Cisco IP Conference Phone 7832 Multiplatform Phones Provisioning Guide

First Published: 2017-08-14

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The following information is for FCC compliance of Class A devices: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case users will be required to correct the interference at their own expense.

The following information is for FCC compliance of Class B devices: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If the equipment causes interference to radio or television reception, which can be determined by turning the equipment off and on, users are encouraged to try to correct the interference by using one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Modifications to this product not authorized by Cisco could void the FCC approval and negate your authority to operate the product

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

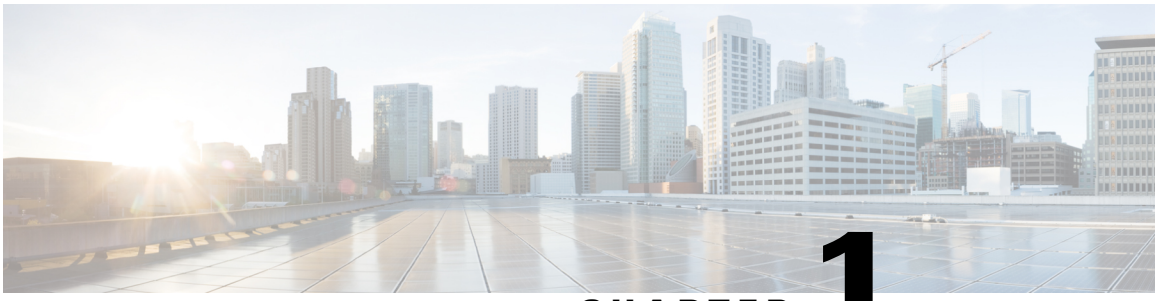
NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: http://www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

CONTENTS



CHAPTER 1

Deployment and Provisioning

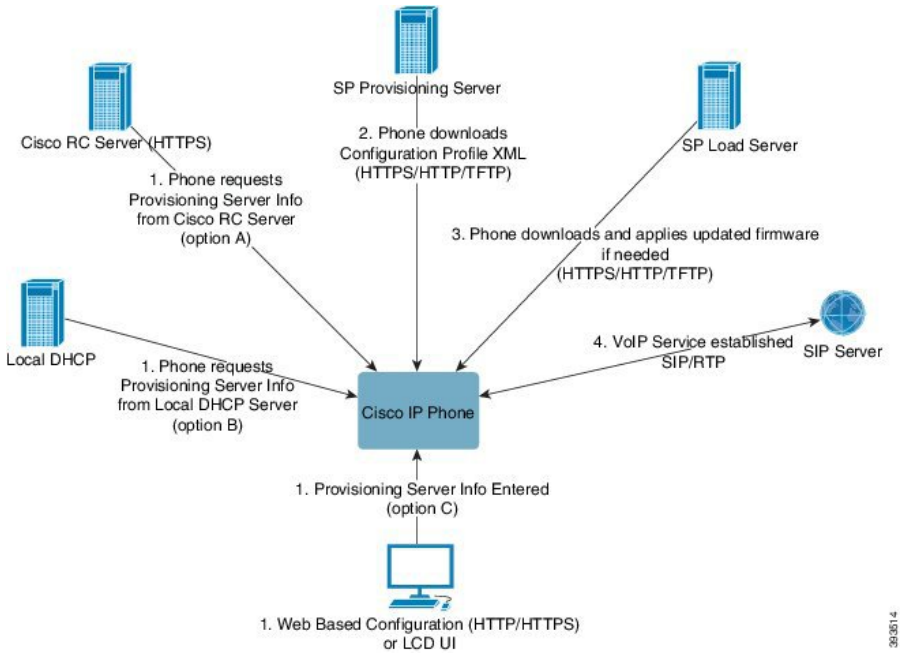
Provisioning Overview

Cisco IP Phones are intended for high-volume deployments by VoIP service providers to customers in home, business, or enterprise environments. Hence, provisioning the Cisco IP Phone via remote management and configuration ensures the proper operation of the phone at the customer site.

The following features support this customized, ongoing configuration:

- Reliable remote control of the Cisco IP Phone
- Encryption of the communication that controls the Cisco IP Phone
- Streamlined endpoint account binding

Phones can be provisioned to download configuration profiles or updated firmware from a remote server. Downloads can happen when the phones are connected to a network, when they are powered up, and at set intervals. Provisioning is typically part of high-volume, Voice-over-IP (VoIP) deployments common to service providers. Configuration profiles or updated firmware are transferred to the device using TFTP, HTTP, or HTTPS.



At a high level, the phone provisioning process is as follows:

1. If phone is not yet configured, provisioning server information is applied to phone via one of the following options:
    1. Downloaded from Cisco EDOS RC server via HTTPS.
    2. Queried from local DHCP server.
    3. Entered via Cisco phone web based configuration utility or Phone UI.
2. Phone downloads and applies configuration XML via HTTPS, HTTP, or TFTP using provisioning server information.
3. Phone downloads and applies updated firmware if needed via HTTPS, HTTP, or TFTP.
4. VOIP service establishing using specified configuration and firmware.

Cisco IP Phones are intended for high-volume deployments by VoIP service providers to residential and small business customers. In business or enterprise environments, Cisco IP Phones can serve as terminal nodes. These devices are widely distributed across the Internet, connected through routers and firewalls at the customer premises.

The Cisco IP Phone can be used as a remote extension of the service provider back-end equipment. Remote management and configuration ensure the proper operation of the Cisco IP Phone at the customer premises.

TR69 Provisioning

The Cisco IP phone allows the administrator to configure the TR69 parameters using the Web UI. For information related to the parameters, refer to the Administration Guide of the corresponding phone series.

The phones support ACS discovery from DHCP Option 43, 60, and 125.

Option 43: Vendor specific information - for ACS URL

Option 60: Vendor class identifier - The phone identifies itself with "dslforum.org" to ACS

Option 125: Vendor specific information - for gateway association

PRC Methods

RPC Methods Supported

The phones support only a limited set of RPC methods as follows:

- GetRPCMethods
- SetParameterValues
- GetParameterValues
- SetParameterAttributes
- GetParameterAttributes
- GetParameterNames
- AddObject
- DeleteObject
- Reboot
- FactoryReset
- Inform
- Download: Download RPC method, the file types supported are:
    - Firmware Upgrade Image
    - Vendor Configuration File
    - Custom CA File
- Transfer Complete

Event Types Supported

The phones support event types based on features and methods supported. Only the following event types are supported:

- Bootstrap
- Boot
- value change
- connection request
- Periodic
- Transfer Complete

- M Download
- M Reboot

Phone Behavior During Times of Network Congestion

Anything that degrades network performance can affect Cisco IP Phone voice and video quality, and in some cases, can cause a call to drop. Sources of network degradation can include, but are not limited to, the following activities:

- Administrative tasks, such as an internal port scan or security scan
- Attacks that occur on your network, such as a Denial of Service attack

Deployment

Cisco IP Phones provide convenient mechanisms for provisioning, based on these deployment models:

- Bulk distribution—The service provider acquires Cisco IP Phones in bulk quantity and either preprovisions them in-house or purchases Remote Customization (RC) units from Cisco. The devices are then issued to the customers as part of a VoIP service contract.
- Retail distribution—The customer purchases the Cisco IP Phone from a retail outlet and requests VoIP service from the service provider. The service provider must then support the secure remote configuration of the device.

Bulk Distribution

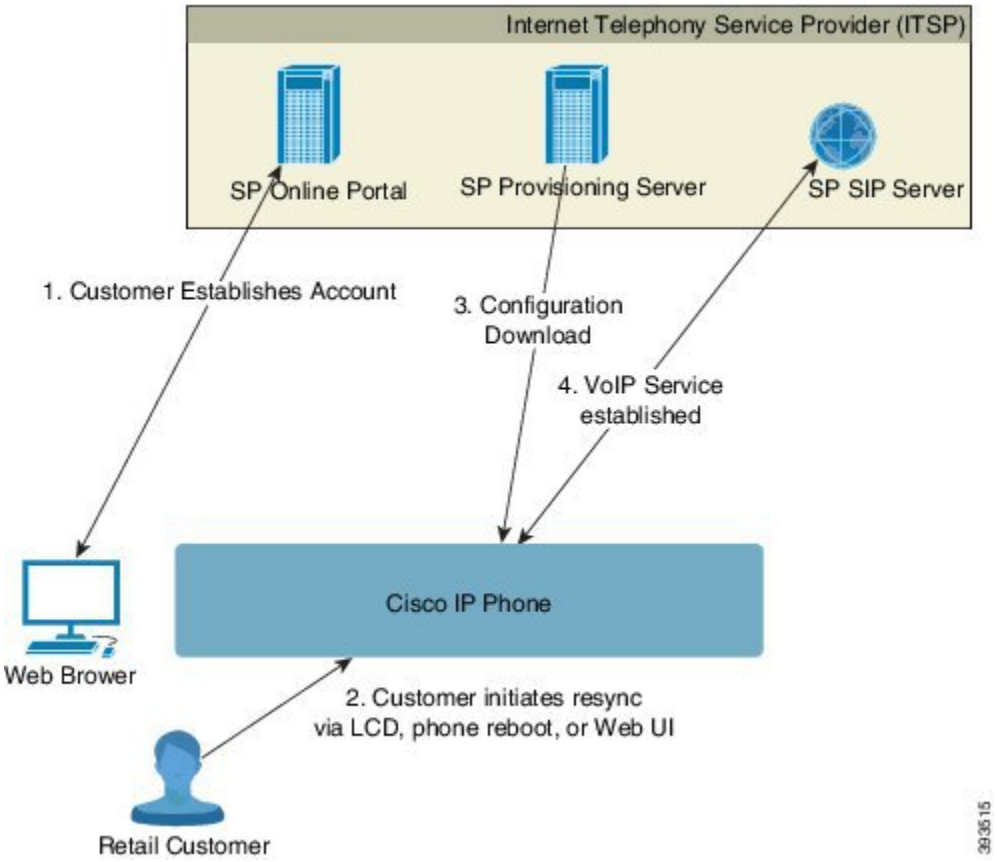
In this model, the service provider issues Cisco IP Phones to its customers as part of a VoIP service contract. The devices are either RC units or preprovisioned in-house.

Cisco preprovisions RC units to resynchronize with a Cisco server that downloads the device profile and firmware updates.

A service provider can preprovision Cisco IP Phones with the desired parameters, including the parameters that control resynchronization, through various methods:

- In-house by using DHCP and TFTP
- Remotely by using TFTP, HTTP, or HTTPS
- A combination of in-house and remote provisioning

Retail Distribution



The Cisco IP Phone includes the web-based configuration utility that displays internal configuration and accepts new configuration parameter values. The server also accepts a special URL command syntax for performing remote profile resync and firmware upgrade operations.

In a retail distribution model, a customer purchases a Cisco IP Phone and subscribes to a particular service. The Internet Telephony Service Provider (ITSP) sets up and maintains a provisioning server, and preprovisions the phone to resynchronize with the service provider server.

The customer signs on to the service and establishes a VoIP account, possibly through an online portal, and binds the device to the assigned service account. The unprovisioned Cisco IP Phone is instructed to resync with a specific provisioning server through a resync URL command. The URL command typically includes an account Customer ID number or alphanumeric code to associate the device with the new account.

In the following example, a device at the DHCP-assigned IP address 192.168.1.102 is instructed to provision itself to the SuperVoIP service:

```
http://192.168.1.102/admin/resync?https://prov.supervoip.com/cisco-init/1234abcd
```

In this example, 1234abcd is the Customer ID number of the new account. The remote provisioning server associates the phone that is performing the resync request with the new account, based on the URL and the

supplied Customer ID. Through this initial resync operation, the phone is configured in a single step. The phone is automatically directed to resync thereafter to a permanent URL on the server. For example:

```
https://prov.supervoip.com/cisco-init
```

For both initial and permanent access, the provisioning server relies on the Cisco IP phone client certificate for authentication. The provisioning server supplies correct configuration parameter values based on the associated service account.

When the device is powered up or a specified time elapses, the Cisco IP Phone resynchronizes and downloads the latest parameters. These parameters can address goals such as setting up a hunt group, setting speed dial numbers, and limiting the features that a user can modify.

Related Topics

In-House Device Preprovisioning, on page 33

Resynchronization Process

The firmware for each Cisco IP Phone includes an administration web server that accepts new configuration parameter values. The Cisco IP Phone may be instructed to resynchronize configuration after reboot, or at scheduled intervals with a specified provisioning server through a resync URL command in the device profile.

By default, the web server is enabled. To disable/enable the Web server, use the resync URL command.

If needed, an immediate resynchronization may be requested via a “resync" action URL. The resync URL command may include an account Customer ID number or alphanumeric code to uniquely associate the device with the user’s account.

Example

```
http://192.168.1.102/admin/resync?https://prov.supervoip.com/cisco-init/1234abcd
```

In this example, a device at the DHCP-assigned IP address 192.168.1.102 is instructed to provision itself to the SuperVoIP service at prov.supervoip.com. The Customer ID number for the new account is 1234abcd. The remote provisioning server associates the Cisco IP Phone that is performing the resync request with the account, based on the URL and Customer ID.

Through this initial resync operation, the Cisco IP Phone is configured in a single step. The phone is automatically directed to resync thereafter to a permanent URL on the server.

For both initial and permanent access, the provisioning server relies on the client certificate for authentication. The server supplies configuration parameter values based on the associated service account.

Provisioning

A Cisco IP Phone can be configured to resynchronize its internal configuration state to match a remote profile periodically and on power-up. The phone contacts a normal provisioning server (NPS) or an access control server (ACS).

By default, a profile resync is only attempted when the Cisco IP Phone is idle. This practice prevents an upgrade that would trigger a software reboot and interrupt a call. If intermediate upgrades are required to reach a current upgrade state from an older release, the upgrade logic can automate multistage upgrades.

Normal Provisioning Server

The Normal Provisioning Server (NPS) can be a TFTP, HTTP, or HTTPS server. A remote firmware upgrade is achieved by using TFTP or HTTP, or HTTPS, because the firmware does not contain sensitive information.

Although HTTPs is recommnded, communication with the NPS does not require the use of a secure protocol because the updated profile can be encrypted by a shared secret key. For more information about utilizing HTTPS, see Communication Encryption, on page 7. Secure first-time provisioning is provided through a mechanism that uses SSL functionality. An unprovisioned Cisco IP Phone can receive a 256-bit symmetric key encrypted profile that is targeted for that device.

Configuration Access Control

The Cisco IP Phone firmware provides mechanisms for restricting end-user access to some parameters. The firmware provides specific privileges for sign-in to an **Admin** account or a **User** account. Each can be independently password protected.

- Admin Account—Allows the service provider full access to all administration web server parameters.
- User Account—Allows the user to configure a subset of the administration web server parameters.

The service provider can restrict the user account in the provisioning profile in the following ways:

- Indicate which configuration parameters are available to the User account when creating the configuration.
- Disable user access to the administration web server.
- Disable user access for LCD GUI.
- Restrict the Internet domains accessed by the device for resync, upgrades, or SIP registration for Line 1.

Related Topics

Element Tag Properties, on page 10
Access Control, on page 12

Communication Encryption

The configuration parameters that are communicated to the device can contain authorization codes or other information that protect the system from unauthorized access. It is in the service provider's interest to prevent unauthorized customer activity. It is in the customer's interest to prevent the unauthorized use of the account. The service provider can encrypt the configuration profile communication between the provisioning server and the device, in addition to restricting access to the administration web server.

Phone Provisioning Practices

Typically, the Cisco IP Phone is configured for provisioning when it first connects to the network. The phone is also provisioned at the scheduled intervals that are set when the service provider or the VAR preprovisions

(configures) the phone. Service providers can authorize VARs or advanced users to manually provision the phone by using the phone keypad. You can also configure provisioning using the Phone Web UI.

Check the **Status** > **Phone Status** > **Provisioning** from the Phone LCD UI, or Provisioning Status in the **Status** tab of the web-based Configuration Utility.

Related Topics

Manually Provision a Phone from the Keypad, on page 8

Manually Provision a Phone from the Keypad

Procedure

Step 1 Press **Settings**.

Step 2 Select **Device administration** > **Profile Rule**.

Step 3 Enter the profile rule using the following format:

```
protocol://server[:port]/profile_pathname
```

For example:

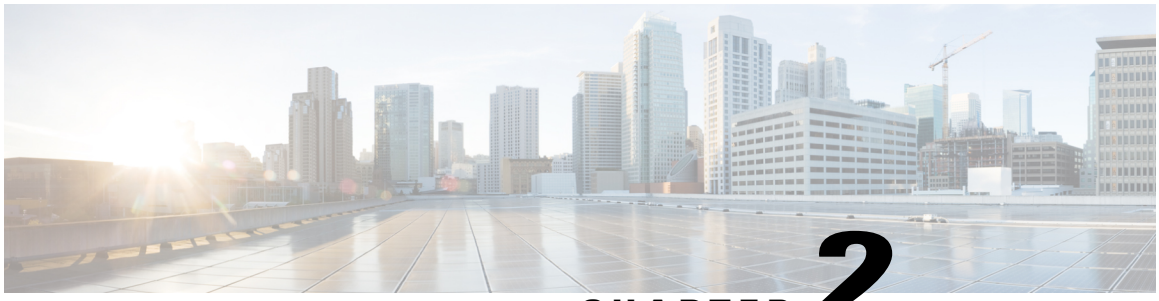
```
tftp://192.168.1.5/CP_x8xx_MPP.cfg
```

If no protocol is specified, TFTP is assumed. If no server-name is specified, the host that requests the URL is used as the server name. If no port is specified, the default port is used (69 for TFTP, 80 for HTTP, or 443 for HTTPS).

Step 4 Press **Resync**.

Related Topics

Phone Provisioning Practices, on page 7



CHAPTER 2

Provisioning Scripts

Provisioning Scripts

The Cisco IP Phone accepts configuration in an XML format.

The examples in this document use configuration profiles with an XML format (XML) syntax. Sample profiles can be found in Sample Configuration Profiles, on page 67

For detailed information about your Cisco IP Phone, refer to the administration guide for your particular device. Each guide describes the parameters that can be configured through the administration web server.

Configuration Profile Formats

The configuration profile defines the parameter values for the Cisco IP Phone.

The configuration profile XM format uses standard XML authoring tools to compile the parameters and values.

Note Only UTF-8 charset is supported. If you modify the profile in an editor, do not change the encoding format; otherwise, the Cisco IP Phone cannot recognize the file.

Each model of Cisco IP Phone has a different feature set and therefore, a different set of parameters.

XML Format (XML) Profile

The Open format profile is a text file with XML-like syntax in a hierarchy of elements, with element attributes and values. This format lets you use standard tools to create the configuration file. A configuration file in this format can be sent from the provisioning server to the Cisco IP Phone during a resync operation. The file can be sent without compilation as a binary object.

The Cisco IP Phone can accept configuration formats that standard tools generate. This feature eases the development of back-end provisioning server software that generates configuration profiles from existing databases.

To protect confidential information in the configuration profile, the provisioning server delivers this type of file to the phone over a channel secured by TLS. Optionally, the file can be compressed by using the gzip deflate algorithm (RFC1951). The file can be encrypted with 256-bit AES symmetric key encryption.

Example: Open Profile Format

```
<flat-profile>
<Resync_On_Reset> Yes </Resync_On_Reset>
<Resync_Periodic> 7200 </Resync_Periodic>
<Profile_Rule> tftp://prov.telco.com:6900/cisco/config/CP_x8xx_MPP.cfg</Profile_Rule>
</flat-profile>
```

The <flat-profile> element tag encloses all parameter elements that the Cisco IP Phone recognizes.

Related Topics

Open Profile (XML) Compression and Encryption, on page 13

Configuration File Components

A configuration file can include these components:

- Element tags
- Attributes
- Parameters
- Formatting features
- XML comments

Element Tag Properties

- The XML provisioning format and the Web UI allow the configuration of the same settings. The XML tag name and the field names in the Web UI are similar but vary due to XML element name restrictions. For example, underscores (_) instead of " ".
- The Cisco IP Phone recognizes elements with proper parameter names that are encapsulated in the special <flat-profile> element.
- Element names are enclosed in angle brackets.
- Most element names are similar to the field names in the administration web pages for the device, with the following modifications:

- Element names may not include spaces or special characters. To derive the element name from the administration web field name, substitute an underscore for every space or the special characters `[`, `]`, `(`, `)`, or `/`.

  Example: The <Resync_On_Reset> element represents the **Resync On Reset** field.

- Each element name must be unique. In the administration web pages, the same fields can appear on multiple web pages, such as the Line, User, and Extension pages. Append `[n]` to the element name to indicate the number that is shown in the page tab.

  Example: The <Dial_Plan_1_> element represents the **Dial Plan** for Line 1.

- Each opening element tag must have a matching closing element tag. For example:

```
<flat-profile>
<Resync_On_Reset> Yes
 </Resync_On_Reset>
<Resync_Periodic> 7200
 </Resync_Periodic>
<Profile_Rule>tftp://prov.telco.com: 6900/cisco/config/CP_x8xx_MPP.cfg
 </Profile_Rule>
</flat-profile>
```

- Element tags are case-sensitive.
- Empty element tags are allowed and will be interpreted as configuring the value to be empty. Enter the opening element tag without a corresponding element tag, and insert a space and a forward slash before the closing angle bracket (>). In this example, Profile Rule B is empty:

```
<Profile_Rule_B />
```

- Use an empty value to set the corresponding parameter to an empty string. Enter an opening and closing element without any value between them. In the following example, the GPP_A parameter is set to an empty string.

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

- Unrecognized element names are ignored.

