Customer Voice Portal的(CVP)安全哈希算法(SHA) 256

Contents

Introduction

本文描述程序以CVP使用SHA256。

Prerequisites

Requirements

Cisco 建议您了解以下主题：

- CVP
- 证书

Components Used

本文的信息根据CVP 10.5。

The information in this document was created from the devices in a specific lab environment.All of the devices used in this document started with a cleared (default) configuration.如果您的网络实际，请保证您了解所有命令的潜在影响。

背景信息

启动2016年1月所有浏览器拒绝了SHA1签名的证书。除非从SHA1移动到SHA256，这没有正确地回报被请求的服务。

使用在算法以及易爆的计算容量的最新发展成果， SHA1天天变得更弱。这导致了SHA1和最后的困境的根本降低冲突电阻。

Configure

认证在CVP操作之间的交换程序控制(OAMP) ：

在OAMP

步骤1.导出OAMP CERT。

c:\Cisco\CVP\jre\bin\keytool.exe -导出- v - keystore .keystore - storetype JCEKS -别名 oamp_certificate -文件oamp_security_76.cer

步骤2.复制OAMP认证到Callserver并且导入。

c:\Cisco\CVP\jre\bin\keytool.exe -导入- trustcacerts - keystore .keystore - storetype JCEKS -别名 orm_oamp_certificate -文件oamp_security_76.cer

在呼叫服务器上

步骤1.导出CALLSERVER CERT。

c:\Cisco\CVP\jre\bin\keytool.exe -导出- v - keystore .ormkeystore - storetype JCEKS -别名 orm_certificate -文件orm_security_108.cer

步骤2.复制CALLSERVER CERT到OAMP并且导入。

c:\Cisco\CVP\jre\bin\keytool.exe -导入- trustcacerts - keystore .keystore - storetype JCEKS -别名 oamp_orm_certificate -文件orm_security_108.cer

步骤3.导出在呼叫服务器keystore的orm认证。

C:\Cisco\CVP\conf\security > c:\Cisco\CVP\jre\bin\keytool.exe -导入- trustcacerts - keystore .keystore - storetype JCEKS -别名vxml_orm_certificate -文件orm_security_108.cer

Verify

如果安全通信建立在组件之间，您能验证。连接对**OAMP页>设备管理> <managed server> >统计数据**

必须显示Stats。

如果安全适当地设置，您能使用JConsole建立连接：

第1.步**在**OAMP的**c:\Cisco\CVP\conf\orm _jmx.conf**看起来象：

```
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>
```

步骤2.打开从命令的jconsole。 请使用命令：

C:\Cisco\CVP\jre\bin >jconsole.exe - J-Djavax.net.ss l.trustStore= C:\Cisco\CVP\conf\security\ .keystore - J-Djavax.net.ss l.trustStorePassword =<oamp安全密码/jconsole client> - J-Djavax.net.ss l.keyStore = C:\Cisco\CVP\conf\security\ .keystore - J-Djavax.net.ss l.keyStorePassword =<oamp安全密码/jconsole client> - J-Djavax.net.ss l.keyStoreType=JCEKS - 调试- J-Djavax.net.ss l.trustStoreType =JCEKS

在**<managed服务器ip>的**键：**<secure在远程进程**字段的**jmx端口eg:2099>**。

> **Note**: JConsole必须连接，不用提示输入应用程序绕过安全的方法。

步骤3. Wireshark，当jconsole连接被调用时。捕获给予您洞察力到协商的详细资料，当安全握手时。

在JMX的跟踪

JMX用途java.util.logging的实施记录调试跟踪。许多这些跟踪关系到内部未曝光的组，但是他们可帮助您了解怎么回事与您的应用程序。

JMX实施有两套日志记录器：

- **javax.management. \ *** 日志记录器与JMX API有关
- **javax.management.remote. \ *** 与JMX远程API有关

您能找到JMX日志记录器更多完整说明这里。

您能激活JMX跟踪用两个不同的方式：

- 静态，与使用**logging.properties文件**
- 动态地，与使用JMXTracing MBean。在Java SE 6中，您能为执行此应用程序，即使JMX连接器在line命令没有被启用。

请使用logging.properties文件

运行您的与这些标志位的应用程序：

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

那里**logging.properties激活**JMX日志记录器的跟踪：

```
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
```