PROVISIONING GUIDE

Cisco Small Business

IP Telephony Devices
Voice System, Voice Gateways, and IP Telephones

Contents

About This Document

This guide describes the provisioning of Cisco Small Business Voice over IP (VoIP) products:

Purpose

The following Cisco Small Business VoIP products can be remotely provisioned or preprovisioned by using the information in this document:

Cisco Small Business Analog Telephone Adapters (ATAs):

- PAP2T
- SPA2102
- SPA3102
- SPA8000
- WRP400

Cisco Small Business IP phones:

- SPA301
- SPA303
- SPA501
- SPA502
- SPA504
- SPA508

- SPA509
- SPA525G/G2

Document Audience

This document is written for service providers who offer services by using Cisco Small Business VoIP products and specifically for administrative staff responsible for remote provisioning and preprovisioning Cisco Small Business devices.

Organization

This document is divided into the following chapters and appendices.

Chapter	Contents
Chapter 1, "Deployment and Provisioning"	Introduces Cisco Small Business VoIP products.
Chapter 2, "Creating Provisioning Scripts"	Describes how to work with Cisco Small Business provisioning scripts and configuration profiles.
Chapter 4, "Provisioning Examples"	Step-by-step procedures for using the scripting language to create a configuration profile.
Chapter 5, "Provisioning Field Reference"	A systematic reference for each parameter on the Provisioning tab of the administration web server.
Appendix A, "Sample Configuration Profiles"	A sample profile that you might find helpful.
Appendix B, "Acronyms"	The expansions for the acronyms used in this document.
Appendix C, "Where to Go From Here"	Links to resources for information and support.

Document Conventions

The following typographic conventions are used in this document.

Typographic Element	Meaning
Boldface	An option on a menu or a literal value to be entered in a field.
<parameter>	Angle brackets (<>) identify parameters that appear on the configuration pages of the administration web server.
Italic	A variable that should be replaced with a literal value.
`Monospaced Font`	A code sample or system output.

1

Deployment and Provisioning

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

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

This customized, ongoing configuration is supported by the following features:

- Reliable remote control of the endpoint
- Encryption of the communication controlling the endpoint
- Streamlined endpoint account binding

This chapter describes the features and functionality available when provisioning Cisco Small Business IP Telephony devices and explains the setup required:

- **Deployment, page 9**
- **Provisioning Overview, page 11**

Deployment

Cisco Small Business IP Telephony devices provide convenient mechanisms for provisioning, based on two deployment models:

- Bulk distribution—The service provider acquires IP Telephony devices in bulk quantity and either preprovisions the IP Telephony Devices in-house or purchases RC units from Cisco. The devices are then issued to its customers as part of a VoIP service contract.
- Retail distribution—The customer purchases the IP Telephony device 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 IP Telephony devices to its customers as part of a VoIP service contract. The devices are either RC units or preprovisioned in-house.

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

A service provider can preprovision IP Telephony devices with the desired parameters, including the parameters that control resynchronization, in-house before distribution by using DHCP and TFTP, remotely by using TFTP, HTTP, or HTTPS, or a combination of in-house and remote provisioning.

RC Unit Deployment

RC units eliminate in-house preprovisioning of IP Telephony devices and reduce the need for the service provider to physically handle the devices prior to shipping them to end customers. It also discourages the use of Cisco Small Business IP Telephony Devices with an inappropriate service provider.

A RC unit is preprovisioned by Cisco with the connection information for the Cisco Small Business provisioning servers. These servers are maintained by Cisco Systems, Inc. for the service provider that purchased the units. The MAC address of each RC unit is associated with a service provider–customizable profile on the Cisco provisioning servers. When the RC unit is connected to the broadband link, it contacts the Cisco Small Business provisioning server and downloads its customized profile.

The service provider works with a Cisco sales engineer to develop a simple provisioning profile. The profile contains minimal information that redirects the device to the service provider provisioning server. This profile is placed on the Cisco RC server by the Cisco Voice Team.

RC Unit Status

The status of an RC unit can be determined by viewing the Info > Product Information page Customization section on the administration web server. An RC unit that has not been provisioned displays **Pending**. An RC unit that has been provisioned displays the name of the company that owns the unit. If the unit is not an RC unit, the page displays **Open**.

The following is a sample template for an RC unit to be preprovisioned by Cisco with the connection information:

```
Restricted Access Domains "domain.com, domain1.com, domain2.com";
Primary_DNS               * "x.y.w.z";
Secondary_DNS             * "a.b.c.d";
Provision_Enable          * "Yes";
Resync_Periodic           * "30";
Resync_Error_Retry_Delay  * "30";
Profile_Rule  * "http://prov.domain.com/sipura/profile?id=$MA";
```

The `Restricted Access Domains` parameter is configured with the actual domain names of up to a maximum of five domains. The Primary_DNS and Secondary_DNS parameters are configured with the actual domain names or IP addresses of the DNS servers available to the RC unit.

Retail Distribution

In a retail distribution model, a customer purchases a Cisco Small Business voice endpoint device and subscribes to a particular service. The an Internet Telephony Service Provider (ITSP) sets up and maintains a provisioning server, and preprovisions the phone to resynchronize with the service provider server. See **In-House Device Preprovisioning, page 56** for more information.

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. When the device is powered up or a set time elapses, the IP Telephony device resynchronizes, downloading the latest parameters. These parameters can include anything from hunt group configuration to preset speed dials to limiting the features that the user can modify.

Resynchronization Process

The firmware for each IP Telephony device includes an administration web server that accepts new configuration parameter values. The IP Telephony device is instructed to resync with a specific provisioning server through a resync URL command in the device profile. The URL command typically includes an account PIN number or alphanumeric code to associate the device with the new account. For 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` and is instructed to provision itself to the SuperVoIP service at `prov.supervoip.com`. The PIN number for the new account is `1234abcd`. The remote provisioning server is configured to associate the IP Telephony device that is performing the resync request with the new account, based on the URL and PIN.

Through this initial resync operation, the IP Telephony device is configured in a single step, and 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 and supplies configuration parameter values based on the associated service account.

Provisioning Overview

An IP Telephony device can be configured to resynchronize its internal configuration state to match a remote profile periodically and on power up by contacting a normal provisioning server (NPS) or an access control server (ACS).

By default, a profile resync is only attempted when the IP Telephony device is idle, because the upgrade might trigger a software reboot interrupting a call. If intermediate upgrades are required to reach a current upgrade state from an older release, the upgrade logic is capable of automating multi-stage upgrades.

NPS

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

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. Secure first-time provisioning is provided through a mechanism that uses SSL functionality. An unprovisioned IP Telephony device can receive a 256-bit symmetric key encrypted profile specifically targeted for that device.

TR-069

The digital subscriber line (DSL) Forum TR-069, CPE WAN Management Protocol (CWMP), is used for communications between a customer premise equipment (CPE) device and an auto-configuration server (ACS). The TR-069 Agent manages a collection of CPE devices, with the primary capability for auto-configuration and dynamic service provisioning, software image management, status and performance monitoring and diagnostics.

It supports multiple scenarios, including:

Device administration: Authenticates administrators, authorizes commands, and provides an audit trail

Remote Access: Works with VPN and other remote network access devices to enforce access policies

Network admission control: Communicates with posture and audit servers to enforce admission control policies

The TR-069 Agent CPE devices must be set up and enabled for TR-069. An ACS used to communicate with the CPE must be TR-069 compliant in order to enable the TR-069 Agent.

The schema for TR-069, TR-098, TR-104 and TR-106 dictionaries are located at BPR_HOME/rdu/templates/cwmp/schema/TemplateDictionarySchema.xsd

Provisioning States

The provisioning process involves these provisioning states.

State	Description
MFG-RESET Manufacturing Reset	The device returns to a fully unprovisioned state; all configurable parameters regain their default values. Manufacturing reset can be performed through the IVR sequence `****RESET#1#`. On phones that do not support IVR, power cycle the phone to reset it to default values. Allowing the end user to perform a manufacturing reset guarantees that the device can always be returned to an accessible state.

State	Description
SP-CUST Service Provider Customization	The Profile_Rule parameter points to a device-specific configuration profile by using a provisioning server that is specific to the service provider. The methods for initiating resynchronization are: ▪ Auto-configuration by using a local DHCP server. A TFTP server name or IPv4 address is specified by DHCP. The TFTP server includes the **Profile_Rule** parameter in the configuration file. ▪ Entering a resync URL. The URL starts a web browser and requests a resync to a specific TFTP server by entering the URL syntax: `http://x.x.x.x/admin/resync?prvserv/device.cfg`, where: *x.x.x.x* is the IP address of the IP Telephony device, *prvserv* is the target TFTP server, and *device.cfg* is the name of the configuration file on the server. ▪ Editing the **Profile_Rule** parameter by opening the provisioning pane on the web interface and entering the TFTP URL in the **Profile_Rule** parameter. For example, *prserv*/spa962.cfg. ▪ Modifying the configuration file **Profile_Rule** and configure it to contact a specific TFTP server and request a configuration file identified by the MAC-address. For example, this entry contacts a provisioning server, requesting a profile unique to the device with a MAC address identified by the `$MA` parameter: `Profile_Rule tftp.callme.com/profile/$MA/spa962.cfg;`

State	Description

SEC-PRV-1 Secure Provisioning—Initial Configuration	An initial, device-unique CFG file is targeted to a IP Telephony device by compiling the CFG file with the SPC `--target` option. This provides an encryption that does not require the exchange of keys.
	
	The initial, device-unique CFG file reconfigures the device profile to enable stronger encryption by programming a 256-bit encryption key and pointing to a randomly-generated TFTP directory. For example, the CFG file might contain:
	
	`Profile_Rule [--key $A] tftp.callme.com/profile/$B/spa962.cfg;`
	`GPP_A 8e4ca259…;  # 256 bit key`
	`GPP_B Gp3sqLn…;  # random CFG file path directory`

SEC-PRV-2 Secure Provisioning—Full Configuration	Profile resync operations subsequent to the initial SEC-PRV-1 provisioning retrieve the 256-bit encrypted CFG files that maintain the IP Telephony device in a state synchronized to the provisioning server. The profile parameters are reconfigured and maintained through this strongly encrypted profile. The encryption key and random directory location in the SEC-PRV-2 configuration can be changed periodically for extra security.

Configuration Access Control

The IP Telephony device firmware provides mechanisms for restricting end-user access to some parameters. The firmware provides specific privileges for login to an **Admin** account or a **User** account. Each can be independently password protected.:

- Admin Account—Allows the service provider full access to all interactive voice response (IVR) functions and to all administration web server parameters.
- User Account—Allows the user of a device to access basic IVR functions and 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. (Described in **"Element Tags" on page 19**.)
- Disable user access to the administration web server.
- Disable the factory reset control by using the IVR.
- Restrict the Internet domains accessed by the device for resync, upgrades, or SIP registration for Line 1.

Communication Encryption

The configuration parameters communicated to the IP Telephony 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 activity by the customer, and 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 IP Telephony device, in addition to restricting access to the administration web server.

2

Creating Provisioning Scripts

IP Telephony devices accept two profile formats, one based on an open, published syntax, and one based on an unpublished binary definition. The Open format uses a simple XML-like syntax. The binary format begins as a Plain-text format file. The SPA Profile Compiler (SPC) converts the Plain-text formatted file to a binary file.

To use the proprietary Plain-text format, you must convert the files using the SPC before they can be uploaded to the device. For more information see **Plain-Text Profile Format, page 25**.

The examples in this document use configuration profiles with Open format (XML-style) syntax. Sample profiles can be found in

This chapter describes the provisioning script in the following sections:

For detailed information about your IP Telephony device, refer to the administration guide for that 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 a specific IP Telephony device. The configuration profile can be in one of two formats:

- Open format uses standard XML authoring tools to compile the parameters and values. To protect confidential information in the configuration profile, this type of file is typically delivered from the provisioning server to the IP Telephony device over a secure channel provided by HTTPS. See **Open Profile (XML-style) Compression and Encryption, page 23**.
- Proprietary Plain-text format uses a proprietary parameter value-pair design. See **Plain-Text Profile Format, page 25**.

NOTE Only UTF-8 charset is supported in plain-text and open format. If you modify the profile in an editor, do not change the encoding format; otherwise, phone cannot recognize the file.

Each model of the IP Telephony device model has a different feature set and therefore a different set of parameters. You can produce a sample configuration profile by following the procedure in **Generating Sample Configuration Files, page 50**.

Open Format (XML-style) Profile

The Open format profile is a text file with XML-like syntax in a hierarchy of elements, with element attributes and values. The Open format lets you use standard tools to create the configuration file. A configuration file in Open format can be sent from the provisioning server to the IP Telephony device during a resync operation without compiling the file as a binary object. An example XML profile can be generated by using the SPA Profile Compiler (SPC). See **Generating Sample Configuration Files, page 50** for more information.

The IP Telephony device can accept configuration formats that are generated by standard tools. This feature eases the development of back-end provisioning server software to generate configuration profiles from existing databases.

To protect confidential information contained in the configuration profile, this file is generally delivered from the provisioning server to the IP Telephony device over a secure channel provided by HTTPS.

Optionally, the file can be compressed by using the gzip deflate algorithm (RFC1951). In addition, the file can be encrypted by using 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/spa504.cfg
  </Profile_Rule>
</flat-profile>
```

The `<flat-profile>` element tag encloses all parameter elements to be recognized by the IP Telephony device.

NOTE IP Telephony devices with firmware versions before 2.0.6 do not support Open format profiles.

Element Tags, Attributes, Parameters, and Formatting

A file can include element tags, attributes, parameters, and formatting features.

Element Tags

The properties of element tags are:

- The IP Telephony device recognizes elements with proper parameter names, when encapsulated in the special <flat-profile> element.
- The <flat-profile> element can be encapsulated within other arbitrary elements.
- Element names are enclosed in angle brackets.
- The element names derive from 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 `/`.

    For example, the Resync On Reset field is represented by the element `<Resync_On_Reset>`.

- Each element name must be unique. In the administration web pages, the same fields might 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.

  For example, the Dial Plan for Line 1 is represented by the element `<Dial_Plan[1]>`

- Each opening element tag must be matched by a corresponding 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/
spa962.cfg
    </Profile_Rule>
</flat-profile>
```

- Element tags are case sensitive.
- Empty element tags are allowed.
- Unrecognized element names are ignored.
- An empty element tag can be used to prevent the overwriting of any user-supplied values during a resync operation. In the following example, the user speed dial settings are unchanged:

```
<Speed_Dial_2_2_  ua=”rw”/>
  <Speed_Dial_3_2_  ua=”rw”/>
  <Speed_Dial_4_2_  ua=”rw”/>
  <Speed_Dial_5_2_  ua=”rw”/>
  <Speed_Dial_6_2_  ua=”rw”/>
  <Speed_Dial_7_2_  ua=”rw”/>
  <Speed_Dial_8_2_  ua=”rw”/>
  <Speed_Dial_9_2_  ua=”rw”/>
</flat-profile>
```

- An empty value can be used 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>
```

User Access

The **ua** attribute controls access by the User account for specific parameters. If the **ua** attribute is not specified in an element tag, the factory default user access is applied for the corresponding parameter is applied. Access by the Admin account is unaffected by this attribute.

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

- na—no access
- ro—read-only
- rw—read/write

The **ua** attribute is illustrated by the following example:

```
<flat-profile>
  <SIP_TOS_DiffServ_Value_1_   ua="na"/>
  <Dial_Plan_1_   ua="ro"/>
  <Dial_Plan_2_   ua="rw"/>
</flat-profile>
```

The value of the **ua** option must be enclosed by double quotes.

Parameter Properties

These properties apply to the parameters:

- Any parameters that are not specified by a profile are left unchanged in the IP Telephony device.
- Unrecognized parameters are ignored.

- The IP Telephony device recognizes arbitrary, configurable aliases for a limited number of parameter names.
- If the Open format profile contains multiple occurrences of the same parameter tag, the last such occurrence overrides any earlier ones. To avoid inadvertently overriding configuration values for a parameter, it is recommended that at most one instance of a parameter be specified in any one profile.

Formatting

These properties apply to the formatting of the strings:

- Comments are allowed by using standard XML syntax.

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

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

Special Character	XML Escape Sequence
& (ampersand)	&amp;
< (less than)	&lt;
> (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 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.
- The firmware does not support the full Unicode character set, but only the ASCII subset.

Open Profile (XML-style) Compression and Encryption

The Open configuration profile can be compressed to reduce the network load on the provisioning server and encrypted to protect confidential information. Compression is not required, but it must precede encryption for the IP Telephony device to recognize a compressed and encrypted Open profile.

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 when compression is applied, the IP Telephony device expects the compressed file to contain a gzip compatible header, as generated by invoking the gzip utility on the original Open profile. The IP Telephony device inspects the downloaded file header to determine the format of the file.

For example, if `profile.xml` is a valid profile, the file `profile.xml.gz` is also accepted. This profile type can be generated with either of the following commands:

```
>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 **Open Profile gzip Compression, page 79**.

Open Profile Encryption by using AES

An Open configuration profile can be encrypted by using symmetric key encryption, whether or not it is compressed. 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 IP Telephony device to recognize a compressed and encrypted Open format profile. A tutorial on encryption is provided in **Profile Encryption by using OpenSSL, page 80**.

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

If the file is encrypted, 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 lower case –k precedes the secret key, which can be any plain text phrase and is used to generate a random 64-bit salt. Then, in combination with the secret specified with 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 to encrypt a configuration profile, the IP Telephony device must be informed of the secret key value to decrypt the file. This value is specified as a qualifier in the pertinent 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. This bootstrap of the secret key can be accomplished securely by using HTTPS.

Preencrypting 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 IP Telephony devices after deployment. This feature reduces the load on the HTTPS server in large scale deployments.

The final file name does not need to follow a specific format, but it is conventional to end the name with the .cfg extension to indicate that it is a configuration profile.

