设置示例

设置示例概述

本节介绍在电话与设置服务器之间传输配置文件的程序示例。

有关创建配置文件的信息，请参阅设置脚本。

基本重新同步

本节介绍电话的基本重新同步功能。

TFTP 重新同步

电话支持适用于检索配置文件的多个网络协议。最基本的配置文件传输协议是 TFTP (RFC1350)。TFTP 广泛用于在专用 LAN 网络内设置网路设备。尽管不建议部署跨 Internet 的远程终端，但 TFTP 对于小型企业内的部署而言非常方便，既适合内部预设置，又利于开发和测试。请参阅内部设备预设置，了解有关内部预设置的详细信息。在下面的程序中，从 TFTP 服务器下载文件后会修改配置文件。

过程

步骤 1 在 LAN 环境中，将 PC 和电话连接到集线器、交换机或小型路由器。

步骤 2 在 PC 上安装并激活 TFTP 服务器。

步骤 3 如示例中所示使用文本编辑器创建配置文件，将 GPP_A 的值设置为 12345678。

```
<flat-profile>
  <GPP_A> 12345678
  </GPP_A>
</flat-profile>
```

步骤 4 在 TFTP 服务器的根目录下，以 `basic.txt` 为名保存配置文件。

您可以验证 TFTP 服务器的配置是否正确：使用 TFTP 客户端（而非电话）请求 `basic.txt` 文件。最好从设置服务器使用在单独主机上运行的 TFTP 客户端。

步骤 5 用 PC web 浏览器打开管理员/高级配置页面。例如，如果电话的 IP 地址是 192.168.1.100：

```
http://192.168.1.100/admin/advanced
```

步骤 6 选择**语音** > **设置**选项卡，然后检查通用参数 GPP_P 到 GPP_A 的值。这些值应该为空。

步骤 7 在 web 浏览器窗口中打开重新同步 URL，将测试电话重新同步到 `basic.txt` 配置文件。

如果 TFTP 服务器的 IP 地址是 192.168.1.200，命令应该与以下示例类似：

```
http://192.168.1.100/admin/resync?tftp://192.168.1.200/basic.txt
```

当电话收到此命令时，位于 192.168.1.100 的设备会从 IP 地址为 192.168.1.200 的 TFTP 服务器请求文件 `basic.txt`。然后，电话将解析下载的文件，并将 GPP_A 参数的值更新为 12345678。

步骤 8 验证参数是否已正确更新：在 PC web 浏览器上刷新配置页面，然后选择**语音** > **设置**选项卡。

GPP_A 参数现应包含值 12345678。

使用系统日志记录消息

当设备即将与设置服务器重新同步以及重新同步完成或失败后，电话会将系统日志消息发送到指定的系统日志服务器。为识别此服务器，您可以访问电话管理网页（请参阅访问电话网页），选择**语音** > **系统**，然后识别**可选网络配置**部分**系统日志服务器**参数中的服务器。将系统日志服务器 IP 地址配置到设备并查看剩余步骤中生成的消息。

过程

步骤 1 在本地 PC 上安装并激活系统日志服务器。

步骤 2 将 PC IP 地址编程到配置文件的 Syslog_Server 参数并提交更改：

```
<Syslog_Server>192.168.1.210</Syslog_Server>
```

步骤 3 单击**系统**选项卡，并在 Syslog_Server 参数中输入本地系统日志服务器的值。

步骤 4 重复执行 TFTP 重新同步 ， 第 1 页中的重新同步操作。

重新同步期间，设备会生成两条系统日志消息。第一条消息说明正在处理请求。第二条消息说明重新同步成功或者失败。

步骤 5 验证您的系统日志服务器收到的消息是否与如下消息类似：

```
CP-88xx-3PCC 00:0e:08:ab:cd:ef -- Successful resync tftp://192.168.1.200/basic.txt
```

使用系统日志服务器的 IP 地址设定 Debug_Server 参数（而非 Syslog_Server 参数），并将 Debug_Level 设定为 0 至 3 之间的值（3 最详细），可以获得详细的消息：

```
<Debug_Server>192.168.1.210</Debug_Server>
<Debug_Level>3</Debug_Level>
```