User Access Attribute

The user access (**ua**) attribute controls may be used to change access by the User account. If the **ua** attribute is not specified, the existing user access setting is retained. This attribute does not affect access by the Admin account.

The **ua** attribute, if present, must have one of the following values:

- na—No access
- ro—Read-only
- rw—Read and write

The following example illustrates the **ua** attribute:

```
<flat-profile>
   <SIP_TOS_DiffServ_Value_1_ ua="na"/>
```

```
   <Dial_Plan_1_ ua="ro"/>
   <Dial_Plan_2_ ua="rw"/>
</flat-profile>
```

Double quotes must enclose the value of the **ua** option.

Access Control

If the <Phone-UI-User-Mode> parameter is enabled, the phone GUI honors the user access attribute of the relevant parameters when the GUI presents a menu item.

For menu entries that are associated with a single configuration parameter:

- Provisioning the parameter with "ua=na" ("ua" stands for "user access") attribute makes the entry disappear.
- Provisioning the parameter with "ua=ro" attribute makes the entry read-only and non-editable.

For menu entries that are associated with multiple configuration parameters:

- Provisioning all concerned parameters with "ua=na" attribute makes the entries disappear.

Parameter Properties

These properties apply to the parameters:

- Any parameters that no profile specifies are left unchanged in the Cisco IP Phone.
- Unrecognized parameters are ignored.
- If the Open format profile contains multiple occurrences of the same parameter tag, the last such occurrence overrides any earlier ones. To avoid inadvertent override of configuration values for a parameter, we recommend that each profile specify at most one instance of a parameter.
- The last profile processed takes precedence. If multiple profiles specify the same configuration parameter, the value of the latter profile takes precedence.

Formatting

These properties apply to the formatting of the strings:

- Comments are allowed through standard XML syntax.

  ```
  <!-- My comment is typed here -->
  ```

- Leading and trailing white space is allowed for readability but is removed from the parameter value.
- New lines within a value are converted to spaces.
- An XML header of the form `<? ?>` is allowed, but the Cisco IP Phone ignores it.
- To enter special characters, use basic XML character escapes, as shown in the following table.

Special Character	XML Escape Sequence
& (ampersand)	&amp;
< (less than)	<

Special Character	XML Escape Sequence
> (greater than)	&gt;
’ (apostrophe)	&apos;
” (double quote)	&quot;

In the following example, character escapes are entered to represent the greater than and less than symbols that are required in a dial plan rule. This example defines an information hotline dial plan that sets the <Dial_Plan_1_> parameter (**Admin Login** > **advanced** > **Voice** > **Ext (n)**) equal to (S0 <:18005551212>).

```
<flat-profile>
 <Dial_Plan_1_>
  (S0 &lt;:18005551212&gt;)
 </Dial_Plan_1_>
</flat-profile>
```

- Numeric character escapes, using decimal and hexadecimal values (s.a. `&#40;` and `&#x2e;`), are translated.

Open Profile (XML) Compression and Encryption

The Open configuration profile can be compressed to reduce the network load on the provisioning server. The profile can also be encrypted to protect confidential information. Compression is not required, but it must precede encryption.

Open Profile Compression

The supported compression method is the gzip deflate algorithm (RFC1951). The gzip utility and the compression library that implements the same algorithm (zlib) are available from Internet sites.

To identify compression, the Cisco IP Phone expects the compressed file to contain a gzip compatible header. Invocation of the gzip utility on the original Open profile generates the header. The Cisco IP Phone inspects the downloaded file header to determine the file format.

For example, if `profile.xml` is a valid profile, the file `profile.xml.gz` is also accepted. Either of the following commands can generate this profile type:

- `>gzip profile.xml`

  Replaces original file with compressed file.

- `>cat profile.xml | gzip > profile.xml.gz`

  Leaves original file in place, produces new compressed file.

A tutorial on compression is provided in the Open Profile gzip Compression, on page 54 section.

Open Profile Encryption by Using AES

Symmetric key encryption can be used to encrypt an Open configuration profile, whether the file is compressed or not. The supported encryption algorithm is the American Encryption Standard (AES), using 256-bit keys, applied in cipher block chaining mode.

Note Compression must precede encryption for the Cisco IP Phone to recognize a compressed and encrypted Open format profile. The Profile Encryption by Using OpenSSL, on page 55 section provides a tutorial on encryption.

The OpenSSL encryption tool, available for download from various Internet sites, can perform the encryption. Support for 256-bit AES encryption may require recompilation of the tool to enable the AES code. The firmware has been tested against version openssl-0.9.7c.

For an encrypted file, the profile expects the file to have the same format as generated by the following command:

```
# example encryption key = SecretPhrase1234

openssl enc –e –aes-256-cbc –k SecretPhrase1234 –in profile.xml –out profile.cfg

# analogous invocation for a compressed xml file

openssl enc –e –aes-256-cbc –k SecretPhrase1234 –in profile.xml.gz –out profile.cfg
```

A lowercase -k precedes the secret key, which can be any plain text phrase, and which is used to generate a random 64-bit salt. With the secret specified by the -k argument, the encryption tool derives a random 128-bit initial vector and the actual 256-bit encryption key.

When this form of encryption is used on a configuration profile, the phone must be informed of the secret key value to decrypt the file. This value is specified as a qualifier in the profile URL. The syntax is as follows, using an explicit URL:

```
[--key “SecretPhrase1234”] http://prov.telco.com/path/profile.cfg
```

This value is programmed by using one of the Profile_Rule parameters. The key must be preprovisioned into the unit at an earlier time. Bootstrap of the secret key can be accomplished securely by using HTTPS.

Pre-encrypting configuration profiles offline, with symmetric key encryption, allows the use of HTTP for resyncing profiles. The provisioning server uses HTTPS to handle initial provisioning of the Cisco IP Phone after deployment. This feature reduces the load on the HTTPS server in large-scale deployments.

The final filename does not require a specific format, but a filename that ends with the `.cfg` extension normally indicates a configuration profile.

Macro Expansion

Several provisioning parameters undergo macro expansion internally prior to being evaluated. This preevaluation step provides greater flexibility in controlling the Cisco IP Phone resync and upgrade activities.

These parameter groups undergo macro expansion before evaluation:

- Resync_Trigger_*
- Profile_Rule*

- Log_xxx_Msg
- Upgrade_Rule

Under certain conditions, some general-purpose parameters (GPP_*) also undergo macro expansion, as explicitly indicated in the Optional Resync Arguments, on page 17 section.

During macro expansion, the contents of the named variables replace expressions of the form $NAME and $(NAME). These variables include general-purpose parameters, several product identifiers, certain event timers, and provisioning state values. For a complete list, see the Macro Expansion Variables, on page 63section.

In the following example, the expression $(MAU) is used to insert the MAC address 000E08012345.

The administrator enters: $(MAU)config.cfg

The resulting macro expansion for a device with MAC address 000E08012345 is:
`000E08012345config.cfg`

If a macro name is not recognized, it remains unexpanded. For example, the name STRANGE is not recognized as a valid macro name, while MAU is recognized as a valid macro name.

The administrator enters: $STRANGE$MAU.cfg

The resulting macro expansion for a device with MAC address 000E08012345 is:
`$STRANGE000E08012345.cfg`

Macro expansion is not applied recursively. For example, $$MAU" expands into $MAU" (the $$ is expanded), and does not result in the MAC address.

The contents of the special purpose parameters, GPP_SA through GPP_SD, are mapped to the macro expressions $SA through $SD. These parameters are only macro expanded as the argument of the **--key** , **--uid**, and **--pwd** options in a resync URL.

Conditional Expressions

Conditional expressions can trigger resync events and select from alternate URLs for resync and upgrade operations.

Conditional expressions consist of a list of comparisons, separated by the **and** operator. All comparisons must be satisfied for the condition to be true.

Each comparison can relate to one of the following three types of literals:

- Integer values
- Software or hardware version numbers
- Doubled-quoted strings

Version Numbers

Multiplatform phones (MPP) phone formal release software version uses the format `sip7832.v1-v2-v3MPP-BN` (BN==Build Number) for the Cisco IP Conference Phone 7832. The comparing string must use the same format. Otherwise, a format parsing error results.

In the software version, v1-v2-v3-v4 can specify different digits and characters, but must start with a numeric digit. When comparing the software version, v1-v2-v3-v4 is compared in sequence, and the leftmost digits take precedence over the latter ones.

If v[x] includes only numeric digits, the digits are compared; if v[x] includes numeric digits + alpha characters, digits are compared first, then characters are compared in alphabetical order.

Example of Valid Version Number

sip78yy.11-0-0MPP-BN

By contrast: 11.0.0 is an invalid format.

Comparison

sip78xx.11-0-0MPP-BN < sip78xx.11-0-0MN-1MPP-BN

Quoted strings can be compared for equality or inequality. Integers and version numbers can also be compared arithmetically. The comparison operators can be expressed as symbols or as acronyms. Acronyms are convenient for expressing the condition in an Open format profile.

Operator	Alternate Syntax	Description	Applicable to Integer and Version Operands	Applicable to Quoted String Operands
=	eq	equal to	Yes	Yes
!=	ne	not equal to	Yes	Yes
<	lt	less than	Yes	No
<=	le	less than or equal to	Yes	No
>	gt	greater than	Yes	No
>=	ge	greater than or equal to	Yes	No
AND		and	Yes	Yes

It is important to enclose macro variables in double quotes where a string literal is expected. Do not do so where a number or version number is expected.

When used in the context of the Profile_Rule* and Upgrade_Rule parameters, conditional expressions must be enclosed within the syntax “(expr)?” as in this upgrade rule example:

```
($SWVER ne sip78xx.11-0-0MPP)? http://ps.tell.com/sw/sip78xx.11-0-0MPP-BN (BN==Build
Number).loads
```

Do not use the preceding syntax with parentheses to configure the Resync_Trigger_* parameters.

URL Syntax

Use Standard URL syntax to specify how to retrieve configuration files and firmware loads in Profile_Rule* and Upgrade_Rule parameters, respectively. The syntax is as follows:

[scheme://] [server [:port]] filepath

Where `scheme` is one of these values:

- tftp
- http
- https

If `scheme` is omitted, tftp is assumed. The server can be a DNS-recognized hostname or a numeric IP address. The port is the destination UDP or TCP port number. The filepath must begin with the root directory (/); it must be an absolute path.

If `server` is missing, the tftp server specified through DHCP (option 66) is used.

If `port` is missing, the standard port for the specified scheme is used. (tftp uses UDP port 69, http uses TCP port 80, https uses TCP port 443.)

A filepath must be present. It need not necessarily refer to a static file, but can indicate dynamic content obtained through CGI.

Macro expansion applies within URLs. The following are examples of valid URLs:

```
/$MA.cfg
/cisco/sip78xx.11-0-0MPP-BN (BN==Build Number).loads
192.168.1.130/profiles/init.cfg
tftp://prov.call.com/cpe/cisco$MA.cfg
http://neptune.speak.net:8080/prov/$D/$E.cfg
https://secure.me.com/profile?Linksys
```

Optional Resync Arguments

Optional arguments, key, uid, and pwd can precede the URLs entered in Profile_Rule* parameters, collectively enclosed by square brackets.

key

The **key** option is used to specify an encryption key. Decryption of profiles that have been encrypted with an explicit key is required. The key itself is specified as a (possibly quoted) string following the term **--key**.

Usage Examples

```
[--key VerySecretValue]
[--key “my secret phrase”]
[--key a37d2fb9055c1d04883a0745eb0917a4]
```

The bracketed optional arguments are macro expanded. Special purpose parameters, GPP_SA through GPP_SD, are macro expanded into macro variables, $SA through $SD, only when they are used as key option arguments. See these examples:

```
[--key $SC]
[--key “$SD”]
```

In Open format profiles, the argument to **--key** must be the same as the argument to the **-k** option that is given to **openssl**.

uid&pwd

The **uid** & **pwd** options can be used to specify userID and Password authentication for the specified URL. The bracketed optional arguments are macro expanded. Special purpose parameters, GPP_SA through GPP_SD, are macro expanded into macro variables, $SA through $SD, only when they are used as key option arguments. See these examples:

```
GPP_SA = MyUserID
GPP_SB = MySecretPassword
```

[--uid $SA -pwd $SB] https://provisioning_server_url/path_to_your_config/your_config.xml

would then expand to:

[--uid MyUserID -pwdMySecretPassword]
https://provisioning_server_url/path_to_your_config/your_config.xml

Apply a Profile to the IP Telephony Device

After you create an XML configuration script, it must be passed to the Cisco IP Phone for application. To apply the configuration, you can either download the configuration file to the IP Phone from a TFTP, HTTP, or HTTPS server using a web browser or by using cURL command line utility.

Download the Configuration File to the IP Phone from a TFTP Server

Complete these steps to download the configuration file to a TFTP server application on your PC.

Procedure

Step 1 Connect your PC to the phone LAN.

Step 2 Run a TFTP server application on the PC and ensure that the configuration file is available in the TFTP root directory.

Step 3 In a web browser, enter the Cisco IP Phone LAN IP address, the IP address of the computer, the filename, and the login credentials. Use this format:
http://<WAN_IP_Address>/admin/resync?tftp://<PC_IP_Address>/<file_name>&xuser=admin&xpassword=<password>

Example:

```
http://192.168.15.1/admin/resync?tftp://192.168.15.100/my_config.xml&xuser=admin&xpassword=admin
```

Download the Configuration File to the IP Phone Using cURL

Complete these steps to download the configuration to the Cisco IP Phone by using cURL. This command-line tool is used to transfer data with a URL syntax. To download cURL, visit:.

https://curl.haxx.se/download.html

Procedure

Step 1 Connect your PC to the LAN port of the Cisco IP Phone.

Step 2 Download the configuration file to the Cisco IP Phone by entering the following cURL command:

```
curl –d @my_config.xml
“http://192.168.15.1/admin/config.xml&xuser=admin&xpassword=admin”
```

Provisioning Parameters

This section describes the provisioning parameters broadly organized according to function:

These provisioning parameter types exist:

- General Purpose
- Enables
- Triggers
- Configurable Schedules
- Profile Rules
- Upgrade Rule

General Purpose Parameters

The general-purpose parameters GPP_* (**Admin Login** > **advanced** > **Voice** > **Provisioning**) are used as free string registers when configuring the Cisco IP Phone to interact with a particular provisioning server solution. The GPP_* parameters are empty by default. They can be configured to contain diverse values, including the following:

- Encryption keys
- URLs
- Multistage provisioning status information
- Post request templates
- Parameter name alias maps
- Partial string values, eventually combined into complete parameter values.

The GPP_* parameters are available for macro expansion within other provisioning parameters. For this purpose, single-letter uppercase macro names (A through P) suffice to identify the contents of GPP_A through GPP_P. Also, the two-letter uppercase macro names SA through SD identify GPP_SA through GPP_SD as a special case when used as arguments of the following URL options:

key, **uid**, and **pwd**

These parameters can be used as variables in provisioning and upgrade rules. They are referenced by prefixing the variable name with a '$' character, such as $GPP_A.

Using General Purpose Parameters

For example, if GPP_A contains the string ABC, and GPP_B contains 123, the expression AB macro expands into ABC123.

Procedure

Step 1 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.

Step 2 Scroll to the **General Purpose Parameters** section.

Step 3 Enter valid values in the fields, GPP A through GPP P.

Step 4 Click **Submit All Changes**.

Enables

The Provision_Enable and Upgrade_Enable parameters control all profile resync and firmware upgrade operations. These parameters control resyncs and upgrades independently of each other. These parameters also control resync and upgrade URL commands that are issued through the administration web server. Both of these parameters are set to **Yes** by default.

The Resync_From_SIP parameter controls requests for resync operations. A SIP NOTIFY event is sent from the service provider proxy server to the Cisco IP Phone. If enabled, the proxy can request a resync. To do so, the proxy sends a SIP NOTIFY message that contains the Event: resync header to the device.

The device challenges the request with a 401 response (authorization refused for used credentials). The device expects an authenticated subsequent request before it honors the resync request from the proxy. The Event: reboot_now and Event: restart_now headers perform cold and warm restarts, respectively, which are also challenged.

The two remaining enables are Resync_On_Reset and Resync_After_Upgrade_Attempt. These parameters determine whether the device performs a resync operation after power-up software reboots and after each upgrade attempt.

When Resync_On_Reset is enabled, the device introduces a random delay that follows the boot-up sequence before the reset is performed. The delay is a random time up to the value that the Resync_Random_Delay (in seconds) specifies. In a pool of phones that power up simultaneously, this delay spreads out the start times of the resync requests from each unit. This feature can be useful in a large residential deployment, in the case of a regional power failure.

Triggers

The Cisco IP Phone allows you to resync at specific intervals or at a specific time.

Resyncing at Specific Intervals

The Cisco IP Phone is designed to resync with the provisioning server periodically. The resync interval is configured in Resync_Periodic (seconds). If this value is left empty, the device does not resync periodically.

The resync typically takes place when the voice lines are idle. If a voice line is active when a resync is due, the Cisco IP Phone delays the resync procedure until the line becomes idle again. A resync can cause configuration parameter values to change.

A resync operation can fail because the Cisco IP Phone is unable to retrieve a profile from the server, the downloaded file is corrupt, or an internal error occurred. The device tries to resync again after a time that is specified in Resync_Error_Retry_Delay (seconds). If Resync_Error_Retry_Delay is set to 0, the device does not try to resync again after a failed resync attempt.

If an upgrade fails, a retry is performed after Upgrade_Error_Retry_Delay seconds.

Two configurable parameters are available to conditionally trigger a resync: Resync_Trigger_1 and Resync_Trigger_2. Each parameter can be programmed with a conditional expression that undergoes macro expansion. When the resync interval expires (time for the next resync) the triggers, if set, will prevent resync unless one or more triggers evaluates to true.

The following example condition triggers a resync. In the example, the last phone upgrade attempt has elapsed more than 5 minutes (300 seconds), and at least 10 minutes (600 seconds) have elapsed since the last resync attempt.

```
$UPGTMR gt 300 and $PRVTMR ge 600
```

Resync at a Specific Time

The Resync_At parameter allows the phone to resync at a specific time. This parameter uses the 24-hour format (hhmm) to specify the time.

The Resync_At_Random_Delay parameter allows the phone to resync at an unspecified delay in time. This parameter uses a positive integer format to specify the time.

Flooding the server with resync requests from multiple phones that are set to resync at the same time should be avoided. To do so, the phone triggers the resync up to 10 minutes after the specified time.

For example, if you set the resync time to 1000 (10 a.m.), the phone triggers the resync anytime between 10:00 a.m. and 10:10 a.m.

By default, this feature is disabled. If the Resync_At parameter is provisioned, the Resync_Periodic parameter is ignored.

Configurable Schedules

You can configure schedules for periodic resyncs, and you can specify the retry intervals for resync and upgrade failures by using these provisioning parameters:

- Resync_Periodic
- Resync_Error_Retry_Delay
- Upgrade_Error_Retry_Delay

Each parameter accepts a single delay value (seconds). The new extended syntax allows for a comma-separated list of consecutive delay elements. The last element in the sequence is implicitly repeated forever. Here is an example:

```
Resync_Periodic=7200
Resync_Error_Retry_Delay=1800,3600,7200,14400
```

In the preceding example, the Cisco IP Phone periodically resyncs every 2 hours. If a resync failure occurs, the device retries at these intervals: 30 minutes, 1 hour, 2 hours, 4 hours. The device continues to try at 4-hour intervals until it resyncs successfully.

Optionally, you can use a plus sign to specify another numeric value that appends a random extra delay, as shown in this example:

```
Resync_Periodic=3600+600
Resync_Error_Retry_Delay=1800+300,3600+600,7200+900
```

In the preceding example, the device periodically resyncs every hour (plus an extra random delay of up to 10 minutes). In the case of a resync failure, the device retries at these intervals: 30 minutes (plus up to 5 minutes). 1 hour (plus up to 10 minutes), 2 hours (plus up to 15 minutes). The device continues to try at 2-hour intervals (plus up to 15 minutes) until it successfully resyncs.

Here is another example:

```
Upgrade_Error_Retry_Delay  =  1800,3600,7200,14400+3600
```

In this example, if a remote upgrade attempt fails, the device retries the upgrade in 30 minutes, then again after one more hour, then in two hours. If the upgrade still fails, the device retries every four to five hours until the upgrade succeeds.

Profile Rules

The Cisco IP Phone provides multiple remote configuration profile parameters (Profile_Rule*). Thus, each resync operation can retrieve multiple files that different servers manage.

In the simplest scenario, the device resyncs periodically to a single profile on a central server, which updates all pertinent internal parameters. Alternatively, the profile can be split between different files. One file is common for all the Cisco IP Phones in a deployment. A separate, unique file is provided for each account. Encryption keys and certificate information can be supplied by still another profile, stored on a separate server.

Whenever a resync operation is due, the Cisco IP Phone evaluates the four Profile_Rule* parameters in sequence:

1. Profile_Rule
2. Profile_Rule_B
3. Profile_Rule_C
4. Profile_Rule_D

Each evaluation can result in a profile retrieval from a remote provisioning server, with a possible update of some number of internal parameters. If an evaluation fails, the resync sequence is interrupted, and is retried again from the beginning specified by the Resync_Error_Retry_Delay parameter (seconds). If all evaluations succeed, the device waits for the second specified by the Resync_Periodic parameter and then performs another resync.