Plain-Text Profile Format

The Plain-text format profile uses a proprietary format that can be encrypted to prevent unauthorized use of confidential information. By convention, the encrypted profile is named with the extension .cfg (for example, spa962.cfg). The SPC tool is used to compile the Plain-text format profile into an encrypted CFG file.

The Plain-text format is an alternative to the Open format and is supported by all firmware releases, and it is the only format recognized by firmware releases prior to version 2.0.6. An example Plain-text format profile can be generated by using the SPA Profile Compiler (SPC). See the **Generating Sample Configuration Files, page 50** for more information.

Plain–Text Syntax

The syntax of the Plain-text format profile accepted by SPC is a series of parameter-value pairs, with the value enclosed in double quotes. Each parameter-value pair is followed by a semicolon (for example, parameter_name “parameter_value”;). If no quoted value is specified for a parameter (or if a parameter specification is missing entirely from the Plain-text format profile) the value of the parameter remains unchanged in the IP Telephony device.

The syntax also controls the User account access to the parameter in the administration web server. An optional exclamation point or question mark, immediately following the parameter name, indicates the parameter should be read-write (!) or user read-only (?) for the User account.

If neither mark is present, the parameter is made inaccessible to the user from the web server pages. Note that this syntax has no effect on the Admin account access to the parameter. If the parameter specification is missing entirely from the Plain-text format profile, the User account access to the parameter remains unchanged in the IP Telephony device.

If the Plain-text format profile contains multiple occurrences of the same parameter-value specification, the last occurrence overrides any earlier ones. To avoid accidentally overwriting configuration values, it is recommended that a profile includes no more than one specification for each parameter.

The element names derive from 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 field name, substitute an underscore for spaces or the following special characters: `[ ] ( ) /`

  For example, the Resync On Reset field is represented by the following element: `<Resync_On_Reset>`

- Each element name must be unique. For fields that are duplicated on multiple Line, User, or Extension pages, you must append [*n*] to indicate the line, user, or extension number.

  For example, the Dial Plan for Line 1 is represented by the following element: `<Dial_Plan[1]>`

The following additional features can be used:

- Comments are delimited by a # character, from the character to the end-of-line.
- Blank lines can be used for readability.

The following string illustrates the format for each parameter-value pair:

```
Parameter_name [ '?' | '!' ] ["quoted_parameter_value_string"] ';'
```

Boolean parameter values are asserted by any one of the values {Yes | yes | Enable | enable | 1}. They are deasserted by any one of the values {No | no | Disable | disable | 0}.

The following are examples of Plain-text format profile entries:

```
# These parameter names are for illustration only

Feature_Enable      ! "Enable" ;   # user read-
write, but force the value to Enable
Another_Parameter   ? "3600"   ;   # user read-only
Hidden_Parameter      "abc123" ;   # user not-accessible
Some_Entry          !          ;   # user read-write, leaves value
unchanged
```

Multiple plain text files can be spliced together to generate the source for the final binary CFG file. This is accomplished by using the **import** directive at the start of a new line followed by one or more spaces and the file name to splice into the stream of parameter-value pairs. File splicing can be nested several files deep.

For example, the file base.txt contains the following:

```
Param1 "base value 1" ;
Param2 "base value 2" ;
```

The file spa1234.txt contains the following lines:

```
import base.txt
Param1 "new value overrides base" ;
Param7 "particular value 7" ;
```

When compiled, spa1234.cfg becomes:

```
Param1 "base value 1" ;
Param2 "base value 2" ;
Param1 "new value overrides base" ;
Param7 "particular value 7" ;
```

Comments

During development and scripting, it is often convenient to temporarily disable a provisioning parameter by entering a # character at the start of the parameter value. This effectively comments-out the remaining text in that parameter.

For example, a Profile_Rule with the value "`# http://192.168.1.200/sample.cfg`" is equivalent to an empty Profile_Rule. The # character comment-mechanism applies to the Profile_Rule*, Upgrade_Rule, and Resync_Trigger_* parameters.

Macro Expansion

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

The parameter groups which undergo macro expansion before evaluation are as follows:

- Resync_Trigger_*
- Profile_Rule*

- Log_Resync_*
- Upgrade_Rule
- Log_Upgrade_*

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

During macro expansion, expressions of the form $NAME and $(NAME) are replaced by the contents of the named variables. 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" section on page 95**.

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

> The administrator enters: `spa$(MAU)config.cfg`
> The resulting macro expansion for a device with MAC address 000E08012345 is: `spa000E08012345config.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: `spa$STRANGE$MAU.cfg`
> The resulting macro expansion for a device with MAC address 000E08012345 is: `spa$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 special purpose parameters (GPP_SA through GPP_SD), whose contents are mapped to the macro expressions $SA through $SD, are only macro expanded as the argument of the **--key** option in a resync URL.

Also, the macro expression can qualify the expansion so that only a substring of the macro variable is used instead of its full value, such as a portion of the MAC address.

The syntax for substring macro expansion is $(NAME:p) and $(NAME:p:q), where p and q are non-negative integers. The resulting expansion results in the macro variable substring starting at character offset p, and of length q (or till end-of-string if q is not specified). Refer to the following examples.

> The administrator enters: `$(MAU:4)`
> The resulting macro expansion for a device with MAC address 000E08012345 is: `08012345`
>
> The administrator enters: `$(MAU:8:2)`
> The resulting macro expansion for a device with MAC address 000E08012345 is: `23`

Conditional Expressions

Conditional expressions can trigger resync events and select from alternative 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 one of three types of literals:

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

Note that version numbers take the form of three non-negative integers separated by periods (major, minor, and build numbers), plus an optional alphanumeric string in parentheses. No spaces are allowed.

The following are examples of valid version numbers:

```
1.0.31(b)
1.0.33
2.0.3(G)
2.0.3(0412s)
2.0.6
```

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 particularly convenient when 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

For legacy support to firmware versions prior to 2.0.6, the not-equal-to operator can also be expressed as a single ! character (in place of the two-character != string).

Conditional expressions typically involve macro-expanded variables. For example:

```
$REGTMR1 gt 300 and $PRVTMR gt 1200 and "$EXTIP" ne ""

$SWVER ge 2.0.6 and "$CCERT" eq "Installed"
```

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.

For legacy support of firmware versions prior to 2.0.6, a relational expression with no left-hand-side operand assumes $SWVER as the implicit left-hand-side. For example, ! 1.0.33 is equivalent to: $SWVER != 1.0.33.

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

```
($SWVER ne 2.0.6)? http://ps.tell.com/sw/spa021024.bin
```

On the other hand, the syntax above using parentheses should not be used when configuring the Resync_Trigger_* parameters.

Assignment Expressions

Arbitrary parameters can be pre-assigned values within the context of Profile_Rule* and Upgrade_Rule parameter. This causes the assignment to be performed before the profile if retrieved.

The syntax for performing these assignments is a list of individual parameter assignments, enclosed within parentheses (assignments)!, with each assignment taking the form:

ParameterXMLName = “Value” ;

Note that the recognized parameter names correspond to the names as for XML-based profiles.

Any parameter can be assigned a new value in this way, and macro-expansion applies. For example, the following is a valid assignment expression:

```
(User_ID_1_ = “uid$B” ; GPP_C = “” ; GPP_D = “$MA” ;)!
```

For conciseness, the general purpose parameters GPP_A through GPP_P can also be referred to by the single lowercase letters a through p. The example above is equivalent to the following:

```
(User_ID_1_ = “uid$B” ; c = “” ; d = “$MA” ;)!
```

White space can be used for readability.

URL Syntax

Standard URL syntax is used 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 the following values:

- tftp
- http
- https

If scheme is omitted, tftp is assumed. The server can be a DNS-recognized host name 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, then the tftp server specified through DHCP (option 66) is used.

If port is missing, then 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:

```
/$MA.cfg
/cisco/spa021025.bin
192.168.1.130/profiles/init.cfg
tftp://prov.call.com/cpe/cisco$MA.cfg
http://neptune.speak.net:8080/prov/$D/$E.cfg
https://secure.me.com/profile?Linksys
```

Optional Resync Arguments

The URLs entered in Profile_Rule* parameters can be preceded by optional arguments, collectively enclosed by square brackets. The options are key, post, and alias.

key

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

Some usage examples:

```
[--key VerySecretValue]
[--key "my secret phrase"]
[--key a37d2fb9055c1d04883a0745eb0917a4]
```

The bracketed optional arguments are macro expanded. In particular, note that the special purpose parameters GPP_SA through GPP_SD are only macro expanded into their macro variables $SA through $SD when used as arguments of the key option, as in the following examples:

```
[--key $SC]
[--key "$SD"]
```

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

In the case of SPC compiled profiles, the argument to **--key** must be the same as the argument to either the **--ascii-key** or the **--hex-key** options.

post

The **post** option provides an alternative access method for the http and https schemes. If left unspecified, the IP Telephony device performs an HTTP GET operation, when contacting the provisioning server. If post is specified, the device performs an HTTP POST operation.

The body of the POST is generated from the contents of one of the general purpose parameters, GPP_A through GPP_P, with macro expansion applied. The GPP_* parameter to use is indicated by a single lowercase letter (a through p) given as argument to the term **--post**.

Using POST provides a convenient alternative to the GET method when arbitrary state or identifying information needs to be supplied from the IP Telephony device to the server, as part of periodic resyncs.

For example, GPP_F could contain the following POST body template:

```
Product = "$PN"; MAC_Addr = "$MA"; Ser_Num = "$SN"; SW_Ver = "$SWVER";
```

Then, a URL option uses the POST method to convey the information to the server in the body of the profile request message (shown here with an accompanying URL):

```
[--post f ] http://ps.one.com/cpe/resyncs?
```

alias

The **alias** option provides a flexible means of recognizing alternative parameter names in Open format profiles. This is useful in cases where part of the configuration profile is obtained from a customer database form that uses different terminology than expected by the IP Telephony device.

For example, a customer Open format profile specifies the SIP registration parameters: name, number, auth-secret, enclosed in an XML element hierarchy as follows:

```
<CPE>
  <SIP-Credentials>
    <name>J. Smith</name>
    <number>14085551234</number>
    <auth-secret>732091751563sfd</auth-secret>
```

```
  <SIP-Credentials>
</CPE>
```

To map these three parameters directly to the Display_Name_1_, User_ID_1_, and Password_1_ parameters (Line 1), enter this mapping in a general purpose parameter (for example, GPP_M):

```
/CPE/SIP-Credentials/name = /flat-profile/Display_Name_1_ ;
/CPE/SIP-Credentials/number = /flat-profile/User_ID_1_ ;
/CPE/SIP-Credentials/auth-secret = /flat-profile/Password_1_ ;
```

Then, request the customer credentials profile with the following URL option (showing an example URL for completeness):

```
[--alias m ] http://acct.voipservice.net/credentials/spa$MA.xml
```

Upon receiving the profile, the IP Telephony device would apply the indicated translations, assigning J. Smith to Display_Name_1_, 14085551234 to User_ID_1_, and 732091751563sfd to Password_1_.

The **alias** option matches only the left-hand-side of an alias as much as specified by the configured alias map. The element itself can be nested further. GPP_M could have contained the following:

```
/SIP-Credentials/name = /flat-profile/Display_Name_1_ ;
/SIP-Credentials/number = /flat-profile/User_ID_1_ ;
/auth-secret = /flat-profile/Password_1_ ;
```

In general, it is best to specify enough enclosing elements to ensure an unambiguous translation.

The **alias** option is designed to recognize a limited number of critical parameters. Up to 30 parameters can be remapped this way.

Combining Options

Multiple URL options can be combined, by enclosing them within the same set of square brackets. The following are examples of valid URL option strings:

```
[--post j --alias k]
[--key “SymmetricSecret” --alias a]
[--key “$SB” --post g]
[--alias a --key abracadabra321 --post c]
```

Using Provisioning Parameters

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

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

General Purpose Parameters

The general purpose parameters GPP_* are used as free string registers when configuring the IP Telephony device 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 upper-case macro names (A through P) are sufficient to identify the contents of GPP_A through GPP_P. Also, the two-letter upper-case macro names SA through SD identify GPP_SA through GPP_SD as a special case when used as arguments of the **key** URL option.

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

Enables

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

In addition, the Resync_From_SIP parameter controls requests for resync operations via a SIP NOTIFY event sent from the service provider proxy server to the IP Telephony device. If enabled, the proxy can request a resync by sending a SIP NOTIFY message containing the Event: resync header to the device.

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

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

When enabling Resync_On_Reset, the device introduces a random delay following the boot-up sequence before actually performing the reset. The delay is a random time up to the value specified in Resync_Random_Delay (in seconds). In a pool of IP Telephony devices, all of which are simultaneously powered up, this introduces a spread in the times at which each unit initiates a resync request to the provisioning server. This feature can be useful in a large residential deployment, in the case of a regional power failures.

Triggers

The IP Telephony device allows you to resync at specific intervals or at a specific time.

Resyncing at Specific Intervals

The IP Telephony device 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. In case a voice line is active when a resync is due, the IP Telephony device delays the resync procedure until the line becomes idle again. However, it waits no longer than Forced_Resync_Delay (seconds). A resync might cause configuration parameter values to change. This, in turn, causes a firmware reboot and terminates any voice connection active at the time of the resync.

If a resync operation fails because the IP Telephony device was unable to retrieve a profile from the server, if the downloaded file is corrupt, or an internal error occurs, the device tries to resync again after a time specified in Resync_Error_Retry_Delay (seconds). If Resync_Error_Retry_Delay is set to 0, the device does not try to resync again following a failed resync attempt.

When upgrading, 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 of these parameters can be programmed with a conditional expression (which undergoes macro expansion). If the condition in any of these parameters evaluates to true, a resync operation is triggered, as though the periodic resync timer had expired.

The following example condition triggers a resync if Line 1 failed to register for more than 5 minutes (300 seconds), and at least 10 minutes (600 seconds) have elapsed since the last resync attempt.

```
$REGTMR1 gt 300 and $PRVTMR ge 600
```

Resyncing at a Specific Time

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

To avoid simultaneously flooding the server with resync requests from multiple phones set to resync at the same time, the phone triggers the resync up to ten 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

Profile resyncs and upgrades provide for automatic retries in case of failure, in addition to periodic configuration updates. Time intervals are specified by using three parameters, which are usually specified as a specific interval duration, in seconds. Starting with firmware version 3, these parameters allow the application-level (macro time scale) retry schedule to be configured. These provisioning parameters are:

- Resync_Periodic
- Resync_Error_Retry_Delay
- Upgrade_Error_Retry_Delay

These parameters accept a single delay value (seconds). The new extended syntax allows for a comma-separated list of consecutive delay elements. Each delay element consists of a deterministic delay value, optionally followed by a plus sign and an additional numeric value that bounds a random extra delay. The last element in the sequence is implicitly repeated forever. For example,

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

In this example, the IP Telephony device periodically resyncs every two hours. In case of a resync failure, the device retries in 30 minutes, then again in 1 more hour, then after two more hours, and then after four more hours, continuing at four-hour intervals until it successfully resyncs.

The following is another example:

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

In this example, the device periodically resyncs every hour (plus an additional random delay of up to 10 minutes). In case of resync failure, the device retries in 30 minutes (plus up to five minutes more).

If it fails again, it waits an additional hour (plus up to 10 minutes). If again unsuccessful, it waits two more hours (plus up to 15 minutes), and so also thereafter, until it successfully resyncs.

The following 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 it still fails, it subsequently retries every four to five hours, until it succeeds.

Profile Rules

The IP Telephony device provides multiple remote configuration profile parameters (Profile_Rule*). This means that each resync operation can retrieve multiple files, potentially managed by different servers.

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 IP Telephony devices in a deployment, while a separate file is provided that is unique for each account. Encryption keys and certificate information could be supplied by still another profile, stored on a separate server.

Whenever a resync operation is due, the IP Telephony device 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 being retrieved from a remote provisioning server, possibly updating 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 a resync again.

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 of the alternatives except the last one must provide a conditional expression. This expression is evaluated and 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 located at the specified URL, and update the internal parameters accordingly.

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

The following are some examples of valid programming for a single Profile_Rule* parameter.

The following example resyncs unconditionally to the profile at the specified URL, performing an HTTP GET request to the remote provisioning server.

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

In the following 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, transmitting the contents of the macro expanded GPP_A (which may provide additional information on the state of the device).

```
($REGTMR1 eq 0)? http://p.tel.com/has-reg.cfg
| [--post a] http://p.tel.com/lost-reg?
```

In the following example, the device resyncs to the same server, but 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 the following 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 the following example, the profile returned by the server is assumed to contain XML element tags that need to be remapped to proper parameter names by the aliases map stored in GPP_B.

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

A resync is typically considered unsuccessful if a requested profile is not received from the server. This default behavior can be overridden by the parameter Resync_Fails_On_FNF. If Resync_Fails_On_FNF is set to No, then 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.

Report Rule

The IP Telephony device provides a mechanism for reporting its current internal configuration to the provisioning server. This is useful for development and debugging. The report syntax is similar to the Open format profile. All provisionable parameters are included, except for the values of passwords, keys, and the GPP_SA to GPP_SD parameters, which are not shown.

The Report_Rule parameter is evaluated like a profile rule parameter. In other words, it accepts a URL, optionally qualified with a bracketed expression. The URL specifies the target destination for the report and an encryption key can be included as an option.

The URL scheme can be TFTP, HTTP, or HTTPS. When using TFTP, the operation performed is TFTP PUT. In the case of HTTP and HTTPS, the operation performed is HTTP POST.

If an encryption key is specified, the report is encrypted using 256-bit AES in CBC mode. The encrypted report can be decrypted with the following OpenSSL (or equivalent) command:

```
openssl enc -d -aes-256-cbc -k secretphrase -in rep.xml.enc -out
rep.xml
```

The following is an example of the corresponding Report_Rule configuration:

```
[ --key secretphrase ] http://prov.serv.net/spa/$MA/rep.xml.enc
```

Once the report rule is configured, an actual report can be generated and transmitted by sending the device a SIP NOTIFY message, with the Event: report type. The SIP NOTIFY request is handled like other SIP notifies, with the device requiring authentication from the requesting server before honoring the request to issue a report. Each SIP NOTIFY report request generates one attempt to transmit the report. Retries are not supported.