可以使用以下参数来配置这些消息的内容：

- Log_Request_Msg
- Log_Success_Msg
- Log_Failure_Msg

如果任何参数被清除，则不会生成相应的系统日志消息。

自动重新同步设备

设备可以定期重新同步到设置服务器，以确保在服务器上进行的配置文件更改会传播到终端设备（而不是将显式重新同步请求发送到端点）。

要使电话定期重新同步到服务器，使用 Profile_Rule 参数定义配置文件 URL，并使用 Resync_Periodic 参数定义重新同步周期。

开始之前

访问电话管理网页。请参阅访问电话网页。

过程

步骤 1 选择**语音** > **部署**。

步骤 2 定义 Profile_Rule 参数。本例假设 TFTP 服务器 IP 地址为 192.168.1.200。

步骤 3 在**重新同步周期**字段中，输入一个较小的值以进行测试，例如 **30** 秒。

步骤 4 单击**提交所有更改**。

使用新的参数设置，电话每分钟会执行两次到 URL 指定的配置文件的重新同步。

步骤 5 查看系统日志跟踪记录中生成的消息（如使用系统日志记录消息，第 2 页部分所述）。

步骤 6 确保**重置时重新同步**字段设置为**是**。

```
<Resync_On_Reset>Yes</Resync_On_Reset>
```

步骤 7 重启电话以强制其重新同步到设置服务器。

如果出于任何原因重新同步操作失败，比如服务器不响应，设备在重试重新同步之前需等待（**重新同步错误重试延迟**中配置的秒数）。如果**重新同步错误重试延迟**设置为0，在尝试重新同步失败后，电话不会尝试重新同步。

步骤 8 （可选）将**重新同步错误重试延迟**字段的值设定为较小的数值，例如 **30**。

```
<Resync_Error_Retry_Delay>30</Resync_Error_Retry_Delay>
```

步骤 9 禁用 TFTP 服务器并查看系统日志输出中的结果。

唯一的配置文件、宏扩展和 HTTP

在每个电话必须为某些参数（例如 User_ID 或 Display_Name）配置不同值的部署中，服务提供商可以为每个部署的设备创建一个唯一的配置文件，并将这些配置文件托管在设置服务器上。然后，必须配置每部电话，以根据预先确定的配置文件命名约定，重新同步到自己的配置文件。

通过使用内置变量的宏扩展，配置文件 URL 语法可包含特定于每部电话的标识信息，例如 MAC 地址或序列号。有了宏扩展之后，便无需在每个配置文件的多个位置指定这些值。

在配置文件规则应用于电话之前，规则会进行宏扩展。宏扩展控制多个值，例如：

- $MA 将扩展到设备的 12 位 MAC 地址（使用小写十六进制数字）。例如 000e08abcdef。
- $SN 将扩展到设备序列号。例如 88012BA01234。

其他值可以通过这种方式宏扩展，包括 GPP_A 到 GPP_P 的所有通用参数。此过程的示例可参见TFTP 重新同步，第 1 页。宏扩展不限于 URL 文件名称，但也可以应用到配置文件规则参数的任何部分。这些参数作为 $A 到 $P 引用。如需适用于宏扩展的完整变量列表，请参阅宏扩展变量。

在此练习中，配置文件特定于 TFTP 服务器上设置的电话。

练习：在 TFTP 服务器上设置特定 IP 电话配置文件

过程

步骤 1 从产品标签获取电话的 MAC 地址。（MAC 地址是使用数字以及小写十六进制数字的号码，例如 000e08aabbcc。

步骤 2 复制 `basic.txt` 配置文件（如TFTP 重新同步，第 1 页中所述）到名为 `CP-xxxx-3PCC macaddress.cfg` 的新文件（将 `xxxx` 替换为型号，`macaddress` 替换为电话的 MAC 地址）。

步骤 3 移动 TFTP 服务器虚拟根目录中的新文件。

步骤 4 访问电话管理网页。请参阅访问电话网页。

步骤 5 选择**语音** > **部署**。

步骤 6 在**配置文件规则**字段输入 `tftp://192.168.1.200/CP-8841-3PCC $MA.cfg`。

```
<Profile_Rule>
  tftp://192.168.1.200/CP-8841-3PCC$MA.cfg
</Profile_Rule>
```

步骤 7 单击**提交所有更改**。这样会立即重新启动并重新同步。

下次重新同步时，电话会将 $MA 宏表达式扩展到它的 MAC 地址，从而检索新文件。

HTTP GET 重新同步

与 TFTP 相比，HTTP 提供一种更为可靠的重新同步机制，因为 HTTP 会建立 TCP 连接，而 TFTP 则使用可靠性较低的 UDP。此外，与 TFTP 服务器相比，HTTP 服务器的过滤和日志记录功能更强。