The contents of each Profile_Rule* parameter consist of a set of alternatives. The alternatives are separated by the | (pipe) character. Each alternative consists of a conditional expression, an assignment expression, a profile URL, and any associated URL options. All these components are optional within each alternative. The following are the valid combinations, and the order in which they must appear, if present:

```
[ conditional-expr ] [ assignment-expr ] [[ options ] URL ]
```

Within each Profile_Rule* parameter, all alternatives except the last one must provide a conditional expression. This expression is evaluated and is processed as follows:

1. Conditions are evaluated from left to right, until one is found that evaluates as true (or until one alternative is found with no conditional expression).
2. Any accompanying assignment expression is evaluated, if present.
3. If a URL is specified as part of that alternative, an attempt is made to download the profile that is located at the specified URL. The system attempts to update the internal parameters accordingly.

If all alternatives have conditional expressions and none evaluates to true (or if the whole profile rule is empty), the entire Profile_Rule* parameter is skipped. The next profile rule parameter in the sequence is evaluated.

Examples of Valid Programming for a Single Profile_Rule* Parameter

This example resyncs unconditionally to the profile at the specified URL, and performs an HTTP GET request to the remote provisioning server:

```
http://remote.server.com/cisco/$MA.cfg
```

In this example, the device resyncs to two different URLs, depending on the registration state of Line 1. In case of lost registration, the device performs an HTTP POST to a CGI script. The device sends the contents of the macro expanded GPP_A, which may provide additional information on the device state:

```
($PRVTMR ge 600)? http://p.tel.com/has-reg.cfg
| [--post a] http://p.tel.com/lost-reg?
```

In this example, the device resyncs to the same server. The device provides additional information if a certificate is not installed in the unit (for legacy pre-2.0 units):

```
("$CCERT" eq "Installed")? https://p.tel.com/config?
| https://p.tel.com/config?cisco$MAU
```

In this example, Line 1 is disabled until GPP_A is set equal to Provisioned through the first URL. Afterwards, it resyncs to the second URL:

```
("$A" ne "Provisioned")? (Line_Enable_1_ = "No";)! https://p.tel.com/init-prov
| https://p.tel.com/configs
```

In this example, the profile that the server returns is assumed to contain XML element tags. These tags must be remapped to proper parameter names by the aliases map stored in GPP_B:

```
[--alias b] https://p.tel.com/account/$PN$MA.xml
```

A resync is typically considered unsuccessful if a requested profile is not received from the server. The Resync_Fails_On_FNF parameter can override this default behavior. If Resync_Fails_On_FNF is set to No, the device accepts a file-not-found response from the server as a successful resync. The default value for Resync_Fails_On_FNF is Yes.

Upgrade Rule

Upgrade rule is to tell the device to activate to a new load and from where to get the load, if necessary. If the load is already on the device, it will not try to get the load. So, validity of the load location does not matter when the desired load is in the inactive partition.

The Upgrade_Rule specifies a firmware load which, if different from the current load, will be downloaded and applied unless limited by a conditional expression or Upgrade_Enable is set to **No**.

The Cisco IP Phone provides one configurable remote upgrade parameter, Upgrade_Rule. This parameter accepts syntax similar to the profile rule parameters. URL options are not supported for upgrades, but conditional expressions and assignment expressions can be used. If conditional expressions are used, the parameter can be populated with multiple alternatives, separated by the | character. The syntax for each alternative is as follows:

```
[ conditional-expr ] [ assignment-expr ] URL
```

As in the case of Profile_Rule* parameters, the Upgrade_Rule parameter evaluates each alternative until a conditional expression is satisfied or an alternative has no conditional expression. The accompanying assignment expression is evaluated, if specified. Then, an upgrade to the specified URL is attempted.

If the Upgrade_Rule contains a URL without a conditional expression, the device upgrades to the firmware image that the URL specifies. After macro expansion and evaluation of the rule, the device does not reattempt to upgrade until the rule is modified or the effective combination of scheme + server + port + filepath is changed.

To attempt a firmware upgrade, the device disables audio at the start of the procedure and reboots at the end of the procedure. The device automatically begins an upgrade that is driven by the contents of Upgrade_Rule only if all voice lines are currently inactive.

For example, for the Cisco IP Conference Phone 7832:

```
http://p.tel.com/firmware/sip7832.11-0-1MPP-BN (BN==Build Number).loads
```

In this example, the Upgrade_Rule upgrades the firmware to the image that is stored at the indicated URL.

Here is another example for the Cisco IP Conference Phone 7832:

```
("$F" ne "beta-customer")? http://p.tel.com/firmware/sip7832.11-0-1MPP-BN (BN==Build
Number).loads
| http://p.tel.com/firmware/sip7832.11-0-1MPP-BN (BN==Build Number).loads
```

This example directs the unit to load one of two images, based on the contents of a general-purpose parameter, GPP_F.

The device can enforce a downgrade limit regarding firmware revision number, which can be a useful customization option. If a valid firmware revision number is configured in the Downgrade_Rev_Limit parameter, the device rejects upgrade attempts for firmware versions earlier than the specified limit.

Data Types

These data types are used with configuration profile parameters:

- {a,b,c,…}—A choice among a, b, c, …

- Bool—Boolean value of either “yes” or “no.”
- CadScript—A miniscript that specifies the cadence parameters of a signal. Up to 127 characters.

  Syntax: $S_1[;S_2]$, where:

  $S_i=D_i(on_{i,1}/off_{i,1}[,on_{i,2}/off_{i,2}[,on_{i,3}/off_{i,3}[,on_{i,4}/off_{i,4}[,on_{i,5}/off_{i,5}[,on_{i,6}/off_{i,6}]]]]])$ and is known as a section. $on_{i,j}$ and $off_{i,j}$ are the on/off duration in seconds of a *segment*. i = 1 or 2, and j = 1 to 6. D_i is the total duration of the section in seconds. All durations can have up to three decimal places to provide 1 ms resolution. The wildcard character “*” stands for infinite duration. The segments within a section are played in order and repeated until the total duration is played.

  Example 1:

```
60(2/4)

Number of Cadence Sections = 1
Cadence Section 1: Section Length = 60 s
Number of Segments = 1
Segment 1: On=2s, Off=4s

Total Ring Length = 60s
```

  Example 2—Distinctive ring (short,short,short,long):

```
60(.2/.2,.2/.2,.2/.2,1/4)

Number of Cadence Sections = 1
Cadence Section 1: Section Length = 60s
Number of Segments = 4
Segment 1: On=0.2s, Off=0.2s
Segment 2: On=0.2s, Off=0.2s
Segment 3: On=0.2s, Off=0.2s
Segment 4: On=1.0s, Off=4.0s

Total Ring Length = 60s
```

- DialPlanScript—Scripting syntax that is used to specify Line 1 and Line 2 dial plans.
- Float<n>—A floating point value with up to n decimal places.
- FQDN—Fully Qualified Domain Name. It can contain up to 63 characters. Examples are as follows:
  - sip.Cisco.com:5060 or 109.12.14.12:12345
  - sip.Cisco.com or 109.12.14.12
- FreqScript—A miniscript that specifics the frequency and level parameters of a tone. Contains up to 127 characters. Syntax: $F_1@L_1[,F_2@L_2[,F_3@L_3[,F_4@L_4[,F_5@L_5[,F_6@L_6]]]]]$, where F_1–F_6 are frequency in Hz (unsigned integers only). L_1–L_6 are corresponding levels in dBm (with up to one decimal place). White spaces before and after the comma are allowed but not recommended.

  Example 1—Call Waiting Tone:

```
440@-10

Number of Frequencies = 1
Frequency 1 = 440 Hz at -10 dBm
```

  Example 2—Dial Tone:

```
350@-19,440@-19
```

```
Number of Frequencies = 2
Frequency 1 = 350 Hz at –19 dBm
Frequency 2 = 440 Hz at –19 dBm
```

- IP— Valid IPv4 Address in the form of x.x.x.x, where x is between 0 and 255. Example: 10.1.2.100.
- UserID—User ID as it appears in a URL; up to 63 characters.
- Phone—A phone number string, such as 14081234567, *69, *72, 345678; or a generic URL, such as, 1234@10.10.10.100:5068 or jsmith@Cisco.com. The string can contain up to 39 characters.
- PhTmplt—A phone number template. Each template may contain one or more patterns that are separated by a comma (,). White space at the beginning of each pattern is ignored. “?” and “*” represent wildcard characters. To represent literally, use %xx. For example, %2a represents *. The template can contain up to 39 characters. Examples: “1408*, 1510*”, “1408123????, 555?1.”.
- Port—TCP/UDP Port number (0-65535). It can be specified in decimal or hex format.
- ProvisioningRuleSyntax—Scripting syntax that is used to define configuration resync and firmware upgrade rules.
- PwrLevel—Power level expressed in dBm with one decimal place, such as –13.5 or 1.5 (dBm).
- RscTmplt—A template of SIP Response Status Code, such as “404, 5*”, “61?”, “407, 408, 487, 481”. It can contain up to 39 characters.
- Sig<n>—Signed n-bit value. It can be specified in decimal or hex format. A “-” sign must precede negative values. A + sign before positive values is optional.
- Star Codes—Activation code for a supplementary service, such as *69. The code can contain up to 7 characters.
- Str<n>—A generic string with up to n nonreserved characters.
- Time<n>—Time duration in seconds, with up to n decimal places. Extra specified decimal places are ignored.
- ToneScript—A miniscript that specifies the frequency, level, and cadence parameters of a call progress tone. Script may contain up to 127 characters. Syntax: FreqScript;Z_1[;Z_2]. The section Z_1 is similar to the S_1 section in a CadScript, except that each on/off segment is followed by a frequency components parameter: $Z_1 = D_1(on_{i,1}/off_{i,1}/f_{i,1}[,on_{i,2}/off_{i,2}/f_{i,2}\ [,on_{i,3}/off_{i,3}/f_{i,3}\ [,on_{i,4}/off_{i,4}/f_{i,4}\ [,on_{i,5}/off_{i,5}/f_{i,5}\ [,on_{i,6}/off_{i,6}/f_{i,6}]]]]])$, where $f_{i,j} = n_1[+n_2]+n_3[+n_4[+n_5[+n_6]]]]$. $1 < n_k < 6$ specifies the frequency components in the FreqScript that are used in that segment. If more than one frequency component is used in a segment, the components are summed together.

  Example 1—Dial tone:

```
350@-19,440@-19;10(*/0/1+2)

Number of Frequencies = 2
Frequency 1 = 350 Hz at –19 dBm
Frequency 2 = 440 Hz at –19 dBm
Number of Cadence Sections = 1
Cadence Section 1: Section Length = 10 s
Number of Segments = 1
Segment 1: On=forever, with Frequencies 1 and 2

Total Tone Length = 10s
```

Example 2—Stutter tone:

```
350@-19,440@-19;2(.1/.1/1+2);10(*/0/1+2)

Number of Frequencies = 2
Frequency 1 = 350 Hz at –19 dBm
Frequency 2 = 440 Hz at –19 dBm
Number of Cadence Sections = 2
Cadence Section 1: Section Length = 2s
Number of Segments = 1
Segment 1: On=0.1s, Off=0.1s with Frequencies 1 and 2
Cadence Section 2: Section Length = 10s
Number of Segments = 1
Segment 1: On=forever, with Frequencies 1 and 2

Total Tone Length = 12s
```

- Uns<n>—Unsigned n-bit value, where n = 8, 16, or 32. It can be specified in decimal or hex format, such as 12 or 0x18, as long as the value can fit into n bits.

Note

- <Par Name> represents a configuration parameter name. In a profile, the corresponding tag is formed by replacing the space with an underscore "_", such as **Par_Name**.
- An empty default value field implies an empty string < "" >.
- The Cisco IP Phone continues to use the last configured values for tags that are not present in a given profile.
- Templates are compared in the order given. The first, *not the closest*, match is selected. The parameter name must match exactly.
- If more than one definition for a parameter is given in a profile, the last such definition in the file is the one that takes effect in the Cisco IP Phone.
- A parameter specification with an empty parameter value forces the parameter back to its default value. To specify an empty string instead, use the empty string "" as the parameter value.

Profile Updates and Firmware Upgrades

The Cisco IP Phone supports secure remote provisioning (configuration) and firmware upgrades. An unprovisioned phone can receive an encrypted profile targeted for that device. The phone does not require an explicit key due to a secure first-time provisioning mechanism that uses SSL functionality.

User intervention is not required to either start or complete a profile update, or firmware upgrade, or if intermediate upgrades are required to reach a future upgrade state from an older release. A profile resync is only attempted when the Cisco IP phone is idle, because a resync can trigger a software reboot and disconnect a call.

General-purpose parameters manage the provisioning process. Each Cisco IP phone can be configured to periodically contact a normal provisioning server (NPS). Communication with the NPS does not require the use of a secure protocol because the updated profile is encrypted by a shared secret key. The NPS can be a standard TFTP, HTTP, or HTTPS server with client certificates.

The administrator can upgrade, reboot, restart, or resync Cisco IP phones by using the phone web user interface. The administrator can also perform these tasks by using a SIP notify message.

Configuration profiles are generated by using common, open-source tools that integrate with service provider provisioning systems.

Related Topics

Allow and Configure Profile Updates, on page 28
Allow and Configure Firmware Upgrades, on page 28

Allow and Configure Profile Updates

Profile updates can be allowed at specified intervals. Updated profiles are sent from a server to the phone by using TFTP, HTTP, or HTTPS.

Procedure

Step 1 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.
Step 2 In the **Configuration Profile** section, choose **Yes** from the **Provision Enable** drop-down list box.
Step 3 Enter the parameters.
Step 4 Click **Submit All Changes**.

Allow and Configure Firmware Upgrades

Firmware updates can be allowed at specified intervals. Updated firmware is sent from a server to the phone by using TFTP or HTTP. Security is less of an issue with a firmware upgrade, because firmware does not contain personal information.

Procedure

Step 1 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.
Step 2 In the **Firmware Upgrade** section, choose **Yes** from the **Upgrade Enable** drop-down list box.
Step 3 Enter the parameters.
Step 4 Click **Submit All Changes**.

Upgrade Firmware by tftp/http/https

The phone supports single one image upgrade by tftp/http/https.

Note A device (with new base and DCU) may not be downgraded to an earlier firmware release, such as 9.3(3). For details, refer to the hardware information and the firmware and hardware compatibility information in the current release note document that is, *Cisco IP Conference Phone 7832 Multiplatform Phones Release Notes*.

Before You Begin

The firmware load file must be downloaded to an accessible server.

Procedure

Step 1 Rename the image as follows:
cp-x8xx-sip.aa-b-cMPP.cop to cp-x8xx-sip.aa-b-cMPP.tar.gz
where:

x8xx is 7832

aa-b-c is the release number, such as 10-4-1

Step 2 Use the tar –xzvf command to untar the tar ball.

Step 3 Copy the folder to a tftp/http/https download directory.

Step 4 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.

Step 5 Find the load filename which ends in **.loads** and append it to the valid URL.

Step 6 Click **Submit All Changes**.

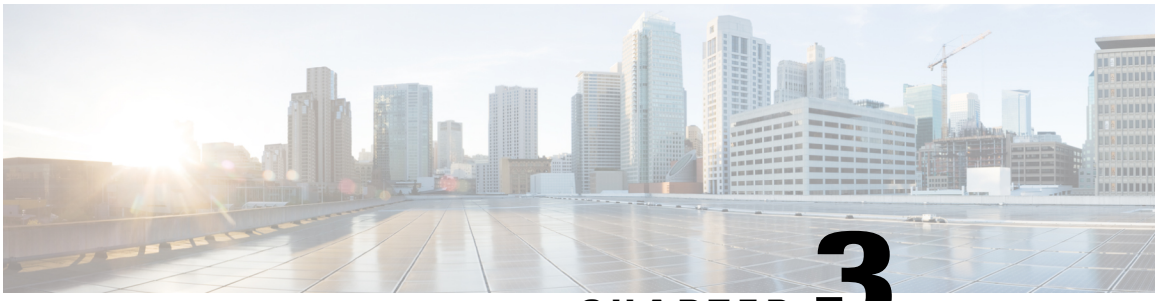
Upgrade Firmware With a Browser Command

An upgrade command entered into the browser address bar can be used to upgrade firmware on a phone. The phone updates only when it is idle. The update is attempted automatically after the call is complete.

Procedure

To upgrade the Cisco IP Phone CP-78xx-3PCC with a URL in a web browser, enter this command:

```
http://<phone_ip>/admin/upgrade?<schema>://<serv_ip[:port]>/filepath
```



CHAPTER 3

In-House Preprovisioning and Provisioning Servers

In-House Preprovisioning and Provisioning Servers

The service provider preprovisions Cisco IP Phones, other than RC units, with a profile. The preprovision profile can comprise a limited set of parameters that resynchronizes the Cisco IP Phone. The profile can also comprise a complete set of parameters that the remote server delivers. By default, the Cisco IP Phone resynchronizes on power-up and at intervals that are configured in the profile. When the user connects the Cisco IP Phone at the customer premises, the device downloads the updated profile and any firmware updates.

This process of preprovisioning, deployment, and remote provisioning can be accomplished in many ways.

Server Preparation and Software Tools

The examples in this chapter require the availability of one or more servers. These servers can be installed and run on a local PC:

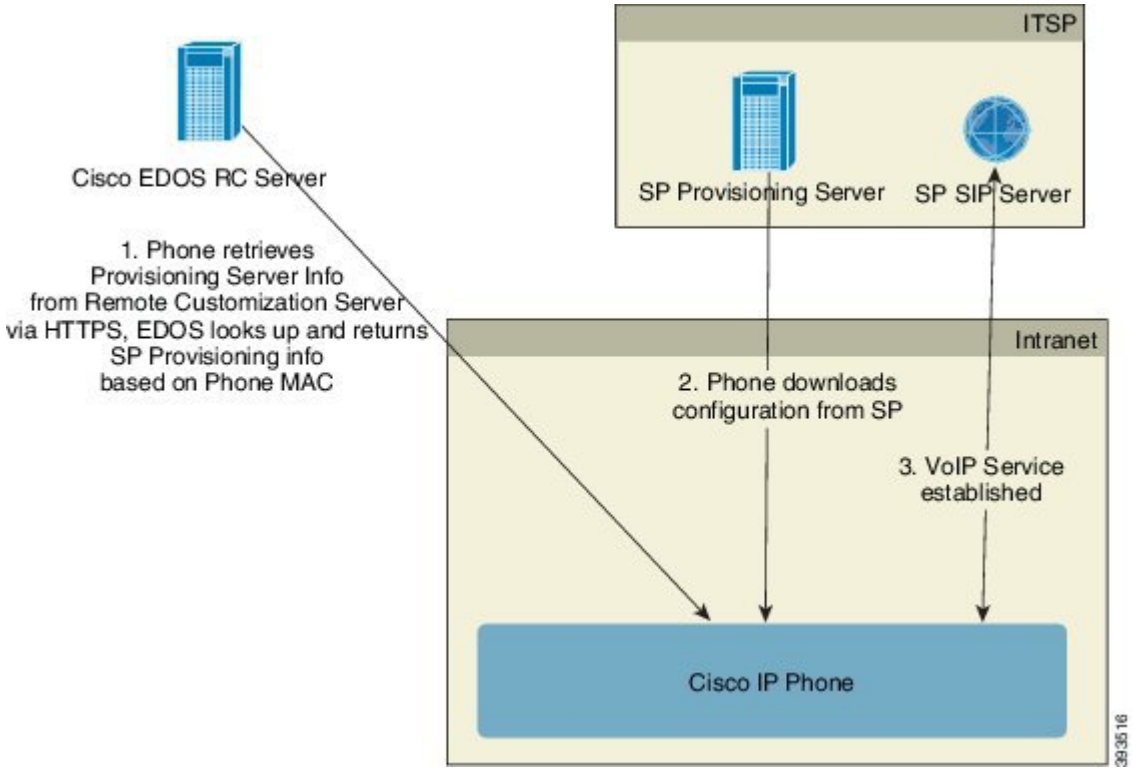
- TFTP (UDP port 69)
- syslog (UDP port 514)
- HTTP (TCP port 80)
- HTTPS (TCP port 443).

To troubleshoot server configuration, it is helpful to install clients for each type of server on a separate server machine. This practice establishes proper server operation, independent of the interaction with the Cisco IP Phones.

We also recommend that you install these software tools:

- To generate configuration profiles, install the open source gzip compression utility.
- For profile encryption and HTTPS operations, install the open source OpenSSL software package.
- To test the dynamic profile generation and one-step remote provisioning using HTTPS, we recommend a scripting language with CGI scripting support. Open source Perl language tools is an example of such a scripting language.
- To verify secure exchanges between provisioning servers and the Cisco IP Phones, install an Ethernet packet sniffer (such as the freely downloadable Ethereal/Wireshark). Capture an Ethernet packet trace of the interaction between the Cisco IP Phone and the provisioning server. To do so, run the packet sniffer on a PC that is connected to a switch with port mirroring enabled. For HTTPS transactions, you can use the ssldump utility.

Remote Customization (RC) Distribution



All Cisco IP Phones contact the Cisco EDOS RC server until they are provisioned initially.

In an RC distribution model, a customer purchases a Cisco IP Phone that has already been associated with a specific Service Provider in the Cisco EDOS RC Server. The Internet Telephony Service Provider (ITSP) sets up and maintains a provisioning server, and registers their provisioning server information with the Cisco EDOS RC Server.

When the Cisco IP Phone is powered on with an internet connection, the customization state for the unprovisioned Cisco IP Phone is **Open**. The phone first queries the local DHCP server for provisioning server

information and sets the customization state of the Cisco IP Phone. If DHCP query is successful, Customization State is set to **Aborted** and RC is not attempted due to DHCP providing the needed provisioning server information.