In addition to reporting the current internal configuration to the provisioning server, the Report Rule has an option for triggering the reporting of configuration changes (deltas) to the server since the last resync, reboot, or upgrade.

The syntax of this option is:

Report Rule: [--delta] *URL*

Where *URL* is the path to where the report is stored on the server.

For example, to store delta configuration changes in a file with a name like SPA504G_*<MAC>*_*<serial#>*.xml, do one of the following:

- On the phone Web GUI, set the **Report Rule** field on the **Configuration Profile** page (Voice tab > Provisioning tab > Configuration Profile) to:

  ```
  [--delta] http://reportTargetServer/reportPath/$PN_$MA_$SN.xml
  ```

- Add the following to your provisioning file:

  ```
  <Report_Rule ua="na">[ --delta ]
  http://reportTargetServer/reportPath/$PN_$MA_$SN.xml
  </Report_Rule>
  ```

Upgrade Rule

The IP Telephony device provides one configurable remote upgrade parameter, Upgrade_Rule. This parameter accepts a syntax similar to the profile rule parameters. URL options 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 specified by the URL. Subsequently, it does not attempt to upgrade again until either the rule itself is modified or the effective combination of scheme + server + port + filepath is changed, following macro expansion and evaluation of the rule.

In order to attempt a firmware upgrade, the device disables audio at the start of the procedure, and reboots at the end of the procedure. For this reason, an upgrade driven by the contents of Upgrade_Rule is only automatically initiated by the device if any voice line is currently inactive.

For example,

```
http://p.tel.com/firmware/spa021025.bin
```

In this example, the Upgrade_Rule upgrades the firmware to the image stored at the indicated URL. The following is another example:

```
("$F" ne "beta-customer")? http://p.tel.com/firmware/spa021025.bin
| http://p.tel.com/firmware/spa-test-0527s.bin
```

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 with respect to firmware revision number. This can be useful as a customization option. If a valid firmware revision number is configured in the parameter Downgrade_Rev_Limit, the device rejects upgrade attempts for firmware versions earlier than the specified limit.

Data Types

The data types used with configuration profile parameters are as follows:

- 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.
- Sig<n>—Signed n-bit value. It can be specified in decimal or hex format. Negative values must be preceded by a "-" sign. A + sign before positive value is optional.
- Str<n>—A generic string with up to n non-reserved characters.
- Float<n>—A floating point value with up to n decimal places.
- Time<n>—Time duration in seconds, with up to n decimal places. Extra decimal places specified are ignored.
- PwrLevel—Power level expressed in dBm with 1 decimal place, such as –13.5 or 1.5 (dBm).
- Bool—Boolean value of either "yes" or "no."
- {a,b,c,...}—A choice among a, b, c, ...
- IP—IP Address in the form of x.x.x.x, where x between 0 and 255. For example 10.1.2.100.
- Port—TCP/UDP Port number (0-65535). It can be specified in decimal of hex format.

- UserID—User ID as appeared in a URL; up to 63 characters.
- FQDN—Fully Qualified Domain Name, such as “sip.Cisco.com:5060”, or “109.12.14.12:12345”. It can contain 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. It can contain up to 39 characters.
- ActCode—Activation code for a supplementary service, such as *69. It can contain up to 7 characters.
- PhTmplt—A phone number template. Each template may contain one or more patterns separated by a “,”. White space at the beginning of each pattern is ignored. “?” and “*” represent wildcard characters. To represent literally use %xx. For example, %2a represents *. It can contain up to 39 characters. Examples: “1408*, 1510*”, “14081234????, 555?1.”.
- RscTmplt—A template of SIP Response Status Code, such as “404, 5*”, “61?”, “407, 408, 487, 481”. It can contain up to 39 characters.
- CadScript—A mini-script 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* and 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
```

- FreqScript—A mini-script that specifics the frequency and level parameters of a tone. Up to 127 characters. Syntax: $F_1@L_1[,F_2@_{L2}[,F_3@L_3[,F_4@L_4[,F_5@L_5[,F_6@L_6]]]]]$, where F_1–F_6 are frequency in Hz (unsigned integers only) and L_1–L_6 are corresponding levels in dBm (with up to 1 decimal places). White spaces before and after the comma are allowed (but not recommended).

  Example 1—Call Waiting Tone:

```
440@-10

Number of Frequencies = 1
   Frequency 2 = 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
```

- ToneScript—A mini-script that specifies the frequency, level and cadence parameters of a call progress tone. 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 $fi,j = n_1[+n_2]+n_3[+n_4[+n_5[+n_6]]]]$ and $1 < n_k < 6$ indicates which of the frequency components given in the FreqScript 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
```

Example 3—SIT tone:

```
985@-16,1428@-16,1777@-16;20(.380/0/1,.380/0/2,.380/0/3,0/4/0)
Number of Frequencies = 3
Frequency 1 = 985 Hz at -16 dBm
Frequency 2 = 1428 Hz at -16 dBm
Frequency 3 = 1777 Hz at -16 dBm
Number of Cadence Sections = 1
Cadence Section 1: Section Length = 20s
Number of Segments = 4
Segment 1: On=0.38s, Off=0s, with Frequency 1
Segment 2: On=0.38s, Off=0s, with Frequency 2
Segment 3: On=0.38s, Off=0s, with Frequency 3
Segment 4: On=0s, Off=4s, with no frequency components
Total Tone Length = 20s
```

- ProvisioningRuleSyntax—Scripting syntax used to define configuration resync and firmware upgrade rules.
- DialPlanScript—Scripting syntax used to specify Line 1 and Line 2 dial plans.


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 IP Telephony device 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 IP Telephony device.
- 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.

SPA Profile Compiler

The SPA Profile Compiler (SPC) is available from Cisco for the Win32 environment and Linux-i386-elf environment. For the OpenBSD environment, the SPC is made available on a case-by-case basis. Contact Cisco for more information.

The SPC can:

- Generate a sample Plain-text format or Open format profile. (See **Generating Sample Configuration Files, page 50**.)
- Compile a Plain-text format profile containing parameter-value pairs into a binary, encrypted CFG file.

Downloading the SPC

The SPC is specific to the device and to the firmware release. It might be necessary to download multiple versions of the SPC to create configuration profiles for different model devices. To download a SPC, do the following:

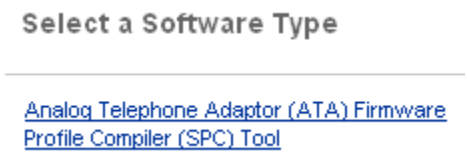
STEP 1 Go to Cisco.com.

STEP 2 Enter the device model number in the search box, and click **Go**.

STEP 3 In the **Filter Results By** list on the left side of the Search Results page, find the **Task**, and click **Download Software**. The page refreshes.

STEP 4 Click the **Download Software...** link for the device (usually one of the first entries in the filtered list). The Download Software... page appears. (If the Select A Product page appear, select the desired device and continue.)

STEP 5 Choose **Profile Compiler (SPC) Tool** in the Select Software Type section.



STEP 6 Click **Download Now** to choose the latest release of the firmware.

STEP 7 Follow the instructions on the screen to complete the download process.

NOTE In the examples in this section, the SPC executable name is represented by `spc`.

Generating Sample Configuration Files

SPC can be used to generate sample configuration source files for both Plain-text format and Open format. The features controlled through the configuration files corresponds to the IP Telephony device model and firmware release. Before generating a sample file, verify that you are using the correct SPC for your device and firmware version.

Syntax (Windows)

```
spcexecuteable --sample-profile outputfile.name
spcexecuteable --sample-xml outputfile.name
```

Where:

Parameter	Description
spcexecuteable	SPC executable for the device. For example, the executable for a SPA525G firmware version 7.4.9 in a Windows environment is `spa525g-sccp-7-4-9-spc-win32-i386`.
--sample-profile **--sample-xml**	Flag to indicate the file type. The `--sample-profile` flag outputs a profile in Plain-text format. The `--sample-xml` flag outputs a profile in Open format.
outputfile.name	File name of the output profile.

Usage Guidelines

IP Telephony devices with firmware versions before 2.0.6 do not support Open format profiles.

Example Commands

To generate a sample plain text configuration profile to be used as source file for a SPA525G in a Windows environment:

```
C:\SPC>spa525g-sccp-7-4-9-spc-win32-i386 --sample-profile plaintext.txt
```

Displays the message: `spc: generated SPA525G-SCCP sample configuration plaintext.txt`

And produces the output file `plaintext.txt`.

To generate an Open format profile to be used as source file for a SPA525G running firmware version 2.0.6 or above in a Windows environment:

```
C:\SPC>spa525g-sccp-7-4-9-spc-win32-i386 --sample-xml xml.txt
```

Displays the message: `spc: generated SPA525G-SCCP sample configuration xml.txt`

And produces the output file `xml.txt`.

Compiling Profiles by using SPC

SPC can use different types of encryption to generate configuration files or none at all.

- Generic files are binary, non-targeted, and without an explicit key. A *scramble* option can be used to encrypt a generic file with a randomizing argument.
- Targeted files are encrypted without an explicit key. The SPC uses the MAC address of the target IP Telephony device to encrypt the file, and only that device can decode it.
- Explicit files are key-based and use AES or RC4 encryption to create the CFG file.

Any combination of scrambling, targeting, or explicit-key encrypting can be applied to a configuration file.

Syntax

```
spcexecuteable --scramble inputfile.txt outputfile.cfg
spcexecuteable --target inputfile.txt outputfile.cfg
```

spcexecuteable **--rc4** *inputfile*.txt *outputfile.cfg*
spcexecuteable **--aes** *inputfile*.txt *outputfile.cfg*

Where:

Parameter	Description
spcexecuteable	SPC executable for the device. For example, the executable for a SPA525G firmware version 7.4.9 in a Windows environment is spa525g-sccp-7-4-9-spc-win32-i386.
--scramble **--target** **--rc4** **--aes**	Flag to indicate the profile type. The subsections in this section describe these profile types.
inputfile.txt	File name of the input profile.
outputfile.cfg	File name of the output profile.

Generic Profile and the Scramble Option

A generic, non-targeted profile is accepted as valid by any IP Telephony device that resyncs to it. The following example command generates a generic CFG file:

```
>spcexecutable spa962.txt spa962.cfg
```

Where spc is the executable. The Plain-text format input file is spa962.txt. The binary output is spa962.cfg.

The **--scramble** option performs encryption that does not require the explicit transmission of a key to the target device. It does require one randomizing argument. For example,

```
>spcexecutable --scramble SecretPhrase spa962.txt spa962.cfg
```

Where spc is the executable. The encryption is indicated by the *--scramble* option. A randomizing argument is represented by the *SecretPhrase*. The Plain-text format input file is spa962.txt. The binary output is spa962.cfg.

Targeted Profile

The **--target** option encrypts the profile without the need to explicitly transmit a key, but only the target IP Telephony device can decode it. Targeted CFG files provide a basic level of security. This command uses the MAC address of the target device as an argument:

```
>spcexecutable --target 000e08aabbcc spa962.txt spa962.cfg
```

Where `spc` is the executable. The encryption is indicated by the *`--target`* option. The MAC address argument is the MAC address `000e08aabbcc`. The Plain-text format input file is `spa962.txt`. The binary output is `spa962.cfg`.

This example, only the IP Telephony device with the MAC address 000e08aabbcc is able to decrypt and resynch to the spa962.cfg profile. If any other IP Telephony device attempts to resync to this file, the device rejects the file as unreadable.

Explicit Key

Explicit key-based encryption requires that the key used to encrypt the file be preprovisioned in the target device, so that the file can be decoded.

Two algorithms are available for this type of encryption:

- RC4 (**--rc4**)
- AES (**--aes**)

The key can be specified either explicitly as a hexadecimal digit sequence (**--hex-key**) or by hashing a secret phrase (**--ascii-key**). With the **--hex-key** option, the key can be up to 256 bits in length. With the **--ascii-key** option, the generated key is 128 bits.

The following example commands illustrate explicit key-based encryption:

```
>spcexecutable --rc4 --ascii-key apple4sale spa962.txt spa962.cfg
>spcexecutable --aes --ascii-key lucky777 spa962.txt spa962.cfg
>spcexecutable --aes --ascii-key "my secret phrase" spa962.txt
spa962.cfg
>spcexecutable --aes --hex-key 8d23fe7...a5c29 spa962.txt spa962.cfg
```

Where `spc` is the executable. The encryption is indicated by the *`--rc4`* or the `--aes` option. Which key type, `--ascii-key` or `--hex-key`. An argument in the form of an ASCII string or hexadecimal digits. The Plain-text format input file is `spa962.txt`. The binary output is `spa962.cfg`.

Status Messages

After each compilation, SPC prints a final status message. Syntax error messages are also printed if a compilation is not successful.

The status and error messages can be suppressed by using the **--quiet** command line option:

```
>spcexecutable --quiet filename.txt filename.cfg
```

Messages can be redirected to a file by using the **--log** *file_name* option:

```
>spcexecutable --log prov.log filename.txt filename.cfg
```

When the messages are redirected, the SPC command is also printed in the log file, preceded by a timestamp.

3

In-house Preprovisioning and Provisioning Servers

Cisco Small Business IP Telephony devices, other than RC units, are preprovisioned by the service provider with a profile. That preprovision profile can range from a a limited set of parameters that resynchronizes the IP Telephony device to another profile with a complete set of parameters delivered by remote server. Or, it can be a complete set of parameters. By default, the IP Telephony device resynchronizes on power up and at intervals configured in the profile. When the user connects the IP Telephony device 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 many ways. This chapter describes the features and functionality available when preprovisioning Cisco Small Business IP Telephony devices in-house and provisioning them remotely:

Server Preparation and Software Tools

The examples presented 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 establishes proper server operation, independent of the interaction with Cisco Small Business VoIP devices.

Cisco also recommends the installation of the following software tools:

- To generate configuration profiles, it is useful to install the open source gzip compression utility.
- For profile encryption and HTTPS operations, install the open source OpenSSL software package.
- To test the dynamic generation of profiles and one-step remote provisioning using HTTPS, a scripting language with CGI scripting support, such as open source Perl language tools, is recommended.
- To verify secure exchanges between provisioning servers and Cisco Small Business voice devices, install an Ethernet packet sniffer (such as the freely downloadable Ethereal/Wireshark). Capture an Ethernet packet trace of the interaction between the IP Telephony device and the provisioning server by running 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.

In-House Device Preprovisioning

With the Cisco factory default configuration, an IP Telephony device automatically tries to resync to a profile on a TFTP server. The information regarding the profile and TFTP server configured for preprovisioning is delivered to the device by a managed DHCP server on a LAN. The service provider connects each new IP Telephony device that LAN and the IP Telephony device automatically resyncs to the local TFTP server, initializing its internal state in preparation for deployment. This preprovisioning profile typically includes the URL of a remote provisioning server that will keep the device updated after it is deployed and connected to the customer network.

The preprovisioned device barcode can be scanned to record its MAC address or serial number before the IP Telephony device is shipped to the customer. This information can be used to create the profile to which the IP Telephony device will resynchronize.

Upon receiving the IP Telephony device, the customer connects it to the broadband link. On power-up the IP Telephony device contacts the provisioning server through the URL configured through preprovisioning to for its resync and updates the profile and firmware as necessary.

Provisioning Server Setup

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

TFTP Provisioning

Cisco Small Business voice devices support TFTP for both provisioning resync and firmware upgrade operations. Once devices are deployed remotely, HTTP is recommended for provisioning as it offers greater reliability, given NAT and router protection mechanisms. TFTP is useful for the in-house preprovisioning of a large number of un-provisioned devices. See **In-House Device Preprovisioning, page 56** for more information.

The IP Telephony device 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 when it is connected to a LAN and powered up.

The Profile_Rule provided with the factory default configuration is */device*.cfg. For example, on a SPA962 the filename is spa962.cfg. If the device has the factory default profile, when powered up it resyncs to this file on the local TFTP server specified by DHCP option 66. (The filepath is relative to the TFTP server virtual root directory.)

Remote Endpoint Control and NAT

The IP Telephony device accesses the Internet through a router by using network address translation (NAT). 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 (1) a dynamic external (public) IP address from your service, (2) a computer running STUN server software, and (3) an edge device with an asymmetric NAT mechanism.

HTTP Provisioning

The IP Telephony device behaves like a browser requesting 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.

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

Alternatively, the requested URL can invoke a CGI script (using the GET method). The configuration profile is generated dynamically by querying a customer database and producing the profile on-the-fly.

When CGI handles resync requests, the IP Telephony device 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 store the status information for later analysis and tracking.

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

For example, the following example is the User-Agent request field from a SPA962:

```
User-Agent: cisco/SPA-962-2.0.5 (88012BA01234)
```

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

HTTPS Provisioning

For increased security managing remotely deployed units, the IP Telephony device supports HTTPS for provisioning. Each IP Telephony device carries a unique SLL Client Certificate (and associated private key), in addition to a Sipura CA server root certificate. The latter allows the IP Telephony device 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 an IP Telephony device 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 IP Telephony device:

```
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 IP Telephony device tests the server IP address against a DNS lookup of the server name specified in the server certificate.

A certificate signing request can be generated by using the OpenSSL utility. 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.

Cisco also provides a Sipura CA Client Root Certificate to the service provider. This root certificate certifies the authenticity of the client certificate carried by each IP Telephony device.

The unique client certificate offered by each device during an HTTPS session carries identifying information 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 a SPA962 client certificate subject field shows these elements:

```
OU=SPA-962, 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 using HTTPS, but are only able to supply a generic client certificate if requested to do so by the server. This generic certificate contains the following information in the identifying fields:

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

To determine if an IP Telephony device 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 verify the client certificate by using the Sipura CA Client Root Certificate supplied by Cisco. It can then provide the certificate information to a CGI for further processing.