在客户端，电话不需要在服务器上进行任何特殊的配置设置，即可使用 HTTP 重新同步。使用 HTTP 搭配 GET 方法的 Profile_Rule 参数语法与用于 TFTP 的语法类似。如果标准 web 浏览器可以从您的 HTTP 服务器检索配置文件，电话应该也能够执行此操作。

练习：*HTTP GET* 重新同步

过程

步骤 1 在本地 PC 或其他可访问主机上安装 HTTP 服务器。

可从 Internet 下载开放源码 Apache 服务器。

步骤 2 将 `basic.txt` 配置文件（如 TFTP 重新同步，第 1 页中所述）安装到所安装服务器的虚拟根目录。

步骤 3 要验证服务器安装和 `basic.txt` 文件访问权限是否适当，请使用 Web 浏览器访问配置文件。

步骤 4 将测试电话的 Profile_Rule 改为指向 HTTP 服务器（而非 TFTP 服务器），从而定期下载其配置文件。

例如，假设 HTTP 服务器位于 192.168.1.300，则输入以下值：

```
<Profile_Rule>
http://192.168.1.200/basic.txt
</Profile_Rule>
```

步骤 5 单击**提交所有更改**。这样会立即重新启动并重新同步。

步骤 6 查看电话发送的系统日志消息。定期重新同步现在应会从 HTTP 服务器获取配置文件。

步骤 7 在 HTTP 服务器日志中，观察标识测试电话的信息如何显示在用户代理的日志中。

此信息都应包括制造商、产品名称、当前固件版本和序列号。

通过 Cisco XML 设置

对于此处指定为 xxxx 的每个电话，您可以通过 Cisco XML 功能进行设置。

您可以将 XML 对象通过 SIP Notify 数据包发送到电话，或通过 HTTP Post 发送到电话的 CGI 接口：`http://IPAddressPhone/CGI/Execute`。

CP-xxxx-3PCC 扩展了 Cisco XML 功能，现支持通过 XML 对象进行设置：

```
<CP-xxxx-3PCCExecute>
        <ExecuteItem URL=Resync:[profile-rule]/>
</CP-xxxx-3PCCExecute>
```

电话收到 XML 对象后，会从 [profile-rule] 下载设置文件。此规则使用宏来简化 XML 服务应用程序的开发。

通过宏扩展解析 URL

服务器上具有多个配置文件的子目录为管理所部署的大量设备提供了便利的方法。配置文件URL可能包含：

- 设置服务器名称或显式 IP 地址。如果配置文件按名称标识设置服务器，电话将执行 DNS 查询以解析名称。
- 使用标准语法 `:port` 在 URL 中指定的非标准服务器端口，位于服务器名称之后。
- 配置文件所处服务器虚拟根目录的子目录，使用标准 URL 表示法指定，通过宏扩展管理。

例如，以下 Profile_Rule 在服务器子目录 `/cisco/config` 中，从在主机 prov.telco.com 上运行的 TFTP 服务器请求配置文件 ($PN.cfg)，侦听端口 6900 上的连接：

```
<Profile_Rule>
tftp://prov.telco.com:6900/cisco/config/$PN.cfg
</Profile_Rule>
```

每个电话的配置文件可以在通用参数中标识，其值通过使用宏扩展在通用配置文件规则中引用。

例如，假设 GPP_B 定义为 `Dj6Lmp23Q`。

Profile_Rule 具有值：

```
tftp://prov.telco.com/cisco/$B/$MA.cfg
```

当设备重新同步和宏扩展时，MAC 地址为 000e08012345 的电话将通过以下 URL 请求名称包含设备 MAC 地址的配置文件：

```
tftp://prov.telco.com/cisco/Dj6Lmp23Q/000e08012345.cfg
```

安全 HTTPS 重新同步

电话上配置了以下机制，用于通过安全的通信流程完成重新同步：

- 基本 HTTPS 重新同步
- HTTPS 与客户端证书验证
- HTTPS 客户端过滤和动态内容

基本 HTTPS 重新同步

HTTPS 会向 HTTP 添加 SSL 用于远程设置，以便:

- 电话可以验证设置服务器。
- 设置服务器可以验证电话。
- 确保电话和设置服务器之间所交换信息的机密性。

SSL 会使用在电话与设置服务器之间预安装的公共/专用密钥对，为电话与服务器之间的每个连接生成和交换秘密（对称）密钥。

在客户端，不需要在服务器上进行任何特殊的配置设置，电话即可使用 HTTPS 重新同步。使用 HTTPS 搭配 GET 方法的 Profile_Rule 参数语法与用于 HTTP 或 TFTP 的语法类似。如果标准 web 浏览器可以从您的 HTTPS 服务器检索配置文件，电话应该也能够执行此操作。

除安装 HTTPS 服务器外，还必须在设置服务器上安装思科签名的 SSL 服务器证书。设备无法重新同步到使用 HTTPS 的服务器，除非服务器提供思科签名的服务器证书。有关为语音产品创建签名 SSL 证书的说明，可参阅 https://supportforums.cisco.com/docs/DOC-9852。

练习：基本 HTTPS 重新同步

过程

步骤 1 通过正常的主机名转换，在 IP 地址为网络 DNS 服务器所知的主机上安装 HTTPS 服务器。

当使用开放源码 mod_ssl 软件包安装时，可以将开放源码 Apache 服务器配置为作为 HTTPS 服务器运行。

步骤 2 生成为服务器签署请求的服务器证书。对于此步骤，您可能需要安装开放源码 OpenSSL 软件包或同等软件。如果使用 OpenSSL，生成基本 CSR 文件的命令如下：

```
openssl req -new -out provserver.csr
```

此命令生成公共/专用密钥对，保存在 `privkey.pem` 文件中。

步骤 3 提交 CSR 文件 (provserver.csr) 供思科签名。

签名的服务器证书 (provserver.cert) 会随 Sipura CA 客户端根证书 spacroot.cert 一起返回。

有关详细信息，请参阅https://supportforums.cisco.com/docs/DOC-9852

步骤 4 将签名的服务器证书、专用密钥对文件和客户端根证书存储在服务器上的适当位置。

如果在 Linux 上安装 Apache，这些位置通常如下所示：

```
# Server Certificate:
SSLCertificateFile /etc/httpd/conf/provserver.cert
# Server Private Key:
SSLCertificateKeyFile /etc/httpd/conf/pivkey.pem
# Certificate Authority:
SSLCACertificateFile /etc/httpd/conf/spacroot.cert
```

步骤 5 重新启动服务器。

步骤 6 将 `basic.txt` 配置文件（如 TFTP 重新同步 ， 第 1 页中所述）安装到 HTTPS 服务器的虚拟根目录。

步骤 7 在本地 PC 上使用标准浏览器从 HTTPS 服务器下载 `basic.txt`，验证服务器操作是否适当。

步骤 8 检查服务器提供的服务器证书。

除非浏览器已预先配置为接受思科作为根CA，否则浏览器可能不会将证书识别为有效。不过，电话期望通过这种方式签署证书。

修改测试设备的 Profile_Rule 以包含对 HTTPS 服务器的引用，例如：

```
<Profile_Rule>
https://my.server.com/basic.txt
</Profile_Rule>
```

此例假设 HTTPS 服务器的名称是 **`my.server.com`**。

步骤 9 单击**提交所有更改**。

步骤 10 查看电话发送的系统日志跟踪记录。

系统日志消息应指明重新同步已从 HTTPS 服务器取得配置文件。

步骤 11 （可选） 使用电话子网上的以太网协议分析器来验证数据包是否已加密。

在此练习中，未启用客户端证书验证。电话与服务器之间的连接已加密。但是，由于知道文件名和目录位置，任何客户端都可以连接到服务器并请求文件，因此传输不安全。为确保重新同步安全，服务器还必须验证客户端，如 HTTPS 与客户端证书验证 ， 第 9 页中的练习所示。

HTTPS 与客户端证书验证

在出厂默认配置中，服务器不会从客户端请求 SSL 客户端证书。传输配置文件不安全，因为任何客户端都可以连接到服务器并请求配置文件。您可以编辑配置以启用客户端验证；在接受连接请求之前，服务器需要客户端证书来验证电话。

由于这一要求，使用缺少适当证书的浏览器无法独立测试重新同步操作。使用 ssldump 实用程序可以在测试电话和服务器之间观察到 HTTPS 连接中的 SSL 密钥交换。实用程序跟踪记录会显示客户端与服务器之间的交互。