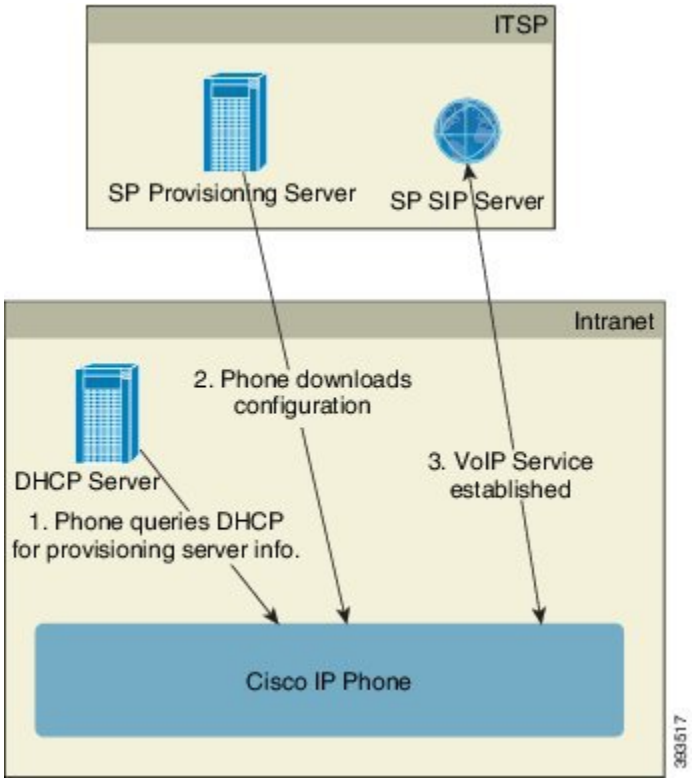
If DHCP server does not provide provisioning server information, the Cisco IP Phone queries the Cisco EDOS RC Server and provides its MAC address and model and the Customization State is set to **Pending**. The Cisco EDOS server responds with the associated service provider's provisioning server information including provisioning server URL and the Cisco IP Phone's Customization State is set to **Custom Pending**. The Cisco IP Phone then performs a resync URL command to retrieve the Service Provider's configuration and, if successful, the Customization State is set to **Acquired**.

If the Cisco EDOS RC Server does not have a service provider associated with the Cisco IP Phone, the customization state of the Cisco IP Phone is set to **Unavailable**. The phone can be manually configured or an association added for the service provider of the phone to the Cisco EDOS Server.

If a phone is provisioned via either the LCD or Web Configuration Utility, prior to the Customization State becoming **Acquired**, the Customization State is set to **Aborted** and the Cisco EDOS Server will not be queried unless the phone is factory reset.

Once the phone has been provisioned, the Cisco EDOS RC Server is not utilized unless the phone is factory reset.

In-House Device Preprovisioning



With the Cisco factory default configuration, a Cisco IP Phone automatically tries to resync to a profile on a TFTP server. A managed DHCP server on a LAN delivers the information about the profile and TFTP server

that is configured for preprovisioning to the device. The service provider connects each new Cisco IP Phone to the LAN. The Cisco IP Phone automatically resyncs to the local TFTP server and initializes its internal state in preparation for deployment. This preprovisioning profile typically includes the URL of a remote provisioning server. The provisioning server keeps the device updated after the device is deployed and connected to the customer network.

The preprovisioned device bar code can be scanned to record its MAC address or serial number before the Cisco IP Phone is shipped to the customer. This information can be used to create the profile to which the Cisco IP Phone resynchronizes.

Upon receiving the Cisco IP Phone, the customer connects it to the broadband link. On power-up, the Cisco IP Phone contacts the provisioning server through the URL that is configured through preprovisioning. The Cisco IP Phone can thus resync and update the profile and firmware, as necessary.

Provisioning Server Setup

This section describes setup requirements for provisioning a Cisco IP Phone by using various servers and different scenarios. For the purposes of this document and for testing, provisioning servers are installed and run on a local PC. Also, generally available software tools are useful for provisioning the Cisco IP Phones.

TFTP Provisioning

Cisco IP Phones support TFTP for both provisioning resync and firmware upgrade operations. When devices are deployed remotely, HTTPS is recommended, but HTTP and TFTP can also be used. This then requires provisioning file encryption to add security, as it offers greater reliability, given NAT and router protection mechanisms. TFTP is useful for the in-house preprovisioning of a large number of unprovisioned devices.

The Cisco IP Phone is able to obtain a TFTP server IP address directly from the DHCP server through DHCP option 66. If a Profile_Rule is configured with the filepath of that TFTP server, the device downloads its profile from the TFTP server. The download occurs when the device is connected to a LAN and powered up.

The Profile_Rule provided with the factory default configuration is *$PN*.cfg, where *$PN* represents the phone model name, such as CP-7832-3PCC. For example, for a CP-7832-3PCC, the filename is CP-7832-3PCC.cfg. For a device with the factory default profile, upon powering up, the device resyncs to this file on the local TFTP server that DHCP option 66 specifies. (The filepath is relative to the TFTP server virtual root directory.)

Related Topics

In-House Device Preprovisioning, on page 33

Remote Endpoint Control and NAT

The Cisco IP Phone is compatible with network address translation (NAT) to access the Internet through a router. For enhanced security, the router might attempt to block unauthorized incoming packets by implementing symmetric NAT, a packet-filtering strategy that severely restricts the packets that are allowed to enter the protected network from the Internet. For this reason, remote provisioning by using TFTP is not recommended.

Voice over IP can co-exist with NAT only when some form of NAT traversal is provided. Configure Simple Traversal of UDP through NAT (STUN). This option requires that the user have:

- A dynamic external (public) IP address from your service
- A computer that is running STUN server software

- An edge device with an asymmetric NAT mechanism

HTTP Provisioning

The Cisco IP Phone behaves like a browser that requests web pages from a remote Internet site. This provides a reliable means of reaching the provisioning server, even when a customer router implements symmetric NAT or other protection mechanisms. HTTP and HTTPS work more reliably than TFTP in remote deployments, especially when the deployed units are connected behind residential firewalls or NAT-enabled routers. HTTP and HTTPs are used interchangeably in the following request type descriptions.

Basic HTTP-based provisioning relies on the HTTP GET method to retrieve configuration profiles. Typically, a configuration file is created for each deployed Cisco IP Phone, and these files are stored within an HTTP server directory. When the server receives the GET request, it simply returns the file that is specified in the GET request header.

Rather than a static profile, the configuration profile can be generated dynamically by querying a customer database and producing the profile on-the-fly.

When the Cisco IP Phone requests a resynch, it can use the HTTP POST method to request the resync configuration data. The device can be configured to convey certain status and identification information to the server within the body of the HTTP POST request. The server uses this information to generate a desired response configuration profile, or to store the status information for later analysis and tracking.

As part of both GET and POST requests, the Cisco IP Phone automatically includes basic identifying information in the User-Agent field of the request header. This information conveys the manufacturer, product name, current firmware version, and product serial number of the device.

The following example is the User-Agent request field from a CP-7832-3PCC:

```
User-Agent: Cisco-CP-7832-3PCC/11.0.1 (00562b043615)
```

When the Cisco IP Phone is configured to resync to a configuration profile by using HTTP, it is recommended that HTTPS be used or the profile be encrypted to protect confidential information. The Cisco IP Phone supports 256-bit AES in CBC mode to decrypt profiles. Encrypted profiles that the Cisco IP Phone downloads by using HTTP avoid the danger of exposing confidential information that is contained in the configuration profile. This resync mode produces a lower computational load on the provisioning server when compared to using HTTPS.

Note The Cisco IP Conference Phone 7832 Multiplatform Phones support HTTP Version 1.0, HTTP Version 1.1, and Chunk Encoding when HTTP Version 1.1 is the negotiated transport protocol.

HTTP Status Code Handling on Resync and Upgrade

The phone supports HTTP response for remote provisioning (Resync). Current phone behavior is categorized in three ways:

- A—Success, where the “Resync Periodic” and “Resync Random Delay” values determine subsequent requests.
- B—Failure when File Not Found or corrupt profile. The “Resync Error Retry Delay” value determines subsequent requests.

- C—Other failure when a bad URL or IP address causes a connection error. The “Resync Error Retry Delay” value determines subsequent requests.

Table 1: Phone Behavior for HTTP Responses

HTTP Status Code	Description	Phone Behavior
301 Moved Permanently	This and future requests should be directed to a new location.	Retry request immediately with new location.
302 Found	Known as Temporarily Moved.	Retry request immediately with new location.
3xx	Other 3xx responses not processed.	C
400 Bad Request	The request cannot be fulfilled due to bad syntax.	C
401 Unauthorized	Basic or digest access authentication challenge.	Immediately retry request with authentication credentials. Maximum 2 retries. Upon failure, the phone behavior is C.
403 Forbidden	Server refuses to respond.	C
404 Not Found	Requested resource not found. Subsequent requests by client are permissible.	B
407 Proxy Authentication Required	Basic or digest access authentication challenge.	Immediately retry request with authentication credentials. Maximum two retries. Upon failure, the phone behavior is C.
4xx	Other client error status codes are not processed.	C
500 Internal Server Error	Generic error message.	Cisco IP Phone behavior is C.
501 Not Implemented	The server does not recognize the request method, or it lacks the ability to fulfill the request.	Cisco IP Phone behavior is C.
502 Bad Gateway	The server is acting as a gateway or proxy and receives an invalid response from the upstream server.	Cisco IP Phone behavior is C.
503 Service Unavailable	The server is currently unavailable (overloaded or down for maintenance). This is a temporary state.	Cisco IP Phone behavior is C.

HTTP Status Code	Description	Phone Behavior
504 Gateway Timeout	The server behaves as a gateway or proxy and does not receive timely response from the upstream server.	C
5xx	Other server error	C

HTTPS Provisioning

The Cisco IP Phone supports HTTPS for provisioning for increased security in managing remotely deployed units. Each Cisco IP Phone carries a unique SLL Client Certificate (and associated private key), in addition to a Sipura CA server root certificate. The latter allows the Cisco IP Phone to recognize authorized provisioning servers, and reject non-authorized servers. On the other hand, the client certificate allows the provisioning server to identify the individual device that issues the request.

For a service provider to manage deployment by using HTTPS, a server certificate must be generated for each provisioning server to which a Cisco IP Phone resyncs by using HTTPS. The server certificate must be signed by the Cisco Server CA Root Key, whose certificate is carried by all deployed units. To obtain a signed server certificate, the service provider must forward a certificate signing request to Cisco, which signs and returns the server certificate for installation on the provisioning server.

The provisioning server certificate must contain the Common Name (CN) field, and the FQDN of the host running the server in the subject. It might optionally contain information following the host FQDN, separated by a slash (/) character. The following examples are of CN entries that are accepted as valid by the Cisco IP Phone:

```
CN=sprov.callme.com
CN=pv.telco.net/mailto:admin@telco.net
CN=prof.voice.com/info@voice.com
```

In addition to verifying the server certificate, the Cisco IP Phone tests the server IP address against a DNS lookup of the server name that is specified in the server certificate.

Get Signed Server Certificate

The OpenSSL utility can generate a certificate signing request. The following example shows the **openssl** command that produces a 1024-bit RSA public/private key pair and a certificate signing request:

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

This command generates the server private key in **privkey.pem** and a corresponding certificate signing request in **provserver.csr**. The service provider keeps the **privkey.pem** secret and submits **provserver.csr** to Cisco for signing. Upon receiving the **provserver.csr** file, Cisco generates **provserver.crt**, the signed server certificate.

To get the signed server certificate:

Procedure

Step 1 Navigate to https://webapps.cisco.com/software/edos/home and log in with your CCO credentials.

Step 2 Select **Certificate Management**.
On the **Sign CSR** tab, the CSR of the previous step is uploaded for signing.

Step 3 From the **Select Product** drop-down list box, select **SPA1xx firmware 1.3.3 and newer/SPA232D firmware 1.3.3 and newer/SPA5xx firmware 7.5.6 and newer/CP-78xx-3PCC/CP-88xx-3PCC**.

Step 4 In the **CSR File** field, click **Browse** and select the CSR for signing.

Step 5 From the **Sign in Duration** drop-down list box, select the applicable duration (for example, 1 year).

Step 6 Click **Sign Certificate Request**.

Step 7 Select one of the following options to receive the signed certificate:

- **Enter Recipient's Email Address**: If you wish to receive the certificate via email, enter your email address in this field.
- **Download**: If you wish to download the signed certificate, select this option.

Step 8 Click **Submit**.
The signed server certificate is then either emailed to the email address previously provided or downloaded.

Sipura CA Client Root Certificate

Cisco also provides a Sipura CA Client Root Certificate to the service provider. This root certificate certifies the authenticity of the client certificate that each Cisco IP Phone carries. Cisco IP Conference Phone 7832 Multiplatform Phones also support third-party signed certificates such as those provided by Verisign, Cybertrust, and so on.

The unique client certificate that each device offers during an HTTPS session carries identifying information that is embedded in its subject field. This information can be made available by the HTTPS server to a CGI script invoked to handle secure requests. In particular, the certificate subject indicates the unit product name (OU element), MAC address (S element), and serial number (L element). The following example from the Cisco IP Phone 7832 Multiplatform Phones client certificate subject field shows these elements:

```
OU=CP-7832-3PCC, L=88012BA01234, S=000e08abcdef
```

Units manufactured before firmware 2.0.x do not contain individual SSL client certificates. When these units are upgraded to a firmware release in the 2.0.x tree, they become capable of connecting to a secure server that is using HTTPS, but are only able to supply a generic client certificate if the server requests them to do so. This generic certificate contains the following information in the identifying fields:

```
OU=cisco.com, L=ciscogeneric, S=ciscogeneric
```

To determine if a Cisco IP Phone carries an individualized certificate, use the $CCERT provisioning macro variable. The variable value expands to either Installed or Not Installed, according to the presence or absence of a unique client certificate. In the case of a generic certificate, it is possible to obtain the serial number of the unit from the HTTP request header in the User-Agent field.

HTTPS servers can be configured to request SSL certificates from connecting clients. If enabled, the server can use the Sipura CA Client Root Certificate that Cisco supplies to verify the client certificate. The server can then provide the certificate information to a CGI for further processing.

The location for certificate storage may vary. For example, in an Apache installation, the file paths for storage of the provisioning server-signed certificate, its associated private key, and the Sipura CA client root certificate are as follows:

```
# Server Certificate:
SSLCertificateFile /etc/httpd/conf/provserver.crt

# Server Private Key:
SSLCertificateKeyFile /etc/httpd/conf/provserver.key

# Certificate Authority (CA):
SSLCACertificateFile /etc/httpd/conf/spacroot.crt
```

For specific information, refer to the documentation for an HTTPS server.

The Cisco Client Certificate Root Authority signs each unique certificate. The corresponding root certificate is made available to service providers for client authentication purposes.

Redundant Provisioning Servers

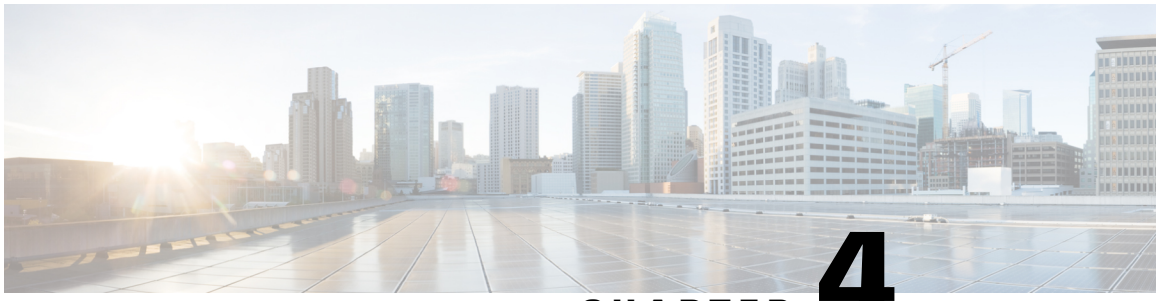
The provisioning server can be specified as an IP address or as a Fully Qualified Domain Name (FQDN). The use of an FQDN facilitates the deployment of redundant provisioning servers. When the provisioning server is identified through an FQDN, the Cisco IP Phone attempts to resolve the FQDN to an IP address through DNS. Only DNS A-records are supported for provisioning; DNS SRV address resolution is not available for provisioning. The Cisco IP Phone continues to process A-records until a server responds. If no server that is associated with the A-records responds, the Cisco IP Phone logs an error to the syslog server.

Syslog Server

If a syslog server is configured on the Cisco IP Phone through use of the <Syslog Server> parameters, the resync and upgrade operations send messages to the syslog server. A message can be generated at the start of a remote file request (configuration profile or firmware load), and at the conclusion of the operation (indicating either success or failure).

The logged messages are configured in the following parameters and macro expanded into the actual syslog messages:

- Log_Request_Msg
- Log_Success_Msg
- Log_Failure_Msg



CHAPTER 4

Provisioning Examples

Provisioning Examples Overview

This chapter provides example procedures for transferring configuration profiles between the Cisco IP Phone and the provisioning server.

For information about creating configuration profiles, refer to Provisioning Scripts, on page 9.

Basic Resync

This section demonstrates the basic resync functionality of the Cisco IP Phones.

TFTP Resync

The Cisco IP Phone supports multiple network protocols for retrieving configuration profiles. The most basic profile transfer protocol is TFTP (RFC1350). TFTP is widely used for the provisioning of network devices within private LAN networks. Although not recommended for the deployment of remote endpoints across the Internet, TFTP can be convenient for deployment within small organizations, for in-house preprovisioning, and for development and testing. See the In-House Device Preprovisioning, on page 33 section for more information on in-house preprovisioning. In the following procedure, a profile is modified after downloading a file from a TFTP server.

Procedure

Step 1 Within a LAN environment, connect a PC and a Cisco IP Phone to a hub, switch, or small router.

Step 2 On the PC, install and activate a TFTP server.

Step 3 Use a text editor to create a configuration profile that sets the value for GPP_A to 12345678 as shown in the example.

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

Step 4 Save the profile with the name `basic.txt` in the root directory of the TFTP server.
You can verify that the TFTP server is properly configured: request the `basic.txt` file by using a TFTP client other than the Cisco IP Phone. Preferably, use a TFTP client that is running on a separate host from the provisioning server.

Step 5 Open the PC web browser on the admin/advanced configuration page. For example, if the IP address of the phone is 192.168.1.100:

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

Step 6 Select the **Provisioning** tab, and inspect the values of the general purpose parameters GPP_A through GPP_P. These should be empty.

Step 7 Resync the test Cisco IP Phone to the `basic.txt` configuration profile by opening the resync URL in a web browser window.
If the IP address of the TFTP server is 192.168.1.200, the command should be similar to the following example:

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

When Cisco IP Phone receives this command, the device at address 192.168.1.100 requests the file `basic.txt` from the TFTP server at IP address 192.168.1.200. The phone then parses the downloaded file and updates the GPP_A parameter with the value 12345678.

Step 8 Verify that the parameter was correctly updated: Refresh the admin/advanced page on the PC web browser and select the **Provisioning** tab on that page.
The GPP_A parameter should now contain the value 12345678.

Logging with syslog

The Cisco IP Phone sends a syslog message to the designated syslog server when the device is about to resync to a provisioning server and after the resync has either completed or failed. This server is identified in the web server administration (**Admin Login** > **advanced** > **Voice** > **System**, Syslog Server parameter). Configure the syslog server IP address into the device and observe the messages that are generated during the remaining procedures.

Procedure

Step 1 Install and activate a syslog server on the local PC.

Step 2 Program the PC IP address into the Syslog_Server parameter of the profile and submit the change:

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

Step 3 Click the **System** tab and enter the value of your local syslog server into the Syslog_Server parameter.

Step 4 Repeat the resync operation as described in the TFTP Resync, on page 41 procedure.
The device generates two syslog messages during the resync. The first message indicates that a request is in progress. The second message marks success or failure of the resync.

Step 5 Verify that your syslog server received messages similar to the following:

```
CP-78xx-3PCC 00:0e:08:ab:cd:ef -- Requesting resync tftp://192.168.1.200/basic.txt
```

Detailed messages are available by programming a Debug_Server parameter (instead of the Syslog_Server parameter) with the IP address of the syslog server, and by setting the Debug_Level to a value between 0 and 3 (3 being the most verbose):

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

The contents of these messages can be configured by using the following parameters:

- Log_Request_Msg
- Log_Success_Msg
- Log_Failure_Msg

If any of these parameters are cleared, the corresponding syslog message is not generated.

Automatic Device Resync

A device can resync periodically to the provisioning server to ensure that any profile changes made on the server are propagated to the endpoint device (as opposed to sending an explicit resync request to the endpoint).

To cause the Cisco IP Phone to periodically resync to a server, a configuration profile URL is defined by using the Profile_Rule parameter, and a resync period is defined by using the Resync_Periodic parameter.

Procedure

Step 1 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.

Step 2 Define the Profile_Rule parameter. This example assumes a TFTP server IP address of 192.168.1.200.

Step 3 In the **Resync Periodic** field, enter a small value for testing, such as **30** seconds.

Step 4 Click **Submit all Changes**.

With the new parameter settings, the Cisco IP Phone resyncs twice a minute to the configuration file that the URL specifies.

Step 5 Observe the resulting messages in the syslog trace (as described in the Logging with syslog, on page 42 section).

Step 6 Ensure that the **Resync On Reset** field is set to **Yes**.

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

Step 7 Power cycle the Cisco IP Phone to force it to resync to the provisioning server.
If the resync operation fails for any reason, such as if the server is not responding, the unit waits (for the number of seconds configured in **Resync Error Retry Delay**) before it attempts to resync again. If **Resync Error Retry Delay** is zero, the Cisco IP Phone does not try to resync after a failed resync attempt.