The location for storing certificates might vary. For example, on an Apache installation the file paths for storing 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
```

Refer to the documentation provided for a HTTPS server for specific information.

Firmware release 2.0.6 and higher supports the following cipher suites for SSL connection to a server by using HTTPS.

Table 1 Cipher Suites Supported for Connecting to an HTTPS Server

Numeric Code	Cipher Suite
0x0039	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
0x0035	TLS_RSA_WITH_AES_256_CBC_SHA
0x0033	TLS_DHE_RSA_WITH_AES_128_CBC_SHA
0x002f	TLS_RSA_WITH_AES_128_CBC_SHA
0x0005	TLS_RSA_WITH_RC4_128_SHA
0x0004	TLS_RSA_WITH_RC4_128_MD5
0x0062	TLS_RSA_EXPORT1024_WITH_RC4_56_SHA
0x0060	TLS_RSA_EXPORT1024_WITH_RC4_56_MD5
0x0003	TLS_RSA_EXPORT_WITH_RC4_40_MD5

Redundant Provisioning Servers

The provisioning server can be specified as an IP address or as a fully qualified domain name (FQDN). The use of a FQDN facilitates the deployment of redundant provisioning servers. When the provisioning server is identified through a FQDN, the IP Telephony device 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 IP Telephony device continues to process A-records until a server responds. If no server associated with the A-records responds, the IP Telephony device logs an error to the syslog server.

Syslog Server

If a syslog server is configured on the IP Telephony device (using the <Syslog_Server> or <Debug_Server> parameters), the resync and upgrade operations log 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 themselves are configured in the following parameters:

- For profile resync:
  - Log_Resync_Request_Msg
  - Log_Resync_Success_Msg
  - Log_Resync_Failure_Msg
- For firmware upgrades:
  - Log_Upgrade_Request_Msg
  - Log_Upgrade_Success_Msg
  - Log_Upgrade_Failure_Msg

These parameters are macro expanded into the actual syslog messages.

As indicated in the lower half of the diagram, a Cisco Small Business Client Certificate Root Authority signs each unique certificate. The corresponding root certificate is made available to service providers for client authentication purposes.

4

Provisioning Examples

This chapter provides example procedures for transferring configuration profiles between the IP Telephony device and the provisioning server:

- **Basic Resync, page 63**
- **Secure HTTPS Resync, page 71**
- **Profile Management, page 79**
- **Plain-Text Profile Resync, page 83**

For information about creating configuration profiles, refer to **Chapter 2, "Creating Provisioning Scripts."**

Basic Resync

This section demonstrates the basic resync functionality of Cisco Small Business VoIP devices.

TFTP Resync

The IP Telephony device supports multiple network protocols for retrieving configuration profiles. The most basic profile transfer protocol is TFTP (RFC1350). TFTP, widely used for the provisioning of network devices within private LAN networks. Although not recommended for the deployment of remote endpoints across the Internet, it can be convenient for deployment within small organizations, for in-house preprovisioning, and for development and testing. See **"In-House Device Preprovisioning" section on page 56** for more information on in-house preprovisioning. In this exercise, an IP Telephony device profile is modified after downloading a file from a TFTP server.

Exercise

STEP 1 Within a LAN environment connect a PC and an IP Telephony device to a hub, switch, or small router.

STEP 2 Connect an analog phone to the Phone 1 port of the IP Telephony device.

STEP 3 On the PC, install and activate a TFTP server.

STEP 4 Using a text editor, 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 5 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 by requesting the basic.txt file by using a TFTP client other than the IP Telephony device. Preferably, use a TFTP client that is running on a separate host from the provisioning server.

STEP 6 Using an analog phone, obtain the IP address of the IP Telephony device (IVR menu ****** 110 #**).

If the configuration has been modified since it was manufactured, perform factory reset on the phone by using the IVR RESET option (******** 73738#).

STEP 7 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 8 Select the Provisioning tab, and inspect the values of the general purpose parameters GPP_A through GPP_P. These should be empty.

STEP 9 Resync the test IP Telephony device to the `basic.txt` configuration profile by opening the resync URL in a web browser window.

Assuming the IP address of the TFTP server is 192.168.1.200 the command should be similar to this example:

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

When IP Telephony device 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. It then parses the downloaded file and updates the GPP_A parameter with the value 12345678.

STEP 10 Verify that the parameter was correctly updated by refreshing the admin/advanced page on the PC web browser and selecting the Provisioning tab on that page.

The GPP_A parameter should now contain the value 12345678.

Logging with syslog

The IP Telephony device 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/advanced, System tab, Syslog_Server parameter). Configure the syslog server IP address into the device and observe the messages generated during the remaining exercises.

Exercise

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 ua="na">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** exercise.

The device generates two syslog messages during the resync. The first indicates that a request is in progress. The second marks success or failure of the resync.

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

```
SPA-962 00:0e:08:ab:cd:ef -- Requesting resync tftp://
192.168.1.200/basic.txt
SPA-962 00:0e:08:ab:cd:ef -- Successful 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 setting the Debug_Level to a value between 0 and 3 (3 being the most verbose):

```
<Debug_Server ua="na">192.168.1.210</Debug_Server>
<Debug_Level ua="na">3</Debug_Level>
```

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

- Log_Resync_Request_Msg
- Log_Resync_Success_Msg
- Log_Resync_Failure_Msg.

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

Automatic Device Resync

A IP Telephony 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 IP Telephony device 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.

Exercise

STEP 1 Using a web browser, open the admin/advanced page Provisioning tab.

STEP 2 Define the Profile_Rule parameter. The example assumes a TFTP server IP address of 192.168.1.200:

```
<Profile_Rule ua="na">tftp://192.168.1.200/basic.txt</
Profile_Rule>
```

STEP 3 In the Resync_Periodic parameter enter a small value for testing, such as **30** seconds:

```
<Resync_Periodic ua="na">30</Resync_Periodic>
```

STEP 4 Click **Submit all Changes**.

With the new parameter settings, the IP Telephony device resyncs to the configuration file specified by the URL twice a minute.

STEP 5 Observe the resulting messages in the syslog trace (as described in the **Logging with syslog** section).

STEP 6 Ensure that the Resync_On_Reset parameter is set to **yes**:

```
<Resync_On_Reset ua="na">Yes</Resync_On_Reset>
```

STEP 7 Power cycle the IP Telephony device. The IP Telephony device resyncs to the provisioning server when it is power-cycled.

If the resync operation fails for any reason, such as if the server is not responding, the unit waits the number of seconds configured in Resync_Error_Retry_Delay before attempting to resync again. If Resync_Error_Retry_Delay is zero, the IP Telephony device does not try to resync after a failed resync attempt.

STEP 8 (Optional) Set the value of Resync_Error_Retry_Delay is set to a small number, such as **30**:

```
<Resync_Error_Retry_Delay ua="na">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 IP Telephony device 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 IP Telephony device, in turn, must be configured to resync to its own profile according a predetermined profile naming convention.

The profile URL syntax can include identifying information specific to each IP Telephony device, 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 being applied to the IP Telephony device. 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** 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 available for macro expansion, see the **"Macro Expansion Variables" section on page 95**.

In this exercise, a profile specific to a IP Telephony device is provisioned on a TFTP server.

Exercise

STEP 1 Obtain the MAC address of the IP Telephony device 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** exercise) to a new file named **`spa_`***`macaddress`***`.cfg`** (replacing *`macaddress`* with the MAC address of the IP Telephony device). For example:

```
spa_000e08abcdef.cfg
```

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

STEP 4 Open the admin/advanced page Provisioning tab.

STEP 5 Enter `tftp://192.168.1.200/spa$MA.cfg` in the Profile_Rule parameter:

```
<Profile_Rule ua="na">
  tftp://192.168.1.200/spa$MA.cfg
</Profile_Rule>
```

STEP 6 Click **Submit All Changes**. This causes an immediate reboot and resync.

When the next resync occurs, the IP Telephony device 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 IP Telephony device 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 used for TFTP. If a standard web browser can retrieve a profile from a your HTTP server, the IP Telephony device should be able to do so as well.

Exercise

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** exercise) onto the virtual root directory of the installed server.

STEP 3 Verify proper server installation (and file access to basic.txt) by accessing the profile by using a web browser.

STEP 4 Modify the Profile_Rule of the test IP Telephony device 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 ua="na">
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 sent by the IP Telephony device. The periodic resyncs should now be obtaining the profile from the HTTP server.

STEP 7 In the HTTP server logs, observe how information identifying the test IP Telephony device appears in the log of user agents.

This should include the manufacturer, product name, current firmware version, and serial number.

URL Resolution by using Macro Expansion

Subdirectories with multiple profiles on the server is 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 IP Telephony device performs a DNS lookup to resolve the name.
- A non-standard server port 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 spa962.cfg, in the server subdirectory /cisco/config, from the TFTP server running on host prov.telco.com listening for a connection on port 6900:

```
<Profile_Rule ua="na">
/tftp://prov.telco.com:6900/cisco/config/spa962.cfg
</Profile_Rule>
```

A profile for each IP Telephony device 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 IP Telephony device 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

This section demonstrates the mechanisms available on the IP Telephony device for resyncing by using a secure communication process. It includes the following topics:

Basic HTTPS Resync

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

- IP Telephony device can authenticate the provisioning server
- provisioning server can authenticate the IP Telephony device
- confidentiality of information exchanged between the IP Telephony device and the provisioning server is ensured.

SSL generates and exchanges secret (symmetric) keys for each connection between the IP Telephony device and the server, using public/private key pairs preinstalled in the IP Telephony device and the provisioning server.

On the client side, the IP Telephony device 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 used for HTTP or TFTP. If a standard web browser can retrieve a profile from a your HTTPS server, the IP Telephony device should be able to do so as well.

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

Exercise

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** exercise) 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 supplied by the server.

The browser probably does not recognize it as valid unless the browser has been preconfigured to accept Cisco as a root CA. However, the IP Telephony devices 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 ua="na">
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 sent by the IP Telephony device.

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 IP Telephony device subnet to verify that the packets are encrypted.

In this exercise, client certificate verification was not enabled. The connection between IP Telephony device 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** 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 IP Telephony device before accepting a connection request.

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

NOTE Both basic and digest authentication are supported on SPA500 Series phones running firmware version 7.4.9c and higher.

Exercise

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** exercise.

STEP 3 Restart the HTTPS server and observe the syslog trace from the IP Telephony device.

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 IP Telephony device 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 containing the profile.

With client authentication enabled, only a IP Telephony device with a MAC address matching a valid client certificate can request the profile from the provisioning server. A request from an ordinary browser or other unauthorized device is rejected by the server.

HTTPS Client Filtering and Dynamic Content

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

The HTTPS server makes the certificate information available to CGI scripts (or compiled CGI programs) 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.

Exercise

STEP 1 Install Perl on the host 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 (as in /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 Open the admin/advanced page, Provisioning tab.

STEP 9 Verify that the GPP_D parameter contains the information captured by the script.

This information contains the product name, MAC address, and serial number if the test device carries a unique certificate from the manufacturer, or else generic strings if it is a unit manufactured before firmware release 2.0.

A similar script could be used to determine information about the resyncing device and then provide it with appropriate configuration parameter values.

HTTPS Certificates

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

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

How HTTPS Works

HTTPS encrypts the communication between a client and a server, 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 single secret key is shared by a client and a server over a secure channel protected by Public/Private key encryption.

Messages encrypted by the secret key can only be decrypted using the same key. HTTPS supports a wide range of symmetric encryption algorithms. The IP Telephony device 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 is an essential capability 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 IP Telephony device supports the RSA algorithm for public/private key cryptography.

SSL Server Certificates

Each secure provisioning server is issued a SSL server certificate, directly signed by Cisco. The firmware running on the IP Telephony device 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 IP Telephony device, or any attempt to spoof the provisioning server. Without such protection, an attacker might be able to reprovision the IP Telephony device, to gain configuration information, or to use a different VoIP service.

Client Certificates

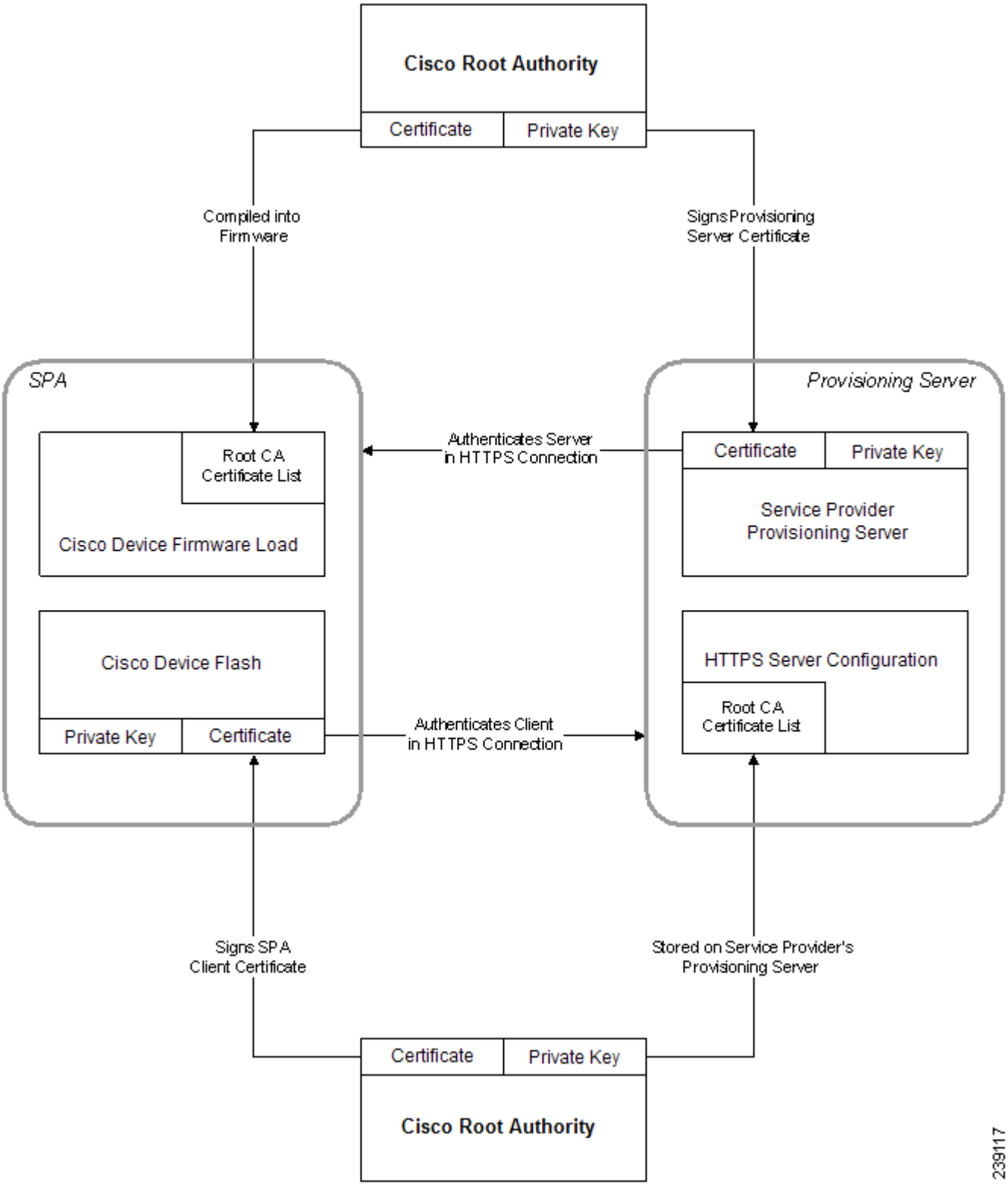
In addition to a direct attack on an IP Telephony device, an attacker might attempt to contact a provisioning server by using a standard web browser or another HTTPS client to obtain the configuration profile from the provisioning server. To prevent this kind of attack, each IP Telephony device also carries a unique client certificate, signed by Cisco, including identifying information about each individual endpoint. A certificate authority root certificate 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 IP Telephony device and its provisioning server. The **"Certificate Authority Flow"** figure 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, allowing the IP Telephony device to authenticate authorized provisioning servers.

Certificate Authority Flow



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 all parameters are individually specified by the profile. To reduce the load on the provisioning server, the IP Telephony device supports compression of the XML file, by using the deflate compression format supported by the gzip utility (RFC 1951).

NOTE Compression must precede encryption for the IP Telephony device 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 IP Telephony device expects the file to contain a valid gzip header.

Additional information on compression is provided in **Open Profile Compression, page 23**.

Exercise

STEP 1 Install gzip on the local PC.

STEP 2 Compress the `basic.txt` configuration profile (described in the **TFTP Resync** exercise) 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 IP Telephony device.

Upon resync, the new file is downloaded by the IP Telephony device and used to update its parameters.

Profile Encryption by using OpenSSL

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

The IP Telephony device supports symmetric key encryption by using the 256-bit AES algorithm. This encryption can be performed by using the open source OpenSSL package. Additional information on encryption is provided in **Open Profile Encryption by using AES, page 24**.

Exercise

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** exercise), 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 created in **Open Profile gzip Compression** 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 IP Telephony device 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 IP Telephony device.

On resync, the new file is downloaded by the IP Telephony device and used to update its parameters.

Partitioned Profiles

An IP Telephony device downloads multiple separate profiles during each resync. This allows managing different kinds of profile information on separate servers and maintaining common configuration parameter values separate from account specific values.

Exercise

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 you can 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 ua="na">tftp://192.168.1.200/basic2.txt
</Profile_Rule_B>
```

STEP 4 Click **Submit All Changes**.

The IP Telephony device 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.

Parameter Name Aliases

When generating an XML profile for the IP Telephony device, it might be convenient to assign names to certain configuration parameters that are different from the canonical names recognized by the IP Telephony device. For example, a customer account database might generate XML element tags for a customer telephone number and SIP registration password with names, such as SIP-number and SIP-password. These names can be mapped to the canonical names (User_ID_1_ and Password_1_) before being applied to Line 1.