练习：HTTPS 与客户端证书验证

过程

步骤 1 在 HTTPS 服务器上启用客户端证书验证。

步骤 2 在 Apache (v.2)，在服务器配置文件中设置以下信息：

```
SSLVerifyClient  require
```

此外，确保已如基本 HTTPS 重新同步，第 7 页的练习中所示存储 spacroot.cert。

步骤 3 重新启动 HTTPS 服务器，从电话查看系统日志跟踪记录。

现在，与服务器的每次重新同步都会执行对称验证，以便在传输配置文件之前验证服务器证书和客户端证书。

步骤 4 使用 ssldump 捕获电话和 HTTPS 服务器之间的重新同步连接。

如果在服务器上正确启用了客户端证书验证，ssldump 跟踪会在包含配置文件的加密数据包之前显示证书的对称交换（首先服务器到客户端，然后客户端到服务器）。

启用客户端验证后，只有具有匹配有效客户端证书的 MAC 地址的电话能从设置服务器请求配置文件。服务器拒绝来自普通浏览器或其他未经授权设备的请求。

HTTPS 客户端过滤和动态内容

如果 HTTPS 服务器配置为要求客户端证书，证书中的信息会标识重新同步电话，并为其提供正确的配置信息。

HTTPS 服务器将证书信息提供给作为重新同步请求一部分调用的 CGI 脚本（或编译的 CGI 程序）。为说明目的，此练习使用开放源码 Perl 脚本语言，并假定将 Apache (v.2) 用作 HTTPS 服务器。

过程

步骤 1 在运行 HTTPS 服务器的主机上安装 Perl。

步骤 2 生成以下 Perl 反射器脚本：

```
#!/usr/bin/perl -wT
use strict;
print “Content-Type: text/plain\n\n”;
print “<flat-profile><GPP_D>”;

print “OU=$ENV{ ‘SSL_CLIENT_I_DN_OU’ },\n”;
print “L=$ENV{ ‘SSL_CLIENT_I_DN_L’ },\n”;
print “S=$ENV{ ‘SSL_CLIENT_I_DN_S’ }\n”;
print “</GPP_D></flat-profile>”;
```

步骤 3 以 `reflect.pl` 为名将此具有执行权限（Linux 上的 chmod 755）的文件保存在 HTTPS 服务器的 CGI 脚本目录中。

步骤 4 验证服务器上的 CGI 脚本的可访问性（即 `/cgi-b i n /...`）。

步骤 5 如以下示例所示，修改测试设备上的 Profile_Rule 以重新同步到反射器脚本：

```
https://prov.server.com/cgi-bin/reflect.pl?
```

步骤 6 单击**提交所有更改**。

步骤 7 查看系统日志跟踪记录，确保重新同步成功。

步骤 8 访问电话管理网页。请参阅访问电话网页。

步骤 9 选择**语音** > **部署**。

步骤 10 验证 GPP_D 参数是否包含脚本捕获的信息。

如果测试设备带有制造商提供的唯一证书，此信息将包含产品名称、MAC 地址和序列号。如果设备是在固件版本 2.0 之前制造的，信息包含通用字符串。

类似脚本可以确定重新同步设备的相关信息，然后为设备提供适当的配置参数值。

HTTPS 证书

电话提供了可靠、安全的设置策略，基于从设备到设置服务器的 HTTPS 请求。向服务器验证电话以及向电话验证服务器时，同时使用服务器证书和客户端证书。

要在电话中使用 HTTPS，您必须生成证书签名请求 (CSR) 并将其提交给思科。电话将生成一个证书，以便在设置服务器上进行安装。当电话寻求与设置服务器建立 HTTPS 连接时，会接受该证书。

HTTPS 方法

HTTPS 会将客户端与服务器之间的通信加密，从而保护来自其他网路设备的消息内容。客户端和服务器之间的通信正文的加密方法基于对称密钥加密。采用对称密钥加密时，客户端和服务器通过受公共/专用密钥加密保护的安全通道共享一个密钥。

用密钥加密的消息只能使用同一个密钥解密。HTTPS 支持广泛的对称加密算法。电话可使用美国加密标准 (AES) 以及 128 位 RC4 实施最多 256 位对称加密。

HTTPS 还可验证参与安全事务的服务器与客户端。此功能可确保设置服务器和单独的客户端不会遭网络上的其他设备欺骗。此功能在远程终端设置中必不可少。

使用公共/专用密钥加密以及包含公共密钥的证书执行服务器和客户端验证。如果文本是使用公共密钥加密，则仅可通过对应的专用密钥解密（反之亦然）。对于公共/专用密钥加密，电话支持 Rivest-Shamir-Adleman (RSA) 算法。

