適用於客戶語音入口網站(CVP)的安全雜湊演演算法(SHA)256

目錄

簡介

本檔案介紹將SHA256與CVP一起使用的程式。

必要條件

需求

思科建議您瞭解以下主題：

- CVP
- 憑證

採用元件

本檔案中的資訊是根據CVP 10.5。

本文中的資訊是根據特定實驗室環境內的裝置所建立。文中使用到的所有裝置皆從已清除（預設）的組態來啟動。如果您的網路運作中，請確保您瞭解任何指令可能造成的影響。

背景資訊

從2016年1月開始，所有瀏覽器都拒絕了SHA1簽名證書。除非您從SHA1移動到SHA256，否則無法正確呈現請求的服務。

隨著計算演算法的發展以及爆炸計算能力的提高，SHA1逐漸變弱。這導致SHA1的基本退化碰撞抵抗並最終死亡。

設定

CVP操作控制檯(OAMP)之間的證書交換過程：

在OAMP上

步驟1.匯出OAMP證書。

c:\Cisco\CVP\jre\bin\keytool.exe -export -v -keystore .keystore -storetype JCEKS -alias oamp_certificate -file oamp_security_76.cer

步驟2.將OAMP憑證複製到Callserver並進行匯入。

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

通話伺服器

步驟1.匯出CALLSERVER證書。

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

步驟2.將CALLSERVER CERT複製到OAMP並匯入。

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

步驟3.匯出呼叫伺服器金鑰庫中的表單證書。

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

驗證

您可以驗證元件之間是否建立了安全通訊。導航到**OAMP Page > Device management > <managed server> > Statistics**

必須顯示統計資訊。

如果安全設定正確，可以使用JConsole建立連線：

步驟1. **c:\Cisco\CVP\conf\orm_jmx.conf** on OAMP如下所示：

```
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>
```

步驟2.從命令開啟jconsole。 使用命令:

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

Remote Process**欄位中的<managed server ip>:<secure jmx port eg:2099>中的密鑰。**

> **注意**:JConsole必須在不提示的情況下進行連線，應用程式才能繞過安全方法。

步驟3.呼叫jconsole連線時Wireshark。通過捕獲，您可以深入瞭解在安全握手時協商的詳細資訊。

JMX中的跟蹤

JMX的實現使用java.util.logging來記錄調試跟蹤。這些跟蹤中有許多涉及內部未暴露的類，但它們可幫助您瞭解應用程式的情況。

JMX實現包含兩組記錄器：

- **javax.management.***:與JMX API相關的所有記錄器
- **javax.management.remote.***:與JMX遠端API具體相關的記錄器

您可以在此處找到有關JMX記錄器的更完整的說明。

您可以通過兩種不同方式啟用JMX跟蹤：

- 靜態使用**logging**.properties檔案
- 使用JMXTracing MBean動態執行。在Java SE 6中，即使命令列上未啟用JMX聯結器，也可以對應用程式執行此操作。

使用logging.properties檔案

使用以下標誌啟動應用程式：

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

其中**logging.properties**為JMX記錄器啟用跟蹤：

```
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
```