In many instances, the back-end provisioning solution used by the service provider can perform this mapping. However, the IP Telephony device itself can remap the parameter names internally. To do this, an alias map is defined and stored in one of the general purpose provisioning parameters. Then, the profile rule which invokes the resync is directed to remap the non-canonical XML elements as specified by the alias map.

Exercise

STEP 1 Generate a profile named customer.XML containing the proprietary customer-account XML form indicated in the following example:

```
<customer-account>
  <SIP-number> 17775551234</SIP-number>
  <SIP-password> 512835907884</SIP-password>
</customer-account>
```

STEP 2 Store the profile in the TFTP server virtual root directory.

STEP 3 Open the web interface on the device to the admin/advanced page, Provisioning tab, and edit GPP_A to contain the alias map (do not enter new lines through the web interface, instead simply enter each alias consecutively):

```
/customer-account/SIP-number = /flat-profile/User_ID_1_ ;
/customer-account/SIP-password = /flat-profile/Password_1_ ;
```

STEP 4 Edit the Profile_Rule to point to the new XML profile, and specify the alias map as a URL option, as follows:

```
[--alias a ] tftp://192.168.1.200/customer.xml
```

STEP 5 Click **Submit All Changes**.

When the IP Telephony device resyncs, it receives the XML profile, remaps the elements, as indicated by the alias map, and populates the User_ID_1_ and Password_1_ parameters.

STEP 6 View the Line 1 tab to verify the new configuration.

NOTE The IP Telephony device supports alias remapping of a limited number of parameters. It is not meant to rename all parameters in its configuration.

Plain-Text Profile Resync

Only firmware releases after version 2.0.6 recognize the XML-based profiles. For firmware releases earlier than 2.0.6, use the SIP Profiler Compiler tool (SPC) to convert a text-based profile definition into a proprietary binary format understood by earlier versions of the firmware. The tool also provides options for encrypting the resulting binary profile.

The text-based profile understood by SPC uses a different syntax from the XML profile presented earlier. It consists of a list of parameter-value pairs, with the value in double quotes. Other minor syntax and parameter naming differences also apply. See **Plain-Text Profile Format, page 25** for more information. See **Generating Sample Configuration Files, page 50** for more information on creating an example Plain-text format profile.

The following exercise specifies values for two parameters that apply to **Line 1**:

Exercise

STEP 1 Obtain the SPC utility from Cisco.

The SPA Profile Compiler (SPC) is available from Cisco for the Win32 environment and Linux-i386-elf environment. For the OpenBSD environment, the SPC is made available on a case-by-case basis. See **Downloading the SPC, page 49** for more information on how to obtain the SPC for each device.

STEP 2 Create a Plain-text format profile named account.txt containing the following parameters:

```
User_ID[1] "17775551234" ;
Password[1] "512835907884" ;
```

STEP 3 Compile the text profile into a binary file, account.cfg by using the following command:

```
>spcexecutable account.txt account.cfg
```

STEP 4 Store account.cfg in the TFTP server virtual root directory.

STEP 5 Modify **profile rule** parameter on the test device to point to the account.cfg profile on the TFTP server:

```
tftp://192.168.1.200/account.cfg
```

STEP 6 Click **Submit All Changes**.

Upon resync, the IP Telephony device retrieves the new file, recognizes the binary format and updates the two specified parameters.

STEP 7 Observe the syslog messages sent by the IP Telephony device during resync.

5

Provisioning Field Reference

This chapter provides a listing of the parameters provided on the administration web server Provisioning tab that can be used in configuration profile scripts. It includes the following sections:

The Provisioning parameters described in this chapter are recognized by the IP Telephony devices beginning with firmware release 2.0.6 and higher unless otherwise indicated.

Delta Configuration Report

When the report rule is set, a SPA phone reports the phone profile to the server upon boot-up or receiving a report SIP NOTIFY message. By default the entire profile is reported.

Password or encryption key–related parameter values are not reported to the server:

```
<Admin_Password> IP Phone admin password
  <User_Password>   IP Phone user password
<PPPOE_Login_Password>
< VPN_Password>
<Access_Password_N>  Camera access password for each camera
profile
```

```
<Password_N>   Sip account user password for each SIP
extension
<SRTP_Private_Key_N> SRTP private key password for each SIP
extension
<Auth_Page_Password_N> Auth page password for each SIP
extension
<PIN_Code>                  BluePhone Pin code
<Directory_Password>  Broadsoft directory
<Password>   LDAP password
```

Deltas Report

SPA phones running firmware version 7.4.9c or higher can report deltas to the server if the **–delta** option is specified in the report rule. For example:

```
Report Rule: [--delta] http://report.com/delta$$MAC.xml
```

NOTE The double hyphen (--) required.

The *main profile* supports all provisionable parameters. Parameters in the main profile include the WiFi profile parameters. The **–delta** option only applies to the main profile. The deltas can be triggered by changes to the phone parameters entered in the LCD screen, the Web GUI, a SIP event, or remote provisioning. The personal address book, call history, Bluetooth profiles, and so forth are not in the main profile and are not reported.

The delta report is generated if the phone detects changes since the last resync, reboot, or upgrade. The report is done in asynchronous manner (with a random delay) and is sent only when the phone is idle. (The phone is considered idle when there is no active call or key press.)

Report Content

An administrator can define the content [**--content**] that is included in the report in the report rule. When **--content** *path* is defined, the main profile, address book and call history are reported to server. Where **p** reports the main profile parameters, **a** reports address book information, **h** reports the call history. This option is only available for UC320W.

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.
Resync_Random_Delay	Prevents an overload of the provisioning server when a large number of devices power-on simultaneously and attempt initial configuration. This delay is effective only on the initial configuration attempt, following a device power-on or reset. The parameter is the maximum time interval that the device waits before making contact with the provisioning server. The actual delay is a pseudo-random number between zero and this value. This parameter is in units of 20 seconds; the default value of 3 represents 60 seconds. This feature is disabled when this parameter is set to zero. The default value is 2 (40 seconds).
Resync At (SPA500 series phones)	The hour and minutes (HHmm) that the device resyncs 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.

Parameter Name	Description and Default Value
Resync_At_Random_Delay (firmware v7.4.9c and higher)	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 resyncs 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 resyncs 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 resyncing. The default value is 3600 seconds.

Parameter Name	Description and Default Value
Resync_Error_Retry_Delay	Resync retry interval (in seconds) applied in case of resync failure. The device has an error retry timer that activates if the previous attempt to sync with the provisioning server fails. The device waits to contact the server again until the timer counts down to zero. This parameter is the value that is initially loaded into the error retry timer. If this parameter is set to zero, the device immediately retries to sync with the provisioning server following a failed attempt. The default value is 3600 seconds.
Forced_Resync_Delay	Maximum delay (in seconds) the IP Telephony device waits before performing a resync. 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 resyncing. 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	Triggers a resync after every firmware upgrade attempt. The default value is Yes.

Parameter Name	Description and Default Value
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	Determines whether a file-not-found response from the provisioning server constitutes a successful or a failed resync. A failed resync activates the error resync timer. The default value is Yes.
Profile_Rule	This parameter is a profile script that evaluates to the provisioning resync command. The command specifies the protocol (TFTP, HTTP, or HTTPS) and an associated URL. If the command is not specified, TFTP is assumed, and the address of the TFTP server is obtained through DHCP option 66. In the URL, either the IP address or the FQDN of the server can be specified. The file name can have macros, such as $MA, which expands to the device MAC address. The default value is /spa$PSN.cfg.
Profile_Rule_B, Profile_Rule_C, Profile_Rule_D	Defines second, third, and fourth resync commands and associated profile URLs. These profile scripts are executed sequentially after the primary Profile Rule resync operation has completed. If a resync is triggered and Profile Rule is blank, Profile Rule B, C, and D are still evaluated and executed. The default value is (empty).
Log_Resync_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 resync $SCHEME:// $SERVIP:$PORT$PATH.

Parameter Name	Description and Default Value
Log_Resync_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_Resync_Failure_Msg	The syslog message that is issued after a failed resync attempt. The default value is $PN $MAC – Resync failed: $ERR.

Parameter Name	Description and Default Value
Report_Rule	The target URL to which configuration reports are sent. This parameter has the same syntax as the Profile_Rule parameter, and resolves to a TCP/IP command with an associated URL. A configuration report is generated in response to an authenticated SIP NOTIFY message, with Event: report. The report is an XML file containing the name and value of all the device parameters. This parameter may optionally contain an encryption key. For example: [--key $K] tftp://ps.callhome.net/$MA/rep.xml.enc Additionally, this parameter can trigger the reporting of configuration changes (deltas) to the server since the last resync, reboot, or upgrade using the --delta option. For example, to store delta configuration changes in a file with a name like SPA504G_<*MAC*>_<*serial#*>.xml, add the following to your provisioning file: [--delta] http://reportTargetServer/reportPath/$PN_$MA_$SN.xml </Report_Rule> The default value is (empty).

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.
Downgrade_Rev_Limit	Enforces a lower limit on the acceptable version number during a firmware upgrade or downgrade. The device does not complete a firmware upgrade operation unless the firmware version is greater than or equal to this parameter. The default value is (empty).
Upgrade_Rule	This parameter is a firmware upgrade script with the same syntax as Profile_Rule. Defines upgrade conditions and associated firmware URLs. The default value is (empty).
Log_Upgrade_Request_Msg	The syslog message that is issued at the start of a firmware upgrade attempt. The default value is $PN $MAC -- Requesting upgrade $SCHEME://$SERVIP:$PORT$PATH.

Parameter Name	Description and Default Value
Log_Upgrade_Success_Msg	The syslog message that is 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	The syslog message that is 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_SA, GPP_SB, GPP_SC, GPP_SD	Special purpose provisioning parameters, designed to hold encryption keys and passwords. To ensure the integrity of the encryption mechanism, these parameters must be kept secret. Therefore these parameters are not displayed on the device configuration web page, and they are not included in the configuration report sent in response to a SIP NOTIFY command. Note that these paramters are not available on the SPA500 Series phones. The default value is (empty).

Parameter Name	Description and Default Value
GPP_A through GPP_P	General purpose provisioning parameters. These parameters can be used as variables in provisioning and upgrade rules. They are referenced by prepending the variable name with a '$' character, such as $GPP_A. The default value is (empty).

Macro Expansion Variables

Certain macro variables are recognized within the following provisioning parameters:

- Profile_Rule
- Profile_Rule_*
- Resync_Trigger_*
- Log_Resync_*
- Upgrade_Rule
- Log_Upgrade_*
- 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 the contents of the special purpose parameters GPP_SA through GPP_SD. These parameters are meant to hold keys or passwords used in provisioning. Note that $SA through $SD are only recognized as arguments to the optional resync URL qualifier **--key**, as in the following example: [--key $SA] http://ps.callme.com/profiles/abcdefg.cfg These variables are not expanded outside of this limited context. Note that these variables are not available on the SPA500 Series phones.
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 SPA962.
PSN	Product Series Number, for example 962.
SN	Serial Number string, for example 88012BA01234.
CCERT	SSL Client Certificate status: Installed or Not Installed.
IP	IP address of the IP Telephony device within its local subnet, for example 192.168.1.100.
EXTIP	External IP of the IP Telephony device, as seen on the Internet, for example 66.43.16.52.
SWVER	Software version string, for example 2.0.6(b).
HWVER	Hardware version string, for example 1.88.1.

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, always expands to true in firmware rev 2.0.6 and above.
SCHEME	File access scheme, one of TFTP, HTTP, or HTTPS, as obtained after parsing resync or upgrade URL.
METH	Deprecated alias for SCHEME, do not use.
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.

Parameter Name	Description and Default Value
UID1	The contents of the Line 1 User_ID configuration parameter (Firmware 2.0.11 and above).
UID2	The contents of the Line 2 User_ID configuration parameter (Firmware 2.0.11 and above).
ISCUST	Value=1 if unit is customized, 0 otherwise; customization status viewable on WebUI Info page.

Internal Error Codes

The IP Telephony device 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.

A

Sample Configuration Profiles

An up-to-date profile template can be obtained from the SPC tool by using the commands:

spcexecuteable **--sample-profile** *outputfile.name*
spcexecuteable **--sample-xml** *outputfile.name*

The first version outputs a Plain-text format template. The second version outputs an Open format (XML-style) template. Sample profiles can be generated by using the SPA Profile Compiler (SPC). See **Generating Sample Configuration Files, page 50** for more information.

Open Format Sample

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<flat-profile xmlns="http://www.sipura.net/xsd/SPA525G-SCCP"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.sipura.net/xsd/SPA525G-SCCP http://
www.sipura.net/xsd/SPA525G-SCCP/SPA525G-SCCP-7-4-9.xsd">


  <!-- System Configuration -->

  <Restricted_Access_Domains ua="na"></Restricted_Access_Domains>
  <Enable_Web_Server ua="na">Yes</Enable_Web_Server>
  <Web_Server_Port ua="na">80</Web_Server_Port>
  <Enable_Web_Admin_Access ua="na">Yes</Enable_Web_Admin_Access>
  <Admin_Password ua="na"></Admin_Password>
  <User_Password ua="rw"></User_Password>
  <SPA525-protocol ua="na">SIP</SPA525-protocol> <!-- options: SIP/SCCP -->
  <SPA525-auto-detect-sccp ua="na">Yes</SPA525-auto-detect-sccp>
  <SPA525-readonly ua="na">No</SPA525-readonly>
  <Phone-UI-user-mode ua="na">No</Phone-UI-user-mode>

  <!-- Power Settings -->

  <PoE_Power_Required ua="na">Normal</PoE_Power_Required> <!-- options:
Normal/Maximum -->

  <!-- Internet Connection Type  -->
```

```
  <Connection_Type ua="rw">DHCP</Connection_Type> <!-- options: DHCP/Static
IP/PPPoE -->

  <!--  Static IP Settings -->

  <Static_IP ua="rw"></Static_IP>
  <NetMask ua="rw"></NetMask>
  <Gateway ua="rw"></Gateway>
  <Ethernet_MTU ua="na">1500</Ethernet_MTU>
  <Duplex_Mode ua="na">Auto</Duplex_Mode> <!-- options: Auto/10Mbps/Duplex/
10Mbps/Half/100Mbps/Duplex/100Mbps/Half -->

  <!--  PPPoE Settings -->

  <PPPoE_Login_Name ua="rw"></PPPoE_Login_Name>
  <PPPoE_Login_Password ua="rw"></PPPoE_Login_Password>
  <PPPoE_Service_Name ua="rw"></PPPoE_Service_Name>

  <!-- Optional Network Configuration -->

  <HostName ua="rw"></HostName>
  <Domain ua="rw"></Domain>
  <Primary_DNS ua="rw"></Primary_DNS>
  <Secondary_DNS ua="rw"></Secondary_DNS>
  <DNS_Server_Order ua="na">Manual,DHCP</DNS_Server_Order> <!-- options:
Manual/Manual,DHCP/DHCP,Manual -->
  <TFTP_Server ua="na"></TFTP_Server>
  <Alternate_TFTP ua="na">No</Alternate_TFTP>
  <Syslog_Server ua="na"></Syslog_Server>
  <Debug_Server ua="na"></Debug_Server>
  <Debug_Level ua="na">3</Debug_Level> <!-- options: 0/1/2/3 -->
  <Layer_2_Logging ua="na">No</Layer_2_Logging>
  <NTP_Enable ua="na">Yes</NTP_Enable>
  <Primary_NTP_Server ua="na"></Primary_NTP_Server>
  <Secondary_NTP_Server ua="na"></Secondary_NTP_Server>
  <Enable_Bonjour ua="na">No</Enable_Bonjour>

  <!-- VLAN Settings -->

  <Enable_VLAN ua="rw">No</Enable_VLAN>
  <VLAN_ID ua="rw">1</VLAN_ID>
  <PC_Port_VLAN_Highest_Priority ua="na">No Limit</
PC_Port_VLAN_Highest_Priority> <!-- options: 0/1/2/3/4/5/6/7/No Limit -->
  <Enable_PC_Port_VLAN_Tagging ua="na">No</Enable_PC_Port_VLAN_Tagging>
  <PC_Port_VLAN_ID ua="na">1</PC_Port_VLAN_ID>
  <Enable_CDP ua="na">Yes</Enable_CDP>
  <Enable_LLDP-MED ua="na">Yes</Enable_LLDP-MED>
  <Network_Startup_Delay ua="na">4</Network_Startup_Delay>

  <!-- Wi-Fi Settings -->

  <SPA525-wifi-on ua="rw">No</SPA525-wifi-on>

  <!-- Bluetooth Settings -->
```

```
  <Enable_BT ua="rw">No</Enable_BT>

  <!-- BluePhone -->

  <Bluetooth_Mode ua="na">0</Bluetooth_Mode> <!-- options: Phone/Handsfree/
Both -->
  <Line ua="na">5</Line> <!-- options: 1/2/3/4/5/Disabled -->
  <Short_Name ua="na"></Short_Name>
  <User_Friendly_ID ua="na"></User_Friendly_ID>
  <PIN_Code ua="na">0000</PIN_Code>

  <!--  VPN Settings -->

  <VPN_Server ua="rw"></VPN_Server>
  <VPN_User_Name ua="rw"></VPN_User_Name>
  <VPN_Password ua="rw"></VPN_Password>
  <VPN_Tunnel_Group ua="na"></VPN_Tunnel_Group>
  <Connect_on_Bootup ua="rw">No</Connect_on_Bootup>