SSL 服务器证书

每个安全设置服务器都会签发直接由思科签名的安全套接字层 (SSL) 服务器证书。电话上运行的固件只会将思科证书识别为有效。当客户端使用 HTTPS 连接到服务器时，它会拒绝并非由思科签名的服务器证书。

这一机制可防止服务提供商未经授权访问电话，或阻止任何企图欺骗设置服务器的行为。如果没有这个保护机制，攻击者可能可以重新设置电话，以获取配置信息或使用不同的 VoIP 服务。如果没有对应于有效服务器证书的专用密钥，攻击者将无法与电话建立通信。

获取服务器证书

过程

步骤 1 同将与您在证书流程方面合作的思科支持人员联系。如果不与特定支持人员合作，通过电子邮件将您的请求发送到 ciscosb-certadmin@cisco.com。

步骤 2 生成将用于 CSR（证书签名请求）的专用密钥。此为专用密钥，您无需将其提供给思科支持部门。使用开放源码“openssl”生成密钥。例如：

```
openssl genrsa -out <file.key> 1024
```

步骤 3 生成 CSR，其中包含标识您的组织和位置的字段。例如：

```
openssl req -new -key <file.key> -out <file.csr>
```

您必须具备以下信息：

- 主题字段 — 输入公用名称 (CN)，必须采用 FQDN（完全限定域名）语法。在 SSL 验证握手过程中，电话会验证其收到的证书是否来自提供该证书的计算机。
- 服务器主机名 — 例如 provserv.domain.com。
- 电子邮件地址—输入电子邮件地址以便客服可以在需要时联系到您。此电子邮件地址会显示在 CSR 中。

步骤 4 通过电子邮件将 CSR（zip 文件格式）发送给思科支持人员或 ciscosb-certadmin@cisco.com。证书由思科签名。思科会将证书发送给您以安装在系统上。

客户端证书

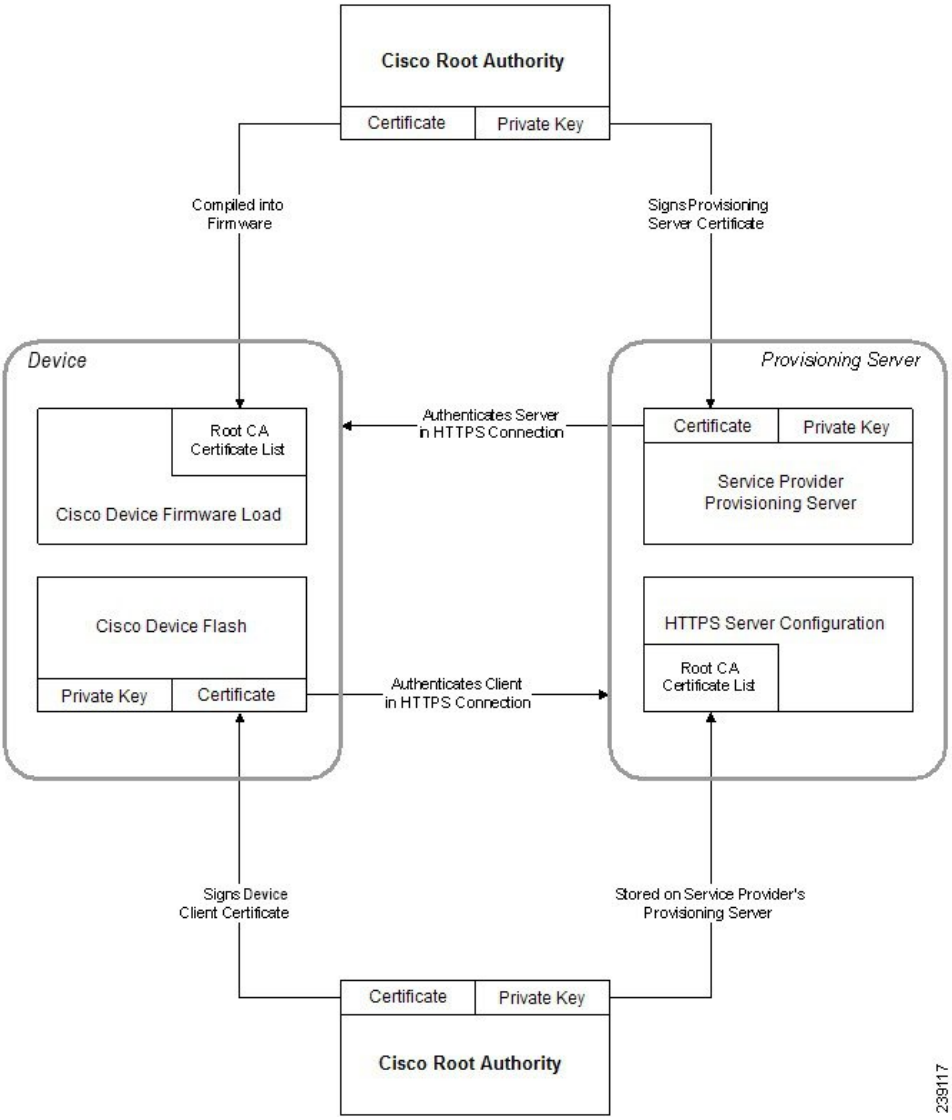
除直接攻击电话外，攻击者还可能尝试通过标准 web 浏览器或另一个 HTTPS 客户端联系设置服务器，以从设置服务器获取配置文件。为防止这类攻击，每部电话也携带由思科签名的唯一客户端证书，其中包含每个单独端点的标识信息。每个服务提供商将获得能够验证设备客户端证书的证书权限根证书。此验证路径允许设置服务器拒绝针对配置文件的未经授权请求。