Step 8 (Optional) Set the value of **Resync Error Retry Delay** field to a small number, such as **30**.

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

Step 9 Disable the TFTP server, and observe the results in the syslog output.

Unique Profiles, Macro Expansion, and HTTP

In a deployment where each Cisco IP Phone must be configured with distinct values for some parameters, such as User_ID or Display_Name, the service provider can create a unique profile for each deployed device and host those profiles on a provisioning server. Each Cisco IP Phone, in turn, must be configured to resync to its own profile according to a predetermined profile naming convention.

The profile URL syntax can include identifying information that is specific to each Cisco IP Phone, such as MAC address or serial number, by using the macro expansion of built-in variables. Macro expansion eliminates the need to specify these values in multiple locations within each profile.

A profile rule undergoes macro expansion before the rule is applied to the Cisco IP Phone. The macro expansion controls a number of values, for example:

- $MA expands to the unit 12-digit MAC address (using lower case hex digits). For example, 000e08abcdef.
- $SN expands to the unit serial number. For example, 88012BA01234.

Other values can be macro expanded in this way, including all the general purpose parameters, GPP_A through GPP_P. An example of this process can be seen in the TFTP Resync, on page 41 section. Macro expansion is not limited to the URL file name, but can also be applied to any portion of the profile rule parameter. These parameters are referenced as $A through $P. For a complete list of variables that are available for macro expansion, see the Macro Expansion Variables, on page 63 section.

In this exercise, a profile specific to the Cisco IP Conference Phone 7832 is provisioned on a TFTP server.

Exercise: Provision a Specific IP Phone Profile on a TFTP Server

Procedure

Step 1 Obtain the MAC address of the phone from its product label. (The MAC address is the number, using numbers and lower–case hex digits, such as 000e08aabbcc.

Step 2 Copy the `basic.txt` configuration file (described in the TFTP Resync, on page 41 section) to a new file named `CP-x8xx-3PCC macaddress.cfg` (replacing `x8xx` with the model number and `macaddress` with the MAC address of the phone).

Step 3 Move the new file in the virtual root directory of the TFTP server.

Step 4 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.

Step 5 Enter `tftp://192.168.1.200/CP-7832-3PCC$MA.cfg` in the **Profile Rule** field.

```
<Profile_Rule>
  tftp://192.168.1.200/CP-7832-3PCC$MA.cfg
</Profile_Rule>
```

Step 6 Click **Submit All Changes**. This causes an immediate reboot and resync.
When the next resync occurs, the Cisco IP Phone retrieves the new file by expanding the $MA macro expression into its MAC address.

HTTP GET Resync

HTTP provides a more reliable resync mechanism than TFTP because HTTP establishes a TCP connection and TFTP uses the less reliable UDP. In addition, HTTP servers offer improved filtering and logging features compared to TFTP servers.

On the client side, the Cisco IP Phone does not require any special configuration setting on the server to be able to resync by using HTTP. The Profile_Rule parameter syntax for using HTTP with the GET method is similar to the syntax that is used for TFTP. If a standard web browser can retrieve a profile from your HTTP server, the Cisco IP Phone should be able to do so as well.

Exercise: HTTP GET Resync

Procedure

Step 1 Install an HTTP server on the local PC or other accessible host. (The open source Apache server can be downloaded from the internet.)

Step 2 Copy the `basic.txt` configuration profile (described in the TFTP Resync, on page 41 section) onto the virtual root directory of the installed server.

Step 3 To verify proper server installation and file access to `basic.txt`, access the profile by using a web browser.

Step 4 Modify the Profile_Rule of the test Cisco IP Phone to point to the HTTP server in place of the TFTP server, so as to download its profile periodically.

For example, assuming the HTTP server is at 192.168.1.300, enter the following value:

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

Step 5 Click **Submit All Changes**. This causes an immediate reboot and resync.

Step 6 Observe the syslog messages that the Cisco IP Phone sends. The periodic resyncs should now be obtaining the profile from the HTTP server.

Step 7 In the HTTP server logs, observe how information that identifies the test Cisco IP Phone appears in the log of user agents.
This information should include the manufacturer, product name, current firmware version, and serial number.

Provisioning Through Cisco XML

For the Cisco IP Conference Phone 7832, designated as x8xx here, provisioning through Cisco XML functions as follows.

You can send an XML object to the phone by a SIP Notify packet or an HTTP Post to the CGI interface of the phone: `http://IPAddressPhone/CGI/Execute`.

The CP-x8xx-3PCC extends the Cisco XML feature to support provisioning via an XML object:

```
<CP-x8xx-3PCCExecute>
        <ExecuteItem URL=Resync:[profile-rule]/>
</CP-x8xx-3PCCExecute>
```

After it receives the XML object, the CP-x8xx-3PCC downloads the provisioning file from [profile-rule]. This rule uses macros to simplify the development of the XML services application.

URL Resolution by Using Macro Expansion

Subdirectories with multiple profiles on the server provide a convenient method for managing a large number of deployed devices. The profile URL can contain:

- A provisioning server name or an explicit IP address. If the profile identifies the provisioning server by name, the Cisco IP Phone performs a DNS lookup to resolve the name.
- A nonstandard server port that is specified in the URL by using the standard syntax `:port` following the server name.
- The subdirectory of the server virtual root directory where the profile is stored, specified by using standard URL notation and managed by macro expansion.

For example, the following Profile_Rule requests the profile CP-7832-3PCC.cfg, in the server subdirectory `/cisco/config`, from the TFTP server that is running on host prov.telco.com listening for a connection on port 6900:

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

A profile for each Cisco IP Phone can be identified in a general purpose parameter, with its value referred within a common profile rule by using macro expansion.

For example, assume GPP_B is defined as `Dj6Lmp23Q`.

The Profile_Rule has the value:

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

When the device resyncs and the macros are expanded, the Cisco IP Phone with a MAC address of 000e08012345 requests the profile with the name that contains the device MAC address at the following URL:

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

Secure HTTPS Resync

These mechanisms are available on the Cisco IP Phone for resyncing by using a secure communication process:

- Basic HTTPS Resync
- HTTPS with Client Certificate Authentication
- HTTPS Client Filtering and Dynamic Content

Related Topics

Basic HTTPS Resync, on page 47
HTTPS with Client Certificate Authentication, on page 49
HTTPS Client Filtering and Dynamic Content, on page 50

Basic HTTPS Resync

HTTPS adds SSL to HTTP for remote provisioning so that the:

- Cisco IP Phone can authenticate the provisioning server.
- Provisioning server can authenticate the Cisco IP Phone.
- Confidentiality of information exchanged between the Cisco IP Phone and the provisioning server is ensured.

SSL generates and exchanges secret (symmetric) keys for each connection between the Cisco IP Phone and the server, using public/private key pairs that are pre-installed in the Cisco IP Phone and the provisioning server.

On the client side, the Cisco IP Phone does not require any special configuration setting on the server to be able to resync using HTTPS. The Profile_Rule parameter syntax for using HTTPS with the GET method is similar to the syntax that is used for HTTP or TFTP. If a standard web browser can retrieve a profile from a your HTTPS server, the Cisco IP Phone should be able to do so as well.

In addition to installing a HTTPS server, a SSL server certificate that Cisco signs must be installed on the provisioning server. The devices cannot resync to a server that is using HTTPS unless the server supplies a Cisco-signed server certificate. Instructions for creating signed SSL Certificates for Voice products can be found at https://supportforums.cisco.com/docs/DOC-9852.

Exercise: Basic HTTPS Resync

Procedure

Step 1 Install an HTTPS server on a host whose IP address is known to the network DNS server through normal hostname translation.
The open source Apache server can be configured to operate as an HTTPS server when installed with the open source mod_ssl package.

Step 2 Generate a server Certificate Signing Request for the server. For this step, you might need to install the open source OpenSSL package or equivalent software. If using OpenSSL, the command to generate the basic CSR file is as follows:

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

This command generates a public/private key pair, which is saved in the `privkey.pem` file.

Step 3 Submit the CSR file (provserver.csr) to Cisco for signing. (See https://supportforums.cisco.com/docs/DOC-9852 for more information.)
A signed server certificate is returned (provserver.cert) along with a Sipura CA Client Root Certificate, spacroot.cert.

Step 4 Store the signed server certificate, the private key pair file, and the client root certificate in the appropriate locations on the server.
In the case of an Apache installation on Linux, these locations are typically as follows:

```
# 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
```

Step 5 Restart the server.

Step 6 Copy the `basic.txt` configuration file (described in the TFTP Resync, on page 41 section) onto the virtual root directory of the HTTPS server.

Step 7 Verify proper server operation by downloading `basic.txt` from the HTTPS server by using a standard browser from the local PC.

Step 8 Inspect the server certificate that the server supplies.
The browser probably does not recognize the certificate as valid unless the browser has been pre-configured to accept Cisco as a root CA. However, the Cisco IP Phones expect the certificate to be signed this way.

Modify the Profile_Rule of the test device to contain a reference to the HTTPS server, for example:

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

This example assumes the name of the HTTPS server is `my.server.com`.

Step 9 Click **Submit All Changes**.

Step 10 Observe the syslog trace that the Cisco IP Phone sends.
The syslog message should indicate that the resync obtained the profile from the HTTPS server.

Step 11 (Optional) Use an Ethernet protocol analyzer on the Cisco IP Phone subnet to verify that the packets are encrypted.
In this exercise, client certificate verification was not enabled. The connection between Cisco IP Phone and server is encrypted. However, the transfer is not secure because any client can connect to the server and request the file, given knowledge of the file name and directory location. For secure resync, the server must also authenticate the client, as demonstrated in the exercise described in the HTTPS with Client Certificate Authentication, on page 49 section.

HTTPS with Client Certificate Authentication

In the factory default configuration, the server does not request a SSL client certificate from a client. Transfer of the profile is not secure because any client can connect to the server and request the profile. You can edit the configuration to enable client authentication; the server requires a client certificate to authenticate the Cisco IP Phone before it accepts a connection request.

Because of this requirement, the resync operation cannot be independently tested by using a browser that lacks the proper credentials. The SSL key exchange within the HTTPS connection between the test Cisco IP Phone and the server can be observed with the ssldump utility. The utility trace shows the interaction between client and server.

Exercise: HTTPS with Client Certificate Authentication

Procedure

Step 1 Enable client certificate authentication on the HTTPS server.

Step 2 In Apache (v.2), set the following in the server configuration file:

```
SSLVerifyClient  require
```

Also, ensure that the spacroot.cert has been stored as shown in the Basic HTTPS Resync, on page 47 exercise.

Step 3 Restart the HTTPS server and observe the syslog trace from the Cisco IP Phone.
Each resync to the server now performs symmetric authentication, so that both the server certificate and the client certificate are verified before the profile is transferred.

Step 4 Use ssldump to capture a resync connection between the Cisco IP Phone and the HTTPS server.
If client certificate verification is properly enabled on the server, the ssldump trace shows the symmetric exchange of certificates (first server-to-client, then client-to-server) before the encrypted packets that contain the profile.

With client authentication enabled, only a Cisco IP Phone with a MAC address that matches a valid client certificate can request the profile from the provisioning server. The server rejects a request from an ordinary browser or other unauthorized device.

HTTPS Client Filtering and Dynamic Content

If the HTTPS server is configured to require a client certificate, the information in the certificate identifies the resyncing Cisco IP Phone and supplies it with the correct configuration information.

The HTTPS server makes the certificate information available to CGI scripts (or compiled CGI programs) that are invoked as part of the resync request. For the purpose of illustration, this exercise uses the open source Perl scripting language, and assumes that Apache (v.2) is used as the HTTPS server.

Procedure

Step 1 Install Perl on the host that is running the HTTPS server.

Step 2 Generate the following Perl reflector script:

```
#!/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>”;
```

Step 3 Save this file with the file name `reflect.pl`, with executable permission (chmod 755 on Linux), in the CGI scripts directory of the HTTPS server.

Step 4 Verify accessibility of CGI scripts on the server (that is, `/cgi-bin/…`).

Step 5 Modify the Profile_Rule on the test device to resync to the reflector script, as in the following example:

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

Step 6 Click **Submit All Changes**.

Step 7 Observe the syslog trace to ensure a successful resync.

Step 8 On the Configuration Utility page, select **Admin Login** > **advanced** > **Voice** > **Provisioning**.

Step 9 Verify that the GPP_D parameter contains the information that the script captured.
This information contains the product name, MAC address, and serial number if the test device carries a unique certificate from the manufacturer. The information contains generic strings if the unit was manufactured before firmware release 2.0.

A similar script can determine information about the resyncing device and then provide the device with appropriate configuration parameter values.

HTTPS Certificate

The Cisco IP Phone provides a reliable and secure provisioning strategy that is based on HTTPS requests from the device to the provisioning server. Both a server certificate and a client certificate are used to authenticate the Cisco IP Phone to the server and the server to the Cisco IP Phone.

To use HTTPS with the phone, you must generate a Certificate Signing Request (CSR) and submit it to Cisco. The Cisco IP Phone generates a certificate for installation on the provisioning server. The Cisco IP Phone accepts the certificate when it seeks to establish an HTTPS connection with the provisioning server.

HTTPS Methodology

HTTPS encrypts the communication between a client and a server, thus protecting the message contents from other network devices. The encryption method for the body of the communication between a client and a server is based on symmetric key cryptography. With symmetric key cryptography, a client and a server share a single secret key over a secure channel that is protected by Public/Private key encryption.

Messages encrypted by the secret key can only be decrypted by using the same key. HTTPS supports a wide range of symmetric encryption algorithms. The Cisco IP Phone implements up to 256-bit symmetric encryption, using the American Encryption Standard (AES), in addition to 128-bit RC4.

HTTPS also provides for the authentication of a server and a client engaged in a secure transaction. This feature ensures that a provisioning server and an individual client cannot be spoofed by other devices on the network. This capability is essential in the context of remote endpoint provisioning.

Server and client authentication is performed by using public/private key encryption with a certificate that contains the public key. Text that is encrypted with a public key can be decrypted only by its corresponding private key (and vice versa). The Cisco IP Phone supports the Rivest-Shamir-Adleman (RSA) algorithm for public/private key cryptography.

SSL Server Certificate

Each secure provisioning server is issued a secure sockets layer (SSL) server certificate that Cisco signs directly. The firmware that runs on the Cisco IP Phone recognizes only a Cisco certificate as valid. When a client connects to a server by using HTTPS, it rejects any server certificate that is not signed by Cisco.

This mechanism protects the service provider from unauthorized access to the Cisco IP Phone, or any attempt to spoof the provisioning server. Without such protection, an attacker might be able to reprovision the Cisco IP Phone, to gain configuration information, or to use a different VoIP service. Without the private key that corresponds to a valid server certificate, the attacker is unable to establish communication with a Cisco IP Phone.

Obtain a Server Certificate

Procedure

Step 1 Contact a Cisco support person who will work with you on the certificate process. If you are not working with a specific support person, email your request to ciscosb-certadmin@cisco.com.

Step 2 Generate a private key that will be used in a CSR (Certificate Signing Request). This key is private and you do not need to provide this key to Cisco support. Use open source "openssl" to generate the key. For example:

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

Step 3 Generate a CSR that contains fields that identify your organization and location. For example:

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

You must have the following information:

- Subject field—Enter the Common Name (CN) that must be a FQDN (Fully Qualified Domain Name) syntax. During SSL authentication handshake, the Cisco IP Phone verifies that the certificate it receives is from the machine that presented it.
- Server hostname—For example, provserv.domain.com.
- Email address—Enter an email address so that customer support can contact you if needed. This email address is visible in the CSR.

Step 4 Email the CSR (in zip file format) to the Cisco support person or to ciscosb-certadmin@cisco.com. The certificate is signed by Cisco. Cisco sends the certificate to you to install on your system.

Client Certificate

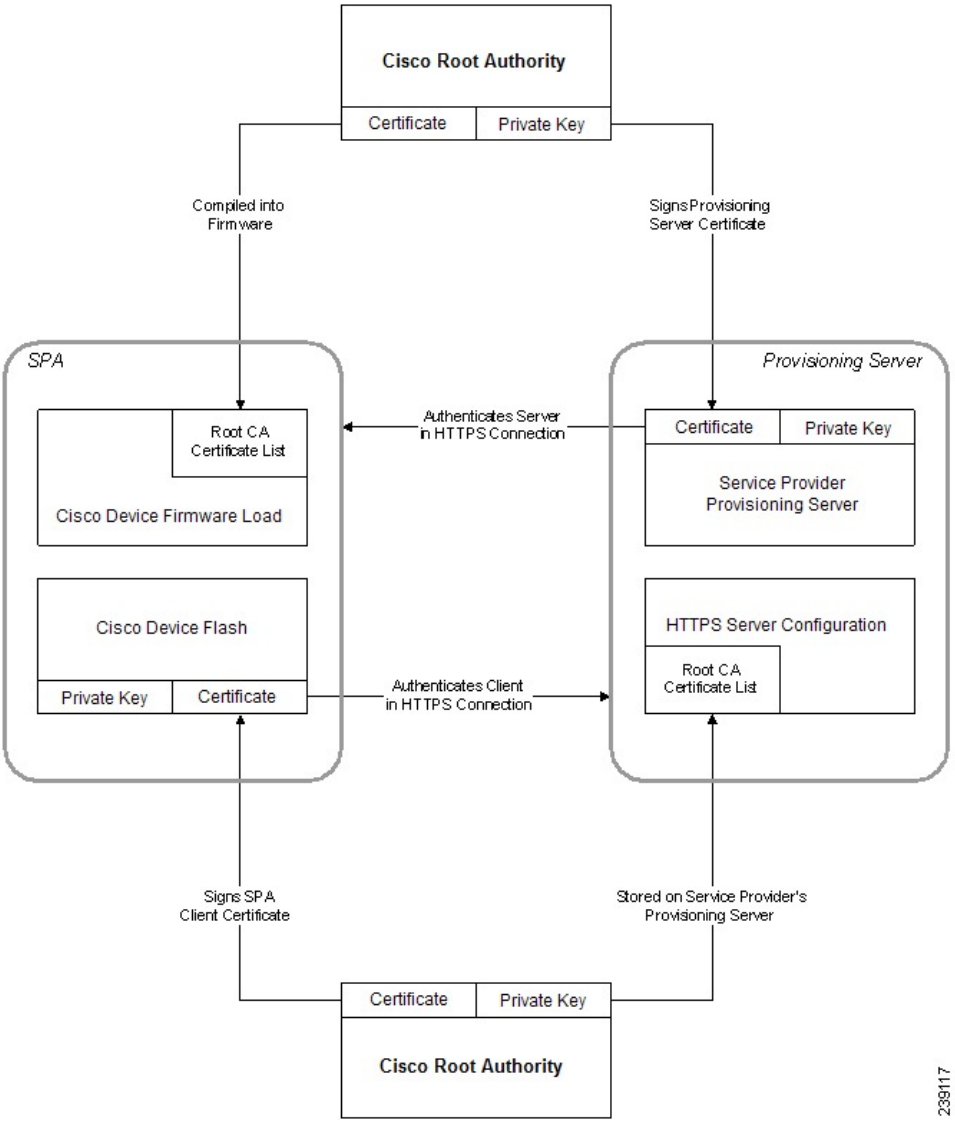
In addition to a direct attack on a Cisco IP Phone, an attacker might attempt to contact a provisioning server through a standard web browser or another HTTPS client to obtain the configuration profile from the provisioning server. To prevent this kind of attack, each Cisco IP Phone also carries a unique client certificate, signed by Cisco, that includes identifying information about each individual endpoint. A certificate authority root certificate that is capable of authenticating the device client certificate is given to each service provider. This authentication path allows the provisioning server to reject unauthorized requests for configuration profiles.

Certificate Structure

The combination of a server certificate and a client certificate ensures secure communication between a remote Cisco IP Phone and its provisioning server. Figure 4-1 illustrates the relationship and placement of certificates, public/private key pairs, and signing root authorities, among the Cisco client, the provisioning server, and the certification authority.

The upper half of the diagram shows the Provisioning Server Root Authority that is used to sign the individual provisioning server certificate. The corresponding root certificate is compiled into the firmware, which allows the Cisco IP Phone to authenticate authorized provisioning servers.

Figure 1: Certificate Authority Flow



Configure a Custom Certificate Authority

Digital certificates can be used to authenticate network devices and users on the network. They can be used to negotiate IPSec sessions between network nodes.

A third party uses a Certificate Authority certificate to validate and authenticate two or more nodes that are attempting to communicate. Each node has a public and private key. The public key encrypts data. The private key decrypts data. Because the nodes have obtained their certificates from the same source, they are assured of their respective identities.

The device can use digital certificates provided by a third-party Certificate Authority (CA) to authenticate IPSec connections.

The phones support a set of preloaded Root Certificate Authority embedded in the firmware:

- Cisco Small Business CA Certificate
- CyberTrust CA Certificate
- Verisign CA certificate
- Sipura Root CA Certificate
- Linksys Root CA Certificate

Procedure

Step 1 On the Configuration Utility page, select **Admin Login** > **advanced** > **Info** > **Status**.

Step 2 Scroll to **Custom CA Status** and see the following fields:

- Custom CA Provisioning Status—Indicates the provisioning status.
  - Last provisioning succeeded on mm/dd/yyyy HH:MM:SS; or
  - Last provisioning failed on mm/dd/yyyy HH:MM:SS
- Custom CA Info—Displays information about the custom CA.
  - Installed—Displays the "CN Value," where "CN Value" is the value of the CN parameter for the Subject field in the first certificate.
  - Not Installed—Displays if no custom CA certificate is installed.