  <!-- Inventory Information -->

  <Asset_ID ua="na"></Asset_ID>

  <!-- 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__HHmm_>
  <Resync_Periodic ua="na">3600</Resync_Periodic>
  <Resync_Error_Retry_Delay ua="na">3600</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_1>
  <Resync_Trigger_2 ua="na"></Resync_Trigger_2>
  <Resync_Fails_On_FNF ua="na">Yes</Resync_Fails_On_FNF>
  <Profile_Rule ua="na">/spa$PSN.cfg</Profile_Rule>
  <Profile_Rule_B ua="na"></Profile_Rule_B>
  <Profile_Rule_C ua="na"></Profile_Rule_C>
  <Profile_Rule_D ua="na"></Profile_Rule_D>
  <DHCP_Option_To_Use ua="na">66,160,159,150</DHCP_Option_To_Use>
  <Transport_Protocol ua="na">https</Transport_Protocol> <!-- options: none/
tftp/http/https -->
  <Log_Resync_Request_Msg ua="na">$PN $MAC -- Requesting resync $SCHEME://
$SERVIP:$PORT$PATH</Log_Resync_Request_Msg>
  <Log_Resync_Success_Msg ua="na">$PN $MAC -- Successful resync $SCHEME://
$SERVIP:$PORT$PATH</Log_Resync_Success_Msg>
  <Log_Resync_Failure_Msg ua="na">$PN $MAC -- Resync failed: $ERR</
Log_Resync_Failure_Msg>
  <Report_Rule ua="na"></Report_Rule>
  <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>
  <Downgrade_Rev_Limit ua="na"></Downgrade_Rev_Limit>
  <Upgrade_Rule ua="na"></Upgrade_Rule>
  <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>
  <License_Keys ua="na"></License_Keys>

  <!-- General Purpose Parameters -->

  <GPP_A ua="na"></GPP_A>
  <GPP_B ua="na"></GPP_B>
  <GPP_C ua="na"></GPP_C>
  <GPP_D ua="na"></GPP_D>
  <GPP_E ua="na"></GPP_E>
  <GPP_F ua="na"></GPP_F>
  <GPP_G ua="na"></GPP_G>
  <GPP_H ua="na"></GPP_H>
  <GPP_I ua="na"></GPP_I>
  <GPP_J ua="na"></GPP_J>
  <GPP_K ua="na"></GPP_K>
  <GPP_L ua="na"></GPP_L>
  <GPP_M ua="na"></GPP_M>
  <GPP_N ua="na"></GPP_N>
  <GPP_O ua="na"></GPP_O>
  <GPP_P ua="na"></GPP_P>
  <GPP_SA ua="na"></GPP_SA>
  <GPP_SB ua="na"></GPP_SB>
  <GPP_SC ua="na"></GPP_SC>
  <GPP_SD ua="na"></GPP_SD>

  <!-- 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_Reg_User_Agent_Name>
  <SIP_Accept_Language ua="na"></SIP_Accept_Language>
  <DTMF_Relay_MIME_Type ua="na">application/dtmf-relay</DTMF_Relay_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>
  <SIP-B_Enable ua="na">No</SIP-B_Enable>
  <Talk_Package ua="na">No</Talk_Package>
  <Hold_Package ua="na">No</Hold_Package>
  <Conference_Package ua="na">No</Conference_Package>
  <Notify_Conference ua="na">No</Notify_Conference>
  <RFC_2543_Call_Hold ua="na">Yes</RFC_2543_Call_Hold>
```

```
  <Random_REG_CID_On_Reboot ua="na">No</Random_REG_CID_On_Reboot>
  <Mark_All_AVT_Packets ua="na">Yes</Mark_All_AVT_Packets>
  <SIP_TCP_Port_Min ua="na">5060</SIP_TCP_Port_Min>
  <SIP_TCP_Port_Max ua="na">5080</SIP_TCP_Port_Max>
  <CTI_Enable ua="na">No</CTI_Enable>
  <Caller_ID_Header ua="na">PAID-RPID-FROM</Caller_ID_Header> <!-- options:
PAID-RPID-FROM/PAID-FROM/RPID-PAID-FROM/RPID-FROM/FROM -->
  <Dialog_SDP_Enable ua="na">No</Dialog_SDP_Enable>

  <!-- 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"></Reg_Retry_Random_Delay>
  <Reg_Retry_Long_Random_Delay ua="na"></Reg_Retry_Long_Random_Delay>
  <Reg_Retry_Intvl_Cap ua="na"></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 -->

  <SIT1_RSC ua="na"></SIT1_RSC>
  <SIT2_RSC ua="na"></SIT2_RSC>
  <SIT3_RSC ua="na"></SIT3_RSC>
  <SIT4_RSC ua="na"></SIT4_RSC>
  <Try_Backup_RSC ua="na"></Try_Backup_RSC>
  <Retry_Reg_RSC ua="na"></Retry_Reg_RSC>

  <!-- 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>
  <No_UDP_Checksum ua="na">No</No_UDP_Checksum>
  <Symmetric_RTP ua="na">No</Symmetric_RTP>
  <Stats_In_BYE ua="na">No</Stats_In_BYE>

  <!-- SDP Payload Types -->

  <AVT_Dynamic_Payload ua="na">101</AVT_Dynamic_Payload>
```

```
  <INFOREQ_Dynamic_Payload ua="na"></INFOREQ_Dynamic_Payload>
  <G726r16_Dynamic_Payload ua="na">98</G726r16_Dynamic_Payload>
  <G726r24_Dynamic_Payload ua="na">97</G726r24_Dynamic_Payload>
  <G726r32_Dynamic_Payload ua="na">2</G726r32_Dynamic_Payload>
  <G726r40_Dynamic_Payload ua="na">96</G726r40_Dynamic_Payload>
  <G729b_Dynamic_Payload ua="na">99</G729b_Dynamic_Payload>
  <L16_Dynamic_Payload ua="na">104</L16_Dynamic_Payload>
  <EncapRTP_Dynamic_Payload ua="na">112</EncapRTP_Dynamic_Payload>
  <RTP-Start-Loopback_Dynamic_Payload ua="na">113</RTP-Start-
Loopback_Dynamic_Payload>
  <RTP-Start-Loopback_Codec ua="na">G711u</RTP-Start-Loopback_Codec> <!--
options: G711u/G711a/G726-32/G729a/G722 -->
  <AVT_Codec_Name ua="na">telephone-event</AVT_Codec_Name>
  <G711u_Codec_Name ua="na">PCMU</G711u_Codec_Name>
  <G711a_Codec_Name ua="na">PCMA</G711a_Codec_Name>
  <G726r32_Codec_Name ua="na">G726-32</G726r32_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>
  <L16_Codec_Name ua="na">L16</L16_Codec_Name>
  <EncapRTP_Codec_Name ua="na">encaprtp</EncapRTP_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"></STUN_Server>
  <EXT_IP ua="na"></EXT_IP>
  <EXT_RTP_Port_Min ua="na"></EXT_RTP_Port_Min>
  <NAT_Keep_Alive_Intvl ua="na">15</NAT_Keep_Alive_Intvl>

  <!-- Linksys Key System Parameters -->

  <Linksys_Key_System ua="na">No</Linksys_Key_System>
  <Multicast_Address ua="na">224.168.168.168:6061</Multicast_Address>
  <Force_LAN_Codec ua="na">none</Force_LAN_Codec> <!-- options: none/G711u/
G711a -->

  <!-- Network Settings -->

  <SCCP_TOS_DiffServ_Value_1_ ua="na">0x68</SCCP_TOS_DiffServ_Value_1_>
  <SCCP_CoS_Value_1_ ua="na">3</SCCP_CoS_Value_1_> <!-- options: 0/1/2/3/4/5/
6/7 -->
  <DTMF_Tx_Method_1_ ua="na">INFO</DTMF_Tx_Method_1_> <!-- options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO -->
  <DTMF_Tx_Volume_for_AVT_Packet_1_ ua="na">0</
DTMF_Tx_Volume_for_AVT_Packet_1_>

  <!-- Network Settings -->
```

```
  <SCCP_TOS_DiffServ_Value_2_ ua="na">0x68</SCCP_TOS_DiffServ_Value_2_>
  <SCCP_CoS_Value_2_ ua="na">3</SCCP_CoS_Value_2_> <!-- options: 0/1/2/3/4/5/
6/7 -->
  <DTMF_Tx_Method_2_ ua="na">INFO</DTMF_Tx_Method_2_> <!-- options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO -->
  <DTMF_Tx_Volume_for_AVT_Packet_2_ ua="na">0</
DTMF_Tx_Volume_for_AVT_Packet_2_>

  <!-- Network Settings -->

  <SCCP_TOS_DiffServ_Value_3_ ua="na">0x68</SCCP_TOS_DiffServ_Value_3_>
  <SCCP_CoS_Value_3_ ua="na">3</SCCP_CoS_Value_3_> <!-- options: 0/1/2/3/4/5/
6/7 -->
  <DTMF_Tx_Method_3_ ua="na">INFO</DTMF_Tx_Method_3_> <!-- options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO -->
  <DTMF_Tx_Volume_for_AVT_Packet_3_ ua="na">0</
DTMF_Tx_Volume_for_AVT_Packet_3_>

  <!-- Network Settings -->

  <SCCP_TOS_DiffServ_Value_4_ ua="na">0x68</SCCP_TOS_DiffServ_Value_4_>
  <SCCP_CoS_Value_4_ ua="na">3</SCCP_CoS_Value_4_> <!-- options: 0/1/2/3/4/5/
6/7 -->
  <DTMF_Tx_Method_4_ ua="na">INFO</DTMF_Tx_Method_4_> <!-- options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO -->
  <DTMF_Tx_Volume_for_AVT_Packet_4_ ua="na">0</
DTMF_Tx_Volume_for_AVT_Packet_4_>

  <!-- Network Settings -->

  <SCCP_TOS_DiffServ_Value_5_ ua="na">0x68</SCCP_TOS_DiffServ_Value_5_>
  <SCCP_CoS_Value_5_ ua="na">3</SCCP_CoS_Value_5_> <!-- options: 0/1/2/3/4/5/
6/7 -->
  <DTMF_Tx_Method_5_ ua="na">INFO</DTMF_Tx_Method_5_> <!-- options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO -->
  <DTMF_Tx_Volume_for_AVT_Packet_5_ ua="na">0</
DTMF_Tx_Volume_for_AVT_Packet_5_>

  <!-- Call Forward -->

  <Cfwd_Setting ua="rw">Yes</Cfwd_Setting>
  <Cfwd_All_Dest ua="rw"></Cfwd_All_Dest>
  <Cfwd_Busy_Dest ua="rw"></Cfwd_Busy_Dest>
  <Cfwd_No_Ans_Dest ua="rw"></Cfwd_No_Ans_Dest>
  <Cfwd_No_Ans_Delay ua="rw">20</Cfwd_No_Ans_Delay>
  <Speakerphone_DTMF_Masking ua="na">No</Speakerphone_DTMF_Masking>

  <!-- Camera Settings -->

  <Enable_Video_VLAN ua="ro">No</Enable_Video_VLAN>
  <Video_VLAN_ID ua="ro">1</Video_VLAN_ID>

  <!-- Camera Profile 1 -->
```

```
  <Camera_Name_1_ ua="ro"></Camera_Name_1_>
  <Access_URL_1_ ua="ro"></Access_URL_1_>
  <Access_User_Name_1_ ua="ro"></Access_User_Name_1_>
  <Access_Password_1_ ua="na"></Access_Password_1_>
  <Associated_Caller_ID_1_ ua="ro"></Associated_Caller_ID_1_>
  <Door_Control_URL_1_ ua="na"></Door_Control_URL_1_>

  <!-- Camera Profile 2 -->

  <Camera_Name_2_ ua="ro"></Camera_Name_2_>
  <Access_URL_2_ ua="ro"></Access_URL_2_>
  <Access_User_Name_2_ ua="ro"></Access_User_Name_2_>
  <Access_Password_2_ ua="na"></Access_Password_2_>
  <Associated_Caller_ID_2_ ua="ro"></Associated_Caller_ID_2_>
  <Door_Control_URL_2_ ua="na"></Door_Control_URL_2_>

  <!-- Camera Profile 3 -->

  <Camera_Name_3_ ua="ro"></Camera_Name_3_>
  <Access_URL_3_ ua="ro"></Access_URL_3_>
  <Access_User_Name_3_ ua="ro"></Access_User_Name_3_>
  <Access_Password_3_ ua="na"></Access_Password_3_>
  <Associated_Caller_ID_3_ ua="ro"></Associated_Caller_ID_3_>
  <Door_Control_URL_3_ ua="na"></Door_Control_URL_3_>

  <!-- Camera Profile 4 -->

  <Camera_Name_4_ ua="ro"></Camera_Name_4_>
  <Access_URL_4_ ua="ro"></Access_URL_4_>
  <Access_User_Name_4_ ua="ro"></Access_User_Name_4_>
  <Access_Password_4_ ua="na"></Access_Password_4_>
  <Associated_Caller_ID_4_ ua="ro"></Associated_Caller_ID_4_>
  <Door_Control_URL_4_ ua="na"></Door_Control_URL_4_>

  <!-- Audio Volume -->

  <Ringer_Volume ua="rw">9</Ringer_Volume>
  <Speaker_Volume ua="rw">11</Speaker_Volume>
  <Handset_Volume ua="rw">9</Handset_Volume>
  <Headset_Volume ua="rw">9</Headset_Volume>
  <Bluetooth_Volume ua="rw">9</Bluetooth_Volume>
  <Handset_Version ua="rw">Auto</Handset_Version> <!-- options: Auto/
Original/v3 -->
  <Deep_Bass ua="rw">Yes</Deep_Bass>

  <!-- Call Record Setting -->

  <Call_Record_Mode ua="na">No</Call_Record_Mode>
  <Call_Record_Beep ua="na">Yes</Call_Record_Beep>

  <!-- LCD Screen -->

  <Screen_Saver_Enable ua="rw">No</Screen_Saver_Enable>
```

```
  <Screen_Saver_Type ua="rw">Black Background</Screen_Saver_Type> <!--
options: Black Background/Gray Background/Black/Gray Rotation/Picture
Rotation/Digital Frame/Download Picture/Clock -->
  <Screen_Saver_Trigger_Time ua="rw">300</Screen_Saver_Trigger_Time>
  <Screen_Saver_Refresh_Time ua="rw">10</Screen_Saver_Refresh_Time>
  <Back_Light_Enable ua="na">No</Back_Light_Enable>
  <Back_Light_Timer__sec_ ua="na">30</Back_Light_Timer__sec_>
  <LCD_Contrast ua="rw">15</LCD_Contrast>
  <Logo_Type ua="na"></Logo_Type> <!-- options: Default/Download BMP Picture/
Text Logo -->
  <Text_Logo ua="na"></Text_Logo>
  <Background_Picture_Type ua="na">Default</Background_Picture_Type> <!--
options: Default/Download BMP Picture -->
  <BMP_Picture_Download_URL ua="na"></BMP_Picture_Download_URL>
  <Station_Name ua="na"></Station_Name>
  <Default_Ring ua="na">Chirp 1</Default_Ring>
  <Default_Ring_1_ ua="na">0</Default_Ring_1_>
  <Use_Default_Ring_1_ ua="na">Yes</Use_Default_Ring_1_>
  <Default_Ring_2_ ua="na">0</Default_Ring_2_>
  <Use_Default_Ring_2_ ua="na">Yes</Use_Default_Ring_2_>
  <Default_Ring_3_ ua="na">0</Default_Ring_3_>
  <Use_Default_Ring_3_ ua="na">Yes</Use_Default_Ring_3_>
  <Default_Ring_4_ ua="na">0</Default_Ring_4_>
  <Use_Default_Ring_4_ ua="na">Yes</Use_Default_Ring_4_>
  <Default_Ring_5_ ua="na">0</Default_Ring_5_>
  <Use_Default_Ring_5_ ua="na">Yes</Use_Default_Ring_5_>

  <!-- Ring Tone -->

  <Ring1 ua="na">n=Cisco Synth;w=file://Cisco_synth_ring1.mp3;c=0</Ring1>
  <Ring2 ua="na">n=Retro;w=file://ringin.726;c=1</Ring2>
  <Ring3 ua="na">n=Office;w=file://thx-short.726;c=1</Ring3>
  <Ring4 ua="na">n=Analog Synth;w=file://Analog1.raw;c=1</Ring4>
  <Ring5 ua="na">n=Are You There;w=file://AreYouThereF.raw;c=1</Ring5>
  <Ring6 ua="na">n=Chime;w=file://Chime.raw;c=1</Ring6>
  <Ring7 ua="na">n=Clock Shop;w=file://ClockShop.raw;c=1</Ring7>
  <Ring8 ua="na">n=Film Score;w=file://FilmScore.raw;c=1</Ring8>
  <Ring9 ua="na">n=Koto Effect;w=file://KotoEffect.raw;c=1</Ring9>
  <Ring10 ua="na">n=Piano;w=file://Piano2.raw;c=1</Ring10>
  <Ring11 ua="na">n=Pulse;w=file://Pulse1.raw;c=1</Ring11>
  <Ring12 ua="na">n=Du-dut;w=file://Ring7.raw;c=1</Ring12>

  <!-- Audio Input Gain (dB) -->

  <Handset_Input_Gain ua="na">0</Handset_Input_Gain> <!-- options: -6/0/6/9 -
->
  <Headset_Input_Gain ua="na">0</Headset_Input_Gain> <!-- options: -6/0/6/9 -
->
  <Speakerphone_Input_Gain ua="na">0</Speakerphone_Input_Gain> <!-- options:
-6/0/6/9 -->
  <Bluetooth_Input_Gain ua="na">0</Bluetooth_Input_Gain> <!-- options: -6/0/
6/9 -->