证书结构

服务器证书和客户端证书相结合，可确保远程电话及其设置服务器之间的通信安全。下图所示为 Cisco 客户端、设置服务器以及证书权限中，证书、公共/专用密钥对以及签名根权限的关系和布局。

图表的上半部分显示设置服务器根权限，用于签署单一设置服务器证书。相应的根证书被编入固件，使得电话能够验证授权的设置服务器。

图 *1:* 证书权限流程图



配置自定义的证书权限

可以使用数字证书验证网路设备和网络上的用户。它们可用于协商网络节点之间的 IPSec 会话。

第三方使用证书权限证书验证和确认尝试通信的两个或多个节点。每个节点都有一个公共和专用密钥。公共密钥加密数据。专用密钥解密数据。由于各个节点是从同一个源获取证书，因此可以确保各自的身份。

设备可以使用第三方证书权限 (CA) 提供的数字证书验证 IPSec 连接。

电话支持嵌入固件的一组预先加载的根证书权限：

- Cisco Small Business CA Certificate
- CyberTrust CA Certificate

- Verisign CA certificate
- Sipura Root CA Certificate
- Linksys Root CA Certificate

开始之前

访问电话管理网页。请参阅访问电话网页。

过程

步骤 1 选择**信息** > **状态**。

步骤 2 滚动至**自定义 CA 状态**并查看以下字段：

- 自定义 CA 设置状态 — 指示设置状态。
  - 上次于 mm/dd/yyyy HH:MM:SS 设置成功；或者
  - 上次于 mm/dd/yyyy HH:MM:SS 设置失败
- 自定义 CA 信息 — 显示自定义 CA 的相关信息。
  - 已安装 — 显示“CN 值”，“CN 值”是第一个证书中主题字段 CN 参数的值。
  - 未安装 — 在未安装自定义 CA 证书时显示。

配置文件管理

本节介绍如何构建配置文件，从而为下载做好准备。为介绍功能，我们将来自本地 PC 的 TFTP 用作重新同步方法，尽管也可以使用 HTTP 或 HTTPS。

通过 Gzip 压缩开放配置文件

如果配置文件单独指定所有参数，XML 格式的配置文件可能会变得非常大。要减少设置服务器上的负载，电话支持使用 gzip 实用程序 (RFC 1951) 支持的 deflate 压缩格式压缩 XML 文件。

注释 电话的压缩必须在加密之前完成，以识别压缩和加密的 XML 配置文件。

要整合到自定义的后端设置服务器解决方案，可以用开放源码 zlib 压缩库代替独立的 gzip 实用程序执行配置文件压缩。不过，电话期望文件包含有效的 gzip 标头。

过程

步骤 1 在本地 PC 上安装 gzip。

步骤 2 通过从命令行调用 gzip 来压缩 `basic.txt` 配置文件（如 TFTP 重新同步， 第 1 页中所述）：

```
gzip basic.txt
```