Profile Management

This section demonstrates the formation of configuration profiles in preparation for downloading. To explain the functionality, TFTP from a local PC is used as the resync method, although HTTP or HTTPS can be used as well.

Open Profile gzip Compression

A configuration profile in XML format can become quite large if the profile specifies all parameters individually. To reduce the load on the provisioning server, the Cisco IP Phone supports compression of the XML file, by using the deflate compression format that the gzip utility (RFC 1951) supports.

Note Compression must precede encryption for the Cisco IP Phone to recognize a compressed and encrypted XML profile.

For integration into customized back-end provisioning server solutions, the open source zlib compression library can be used in place of the standalone gzip utility to perform the profile compression. However, the Cisco IP Phone expects the file to contain a valid gzip header.

Procedure

Step 1 Install gzip on the local PC.

Step 2 Compress the `basic.txt` configuration profile (described in the TFTP Resync, on page 41 section) by invoking gzip from the command line:

```
gzip basic.txt
```

This generates the deflated file `basic.txt.gz`.

Step 3 Save the `basic.txt.gz` file in the TFTP server virtual root directory.

Step 4 Modify the Profile_Rule on the test device to resync to the deflated file in place of the original XML file, as shown in the following example:

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

Step 5 Click **Submit All Changes.**

Step 6 Observe the syslog trace from the Cisco IP Phone.
Upon resync, the Cisco IP Phone downloads the new file and uses it to update its parameters.

Related Topics

Open Profile Compression, on page 13

Profile Encryption by Using OpenSSL

A compressed or uncompressed profile can be encrypted (however, a file must be compressed before it is encrypted). Encryption is useful when the confidentiality of the profile information is of particular concern, such as when TFTP or HTTP is used for communication between the Cisco IP Phone and the provisioning server.

The Cisco IP Phone supports symmetric key encryption by using the 256-bit AES algorithm. This encryption can be performed by using the open source OpenSSL package.

Procedure

Step 1 Install OpenSSL on a local PC. This might require that the OpenSSL application be recompiled to enable AES.

Step 2 Using the `basic.txt` configuration file (described in the TFTP Resync, on page 41 section), generate an encrypted file with the following command:

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

The compressed `basic.txt.gz` file that was created in Open Profile gzip Compression, on page 54 also can be used, because the XML profile can be both compressed and encrypted.

Step 3 Store the encrypted `basic.cfg` file in the TFTP server virtual root directory.

Step 4 Modify the Profile_Rule on the test device to resync to the encrypted file in place of the original XML file. The encryption key is made known to the Cisco IP Phone with the following URL option:

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

Step 5 Click **Submit All Changes**.

Step 6 Observe the syslog trace from the Cisco IP Phone.
Upon resync, the Cisco IP Phone downloads the new file and uses it to update its parameters.

Related Topics

Open Profile Encryption by Using AES, on page 14

Partitioned Profiles

A Cisco IP Phone downloads multiple separate profiles during each resync. This practice allows management of different kinds of profile information on separate servers and maintenance of common configuration parameter values that are separate from account specific values.

Procedure

Step 1 Create a new XML profile, `basic2.txt`, that specifies a value for a parameter that makes it distinct from the earlier exercises. For instance, to the `basic.txt` profile, add the following:

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

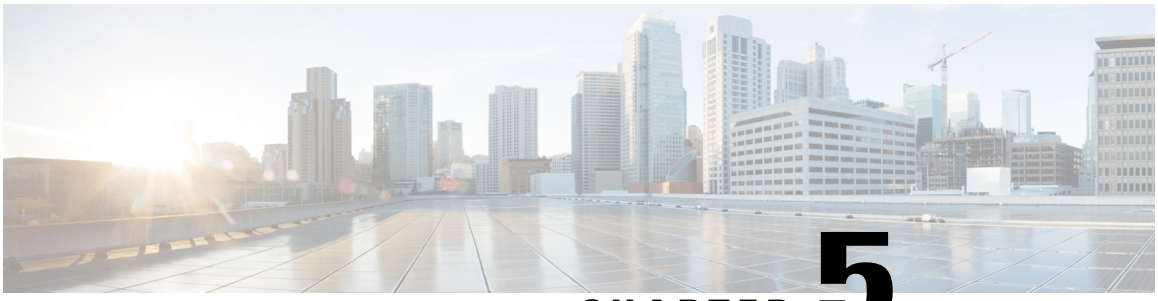
Step 2 Store the `basic2.txt` profile in the virtual root directory of the TFTP server.

Step 3 Leave the first profile rule from the earlier exercises in the folder, but configure the second profile rule (Profile_Rule_B) to point to the new file:

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

Step 4 Click **Submit All Changes**.
The Cisco IP Phone now resyncs to both the first and second profiles, in that order, whenever a resync operation is due.

Step 5 Observe the syslog trace to confirm the expected behavior.



CHAPTER 5

Provisioning Parameters

Provisioning Parameters Overview

This chapter describes the provisioning parameters that can be used in configuration profile scripts.

Configuration Profile Parameters

The following table defines the function and usage of each parameter in the **Configuration Profile Parameters** section under the **Provisioning** tab.

Parameter Name	Description and Default Value
Provision Enable	Controls all resync actions independently of firmware upgrade actions. Set to **Yes** to enable remote provisioning. The default value is Yes.
Resync On Reset	Triggers a resync after every reboot except for reboots caused by parameter updates and firmware upgrades. The default value is Yes.

Parameter Name	Description and Default Value
Resync Random Delay	A random delay following the boot-up sequence before performing the reset, specified in seconds. In a pool of IP Telephony devices that are scheduled to simultaneously power up, this introduces a spread in the times at which each unit sends a resync request to the provisioning server. This feature can be useful in a large residential deployment, in the case of a regional power failure. The default value is 2.
Resync At (HHmm)	The hour and minutes (HHmm) that the device resynchronizes with the provisioning server. The default value is empty. If the value is invalid, the parameter is ignored. If this parameter is set with a valid value, the Resync Periodic parameter is ignored.
Resync At Random Delay	Prevents an overload of the provisioning server when a large number of devices power-on simultaneously. To avoid flooding resync requests to the server from multiple phones, the phone resynchronizes in the range between the hours and minutes, and the hours and minutes plus the random delay (hhmm, hhmm+random_delay). For example, if the random delay = (Resync At Random Delay + 30)/60 minutes. The input value in seconds is converted to minutes, rounding up to the next minute to calculate the final random_delay interval. This feature is disabled when this parameter is set to zero. The default value is 600 seconds (10 minutes). If the parameter value is set to less than 600, the default value is used.
Resync Periodic	The time interval between periodic resynchronizes with the provisioning server. The associated resync timer is active only after the first successful sync with the server. Set this parameter to zero to disable periodic resynchronization. The default value is 3600 seconds.
Resync Error Retry Delay	If a resync operation fails because the IP Telephony device was unable to retrieve a profile from the server, or the downloaded file is corrupt, or an internal error occurs, the device tries to resync again after a time specified in seconds. If the delay is set to 0, the device does not try to resync again following a failed resync attempt.

Parameter Name	Description and Default Value
Forced Resync Delay	Maximum delay (in seconds) the Cisco IP Phone waits before performing a resynchronization. The device does not resync while one of its phone lines is active. Because a resync can take several seconds, it is desirable to wait until the device has been idle for an extended period before resynchronizing. This allows a user to make calls in succession without interruption. The device has a timer that begins counting down when all of its lines become idle. This parameter is the initial value of the counter. Resync events are delayed until this counter decrements to zero. The default value is 14,400 seconds.
Resync From SIP	Enables a resync to be triggered via a SIP NOTIFY message. The default value is Yes.
Resync After Upgrade Attempt	Enables or disables the resync operation after any upgrade occurs. If Yes is selected, sync is triggered. The default value is Yes.
Resync Trigger 1, Resync Trigger 2	Configurable resync trigger conditions. A resync is triggered when the logic equation in these parameters evaluates to TRUE. The default value is (empty).
Resync Fails On FNF	A resync is considered unsuccessful if a requested profile is not received from the server. This can be overridden by this parameter. When it is set to **no**, the device accepts a `file-not-found` response from the server as a successful resync. The default value is Yes.
Profile Rule Profile Rule B Profile Rule C Profile Rule D	Remote configuration profile rules evaluated in sequence. Each resync operation can retrieve multiple files, potentially managed by different servers. The default value is /$PSN.xml.
DHCP Option To Use	DHCP options, delimited by commas, used to retrieve firmware and profiles. The default value is 66,160,159,150,60,43,125.
Log Request Msg	This parameter contains the message that is sent to the syslog server at the start of a resync attempt. The default value is `$PN $MAC –Requesting % $SCHEME://$SERVIP:$PORT$PATH`.

Parameter Name	Description and Default Value
Log Success Msg	The syslog message that is issued upon successful completion of a resync attempt. The default value is `$PN $MAC –Successful Resync % $SCHEME://$SERVIP:$PORT$PATH -- $ERR`.
Log Failure Msg	The syslog message that is issued after a failed resync attempt. The default value is `$PN $MAC -- Resync failed: $ERR`.
User Configurable Resync	Allows a user to resynch the phone from the IP phone screen. The default value is Yes.

Firmware Upgrade Parameters

The following table defines the function and usage of each parameter in the **Firmware Upgrade** section of the **Provisioning** tab.

Parameter Name	Description and Default Value
Upgrade Enable	Enables firmware upgrade operations independently of resync actions. The default value is Yes.
Upgrade Error Retry Delay	The upgrade retry interval (in seconds) applied in case of upgrade failure. The device has a firmware upgrade error timer that activates after a failed firmware upgrade attempt. The timer is initialized with the value in this parameter. The next firmware upgrade attempt occurs when this timer counts down to zero. The default value is 3600 seconds.
Upgrade Rule	A firmware upgrade script that defines upgrade conditions and associated firmware URLs. It uses the same syntax as Profile Rule. Use the following format to enter the upgrade rule: For example: tftp://192.168.1.5/image/sip7832.11-0-1MPP-BN.loads If no protocol is specified, TFTP is assumed. If no server-name is specified, the host that requests the URL is used as the server name. If no port is specified, the default port is used (69 for TFTP, 80 for HTTP, or 443 for HTTPS). The default value is blank.
Log Upgrade Request Msg	Syslog message issued at the start of a firmware upgrade attempt. Default: `$PN $MAC -- Requesting upgrade $SCHEME://$SERVIP:$PORT$PATH`

Parameter Name	Description and Default Value
Log Upgrade Success Msg	Syslog message issued after a firmware upgrade attempt completes successfully. The default value is `$PN $MAC -- Successful upgrade $SCHEME://$SERVIP:$PORT$PATH -- $ERR`
Log Upgrade Failure Msg	Syslog message issued after a failed firmware upgrade attempt. The default value is `$PN $MAC -- Upgrade failed: $ERR`

General Purpose Parameters

The following table defines the function and usage of each parameter in the **General Purpose Parameters** section of the **Provisioning** tab.

Parameter Name	Description and Default Value
GPP A - GPP P	The general purpose parameters GPP_* are used as free string registers when configuring the Cisco IP phones to interact with a particular provisioning server solution. They can be configured to contain diverse values, including the following: • Encryption keys. • URLs. • Multistage provisioning status information. • Post request templates. • Parameter name alias maps. • Partial string values, eventually combined into complete parameter values. The default value is blank.

Macro Expansion Variables

Certain macro variables are recognized within the following provisioning parameters:

- Profile_Rule
- Profile_Rule_*
- Resync_Trigger_*
- Upgrade_Rule
- Log_*

- GPP_* (under specific conditions)

Within these parameters, syntax types, such as $NAME or $(NAME), are recognized and expanded.

Macro variable substrings can be specified with the notation $(NAME:p) and $(NAME:p:q), where p and q are non-negative integers (available in revision 2.0.11 and above). The resulting macro expansion is the substring starting at character offset p, with length q (or else till end-of-string if q is not specified). For example, if GPP_A contains ABCDEF, then $(A:2) expands to CDEF, and $(A:2:3) expands to CDE.

An unrecognized name is not translated, and the $NAME or $(NAME) form remains unchanged in the parameter value after expansion.

Parameter Name	Description and Default Value
$	The form $$ expands to a single $ character.
A through P	Replaced by the contents of the general purpose parameters GPP_A through GPP_P.
SA through SD	Replaced by special purpose parameters GPP_SA through GPP_SD. These parameters hold keys or passwords used in provisioning. **Note** $SA through $SD are recognized as arguments to the optional resync URL qualifier, --key.
MA	MAC address using lower case hex digits, for example, 000e08aabbcc.
MAU	MAC address using upper case hex digits, for example 000E08AABBCC.
MAC	MAC address using lower case hex digits, and colons to separate hex digit pairs, for example 00:0e:08:aa:bb:cc.
PN	Product Name. For example, CP-7832-3PCC.
PSN	Product Series Number. For example, V03.
SN	Serial Number string, for example 88012BA01234.
CCERT	SSL Client Certificate status: Installed or Not Installed.
IP	IP address of the Cisco IP Phone within its local subnet, for example 192.168.1.100.
EXTIP	External IP of the Cisco IP Phone, as seen on the Internet, for example 66.43.16.52.
SWVER	Software version string. For example, sip78xx.11-0-1MPP.
HWVER	Hardware version string, for example 2.0.1

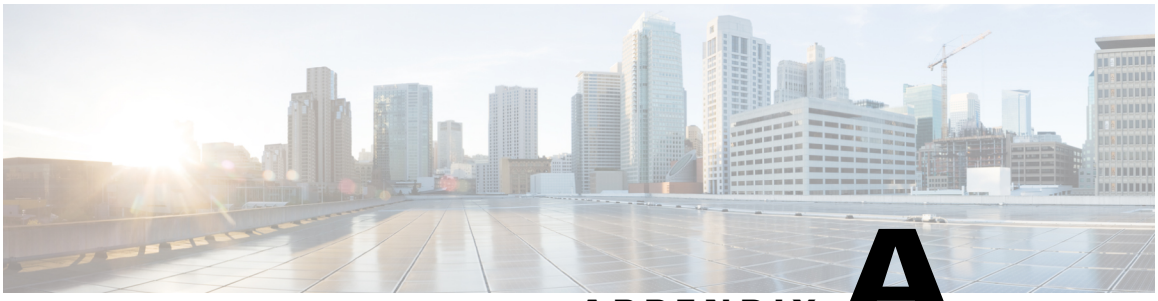
Parameter Name	Description and Default Value
PRVST	Provisioning State (a numeric string): -1 = explicit resync request 0 = power-up resync 1 = periodic resync 2 = resync failed, retry attempt
UPGST	Upgrade State (a numeric string): 1 = first upgrade attempt 2 = upgrade failed, retry attempt
UPGERR	Result message (ERR) of previous upgrade attempt; for example http_get failed.
PRVTMR	Seconds since last resync attempt.
UPGTMR	Seconds since last upgrade attempt.
REGTMR1	Seconds since Line 1 lost registration with SIP server.
REGTMR2	Seconds since Line 2 lost registration with SIP server.
UPGCOND	Legacy macro name.
SCHEME	File access scheme, one of TFTP, HTTP, or HTTPS, as obtained after parsing resync or upgrade URL.
SERV	Request target server host name, as obtained after parsing resync or upgrade URL.
SERVIP	Request target server IP address, as obtained after parsing resync or upgrade URL, possibly following DNS lookup.
PORT	Request target UDP/TCP port, as obtained after parsing resync or upgrade URL.
PATH	Request target file path, as obtained after parsing resync or upgrade URL.
ERR	Result message of resync or upgrade attempt. Only useful in generating result syslog messages. The value is preserved in the UPGERR variable in the case of upgrade attempts.
UIDn	The contents of the Line n UserID configuration parameter.
EMS	Extension Mobility Status

Parameter Name	Description and Default Value
MUID	Extension Mobility User ID
MPWD	Extension Mobility Password

Internal Error Codes

The Cisco IP Phone defines a number of internal error codes (X00–X99) to facilitate configuration in providing finer control over the behavior of the unit under certain error conditions.

Parameter Name	Description and Default Value
X00	Transport layer (or ICMP) error when sending a SIP request.
X20	SIP request times out while waiting for a response.
X40	General SIP protocol error (for example, unacceptable codec in SDP in 200 and ACK messages, or times out while waiting for ACK).
X60	Dialed number invalid according to given dial plan.



