.	1 giugno 2021
Aggiunta nuova versione	Lancio versione 3.3.1 del client AWS CloudHSM.	26 Aprile 2021
Aggiunta nuova versione	Lancio versione 5.0.1 del client AWS CloudHSM.	8 aprile 2021
Aggiunta nuova versione	Lancio versione 5.0.0 del client AWS CloudHSM.	12 marzo 2021
Aggiunti nuovi contenuti	Aggiunta interfaccia endpoint VPC, una funzionalità AWS che ti permette di creare una connessione privata tra il tuo VPC e AWS CloudHSM senza richiedere l'accesso tramite Internet, mediante dispositivo NAT, una connessione VPN o una connessione AWS Direct Connect.	10 febbraio 2021
Aggiunta nuova versione	Lancio versione 3.3.0 del client AWS CloudHSM.	3 febbraio 2021
Aggiungi nuovi contenuti	È stata aggiunta la conservazione gestita dei backup, una funzionalità che elimina automaticamente i vecchi backup.	18 novembre 2020

Aggiungi nuovi contenuti	È stato aggiunto un rapporto di conformità che analizza l'implementazione della libreria PKCS #11 del Client SDK 3.2.1 AWS CloudHSM con lo standard PKCS #11.	29 ottobre 2020
Aggiunta nuova versione	Lancio versione 3.2.1 del client 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	Lancio versione 3.2.0 del client AWS CloudHSM.	31 agosto 2020
Aggiunta nuova versione	Lancio versione 3.1.2 del client AWS CloudHSM.	30 luglio 2020
Aggiunta nuova versione	Lancio versione 3.1.1 del client AWS CloudHSM.	3 giugno 2020
Aggiunta nuova versione	Lancio versione 3.1.0 del client AWS CloudHSM.	21 maggio 2020
Aggiunta nuova versione	Lancio versione 3.0.1 del client AWS CloudHSM.	20 aprile 2020
Aggiunta nuova versione	Lancio versione 3.0.0 del client AWS CloudHSM per la piattaforma Windows Server.	30 ottobre 2019
Aggiunta nuova versione	Lancio versione 3.0.0 del client AWS CloudHSM per tutte le piattaforme, ad eccezione di Windows.	22 ottobre 2019

Aggiunta nuova versione	Lancio versione 2.0.4 del client AWS CloudHSM.	26 agosto 2019
Aggiunta nuova versione	Lancio versione 2.0.3 del client AWS CloudHSM.	13 maggio 2019
Aggiunta nuova versione	Lancio versione 2.0.1 del client AWS CloudHSM.	21 marzo 2019
Aggiunta nuova versione	Lancio versione 2.0.0 del client AWS CloudHSM.	6 febbraio 2019
Aggiunta del supporto regionale	È stato aggiunto il AWS CloudHSM supporto per le regioni UE (Stoccolma) e AWS GovCloud (Stati Uniti orientali).	19 dicembre 2018
Aggiunta nuova versione	Lancio versione 1.1.2 del client AWS CloudHSM 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	Lancio versione 1.1.2 del client AWS CloudHSM per piattaforme Linux.	8 novembre 2018
Aggiunta del supporto regionale	Aggiunto il supporto di AWS CloudHSM per le regioni UE (Parigi) e Asia Pacifico (Seoul).	24 ottobre 2018
Aggiunti nuovi contenuti	Aggiunta possibilità di eliminare e ripristinare i backup di AWS CloudHSM.	10 settembre 2018

Aggiunti nuovi contenuti	È stata aggiunta la consegna automatica dei log di controllo ad Amazon CloudWatch Logs.	13 agosto 2018
Aggiunti nuovi contenuti	Aggiunta possibilità di copiare un backup del cluster AWS CloudHSM tra regioni.	30 luglio 2018
Aggiunta del supporto regionale	Aggiunta del supporto di AWS CloudHSM per la regione UE (Londra).	13 giugno 2018
Aggiunti nuovi contenuti	Aggiunta del supporto alla libreria e al client AWS CloudHSM per Amazon Linux 2, Red Hat Enterprise Linux (RHEL) 6, Red Hat Enterpris e Linux (RHEEL) 7, CentOS 6, CentOS 7 e Ubuntu 16.04 LTS.	10 maggio 2018
Aggiunta nuova versione	Aggiunta di un client AWS CloudHSM Windows.	30 Aprile 2018

Aggiornamenti precedenti

Nella seguente tabella sono descritte le modifiche importanti apportate alla AWS CloudHSM prima del 2018.

Modifica	Descrizione	Data
Nuovo contenuto	Aggiunta dell'autenticazione del quorum (controllo accessi "M of N") per crypto officer (CO). Per ulteriori informazi oni, vedi Utilizzo di CloudHSM Management Utility (CMU)	9 novembre 2017

Modifica	Descrizione	Data
	per gestire l'autenticazione del quorum (controllo dell'accesso "M of N").	
Aggiornamento	Aggiunta di documentazione sull'utilizzo dello strumento a riga di comando key_mgmt_util. Per ulteriori informazioni, vedi Riferimento al comando key_mgmt_util.	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.