  <!-- Call Progress Tones -->
```

```
  <Dial_Tone ua="na">350@-19,440@-19;*(*/0/1+2)</Dial_Tone>
  <Bluetooth_Dial_Tone ua="na">350@-19,440@-19;1(0/*/0);*(*/0/1+2)</
Bluetooth_Dial_Tone>
  <Outside_Dial_Tone ua="na">420@-16;*(*/0/1)</Outside_Dial_Tone>
  <Prompt_Tone ua="na">520@-19,620@-19;*(*/0/1+2)</Prompt_Tone>
  <Busy_Tone ua="na">480@-19,620@-19;*(.5/.5/1+2)</Busy_Tone>
  <Reorder_Tone ua="na">480@-19,620@-19;*(.25/.25/1+2)</Reorder_Tone>
  <Off_Hook_Warning_Tone ua="na">480@-10,620@0;*(.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>
  <SIT1_Tone ua="na">985@-16,1428@-16,1777@-16;20(.380/0/1,.380/0/2,.380/0/
3,0/4/0)</SIT1_Tone>
  <SIT2_Tone ua="na">914@-16,1371@-16,1777@-16;20(.274/0/1,.274/0/2,.380/0/
3,0/4/0)</SIT2_Tone>
  <SIT3_Tone ua="na">914@-16,1371@-16,1777@-16;20(.380/0/1,.380/0/2,.380/0/
3,0/4/0)</SIT3_Tone>
  <SIT4_Tone ua="na">985@-16,1371@-16,1777@-16;20(.380/0/1,.274/0/2,.380/0/
3,0/4/0)</SIT4_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>
  <Low_Battery_Tone ua="na">1600@-16;2(.05/0.05/1,.05/.05/1,0/.5/0)</
Low_Battery_Tone>
  <Alert_Tone ua="na">600@-19;.2(.05/0.05/1)</Alert_Tone>
  <System_Beep ua="na">600@-16;.1(.05/0.05/1)</System_Beep>

  <!-- 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>
  <Call_Back_Expires ua="na">1800</Call_Back_Expires>
  <Call_Back_Retry_Intvl ua="na">30</Call_Back_Retry_Intvl>
  <Call_Back_Delay ua="na">.5</Call_Back_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">*98</Blind_Transfer_Code>
<Call_Back_Act_Code ua="na">*66</Call_Back_Act_Code>
<Call_Back_Deact_Code ua="na">*86</Call_Back_Deact_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">*67</Block_CID_Act_Code>
<Block_CID_Deact_Code ua="na">*68</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">*38</Call_Park_Code>
<Call_Pickup_Code ua="na">*36</Call_Pickup_Code>
<Call_UnPark_Code ua="na">*39</Call_UnPark_Code>
<Group_Call_Pickup_Code ua="na">*37</Group_Call_Pickup_Code>
<Media_Loopback_Code ua="na">*03</Media_Loopback_Code>
<Referral_Services_Codes ua="na"></Referral_Services_Codes>
<Feature_Dial_Services_Codes ua="na"></Feature_Dial_Services_Codes>

<!-- Vertical Service Announcement Codes -->

<Service_Annc_Base_Number ua="na"></Service_Annc_Base_Number>
<Service_Annc_Extension_Codes ua="na"></Service_Annc_Extension_Codes>

<!-- 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_L16_Code ua="na">*01016</Prefer_L16_Code>
<Force_L16_Code ua="na">*02016</Force_L16_Code>
<Prefer_G726r32_Code ua="na">*0172632</Prefer_G726r32_Code>
<Force_G726r32_Code ua="na">*0272632</Force_G726r32_Code>
<Prefer_G729a_Code ua="na">*01729</Prefer_G729a_Code>
```

```
  <Force_G729a_Code ua="na">*02729</Force_G729a_Code>

  <!-- Time -->

  <Time_Zone ua="na">GMT-08:00</Time_Zone> <!-- options: GMT-12:00/GMT-11:00/
GMT-10:00/GMT-09:00/GMT-08:00/GMT-07:00/GMT-06:00/GMT-05:00/GMT-04:00/GMT-
03:30/GMT-03:00/GMT-02:00/GMT-01:00/GMT/GMT+01:00/GMT+02:00/GMT+03:00/
GMT+03:30/GMT+04:00/GMT+05:00/GMT+05:30/GMT+05:45/GMT+06:00/GMT+06:30/
GMT+07:00/GMT+08:00/GMT+09:00/GMT+09:30/GMT+10:00/GMT+11:00/GMT+12:00/
GMT+13:00 -->
  <Time_Offset__HH_mm_ ua="na"></Time_Offset__HH_mm_>
  <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 -->

  <SCCP_Dictionary_Server_Script_1 ua="na"></SCCP_Dictionary_Server_Script_1>
  <Language_Selection ua="na">English-US</Language_Selection>

  <!-- Miscellaneous -->

  <DTMF_Playback_Level ua="na">-16</DTMF_Playback_Level>
  <DTMF_Playback_Length ua="na">.1</DTMF_Playback_Length>
  <Inband_DTMF_Boost ua="na">12dB</Inband_DTMF_Boost> <!-- options: 0dB/3dB/
6dB/9dB/12dB/15dB/18dB -->
  <SCCP_Dictionary_Server_Script_2 ua="na"></SCCP_Dictionary_Server_Script_2>
  <SCCP_Language_Selection_2 ua="na"></SCCP_Language_Selection_2>

  <!-- Locale Settings -->


  <!-- General -->

  <Subscribe_Expires ua="na">1800</Subscribe_Expires>
  <Subscribe_Retry_Interval ua="na">30</Subscribe_Retry_Interval>
  <Unit_1_Enable ua="na">Yes</Unit_1_Enable>
  <Subscribe_Delay ua="na">1</Subscribe_Delay>
  <Unit_2_Enable ua="na">Yes</Unit_2_Enable>
  <Server_Type ua="na">Broadsoft</Server_Type> <!-- options: Broadsoft/
SPA9000/Asterisk/RFC3265_4235/Sylantro -->
  <Test_Mode_Enable ua="na">No</Test_Mode_Enable>
  <Attendant_Console_Call_Pickup_Code ua="na">*98</
Attendant_Console_Call_Pickup_Code>

  <!-- Unit 1 -->

  <Unit_1_Key_1 ua="na"></Unit_1_Key_1>
  <Unit_1_Key_2 ua="na"></Unit_1_Key_2>
  <Unit_1_Key_3 ua="na"></Unit_1_Key_3>
  <Unit_1_Key_4 ua="na"></Unit_1_Key_4>
  <Unit_1_Key_5 ua="na"></Unit_1_Key_5>
  <Unit_1_Key_6 ua="na"></Unit_1_Key_6>
  <Unit_1_Key_7 ua="na"></Unit_1_Key_7>
```

```
<Unit_1_Key_8 ua="na"></Unit_1_Key_8>
<Unit_1_Key_9 ua="na"></Unit_1_Key_9>
<Unit_1_Key_10 ua="na"></Unit_1_Key_10>
<Unit_1_Key_11 ua="na"></Unit_1_Key_11>
<Unit_1_Key_12 ua="na"></Unit_1_Key_12>
<Unit_1_Key_13 ua="na"></Unit_1_Key_13>
<Unit_1_Key_14 ua="na"></Unit_1_Key_14>
<Unit_1_Key_15 ua="na"></Unit_1_Key_15>
<Unit_1_Key_16 ua="na"></Unit_1_Key_16>
<Unit_1_Key_17 ua="na"></Unit_1_Key_17>
<Unit_1_Key_18 ua="na"></Unit_1_Key_18>
<Unit_1_Key_19 ua="na"></Unit_1_Key_19>
<Unit_1_Key_20 ua="na"></Unit_1_Key_20>
<Unit_1_Key_21 ua="na"></Unit_1_Key_21>
<Unit_1_Key_22 ua="na"></Unit_1_Key_22>
<Unit_1_Key_23 ua="na"></Unit_1_Key_23>
<Unit_1_Key_24 ua="na"></Unit_1_Key_24>
<Unit_1_Key_25 ua="na"></Unit_1_Key_25>
<Unit_1_Key_26 ua="na"></Unit_1_Key_26>
<Unit_1_Key_27 ua="na"></Unit_1_Key_27>
<Unit_1_Key_28 ua="na"></Unit_1_Key_28>
<Unit_1_Key_29 ua="na"></Unit_1_Key_29>
<Unit_1_Key_30 ua="na"></Unit_1_Key_30>
<Unit_1_Key_31 ua="na"></Unit_1_Key_31>
<Unit_1_Key_32 ua="na"></Unit_1_Key_32>

<!-- Unit 2 -->

<Unit_2_Key_1 ua="na"></Unit_2_Key_1>
<Unit_2_Key_2 ua="na"></Unit_2_Key_2>
<Unit_2_Key_3 ua="na"></Unit_2_Key_3>
<Unit_2_Key_4 ua="na"></Unit_2_Key_4>
<Unit_2_Key_5 ua="na"></Unit_2_Key_5>
<Unit_2_Key_6 ua="na"></Unit_2_Key_6>
<Unit_2_Key_7 ua="na"></Unit_2_Key_7>
<Unit_2_Key_8 ua="na"></Unit_2_Key_8>
<Unit_2_Key_9 ua="na"></Unit_2_Key_9>
<Unit_2_Key_10 ua="na"></Unit_2_Key_10>
<Unit_2_Key_11 ua="na"></Unit_2_Key_11>
<Unit_2_Key_12 ua="na"></Unit_2_Key_12>
<Unit_2_Key_13 ua="na"></Unit_2_Key_13>
<Unit_2_Key_14 ua="na"></Unit_2_Key_14>
<Unit_2_Key_15 ua="na"></Unit_2_Key_15>
<Unit_2_Key_16 ua="na"></Unit_2_Key_16>
<Unit_2_Key_17 ua="na"></Unit_2_Key_17>
<Unit_2_Key_18 ua="na"></Unit_2_Key_18>
<Unit_2_Key_19 ua="na"></Unit_2_Key_19>
<Unit_2_Key_20 ua="na"></Unit_2_Key_20>
<Unit_2_Key_21 ua="na"></Unit_2_Key_21>
<Unit_2_Key_22 ua="na"></Unit_2_Key_22>
<Unit_2_Key_23 ua="na"></Unit_2_Key_23>
<Unit_2_Key_24 ua="na"></Unit_2_Key_24>
<Unit_2_Key_25 ua="na"></Unit_2_Key_25>
<Unit_2_Key_26 ua="na"></Unit_2_Key_26>
<Unit_2_Key_27 ua="na"></Unit_2_Key_27>
```

```
  <Unit_2_Key_28 ua="na"></Unit_2_Key_28>
  <Unit_2_Key_29 ua="na"></Unit_2_Key_29>
  <Unit_2_Key_30 ua="na"></Unit_2_Key_30>
  <Unit_2_Key_31 ua="na"></Unit_2_Key_31>
  <Unit_2_Key_32 ua="na"></Unit_2_Key_32>
  <SPA525-SSID ua="na">~</SPA525-SSID>
  <SPA525-Encryption-type ua="na">~</SPA525-Encryption-type>
  <SPA525-Encryption-key ua="na">~</SPA525-Encryption-key>
  <SPA525-Encryption-key-code ua="na"></SPA525-Encryption-key-code>
  <Protect_IVR_FactoryReset ua="na">No</Protect_IVR_FactoryReset>

  <profileName_1_ >cisco-voice</profileName_1_>
  <ssid_1_ >cisco-voice</ssid_1_>
  <securityMode_1_ >Disable</securityMode_1_> <!-- options: Disable/WEP/WPA-
PSK/WPA2-PSK/WPA-ENT/WPA2-ENT -->
  <cipherType_1_ ></cipherType_1_> <!-- 0:TKIP, 1:AES -->
  <wepKeyId_1_ ></wepKeyId_1_> <!-- [0-4] -->
  <wepKeyLen_1_ ></wepKeyLen_1_> <!-- 0:10 hex, 1:26 hex, 2:5 ascii, 3:13
ascii -->
  <wepKey0_1_ ></wepKey0_1_>
  <wepKey1_1_ ></wepKey1_1_>
  <wepKey2_1_ ></wepKey2_1_>
  <wepKey3_1_ ></wepKey3_1_>
  <pskKey_1_ ></pskKey_1_>
  <eapType_1_ ></eapType_1_> <!-- 1:TLS, 2:TTLS, 3:LEAP -->
  <identity_1_ ></identity_1_>
  <password_1_ ></password_1_>
  <anonymousIdentity_1_ ></anonymousIdentity_1_>
  <serverCert_1_ ></serverCert_1_>
  <clientCert_1_ ></clientCert_1_>
  <ttlsAuthProto_1_ ></ttlsAuthProto_1_> <!-- 0:MD5, 1:MSCHV2, 2:MSCHAP,
3:PAP, 4:CHAP -->
  <profileLock_1_ ></profileLock_1_> <!-- Yes/No -->
  <profileEnable_1_ >Yes</profileEnable_1_> <!-- Yes/No -->
  <profileName_2_ ></profileName_2_>
  <ssid_2_ ></ssid_2_>
  <securityMode_2_ >Disable</securityMode_2_>
  <cipherType_2_ ></cipherType_2_>
  <wepKeyId_2_ ></wepKeyId_2_>
  <wepKeyLen_2_ ></wepKeyLen_2_>
  <wepKey0_2_ ></wepKey0_2_>
  <wepKey1_2_ ></wepKey1_2_>
  <wepKey2_2_ ></wepKey2_2_>
  <wepKey3_2_ ></wepKey3_2_>
  <pskKey_2_ ></pskKey_2_>
  <eapType_2_ ></eapType_2_>
  <identity_2_ ></identity_2_>
  <password_2_ ></password_2_>
  <anonymousIdentity_2_ ></anonymousIdentity_2_>
  <serverCert_2_ ></serverCert_2_>
  <clientCert_2_ ></clientCert_2_>
  <ttlsAuthProto_2_ ></ttlsAuthProto_2_>
  <profileLock_2_ ></profileLock_2_>
  <profileEnable_2_ ></profileEnable_2_>
  <profileName_3_ ></profileName_3_>
```

```
  <ssid_3_ ></ssid_3_>
  <securityMode_3_ >Disable</securityMode_3_>
  <cipherType_3_ ></cipherType_3_>
  <wepKeyId_3_ ></wepKeyId_3_>
  <wepKeyLen_3_ ></wepKeyLen_3_>
  <wepKey0_3_ ></wepKey0_3_>
  <wepKey1_3_ ></wepKey1_3_>
  <wepKey2_3_ ></wepKey2_3_>
  <wepKey3_3_ ></wepKey3_3_>
  <pskKey_3_ ></pskKey_3_>
  <eapType_3_ ></eapType_3_>
  <identity_3_ ></identity_3_>
  <password_3_ ></password_3_>
  <anonymousIdentity_3_ ></anonymousIdentity_3_>
  <serverCert_3_ ></serverCert_3_>
  <clientCert_3_ ></clientCert_3_>
  <ttlsAuthProto_3_ ></ttlsAuthProto_3_>
  <profileLock_3_ ></profileLock_3_>
  <profileEnable_3_ ></profileEnable_3_>

</flat-profile>
```

Plain Text Sample One

```
# ***
# *** SPA525G-SCCP 7.4.9 Configuration Parameters
# ***


# *** System Configuration

Restricted_Access_Domains                       "" ;
Enable_Web_Server                               "Yes" ;
Web_Server_Port                                 "80" ;
Enable_Web_Admin_Access                         "Yes" ;
Admin_Password                                  "" ;
User_Password                                  ! "" ;
SPA525-protocol                                 "SIP" ;  # options: SIP/SCCP
SPA525-auto-detect-sccp                         "Yes" ;
SPA525-readonly                                 "No" ;
Phone-UI-user-mode                              "No" ;

# *** Power Settings

PoE_Power_Required                              "Normal" ;  # options: Normal/
Maximum

# *** Internet Connection Type

Connection_Type                                ! "DHCP" ;  # options: DHCP/
Static IP/PPPoE
```

```
# ***  Static IP Settings

Static_IP                                       ! "" ;
NetMask                                         ! "" ;
Gateway                                         ! "" ;
Ethernet_MTU                                      "1500" ;
Duplex_Mode                                       "Auto" ;  # options: Auto/
10Mbps/Duplex/10Mbps/Half/100Mbps/Duplex/100Mbps/Half

# ***  PPPoE Settings

PPPoE_Login_Name                                ! "" ;
PPPoE_Login_Password                            ! "" ;
PPPoE_Service_Name                              ! "" ;

# *** Optional Network Configuration

HostName                                        ! "" ;
Domain                                          ! "" ;
Primary_DNS                                     ! "" ;
Secondary_DNS                                   ! "" ;
DNS_Server_Order                                  "Manual,DHCP" ;  # options:
Manual/Manual,DHCP/DHCP,Manual
TFTP_Server                                       "" ;
Alternate_TFTP                                    "No" ;
Syslog_Server                                     "" ;
Debug_Server                                      "" ;
Debug_Level                                       "3" ;  # options: 0/1/2/3
Layer_2_Logging                                   "No" ;
NTP_Enable                                        "Yes" ;
Primary_NTP_Server                                "" ;
Secondary_NTP_Server                              "" ;
Enable_Bonjour                                    "No" ;

# *** VLAN Settings

Enable_VLAN                                     ! "No" ;
VLAN_ID                                         ! "1" ;
PC_Port_VLAN_Highest_Priority                     "No Limit" ;  # options: 0/1/
2/3/4/5/6/7/No Limit
Enable_PC_Port_VLAN_Tagging                       "No" ;
PC_Port_VLAN_ID                                   "1" ;
Enable_CDP                                        "Yes" ;
Enable_LLDP-MED                                   "Yes" ;
Network_Startup_Delay                             "4" ;

# *** Wi-Fi Settings

SPA525-wifi-on                                  ! "No" ;

# *** Bluetooth Settings

Enable_BT                                       ! "No" ;

# *** BluePhone
```

```
Bluetooth_Mode                                  "0" ;  # options: Phone/
Handsfree/Both
Line                                            "5" ;  # options: 1/2/3/4/5/
Disabled
Short_Name                                      "" ;
User_Friendly_ID                                "" ;
PIN_Code                                        "0000" ;

# ***  VPN Settings

VPN_Server                                     ! "" ;
VPN_User_Name                                  ! "" ;
VPN_Password                                   ! "" ;
VPN_Tunnel_Group                                "" ;
Connect_on_Bootup                              ! "No" ;