APPENDIX A

Sample Configuration Profiles

XML Open Format Sample

```
<?xml version="1.0" encoding="UTF-8"?>
<flat-profile>
<!-- System Configuration -->
<Restricted_Access_Domains ua="na"/>
<Enable_Web_Server ua="na">Yes</Enable_Web_Server>
<Enable_Protocol ua="na">Http</Enable_Protocol>
<Enable_Direct_Action_Url ua="na">Yes</Enable_Direct_Action_Url>
<Session_Max_Timeout ua="na">3600</Session_Max_Timeout>
<Session_Idle_Timeout ua="na">3600</Session_Idle_Timeout>
<Web_Server_Port ua="na">80</Web_Server_Port>
<Enable_Web_Admin_Access ua="na">Yes</Enable_Web_Admin_Access>
<!-- <Admin_Password ua="na"/> -->
<!-- <User_Password ua="rw"/> -->
<Phone-UI-readonly ua="na">No</Phone-UI-readonly>
<Phone-UI-User-Mode ua="na">No</Phone-UI-User-Mode>
<!-- Power Settings -->
<!-- Network Settings -->
<IP_Mode ua="na">IPv4 Only</IP_Mode>
<!-- IPv4 Settings  -->
<Connection_Type ua="rw">DHCP</Connection_Type>
<Static_IP ua="rw"/>
<NetMask ua="rw"/>
<Gateway ua="rw"/>
<Primary_DNS ua="rw">64.101.155.135</Primary_DNS>
<Secondary_DNS ua="rw"/>
<!-- IPv6 Settings -->
<!-- 802.1X Authentication -->
<Enable_802.1X_Authentication ua="rw">No</Enable_802.1X_Authentication>
<!-- Optional Network Configuration -->
<Host_Name ua="rw"/>
<Domain ua="rw"/>
<DNS_Server_Order ua="na">Manual,DHCP</DNS_Server_Order>
<DNS_Query_Mode ua="na">Parallel</DNS_Query_Mode>
<DNS_Caching_Enable ua="na">Yes</DNS_Caching_Enable>
<Switch_Port_Config ua="na">AUTO</Switch_Port_Config>
<Syslog_Server ua="na"/>
<Debug_Level ua="na">NOTICE</Debug_Level>
<Primary_NTP_Server ua="rw"/>
<Secondary_NTP_Server ua="rw"/>
<Enable_SSLv3 ua="na">No</Enable_SSLv3>
<!-- VLAN Settings -->
<Enable_VLAN ua="rw">No</Enable_VLAN>
<VLAN_ID ua="rw">1</VLAN_ID>
```

```
<Enable_CDP ua="na">Yes</Enable_CDP>
<Enable_LLDP-MED ua="na">Yes</Enable_LLDP-MED>
<Network_Startup_Delay ua="na">3</Network_Startup_Delay>
<!-- Inventory Settings -->
<Asset_ID ua="na"/>
<!-- SIP Parameters -->
<Max_Forward ua="na">70</Max_Forward>
<Max_Redirection ua="na">5</Max_Redirection>
<Max_Auth ua="na">2</Max_Auth>
<SIP_User_Agent_Name ua="na">$VERSION</SIP_User_Agent_Name>
<SIP_Server_Name ua="na">$VERSION</SIP_Server_Name>
<SIP_Reg_User_Agent_Name ua="na"/>
<SIP_Accept_Language ua="na"/>
<DTMF_Relay_MIME_Type ua="na">application/dtmf-relay</DTMF_Relay_MIME_Type>
<Hook_Flash_MIME_Type ua="na">application/hook-flash</Hook_Flash_MIME_Type>
<Remove_Last_Reg ua="na">No</Remove_Last_Reg>
<Use_Compact_Header ua="na">No</Use_Compact_Header>
<Escape_Display_Name ua="na">No</Escape_Display_Name>
<Talk_Package ua="na">No</Talk_Package>
<Hold_Package ua="na">No</Hold_Package>
<Conference_Package ua="na">No</Conference_Package>
<RFC_2543_Call_Hold ua="na">Yes</RFC_2543_Call_Hold>
<Random_REG_CID_on_Reboot ua="na">No</Random_REG_CID_on_Reboot>
<SIP_TCP_Port_Min ua="na">5060</SIP_TCP_Port_Min>
<SIP_TCP_Port_Max ua="na">5080</SIP_TCP_Port_Max>
<Caller_ID_Header ua="na">PAID-RPID-FROM</Caller_ID_Header>
<Hold_Target_Before_Refer ua="na">No</Hold_Target_Before_Refer>
<Dialog_SDP_Enable ua="na">No</Dialog_SDP_Enable>
<Keep_Referee_When_Refer_Failed ua="na">No</Keep_Referee_When_Refer_Failed>
<Display_Diversion_Info ua="na">No</Display_Diversion_Info>
<Display_Anonymous_From_Header ua="na">No</Display_Anonymous_From_Header>
<Sip_Accept_Encoding ua="na">none</Sip_Accept_Encoding>
<Disable_Local_Name_To_Header ua="na">No</Disable_Local_Name_To_Header>
<!-- SIP Timer Values (sec) -->
<SIP_T1 ua="na">.5</SIP_T1>
<SIP_T2 ua="na">4</SIP_T2>
<SIP_T4 ua="na">5</SIP_T4>
<SIP_Timer_B ua="na">16</SIP_Timer_B>
<SIP_Timer_F ua="na">16</SIP_Timer_F>
<SIP_Timer_H ua="na">16</SIP_Timer_H>
<SIP_Timer_D ua="na">16</SIP_Timer_D>
<SIP_Timer_J ua="na">16</SIP_Timer_J>
<INVITE_Expires ua="na">240</INVITE_Expires>
<ReINVITE_Expires ua="na">30</ReINVITE_Expires>
<Reg_Min_Expires ua="na">1</Reg_Min_Expires>
<Reg_Max_Expires ua="na">7200</Reg_Max_Expires>
<Reg_Retry_Intvl ua="na">30</Reg_Retry_Intvl>
<Reg_Retry_Long_Intvl ua="na">1200</Reg_Retry_Long_Intvl>
<Reg_Retry_Random_Delay ua="na">0</Reg_Retry_Random_Delay>
<Reg_Retry_Long_Random_Delay ua="na">0</Reg_Retry_Long_Random_Delay>
<Reg_Retry_Intvl_Cap ua="na">0</Reg_Retry_Intvl_Cap>
<Sub_Min_Expires ua="na">10</Sub_Min_Expires>
<Sub_Max_Expires ua="na">7200</Sub_Max_Expires>
<Sub_Retry_Intvl ua="na">10</Sub_Retry_Intvl>
<!-- Response Status Code Handling -->
<Try_Backup_RSC ua="na"/>
<Retry_Reg_RSC ua="na"/>
<!-- RTP Parameters -->
<RTP_Port_Min ua="na">16384</RTP_Port_Min>
<RTP_Port_Max ua="na">16482</RTP_Port_Max>
<RTP_Packet_Size ua="na">0.030</RTP_Packet_Size>
<Max_RTP_ICMP_Err ua="na">0</Max_RTP_ICMP_Err>
<RTCP_Tx_Interval ua="na">0</RTCP_Tx_Interval>
<!-- SDP Payload Types -->
<G722.2_Dynamic_Payload ua="na">96</G722.2_Dynamic_Payload>
<iLBC_Dynamic_Payload ua="na">97</iLBC_Dynamic_Payload>
<OPUS_Dynamic_Payload ua="na">99</OPUS_Dynamic_Payload>
<AVT_Dynamic_Payload ua="na">101</AVT_Dynamic_Payload>
<INFOREQ_Dynamic_Payload ua="na"/>
<G711u_Codec_Name ua="na">PCMU</G711u_Codec_Name>
<G711a_Codec_Name ua="na">PCMA</G711a_Codec_Name>
<G729a_Codec_Name ua="na">G729a</G729a_Codec_Name>
<G729b_Codec_Name ua="na">G729ab</G729b_Codec_Name>
```

```
<G722_Codec_Name ua="na">G722</G722_Codec_Name>
<G722.2_Codec_Name ua="na">AMR-WB</G722.2_Codec_Name>
<iLBC_Codec_Name ua="na">iLBC</iLBC_Codec_Name>
<OPUS_Codec_Name ua="na">OPUS</OPUS_Codec_Name>
<AVT_Codec_Name ua="na">telephone-event</AVT_Codec_Name>
<!-- NAT Support Parameters -->
<Handle_VIA_received ua="na">No</Handle_VIA_received>
<Handle_VIA_rport ua="na">No</Handle_VIA_rport>
<Insert_VIA_received ua="na">No</Insert_VIA_received>
<Insert_VIA_rport ua="na">No</Insert_VIA_rport>
<Substitute_VIA_Addr ua="na">No</Substitute_VIA_Addr>
<Send_Resp_To_Src_Port ua="na">No</Send_Resp_To_Src_Port>
<STUN_Enable ua="na">No</STUN_Enable>
<STUN_Test_Enable ua="na">No</STUN_Test_Enable>
<STUN_Server ua="na"/>
<EXT_IP ua="na"/>
<EXT_RTP_Port_Min ua="na">0</EXT_RTP_Port_Min>
<NAT_Keep_Alive_Intvl ua="na">15</NAT_Keep_Alive_Intvl>
<Redirect_Keep_Alive ua="na">No</Redirect_Keep_Alive>
<!-- Configuration Profile -->
<Provision_Enable ua="na">Yes</Provision_Enable>
<Resync_On_Reset ua="na">Yes</Resync_On_Reset>
<Resync_Random_Delay ua="na">2</Resync_Random_Delay>
<Resync_At__HHmm_ ua="na"/>
<Resync_At_Random_Delay ua="na">600</Resync_At_Random_Delay>
<Resync_Periodic ua="na">7200</Resync_Periodic>
<Resync_Error_Retry_Delay ua="na">30</Resync_Error_Retry_Delay>
<Forced_Resync_Delay ua="na">14400</Forced_Resync_Delay>
<Resync_From_SIP ua="na">Yes</Resync_From_SIP>
<Resync_After_Upgrade_Attempt ua="na">Yes</Resync_After_Upgrade_Attempt>
<Resync_Trigger_1 ua="na"/>
<Resync_Trigger_2 ua="na"/>
<Resync_Fails_On_FNF ua="na">Yes</Resync_Fails_On_FNF>
<Profile_Rule ua="na">https://edos1.sandbox.cisco.com/cisco/RC_$MAU.xml</Profile_Rule>
<Profile_Rule_B ua="na"/>
<Profile_Rule_C ua="na"/>
<Profile_Rule_D ua="na"/>
<DHCP_Option_To_Use ua="na">66,160,159,150,60,43,125</DHCP_Option_To_Use>
<Log_Request_Msg ua="na">$PN $MAC -- Requesting resync
$SCHEME://$SERVIP:$PORT$PATH</Log_Request_Msg>
<Log_Success_Msg ua="na">$PN $MAC -- Successful resync
$SCHEME://$SERVIP:$PORT$PATH</Log_Success_Msg>
<Log_Failure_Msg ua="na">$PN $MAC -- Resync failed: $ERR</Log_Failure_Msg>
<User_Configurable_Resync ua="na">Yes</User_Configurable_Resync>
<!-- Firmware Upgrade -->
<Upgrade_Enable ua="na">Yes</Upgrade_Enable>
<Upgrade_Error_Retry_Delay ua="na">3600</Upgrade_Error_Retry_Delay>
<Upgrade_Rule ua="na"/>
<Log_Upgrade_Request_Msg ua="na">$PN $MAC -- Requesting upgrade
$SCHEME://$SERVIP:$PORT$PATH</Log_Upgrade_Request_Msg>
<Log_Upgrade_Success_Msg ua="na">$PN $MAC -- Successful upgrade $SCHEME://$SERVIP:$PORT$PATH
 -- $ERR</Log_Upgrade_Success_Msg>
<Log_Upgrade_Failure_Msg ua="na">$PN $MAC -- Upgrade failed: $ERR</Log_Upgrade_Failure_Msg>
<!-- CA Settings -->
<Custom_CA_Rule ua="na"/>
<!-- HTTP Settings -->
<HTTP_User_Agent_Name ua="na">$VERSION ($MA)</HTTP_User_Agent_Name>
<!-- Problem Report Tool -->
<PRT_Upload_Rule ua="na"/>
<PRT_Upload_Method ua="na">POST</PRT_Upload_Method>
<!-- General Purpose Parameters -->
<GPP_A ua="na"/>
<GPP_B ua="na"/>
<GPP_C ua="na"/>
<GPP_D ua="na"/>
<GPP_E ua="na"/>
<GPP_F ua="na"/>
<GPP_G ua="na"/>
<GPP_H ua="na"/>
<GPP_I ua="na"/>
<GPP_J ua="na"/>
<GPP_K ua="na"/>
<GPP_L ua="na"/>
```

```
<GPP_M ua="na"/>
<GPP_N ua="na"/>
<GPP_O ua="na"/>
<GPP_P ua="na"/>
<!-- Call Progress Tones -->
<Dial_Tone ua="na">350@-19,440@-19;10(*/0/1+2)</Dial_Tone>
<Outside_Dial_Tone ua="na">420@-16;10(*/0/1)</Outside_Dial_Tone>
<Prompt_Tone ua="na">520@-19,620@-19;10(*/0/1+2)</Prompt_Tone>
<Busy_Tone ua="na">480@-19,620@-19;10(.5/.5/1+2)</Busy_Tone>
<Reorder_Tone ua="na">480@-19,620@-19;10(.25/.25/1+2)</Reorder_Tone>
<Off_Hook_Warning_Tone ua="na">480@-10,620@0;10(.125/.125/1+2)</Off_Hook_Warning_Tone>
<Ring_Back_Tone ua="na">440@-19,480@-19;*(2/4/1+2)</Ring_Back_Tone>
<Call_Waiting_Tone ua="na">440@-10;30(.3/9.7/1)</Call_Waiting_Tone>
<Confirm_Tone ua="na">600@-16;1(.25/.25/1)</Confirm_Tone>
<MWI_Dial_Tone ua="na">350@-19,440@-19;2(.1/.1/1+2);10(*/0/1+2)</MWI_Dial_Tone>
<Cfwd_Dial_Tone ua="na">350@-19,440@-19;2(.2/.2/1+2);10(*/0/1+2)</Cfwd_Dial_Tone>
<Holding_Tone ua="na">600@-19;25(.1/.1/1,.1/.1/1,.1/9.5/1)</Holding_Tone>
<Conference_Tone ua="na">350@-19;20(.1/.1/1,.1/9.7/1)</Conference_Tone>
<Secure_Call_Indication_Tone
ua="na">397@-19,507@-19;15(0/2/0,.2/.1/1,.1/2.1/2)</Secure_Call_Indication_Tone>
<Page_Tone ua="na">600@-16;.3(.05/0.05/1)</Page_Tone>
<Alert_Tone ua="na">600@-19;.2(.05/0.05/1)</Alert_Tone>
<Mute_Tone ua="na">600@-19;.2(.1/0.1/1)</Mute_Tone>
<Unmute_Tone ua="na">600@-19;.3(.1/0.1/1)</Unmute_Tone>
<System_Beep ua="na">600@-16;.1(.05/0.05/1)</System_Beep>
<Call_Pickup_Tone ua="na">440@-10;30(.3/9.7/1)</Call_Pickup_Tone>
<!-- Distinctive Ring Patterns -->
<Cadence_1 ua="na">60(2/4)</Cadence_1>
<Cadence_2 ua="na">60(.3/.2,1/.2,.3/4)</Cadence_2>
<Cadence_3 ua="na">60(.8/.4,.8/4)</Cadence_3>
<Cadence_4 ua="na">60(.4/.2,.3/.2,.8/4)</Cadence_4>
<Cadence_5 ua="na">60(.2/.2,.2/.2,.2/.2,1/4)</Cadence_5>
<Cadence_6 ua="na">60(.2/.4,.2/.4,.2/4)</Cadence_6>
<Cadence_7 ua="na">60(4.5/4)</Cadence_7>
<Cadence_8 ua="na">60(0.25/9.75)</Cadence_8>
<Cadence_9 ua="na">60(.4/.2,.4/2)</Cadence_9>
<!-- Control Timer Values (sec) -->
<Reorder_Delay ua="na">255</Reorder_Delay>
<Interdigit_Long_Timer ua="na">10</Interdigit_Long_Timer>
<Interdigit_Short_Timer ua="na">3</Interdigit_Short_Timer>
<!-- Vertical Service Activation Codes -->
<Call_Return_Code ua="na">*69</Call_Return_Code>
<Blind_Transfer_Code ua="na">*95</Blind_Transfer_Code>
<Cfwd_All_Act_Code ua="na">*72</Cfwd_All_Act_Code>
<Cfwd_All_Deact_Code ua="na">*73</Cfwd_All_Deact_Code>
<Cfwd_Busy_Act_Code ua="na">*90</Cfwd_Busy_Act_Code>
<Cfwd_Busy_Deact_Code ua="na">*91</Cfwd_Busy_Deact_Code>
<Cfwd_No_Ans_Act_Code ua="na">*92</Cfwd_No_Ans_Act_Code>
<Cfwd_No_Ans_Deact_Code ua="na">*93</Cfwd_No_Ans_Deact_Code>
<CW_Act_Code ua="na">*56</CW_Act_Code>
<CW_Deact_Code ua="na">*57</CW_Deact_Code>
<CW_Per_Call_Act_Code ua="na">*71</CW_Per_Call_Act_Code>
<CW_Per_Call_Deact_Code ua="na">*70</CW_Per_Call_Deact_Code>
<Block_CID_Act_Code ua="na">*61</Block_CID_Act_Code>
<Block_CID_Deact_Code ua="na">*62</Block_CID_Deact_Code>
<Block_CID_Per_Call_Act_Code ua="na">*81</Block_CID_Per_Call_Act_Code>
<Block_CID_Per_Call_Deact_Code ua="na">*82</Block_CID_Per_Call_Deact_Code>
<Block_ANC_Act_Code ua="na">*77</Block_ANC_Act_Code>
<Block_ANC_Deact_Code ua="na">*87</Block_ANC_Deact_Code>
<DND_Act_Code ua="na">*78</DND_Act_Code>
<DND_Deact_Code ua="na">*79</DND_Deact_Code>
<Secure_All_Call_Act_Code ua="na">*16</Secure_All_Call_Act_Code>
<Secure_No_Call_Act_Code ua="na">*17</Secure_No_Call_Act_Code>
<Secure_One_Call_Act_Code ua="na">*18</Secure_One_Call_Act_Code>
<Secure_One_Call_Deact_Code ua="na">*19</Secure_One_Call_Deact_Code>
<Paging_Code ua="na">*96</Paging_Code>
<Call_Park_Code ua="na">*68</Call_Park_Code>
<Call_Pickup_Code ua="na">*97</Call_Pickup_Code>
<Call_Unpark_Code ua="na">*88</Call_Unpark_Code>
<Group_Call_Pickup_Code ua="na">*98</Group_Call_Pickup_Code>
<Referral_Services_Codes ua="na"/>
<Feature_Dial_Services_Codes ua="na"/>
<!-- Vertical Service Announcement Codes -->
```

