SHA (Secure Hash Algorithm) 256 per Customer Voice Portal (CVP)

Sommario

Introduzione

Questo documento descrive la procedura per utilizzare SHA256 con CVP.

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- CVP
- Certificati

Componenti usati

Le informazioni di questo documento si basano su CVP 10.5.

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Premesse

A partire da gennaio 2016 tutti i browser hanno rifiutato i certificati firmati SHA1. Il rendering dei servizi richiesti non è stato eseguito correttamente, a meno che non si passi da SHA1 a SHA256.

Con i recenti sviluppi negli algoritmi computazionali e la capacità di calcolo esplosiva, SHA1 è diventato più debole giorno dopo giorno. Ciò ha portato a un degrado fondamentale della resistenza alla collisione dello SHA1 e alla sua morte.

Configurazione

Procedura di scambio certificati tra CVP Operations Console (OAMP):

Su OAMP

Passaggio 1. Esportare il certificato OAMP.

c:\Cisco\CVP\jre\bin\keytool.exe -export -v -keystore .keystore -storetype JCEKS -alias certificato_oamp -file_sicurezza_76.cer

Passaggio 2. Copiare il certificato OAMP nel server di chiamata e importarlo.

c:\Cisco\CVP\jre\bin\keytool.exe -import -trustcacerts -keystore .keystore -storetype JCEKS -alias orm_oamp_certificate -file oamp_security_76.cer

Server On Call

Passaggio 1. Esportare CALLSERVER CERT.

c:\Cisco\CVP\jre\bin\keytool.exe -export -v -keystore .ormkeystore -storetype JCEKS -alias certificato_orm -file orm_security_108.cer

Passaggio 2. Copiare il CERTIFICATO CALLSERVER in OAMP e importarlo.

c:\Cisco\CVP\jre\bin\keytool.exe -import -trustcacerts -keystore .keystore -storetype JCEKS -alias orm_certificate -file orm_security_108.cer

Passaggio 3. Esportare il certificato del modulo nel keystore di Call Server.

C:\Cisco\CVP\conf\security>c:\Cisco\CVP\jre\bin\keytool.exe -import -trustcacerts -keystore .keystore -storetype JCEKS -alias vxml_orm_certificate -file orm_security_108.cer

Verifica

Èpossibile verificare se la comunicazione protetta viene stabilita tra i componenti. Passare a **Pagina OAMP > Gestione dispositivi > <server gestito> > Statistiche**

Ènecessario visualizzare le statistiche.

Èpossibile utilizzare JConsole per stabilire una connessione se la protezione è impostata correttamente:

Passaggio 1. **c:\Cisco\CVP\conf\orm_jmx.conf** su OAMP ha il seguente aspetto:

```
javax.net.debug = all
com.sun.management.jmxremote.ssl.need.client.auth = false
com.sun.management.jmxremote.authenticate = false
com.sun.management.jmxremote.port = 2099
com.sun.management.jmxremote.ssl = true
javax.net.ssl.keyStore=C:\Cisco\CVP\conf\security\.ormkeystore
```

```
javax.net.ssl.keyStorePassword=<local security password>
```

Passaggio 2. Aprire jconsole dal comando. Utilizzare il comando:

C:\Cisco\CVP\jre\bin>jconsole.exe -J-Djavax.net.ssl.trustStore=C:\Cisco\CVP\conf\security\.keystore -J-Djavax.net.ssl.trustStorePassword=<password di sicurezza oamp/client jconsole> -J-Djavax.net.ssl.keyStore=C:\Cisco\CVP\conf\security\.keystore -J-Djavax.net.ssl.keyStorePassword=<password di sicurezza oamp/client jconsole> -J-Djavax.net.ssl.keyStoreType=JCEKS -debug -J Djavax.net.ssl.trustStoreType=JCEKS

Chiave nel campo **<managed server ip>:<secure jmx port eg:2099>** in **Remote Process**.

> **Nota**: JConsole deve connettersi senza richiedere all'applicazione di ignorare il metodo sicuro.

Passaggio 3. Wireshark durante il richiamo della connessione jconsole. L'acquisizione fornisce informazioni dettagliate sui dettagli negoziati durante l'handshake di protezione.

Tracce in JMX

L'implementazione di JMX utilizza java.util.logging per registrare le tracce di debug. Molte di queste tracce riguardano classi interne non esposte, ma possono aiutare a comprendere cosa sta succedendo con l'applicazione.

L'implementazione JMX prevede due tipi di logger:

- **`javax.management.\*`**: tutti i logger correlati all'API JMX
- **`javax.management.remote.\*`**: logger correlati in modo specifico all'API remota JMX

Qui è possibile trovare una descrizione più completa dei logger JMX.

Èpossibile attivare le tracce JMX in due modi:

- Staticamente, con l'utilizzo di un file **logging.properties**
- Dinamicamente, con l'uso di un MBean di JMXTrace. In Java SE 6, è possibile eseguire questa operazione per un'applicazione, anche se il connettore JMX non è abilitato sulla riga di comando.

Utilizzare un file logging.properties

Avviare l'applicazione con i seguenti flag:

```
java -Djava.util.logging.config.file=<logging.properties> ....
```

dove **logging.properties** attiva le tracce per i logger JMX:

```
handlers= java.util.logging.ConsoleHandler
.level=INFO
```

```
java.util.logging.FileHandler.pattern = %h/java%u.log
java.util.logging.FileHandler.limit = 50000
java.util.logging.FileHandler.count = 1
java.util.logging.FileHandler.formatter = java.util.logging.XMLFormatter

java.util.logging.ConsoleHandler.level = FINEST
java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter

// Use FINER or FINEST for javax.management.remote.level - FINEST is
// very verbose...
//
javax.management.level=FINEST
javax.management.remote.level=FINER
```