这将生成压缩的文件 `basic.txt.gz`。

步骤 3 在 TFTP 服务器虚拟根目录下保存 `basic.txt.gz` 文件。

步骤 4 修改测试设备上的Profile_Rule，以便重新同步到替代原始XML文件的压缩文件，如以下示例所示：

```
tftp://192.168.1.200/basic.txt.gz
```

步骤 5 单击**提交所有更改**。

步骤 6 查看来自电话的系统日志跟踪记录。

在重新同步时，电话会下载新文件，并用其更新自己的参数。

相关主题

开放配置文件压缩

使用 OpenSSL 加密配置文件

压缩或未压缩的配置文件都可加密（但文件必须先加密才能压缩）。当需要特别注意配置文件信息的机密性（例如将 TFTP 或 HTTP 用于电话与设置服务器之间的通信）时，加密非常有用。

电话支持使用 256 位 AES 算法的对称密钥加密。可以使用开放源码 OpenSSL 软件包来执行这种加密。

过程

步骤 1 在本地 PC 上安装 OpenSSL。这可能需要重新编译 OpenSSL 应用程序以启用 AES。

步骤 2 使用 `basic.txt` 配置文件（如TFTP 重新同步， 第 1 页中所述），通过以下命令生成加密的文件：

```
>openssl enc -aes-256-cbc -k MyOwnSecret -in basic.txt -out basic.cfg
```

也可以使用在通过 Gzip 压缩开放配置文件， 第 14 页中创建的压缩文件 `basic.txt.gz`，因为 XML 配置文件可以同时压缩和加密。

步骤 3 在 TFTP 服务器虚拟根目录下保存加密的 `basic.cfg` 文件。

步骤 4 在测试设备上修改 Profile_Rule，以便重新同步取代原始 XML 文件的加密文件。加密密钥通过以下 URL 选项为电话所知：

```
[--key MyOwnSecret ] tftp://192.168.1.200/basic.cfg
```

步骤 5 单击**提交所有更改**。

步骤 6 查看来自电话的系统日志跟踪记录。

在重新同步时，电话会下载新文件，并用其更新自己的参数。

相关主题

AES-256-CBC 加密

创建分区的配置文件

在每个重新同步期间，电话会下载多个独立的配置文件。这种做法允许在单独的服务器上管理不同类型的配置文件信息，并且可以维护与帐户特定值不同的通用配置参数值。

过程

步骤 1 创建新的 XML 配置文件 `basic2.txt`，采用与之前练习不同的做法为参数指定值。例如，转到 `basic.txt` 配置文件，添加以下内容：

```
<GPP_B>ABCD</GPP_B>
```

步骤 2 在 TFTP 服务器的虚拟根目录中存储 `basic2.txt` 配置文件。

步骤 3 保留文件夹中在之前练习中建立的第一个配置文件规则，但配置第二个配置文件规则(Profile_Rule_B)，使其指向新的文件：

```
<Profile_Rule_B>tftp://192.168.1.200/basic2.txt
</Profile_Rule_B>
```

步骤 4 单击**提交所有更改**。

无论何时重新同步操作到期，电话现在都会按顺序重新同步第一个和第二个配置文件。

步骤 5 观察系统日志跟踪记录以确认预期的行为。

设置电话隐私标头

SIP 消息中的用户隐私标头设置来自受信任的网络的用户隐私需求。

您可以使用 XML 标记在 `config.xml` 文件中设置每个线路分机的用户隐私标头值。

隐私标头选项包括：

- Disabled（默认值）
- none—用户请求隐私服务不对此 SIP 消息应用隐私功能。
- header—用户需要隐私服务来遮盖无法清除标识信息的标头。
- session—用户请求隐私服务为会话提供匿名性。
- user—用户请求仅通过中间人的隐私级别。
- id—用户请求系统替换不显示 IP 地址或主机名的 id。

过程

步骤 1 在文本编辑器或 XML 编辑器中编辑电话 `config.xml` 文件。

步骤 2 插入 **<Privacy_Header_N_ ua="na">值</Privacy_Header_N_>** 标记，其中 N 是线路分机号码 (1 - 10)，并使用下列值之一。

- 默认值：**Disabled**
- **none**
- **header**
- 会话
- 用户
- **id**

步骤 3 （可选） 使用相同的标记及所需的线路分机号码部署任何其他线路分机。

步骤 4 保存对 `config.xml` 文件的更改。