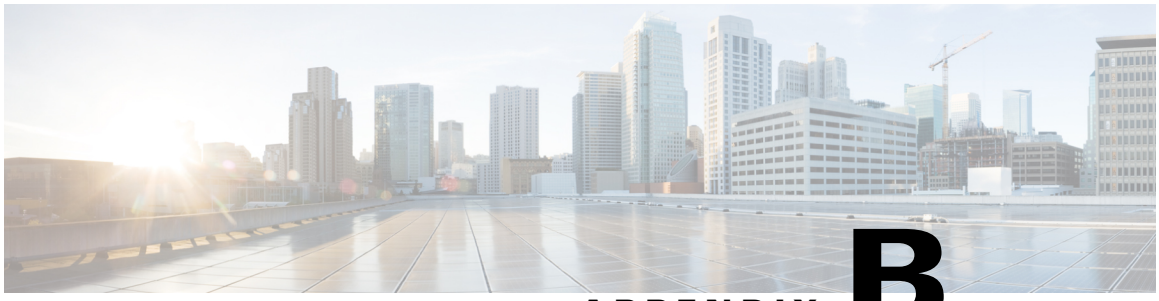
```
<Service_Annc_Base_Number ua="na"/>
<Service_Annc_Extension_Codes ua="na"/>
<!-- Outbound Call Codec Selection Codes -->
<Prefer_G711u_Code ua="na">*017110</Prefer_G711u_Code>
<Force_G711u_Code ua="na">*027110</Force_G711u_Code>
<Prefer_G711a_Code ua="na">*017111</Prefer_G711a_Code>
<Force_G711a_Code ua="na">*027111</Force_G711a_Code>
<Prefer_G722_Code ua="na">*01722</Prefer_G722_Code>
<Force_G722_Code ua="na">*02722</Force_G722_Code>
<Prefer_G722.2_Code ua="na">*01724</Prefer_G722.2_Code>
<Force_G722.2_Code ua="na">*02724</Force_G722.2_Code>
<Prefer_G729a_Code ua="na">*01729</Prefer_G729a_Code>
<Force_G729a_Code ua="na">*02729</Force_G729a_Code>
<Prefer_iLBC_Code ua="na">*01016</Prefer_iLBC_Code>
<Force_iLBC_Code ua="na">*02016</Force_iLBC_Code>
<Prefer_OPUS_Code ua="na">*01056</Prefer_OPUS_Code>
<Force_OPUS_Code ua="na">*02056</Force_OPUS_Code>
<!-- Time -->
<Set_Local_Date__mm_dd_yyyy_ ua="na"/>
<Set_Local_Time__HH_mm_ ua="na"/>
<Time_Zone ua="na">GMT-08:00</Time_Zone>
<Time_Offset__HH_mm_ ua="na"/>
<Ignore_DHCP_Time_Offset ua="na">Yes</Ignore_DHCP_Time_Offset>
<Daylight_Saving_Time_Rule
ua="na">start=3/-1/7/2;end=10/-1/7/2;save=1</Daylight_Saving_Time_Rule>
<Daylight_Saving_Time_Enable ua="na">Yes</Daylight_Saving_Time_Enable>
<!-- Language -->
<Dictionary_Server_Script ua="na"/>
<Language_Selection ua="na">English-US</Language_Selection>
<Locale ua="na">en-US</Locale>
<!-- General -->
<Station_Name ua="na"/>
<Station_Display_Name ua="na"/>
<Voice_Mail_Number ua="na"/>
<!-- Handsfree -->
<Bluetooth_Mode ua="na">Phone</Bluetooth_Mode>
<Line ua="na">5</Line>
<!-- Line Key 1 -->
<Extension_1_ ua="na">1</Extension_1_>
<Short_Name_1_ ua="na">$USER</Short_Name_1_>
<Share_Call_Appearance_1_ ua="na">private</Share_Call_Appearance_1_>
<Extended_Function_1_ ua="na"/>
<!-- Miscellaneous Line Key Settings -->
<Line_ID_Mapping ua="na">Horizontal First</Line_ID_Mapping>
<SCA_Barge-In-Enable ua="na">No</SCA_Barge-In-Enable>
<SCA_Sticky_Auto_Line_Seize ua="na">No</SCA_Sticky_Auto_Line_Seize>
<Call_Appearances_Per_Line ua="na">2</Call_Appearances_Per_Line>
<!-- Supplementary Services -->
<Conference_Serv ua="na">Yes</Conference_Serv>
<Attn_Transfer_Serv ua="na">Yes</Attn_Transfer_Serv>
<Blind_Transfer_Serv ua="na">Yes</Blind_Transfer_Serv>
<DND_Serv ua="na">Yes</DND_Serv>
<Block_ANC_Serv ua="na">Yes</Block_ANC_Serv>
<Block_CID_Serv ua="na">Yes</Block_CID_Serv>
<Secure_Call_Serv ua="na">Yes</Secure_Call_Serv>
<Cfwd_All_Serv ua="na">Yes</Cfwd_All_Serv>
<Cfwd_Busy_Serv ua="na">Yes</Cfwd_Busy_Serv>
<Cfwd_No_Ans_Serv ua="na">Yes</Cfwd_No_Ans_Serv>
<Paging_Serv ua="na">Yes</Paging_Serv>
<Call_Park_Serv ua="na">Yes</Call_Park_Serv>
<Call_Pick_Up_Serv ua="na">Yes</Call_Pick_Up_Serv>
<ACD_Login_Serv ua="na">No</ACD_Login_Serv>
<Group_Call_Pick_Up_Serv ua="na">Yes</Group_Call_Pick_Up_Serv>
<Service_Annc_Serv ua="na">No</Service_Annc_Serv>
<!-- Ringtone -->
<Ring1 ua="na">n=Sunrise;w=file://Sunrise.rwb;c=1</Ring1>
<Ring2 ua="na">n=Chirp 1;w=file://chirp1.raw;c=1</Ring2>
<Ring3 ua="na">n=Chirp 2;w=file://chirp2.raw;c=1</Ring3>
<Ring4 ua="na">n=Delight;w=file://Delight.rwb;c=1</Ring4>
<Ring5 ua="na">n=Evolve;w=file://Evolve.rwb;c=1</Ring5>
<Ring6 ua="na">n=Mellow;w=file://Mellow.rwb;c=1</Ring6>
<Ring7 ua="na">n=Mischief;w=file://Mischief.rwb;c=1</Ring7>
<Ring8 ua="na">n=Reflections;w=file://Reflections.rwb;c=1</Ring8>
```

```
<Ring9 ua="na">n=Ringer;w=file://Ringer.rwb;c=1</Ring9>
<Ring10 ua="na">n=Ascent;w=file://Ascent.rwb;c=1</Ring10>
<Ring11 ua="na">n=Are you there;w=file://AreYouThereF.raw;c=1</Ring11>
<Ring12 ua="na">n=Chime;w=file://Chime.raw;c=1</Ring12>
<Silent_Ring_Duration ua="na">60</Silent_Ring_Duration>
<!-- Extension Mobility -->
<EM_Enable ua="na">No</EM_Enable>
<EM_User_Domain ua="na"/>
<Inactivity_Timer_m_ ua="na">480</Inactivity_Timer_m_>
<Countdown_Timer_s_ ua="na">10</Countdown_Timer_s_>
<!-- Broadsoft Settings -->
<Directory_Enable ua="na">No</Directory_Enable>
<XSI_Host_Server ua="na"/>
<Directory_Name ua="na"/>
<Directory_Type ua="na">Enterprise</Directory_Type>
<Directory_User_ID ua="na"/>
<!-- <Directory_Password ua="na"/> -->
<!-- XML Service -->
<XML_Directory_Service_Name ua="na"/>
<XML_Directory_Service_URL ua="na"/>
<XML_Application_Service_Name ua="na"/>
<XML_Application_Service_URL ua="na"/>
<XML_User_Name ua="na"/>
<!-- <XML_Password ua="na"/> -->
<CISCO_XML_EXE_Enable ua="na">No</CISCO_XML_EXE_Enable>
<CISCO_XML_EXE_Auth_Mode ua="na">Local Credential</CISCO_XML_EXE_Auth_Mode>
<!-- Multiple Paging Group Parameters -->
<Group_Paging_Script
ua="na">pggrp=224.168.168.168:34560;name=All;num=800;listen=yes;</Group_Paging_Script>
<!-- LDAP -->
<LDAP_Dir_Enable ua="na">No</LDAP_Dir_Enable>
<LDAP_Corp_Dir_Name ua="na"/>
<LDAP_Server ua="na"/>
<LDAP_Search_Base ua="na"/>
<LDAP_Client_DN ua="na"/>
<LDAP_Username ua="na"/>
<!-- <LDAP_Password ua="na"/> -->
<LDAP_Auth_Method ua="na">None</LDAP_Auth_Method>
<LDAP_Last_Name_Filter ua="na"/>
<LDAP_First_Name_Filter ua="na"/>
<LDAP_Search_Item_3 ua="na"/>
<LDAP_Item_3_Filter ua="na"/>
<LDAP_Search_Item_4 ua="na"/>
<LDAP_Item_4_Filter ua="na"/>
<LDAP_Display_Attrs ua="na"/>
<LDAP_Number_Mapping ua="na"/>
<!-- Programmable Softkeys -->
<Programmable_Softkey_Enable ua="na">No</Programmable_Softkey_Enable>
<Idle_Key_List ua="na">recents;newcall;favorites;dir;settings;</Idle_Key_List>
<Missed_Call_Key_List ua="na">lcr|1;miss|4;</Missed_Call_Key_List>
<Off_Hook_Key_List ua="na">option;recents;cancel;dir;</Off_Hook_Key_List>
<Dialing_Input_Key_List
ua="na">option|1;call|2;delchar|3;cancel|4;left|5;right|6;</Dialing_Input_Key_List>
<Progressing_Key_List ua="na">endcall|2;</Progressing_Key_List>
<Connected_Key_List
ua="na">hold|1;endcall|2;conf|3;xfer|4;confLx;dir;settings;</Connected_Key_List>
<Start-Xfer_Key_List ua="na">hold|1;endcall|2;xfer|3;settings</Start-Xfer_Key_List>
<Start-Conf_Key_List ua="na">hold|1;endcall|2;conf|3;settings</Start-Conf_Key_List>
<Conferencing_Key_List ua="na">hold|1;endcall|2;join|3;settings;</Conferencing_Key_List>
<Releasing_Key_List ua="na">endcall|2;</Releasing_Key_List>
<Hold_Key_List ua="na">resume|1;endcall|2;newcall|3;recents;dir;settings;</Hold_Key_List>
<Ringing_Key_List ua="na">answer|1;ignore|2;</Ringing_Key_List>
<Shared_Active_Key_List
ua="na">newcall|1;barge|2;recents;favorites;dir;settings</Shared_Active_Key_List>
<Shared_Held_Key_List
ua="na">resume|1;newcall;barge|2;recents;favorites;dir;settings</Shared_Held_Key_List>
<PSK_1 ua="na"/>
<PSK_2 ua="na"/>
<PSK_3 ua="na"/>
<PSK_4 ua="na"/>
<PSK_5 ua="na"/>
<PSK_6 ua="na"/>
<PSK_7 ua="na"/>
```

```
<PSK_8 ua="na"/>
<PSK_9 ua="na"/>
<PSK_10 ua="na"/>
<PSK_11 ua="na"/>
<PSK_12 ua="na"/>
<PSK_13 ua="na"/>
<PSK_14 ua="na"/>
<PSK_15 ua="na"/>
<PSK_16 ua="na"/>
<!-- General -->
<Line_Enable_1_ ua="na">Yes</Line_Enable_1_>
<!-- Share Line Appearance -->
<Share_Ext_1_ ua="na">No</Share_Ext_1_>
<Shared_User_ID_1_ ua="na"/>
<Subscription_Expires_1_ ua="na">3600</Subscription_Expires_1_>
<Restrict_MWI_1_ ua="na">No</Restrict_MWI_1_>
<!-- NAT Settings -->
<NAT_Mapping_Enable_1_ ua="na">No</NAT_Mapping_Enable_1_>
<NAT_Keep_Alive_Enable_1_ ua="na">No</NAT_Keep_Alive_Enable_1_>
<NAT_Keep_Alive_Msg_1_ ua="na">$NOTIFY</NAT_Keep_Alive_Msg_1_>
<NAT_Keep_Alive_Dest_1_ ua="na">$PROXY</NAT_Keep_Alive_Dest_1_>
<!-- Network Settings -->
<SIP_TOS_DiffServ_Value_1_ ua="na">0x68</SIP_TOS_DiffServ_Value_1_>
<RTP_TOS_DiffServ_Value_1_ ua="na">0xb8</RTP_TOS_DiffServ_Value_1_>
<!-- SIP Settings -->
<SIP_Transport_1_ ua="na">UDP</SIP_Transport_1_>
<SIP_Port_1_ ua="na">5060</SIP_Port_1_>
<SIP_100REL_Enable_1_ ua="na">No</SIP_100REL_Enable_1_>
<EXT_SIP_Port_1_ ua="na">0</EXT_SIP_Port_1_>
<Auth_Resync-Reboot_1_ ua="na">Yes</Auth_Resync-Reboot_1_>
<SIP_Proxy-Require_1_ ua="na"/>
<SIP_Remote-Party-ID_1_ ua="na">No</SIP_Remote-Party-ID_1_>
<Referor_Bye_Delay_1_ ua="na">4</Referor_Bye_Delay_1_>
<Refer-To_Target_Contact_1_ ua="na">No</Refer-To_Target_Contact_1_>
<Referee_Bye_Delay_1_ ua="na">0</Referee_Bye_Delay_1_>
<Refer_Target_Bye_Delay_1_ ua="na">0</Refer_Target_Bye_Delay_1_>
<Sticky_183_1_ ua="na">No</Sticky_183_1_>
<Auth_INVITE_1_ ua="na">No</Auth_INVITE_1_>
<Ntfy_Refer_On_1xx-To-Inv_1_ ua="na">Yes</Ntfy_Refer_On_1xx-To-Inv_1_>
<Set_G729_annexb_1_ ua="na">yes</Set_G729_annexb_1_>
<Set_iLBC_mode_1_ ua="na">20</Set_iLBC_mode_1_>
<Voice_Quality_Report_Address_1_ ua="na"/>
<VQ_Report_Interval ua="na">0</VQ_Report_Interval>
<User_Equal_Phone_1_ ua="na">No</User_Equal_Phone_1_>
<!-- Call Feature Settings -->
<Blind_Attn-Xfer_Enable_1_ ua="na">No</Blind_Attn-Xfer_Enable_1_>
<Message_Waiting_1_ ua="na">No</Message_Waiting_1_>
<Auth_Page_1_ ua="na">No</Auth_Page_1_>
<Default_Ring_1_ ua="rw">1</Default_Ring_1_>
<Auth_Page_Realm_1_ ua="na"/>
<Conference_Bridge_URL_1_ ua="na"/>
<!-- <Auth_Page_Password_1_ ua="na"/> -->
<Mailbox_ID_1_ ua="na"/>
<Voice_Mail_Server_1_ ua="na"/>
<Voice_Mail_Subscribe_Interval_1_ ua="na">86400</Voice_Mail_Subscribe_Interval_1_>
<Broadsoft_ACD_1_ ua="na">No</Broadsoft_ACD_1_>
<Auto_Ans_Page_On_Active_Call_1_ ua="na">Yes</Auto_Ans_Page_On_Active_Call_1_>
<Feature_Key_Sync_1_ ua="na">No</Feature_Key_Sync_1_>
<Call_Park_Monitor_Enable_1_ ua="na">No</Call_Park_Monitor_Enable_1_>
<Enable_Broadsoft_Hoteling_1_ ua="na">No</Enable_Broadsoft_Hoteling_1_>
<Hoteling_Subscription_Expires_1_ ua="na">3600</Hoteling_Subscription_Expires_1_>
<!-- Proxy and Registration -->
<Proxy_1_ ua="na"/>
<Outbound_Proxy_1_ ua="na"/>
<Alternate_Proxy_1_ ua="na"/>
<Alternate_Outbound_Proxy_1_ ua="na"/>
<Use_OB_Proxy_In_Dialog_1_ ua="na">Yes</Use_OB_Proxy_In_Dialog_1_>
<Register_1_ ua="na">Yes</Register_1_>
<Make_Call_Without_Reg_1_ ua="na">No</Make_Call_Without_Reg_1_>
<Register_Expires_1_ ua="na">3600</Register_Expires_1_>
<Ans_Call_Without_Reg_1_ ua="na">No</Ans_Call_Without_Reg_1_>
<Use_DNS_SRV_1_ ua="na">No</Use_DNS_SRV_1_>
<DNS_SRV_Auto_Prefix_1_ ua="na">Yes</DNS_SRV_Auto_Prefix_1_>
```

```
<Proxy_Fallback_Intvl_1_ ua="na">3600</Proxy_Fallback_Intvl_1_>
<Proxy_Redundancy_Method_1_ ua="na">Normal</Proxy_Redundancy_Method_1_>
<Dual_Registration_1_ ua="na">No</Dual_Registration_1_>
<Auto_Register_When_Failover_1_ ua="na">No</Auto_Register_When_Failover_1_>
<!-- Subscriber Information -->
<Display_Name_1_ ua="na"/>
<User_ID_1_ ua="na"/>
<!-- <Password_1_ ua="na"/> -->
<Auth_ID_1_ ua="na"/>
<Reversed_Auth_Realm_1_ ua="na"/>
<SIP_URI_1_ ua="na"/>
<!-- Audio Configuration -->
<Preferred_Codec_1_ ua="na">G711u</Preferred_Codec_1_>
<Use_Pref_Codec_Only_1_ ua="na">No</Use_Pref_Codec_Only_1_>
<Second_Preferred_Codec_1_ ua="na">Unspecified</Second_Preferred_Codec_1_>
<Third_Preferred_Codec_1_ ua="na">Unspecified</Third_Preferred_Codec_1_>
<G711u_Enable_1_ ua="na">Yes</G711u_Enable_1_>
<G711a_Enable_1_ ua="na">Yes</G711a_Enable_1_>
<G729a_Enable_1_ ua="na">Yes</G729a_Enable_1_>
<G722_Enable_1_ ua="na">Yes</G722_Enable_1_>
<G722.2_Enable_1_ ua="na">Yes</G722.2_Enable_1_>
<iLBC_Enable_1_ ua="na">Yes</iLBC_Enable_1_>
<OPUS_Enable_1_ ua="na">Yes</OPUS_Enable_1_>
<Silence_Supp_Enable_1_ ua="na">No</Silence_Supp_Enable_1_>
<DTMF_Tx_Method_1_ ua="na">Auto</DTMF_Tx_Method_1_>
<Codec_Negotiation_1_ ua="na">Default</Codec_Negotiation_1_>
<Encryption_Method_1_ ua="na">AES 128</Encryption_Method_1_>
<!-- Dial Plan -->
<Dial_Plan_1_
ua="na">(*xx|[3469]11|0|00|[2-9]xxxxxx|1xxx[2-9]xxxxxxS0|xxxxxxxxxxx.)</Dial_Plan_1_>
<Caller_ID_Map_1_ ua="na"/>
<Enable_URI_Dialing_1_ ua="na">No</Enable_URI_Dialing_1_>
<Emergency_Number_1_ ua="na"/>
<!-- Hold Reminder -->
<Hold_Reminder_Timer ua="rw">0</Hold_Reminder_Timer>
<Hold_Reminder_Ring ua="rw">2</Hold_Reminder_Ring>
<!-- Call Forward -->
<Cfwd_Setting ua="rw">Yes</Cfwd_Setting>
<Cfwd_All_Dest ua="rw"/>
<Cfwd_Busy_Dest ua="rw"/>
<Cfwd_No_Ans_Dest ua="rw"/>
<Cfwd_No_Ans_Delay ua="rw">20</Cfwd_No_Ans_Delay>
<!-- Speed Dial -->
<Speed_Dial_2_Name ua="rw"/>
<Speed_Dial_2_Number ua="rw"/>
<Speed_Dial_3_Name ua="rw"/>
<Speed_Dial_3_Number ua="rw"/>
<Speed_Dial_4_Name ua="rw"/>
<Speed_Dial_4_Number ua="rw"/>
<Speed_Dial_5_Name ua="rw"/>
<Speed_Dial_5_Number ua="rw"/>
<Speed_Dial_6_Name ua="rw"/>
<Speed_Dial_6_Number ua="rw"/>
<Speed_Dial_7_Name ua="rw"/>
<Speed_Dial_7_Number ua="rw"/>
<Speed_Dial_8_Name ua="rw"/>
<Speed_Dial_8_Number ua="rw"/>
<Speed_Dial_9_Name ua="rw"/>
<Speed_Dial_9_Number ua="rw"/>
<!-- Supplementary Services -->
<CW_Setting ua="rw">Yes</CW_Setting>
<Block_CID_Setting ua="rw">No</Block_CID_Setting>
<Block_ANC_Setting ua="rw">No</Block_ANC_Setting>
<DND_Setting ua="rw">No</DND_Setting>
<Secure_Call_Setting ua="na">No</Secure_Call_Setting>
<Auto_Answer_Page ua="na">Yes</Auto_Answer_Page>
<Time_Format ua="na">12hr</Time_Format>
<Date_Format ua="na">month/day</Date_Format>
<Miss_Call_Shortcut ua="na">No</Miss_Call_Shortcut>
<Handset_LED_Alert ua="rw">Voicemail</Handset_LED_Alert>
<Alert_Tone_Off ua="rw">No</Alert_Tone_Off>
<Log_Missed_Calls_for_EXT_1 ua="na">Yes</Log_Missed_Calls_for_EXT_1>
<Shared_Line_DND_Cfwd_Enable ua="na">Yes</Shared_Line_DND_Cfwd_Enable>
```

```
<!-- Camera Profile 1 -->
<!-- Camera Profile 2 -->
<!-- Camera Profile 3 -->
<!-- Camera Profile 4 -->
<!-- Audio Volume -->
<Ringer_Volume ua="rw">9</Ringer_Volume>
<Speaker_Volume ua="rw">11</Speaker_Volume>
<Ehook_Enable ua="na">No</Ehook_Enable>
<!-- Screen -->
<Screen_Saver_Enable ua="rw">No</Screen_Saver_Enable>
<Screen_Saver_Type ua="rw">Clock</Screen_Saver_Type>
<Screen_Saver_Wait ua="rw">300</Screen_Saver_Wait>
<Screen_Saver_Refresh_Period ua="rw">10</Screen_Saver_Refresh_Period>
<Back_Light_Timer ua="rw">30s</Back_Light_Timer>
<Boot_Display ua="na">Default</Boot_Display>
<Text_Logo ua="na"/>
<Phone_Background ua="rw">Default</Phone_Background>
<Picture_Download_URL ua="rw"/>
<Logo_URL ua="rw"/>
<!-- General -->
<Server_Type ua="na">Broadsoft</Server_Type>
<BXfer_To_Starcode_Enable ua="na">No</BXfer_To_Starcode_Enable>
<BXfer_On_Speed_Dial_Enable ua="na">No</BXfer_On_Speed_Dial_Enable>
<!-- TR-069 -->
<Enable_TR-069 ua="na">No</Enable_TR-069>
<ACS_URL ua="na"/>
<ACS_Username ua="na"/>
<!-- <ACS_Password ua="na"/> -->
<ACS_URL_In_Use ua="ro"/>
<Connection_Request_Username ua="na"/>
<!-- <Connection_Request_Password ua="na"/> -->
<Connection_Request_URL ua="ro"/>
<Periodic_Inform_Interval ua="na">20</Periodic_Inform_Interval>
<Periodic_Inform_Enable ua="na">Yes</Periodic_Inform_Enable>
<TR-069_Traceability ua="na">No</TR-069_Traceability>
<CWMP_V1.2_Support ua="na">Yes</CWMP_V1.2_Support>
<TR-069_VoiceObject_Init ua="na">Yes</TR-069_VoiceObject_Init>
<TR-069_DHCPOption_Init ua="na">Yes</TR-069_DHCPOption_Init>
<TR-069_Fallback_Support ua="na">No</TR-069_Fallback_Support>
<BACKUP_ACS_URL ua="na"/>
<BACKUP_ACS_User ua="na"/>
<!-- <BACKUP_ACS_Password ua="na"/> -->
</flat-profile>
```



APPENDIX B

Acronyms

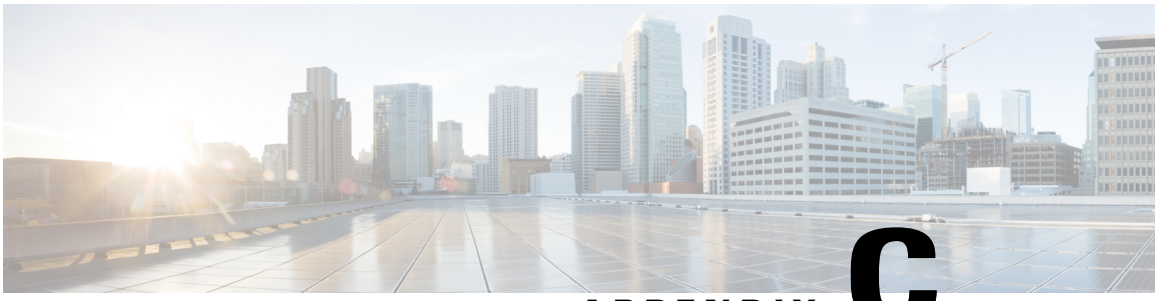
Acronyms

A/D	Analog To Digital Converter
ANC	Anonymous Call
B2BUA	Back to Back User Agent
Bool	Boolean Values. Specified as yes and no, or 1 and 0 in the profile
CA	Certificate Authority
CAS	CPE Alert Signal
CDR	Call Detail Record
CID	Caller ID
CIDCW	Call Waiting Caller ID
CNG	Comfort Noise Generation
CPC	Calling Party Control
CPE	Customer Premises Equipment
CWCID	Call Waiting Caller ID
CWT	Call Waiting Tone
D/A	Digital to Analog Converter

dB	decibel
dBm	dB with respect to 1 milliwatt
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DRAM	Dynamic Random Access Memory
DSL	Digital Subscriber Loop
DSP	Digital Signal Processor
DTAS	Data Terminal Alert Signal (same as CAS)
DTMF	Dual Tone Multiple Frequency
FQDN	Fully Qualified Domain Name
FSK	Frequency Shift Keying
FXS	Foreign eXchange Station
GW	Gateway
ITU	International Telecommunication Union
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP over SSL
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
ILEC	Incumbent Local Exchange Carrier
IP	Internet Protocol
ISP	Internet Service Provider
ITSP	Internet Telephony Service Provider
IVR	Interactive Voice Response
LAN	Local Area Network

LBR	Low Bit Rate
LBRC	Low Bit Rate Codec
MC	Mini-Certificate
MGCP	Media Gateway Control Protocol
MOH	Music On Hold
MOS	Mean Opinion Score (1-5, the higher the better)
MPP	Multiplatform Phones
ms	Millisecond
MSA	Music Source Adaptor
MWI	Message Waiting Indication
OSI	Open Switching Interval
PCB	Printed Circuit Board
PR	Polarity Reversal
PS	Provisioning Server
PSQM	Perceptual Speech Quality Measurement (1-5, the lower the better)
PSTN	Public Switched Telephone Network
NAT	Network Address Translation
OOB	Out-of-band
REQT	(SIP) Request Message
RESP	(SIP) Response Message
RSC	(SIP) Response Status Code, such as 404, 302, 600
RTP	Real Time Protocol
RTT	Round Trip Time
SAS	Streaming Audio Server
SDP	Session Description Protocol

SDRAM	Synchronous DRAM
sec	seconds
SIP	Session Initiation Protocol
SLA	Shared line appearance
SLIC	Subscriber Line Interface Circuit
SP	Service Provider
SSL	Secure Socket Layer
TFTP	Trivial File Transfer Protocol
TCP	Transmission Control Protocol
UA	User Agent
uC	Micro-controller
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VM	Voicemail
VMWI	Visual Message Waiting Indication/Indicator
VQ	Voice Quality
WAN	Wide Area Network
XML	Extensible Markup Language



APPENDIX C

Related Documentation

Related Documentation

Use the following sections to obtain related information.

Cisco IP Phone 7800 Series Documentation

Refer to publications that are specific to your language, phone model, and call control system. Navigate from the following documentation URL:

https://www.cisco.com/c/en/us/support/collaboration-endpoints/unified-ip-phone-7800-series/tsd-products-support-general-information.html

Cisco IP Phone Firmware Support Policy

For information on the support policy for Cisco IP Phones, see http://www.cisco.com/c/en/us/support/docs/collaboration-endpoints/unified-ip-phone-7900-series/116684-technote-ipphone-00.html.

Documentation, Service Requests, and Additional Information

For information about how to obtain documentation, submit a service request, and gather additional information, see the monthly publication What's New in Cisco Product Documentation. This publication also lists all new and revised Cisco technical documentation at https://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html.

Subscribe to the What's New in Cisco Product Documentation as a Really Simple Syndication (RSS) feed. Use a reader application to set content for direct delivery to your desktop. The RSS feeds are a free service, and Cisco currently supports RSS Version 2.0.