# *** Inventory Information

Asset_ID                                        "" ;

# *** Configuration Profile

Provision_Enable                                "Yes" ;
Resync_On_Reset                                 "Yes" ;
Resync_Random_Delay                             "2" ;
Resync_At_(HHmm)                                "" ;
Resync_Periodic                                 "3600" ;
Resync_Error_Retry_Delay                        "3600" ;
Forced_Resync_Delay                             "14400" ;
Resync_From_SIP                                 "Yes" ;
Resync_After_Upgrade_Attempt                    "Yes" ;
Resync_Trigger_1                                "" ;
Resync_Trigger_2                                "" ;
Resync_Fails_On_FNF                             "Yes" ;
Profile_Rule                                    "/spa$PSN.cfg" ;
Profile_Rule_B                                  "" ;
Profile_Rule_C                                  "" ;
Profile_Rule_D                                  "" ;
DHCP_Option_To_Use                              "66,160,159,150" ;
Transport_Protocol                              "https" ;  # options: none/
tftp/http/https
Log_Resync_Request_Msg                         "$PN $MAC -- Requesting resync
$SCHEME://$SERVIP:$PORT$PATH" ;
Log_Resync_Success_Msg                         "$PN $MAC -- Successful resync
$SCHEME://$SERVIP:$PORT$PATH" ;
Log_Resync_Failure_Msg                          "$PN $MAC -- Resync failed:
$ERR" ;
Report_Rule                                     "" ;
User_Configurable_Resync                        "Yes" ;

# *** Firmware Upgrade

Upgrade_Enable                                  "Yes" ;
Upgrade_Error_Retry_Delay                       "3600" ;
```

```
Downgrade_Rev_Limit                              "" ;
Upgrade_Rule                                     "" ;
Log_Upgrade_Request_Msg                         "$PN $MAC -- Requesting upgrade
$SCHEME://$SERVIP:$PORT$PATH" ;
Log_Upgrade_Success_Msg                         "$PN $MAC -- Successful upgrade
$SCHEME://$SERVIP:$PORT$PATH -- $ERR" ;
Log_Upgrade_Failure_Msg                           "$PN $MAC -- Upgrade failed:
$ERR" ;
License_Keys                                     "" ;

# *** General Purpose Parameters

GPP_A                                            "" ;
GPP_B                                            "" ;
GPP_C                                            "" ;
GPP_D                                            "" ;
GPP_E                                            "" ;
GPP_F                                            "" ;
GPP_G                                            "" ;
GPP_H                                            "" ;
GPP_I                                            "" ;
GPP_J                                            "" ;
GPP_K                                            "" ;
GPP_L                                            "" ;
GPP_M                                            "" ;
GPP_N                                            "" ;
GPP_O                                            "" ;
GPP_P                                            "" ;
GPP_SA                                           "" ;
GPP_SB                                           "" ;
GPP_SC                                           "" ;
GPP_SD                                           "" ;

# *** SIP Parameters

Max_Forward                                      "70" ;
Max_Redirection                                  "5" ;
Max_Auth                                         "2" ;
SIP_User_Agent_Name                              "$VERSION" ;
SIP_Server_Name                                  "$VERSION" ;
SIP_Reg_User_Agent_Name                          "" ;
SIP_Accept_Language                              "" ;
DTMF_Relay_MIME_Type                             "application/dtmf-relay" ;
Remove_Last_Reg                                  "No" ;
Use_Compact_Header                               "No" ;
Escape_Display_Name                              "No" ;
SIP-B_Enable                                     "No" ;
Talk_Package                                     "No" ;
Hold_Package                                     "No" ;
Conference_Package                               "No" ;
Notify_Conference                                "No" ;
RFC_2543_Call_Hold                               "Yes" ;
Random_REG_CID_On_Reboot                         "No" ;
Mark_All_AVT_Packets                             "Yes" ;
SIP_TCP_Port_Min                                 "5060" ;
```

```
SIP_TCP_Port_Max                                "5080" ;
CTI_Enable                                      "No" ;
Caller_ID_Header                               "PAID-RPID-FROM" ;  # options:
PAID-RPID-FROM/PAID-FROM/RPID-PAID-FROM/RPID-FROM/FROM
Dialog_SDP_Enable                               "No" ;

# *** SIP Timer Values (sec)

SIP_T1                                          ".5" ;
SIP_T2                                          "4" ;
SIP_T4                                          "5" ;
SIP_Timer_B                                     "16" ;
SIP_Timer_F                                     "16" ;
SIP_Timer_H                                     "16" ;
SIP_Timer_D                                     "16" ;
SIP_Timer_J                                     "16" ;
INVITE_Expires                                  "240" ;
ReINVITE_Expires                                "30" ;
Reg_Min_Expires                                 "1" ;
Reg_Max_Expires                                 "7200" ;
Reg_Retry_Intvl                                 "30" ;
Reg_Retry_Long_Intvl                            "1200" ;
Reg_Retry_Random_Delay                          "" ;
Reg_Retry_Long_Random_Delay                     "" ;
Reg_Retry_Intvl_Cap                             "" ;
Sub_Min_Expires                                 "10" ;
Sub_Max_Expires                                 "7200" ;
Sub_Retry_Intvl                                 "10" ;

# *** Response Status Code Handling

SIT1_RSC                                        "" ;
SIT2_RSC                                        "" ;
SIT3_RSC                                        "" ;
SIT4_RSC                                        "" ;
Try_Backup_RSC                                  "" ;
Retry_Reg_RSC                                   "" ;

# *** RTP Parameters

RTP_Port_Min                                    "16384" ;
RTP_Port_Max                                    "16482" ;
RTP_Packet_Size                                 "0.030" ;
Max_RTP_ICMP_Err                                "0" ;
RTCP_Tx_Interval                                "0" ;
No_UDP_Checksum                                 "No" ;
Symmetric_RTP                                   "No" ;
Stats_In_BYE                                    "No" ;

# *** SDP Payload Types

AVT_Dynamic_Payload                             "101" ;
INFOREQ_Dynamic_Payload                         "" ;
G726r16_Dynamic_Payload                         "98" ;
G726r24_Dynamic_Payload                         "97" ;
```

```
G726r32_Dynamic_Payload                    "2" ;
G726r40_Dynamic_Payload                    "96" ;
G729b_Dynamic_Payload                      "99" ;
L16_Dynamic_Payload                        "104" ;
EncapRTP_Dynamic_Payload                   "112" ;
RTP-Start-Loopback_Dynamic_Payload         "113" ;
RTP-Start-Loopback_Codec                   "G711u" ;  # options: G711u/
G711a/G726-32/G729a/G722
AVT_Codec_Name                             "telephone-event" ;
G711u_Codec_Name                           "PCMU" ;
G711a_Codec_Name                           "PCMA" ;
G726r32_Codec_Name                         "G726-32" ;
G729a_Codec_Name                           "G729a" ;
G729b_Codec_Name                           "G729ab" ;
G722_Codec_Name                            "G722" ;
L16_Codec_Name                             "L16" ;
EncapRTP_Codec_Name                        "encaprtp" ;

# *** NAT Support Parameters

Handle_VIA_received                        "No" ;
Handle_VIA_rport                           "No" ;
Insert_VIA_received                        "No" ;
Insert_VIA_rport                           "No" ;
Substitute_VIA_Addr                        "No" ;
Send_Resp_To_Src_Port                      "No" ;
STUN_Enable                                "No" ;
STUN_Test_Enable                           "No" ;
STUN_Server                                "" ;
EXT_IP                                     "" ;
EXT_RTP_Port_Min                           "" ;
NAT_Keep_Alive_Intvl                       "15" ;

# *** Linksys Key System Parameters

Linksys_Key_System                         "No" ;
Multicast_Address                          "224.168.168.168:6061" ;
Force_LAN_Codec                            "none" ;  # options: none/
G711u/G711a

# *** Network Settings

SCCP_TOS/DiffServ_Value[1]                 "0x68" ;
SCCP_CoS_Value[1]                          "3" ;  # options: 0/1/2/3/4/
5/6/7
DTMF_Tx_Method[1]                          "INFO" ;  # options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO
DTMF_Tx_Volume_for_AVT_Packet[1]           "0" ;

# *** Network Settings

SCCP_TOS/DiffServ_Value[2]                 "0x68" ;
SCCP_CoS_Value[2]                          "3" ;  # options: 0/1/2/3/4/
5/6/7
```

```
DTMF_Tx_Method[2]                          "INFO" ;  # options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO
DTMF_Tx_Volume_for_AVT_Packet[2]           "0" ;

# *** Network Settings

SCCP_TOS/DiffServ_Value[3]                 "0x68" ;
SCCP_CoS_Value[3]                          "3" ;  # options: 0/1/2/3/4/
5/6/7
DTMF_Tx_Method[3]                          "INFO" ;  # options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO
DTMF_Tx_Volume_for_AVT_Packet[3]           "0" ;

# *** Network Settings

SCCP_TOS/DiffServ_Value[4]                 "0x68" ;
SCCP_CoS_Value[4]                          "3" ;  # options: 0/1/2/3/4/
5/6/7
DTMF_Tx_Method[4]                          "INFO" ;  # options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO
DTMF_Tx_Volume_for_AVT_Packet[4]           "0" ;

# *** Network Settings

SCCP_TOS/DiffServ_Value[5]                 "0x68" ;
SCCP_CoS_Value[5]                          "3" ;  # options: 0/1/2/3/4/
5/6/7
DTMF_Tx_Method[5]                          "INFO" ;  # options: InBand/
AVT/INFO/Auto/InBand+INFO/AVT+INFO
DTMF_Tx_Volume_for_AVT_Packet[5]           "0" ;

# *** Call Forward

Cfwd_Setting                             ! "Yes" ;
Cfwd_All_Dest                            ! "" ;
Cfwd_Busy_Dest                           ! "" ;
Cfwd_No_Ans_Dest                         ! "" ;
Cfwd_No_Ans_Delay                        ! "20" ;
Speakerphone_DTMF_Masking                  "No" ;

# *** Camera Settings

Enable_Video_VLAN                        ? "No" ;
Video_VLAN_ID                            ? "1" ;

# *** Camera Profile 1

Camera_Name[1]                           ? "" ;
Access_URL[1]                            ? "" ;
Access_User_Name[1]                      ? "" ;
Access_Password[1]                         "" ;
Associated_Caller_ID[1]                  ? "" ;
Door_Control_URL[1]                        "" ;

# *** Camera Profile 2
```

```
Camera_Name[2]                                  ? "" ;
Access_URL[2]                                   ? "" ;
Access_User_Name[2]                             ? "" ;
Access_Password[2]                                "" ;
Associated_Caller_ID[2]                         ? "" ;
Door_Control_URL[2]                               "" ;

# *** Camera Profile 3

Camera_Name[3]                                  ? "" ;
Access_URL[3]                                   ? "" ;
Access_User_Name[3]                             ? "" ;
Access_Password[3]                                "" ;
Associated_Caller_ID[3]                         ? "" ;
Door_Control_URL[3]                               "" ;

# *** Camera Profile 4

Camera_Name[4]                                  ? "" ;
Access_URL[4]                                   ? "" ;
Access_User_Name[4]                             ? "" ;
Access_Password[4]                                "" ;
Associated_Caller_ID[4]                         ? "" ;
Door_Control_URL[4]                               "" ;

# *** Audio Volume

Ringer_Volume                                   ! "9" ;
Speaker_Volume                                  ! "11" ;
Handset_Volume                                  ! "9" ;
Headset_Volume                                  ! "9" ;
Bluetooth_Volume                                ! "9" ;
Handset_Version                                 ! "Auto" ;  # options: Auto/
Original/v3
Deep_Bass                                       ! "Yes" ;

# *** Call Record Setting

Call_Record_Mode                                  "No" ;
Call_Record_Beep                                  "Yes" ;

# *** LCD Screen

Screen_Saver_Enable                             ! "No" ;
Screen_Saver_Type                             ! "Black Background" ;  # options:
Black Background/Gray Background/Black/Gray Rotation/Picture Rotation/Digital
Frame/Download Picture/Clock
Screen_Saver_Trigger_Time                       ! "300" ;
Screen_Saver_Refresh_Time                       ! "10" ;
Back_Light_Enable                                 "No" ;
Back_Light_Timer_(sec)                            "30" ;
LCD_Contrast                                    ! "15" ;
Logo_Type                                         "" ;  # options: Default/
Download BMP Picture/Text Logo
```

```
Text_Logo                                    "" ;
Background_Picture_Type                      "Default" ;  # options:
Default/Download BMP Picture
BMP_Picture_Download_URL                     "" ;
Station_Name                                 "" ;
Default_Ring                                 "Chirp 1" ;
Default_Ring[1]                              "0" ;
Use_Default_Ring[1]                          "Yes" ;
Default_Ring[2]                              "0" ;
Use_Default_Ring[2]                          "Yes" ;
Default_Ring[3]                              "0" ;
Use_Default_Ring[3]                          "Yes" ;
Default_Ring[4]                              "0" ;
Use_Default_Ring[4]                          "Yes" ;
Default_Ring[5]                              "0" ;
Use_Default_Ring[5]                          "Yes" ;

# *** Ring Tone

Ring1                                        "n=Cisco Synth;w=file://
Cisco_synth_ring1.mp3;c=0" ;
Ring2                                   "n=Retro;w=file://ringin.726;c=1"
;
Ring3                                        "n=Office;w=file://thx-
short.726;c=1" ;
Ring4                                        "n=Analog Synth;w=file://
Analog1.raw;c=1" ;
Ring5                                        "n=Are You There;w=file://
AreYouThereF.raw;c=1" ;
Ring6                                    "n=Chime;w=file://Chime.raw;c=1"
;
Ring7                                        "n=Clock Shop;w=file://
ClockShop.raw;c=1" ;
Ring8                                        "n=Film Score;w=file://
FilmScore.raw;c=1" ;
Ring9                                        "n=Koto Effect;w=file://
KotoEffect.raw;c=1" ;
Ring10                                       "n=Piano;w=file://
Piano2.raw;c=1" ;
Ring11                                       "n=Pulse;w=file://
Pulse1.raw;c=1" ;
Ring12                                       "n=Du-dut;w=file://
Ring7.raw;c=1" ;

# *** Audio Input Gain (dB)

Handset_Input_Gain                           "0" ;  # options: -6/0/6/9
Headset_Input_Gain                           "0" ;  # options: -6/0/6/9
Speakerphone_Input_Gain                      "0" ;  # options: -6/0/6/9
Bluetooth_Input_Gain                         "0" ;  # options: -6/0/6/9

# *** Call Progress Tones

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

```
Bluetooth_Dial_Tone                              "350@-19,440@-19;1(0/*/0);*(*/
0/1+2)" ;
Outside_Dial_Tone                                 "420@-16;*(*/0/1)" ;
Prompt_Tone                                      "520@-19,620@-19;*(*/0/1+2)" ;
Busy_Tone                                       "480@-19,620@-19;*(.5/.5/1+2)" ;
Reorder_Tone                                      "480@-19,620@-19;*(.25/.25/
1+2)" ;
Off_Hook_Warning_Tone                             "480@-10,620@0;*(.125/.125/
1+2)" ;
Ring_Back_Tone                                   "440@-19,480@-19;*(2/4/1+2)" ;
Call_Waiting_Tone                                 "440@-10;30(.3/9.7/1)" ;
Confirm_Tone                                      "600@-16;1(.25/.25/1)" ;
SIT1_Tone                                         "985@-16,1428@-16,1777@-
16;20(.380/0/1,.380/0/2,.380/0/3,0/4/0)" ;
SIT2_Tone                                         "914@-16,1371@-16,1777@-
16;20(.274/0/1,.274/0/2,.380/0/3,0/4/0)" ;
SIT3_Tone                                         "914@-16,1371@-16,1777@-
16;20(.380/0/1,.380/0/2,.380/0/3,0/4/0)" ;
SIT4_Tone                                         "985@-16,1371@-16,1777@-
16;20(.380/0/1,.274/0/2,.380/0/3,0/4/0)" ;
MWI_Dial_Tone                                     "350@-19,440@-19;2(.1/.1/
1+2);10(*/0/1+2)" ;
Cfwd_Dial_Tone                                    "350@-19,440@-19;2(.2/.2/
1+2);10(*/0/1+2)" ;
Holding_Tone                                     "600@-19;25(.1/.1/1,.1/.1/1,.1/
9.5/1)" ;
Conference_Tone                                 "350@-19;20(.1/.1/1,.1/9.7/1)" ;
Secure_Call_Indication_Tone                       "397@-19,507@-19;15(0/2/0,.2/
.1/1,.1/2.1/2)" ;
Page_Tone                                         "600@-16;.3(.05/0.05/1)" ;
Low_Battery_Tone                                  "1600@-16;2(.05/0.05/1,.05/
.05/1,0/.5/0)" ;
Alert_Tone                                        "600@-19;.2(.05/0.05/1)" ;
System_Beep                                       "600@-16;.1(.05/0.05/1)" ;

# *** Distinctive Ring Patterns

Cadence_1                                         "60(2/4)" ;
Cadence_2                                         "60(.3/.2,1/.2,.3/4)" ;
Cadence_3                                         "60(.8/.4,.8/4)" ;
Cadence_4                                         "60(.4/.2,.3/.2,.8/4)" ;
Cadence_5                                        "60(.2/.2,.2/.2,.2/.2,1/4)" ;
Cadence_6                                         "60(.2/.4,.2/.4,.2/4)" ;
Cadence_7                                         "60(4.5/4)" ;
Cadence_8                                         "60(0.25/9.75)" ;
Cadence_9                                         "60(.4/.2,.4/2)" ;

# *** Control Timer Values (sec)

Reorder_Delay                                     "255" ;
Call_Back_Expires                                 "1800" ;
Call_Back_Retry_Intvl                             "30" ;
Call_Back_Delay                                   ".5" ;
Interdigit_Long_Timer                             "10" ;
Interdigit_Short_Timer                            "3" ;
```

```
# *** Vertical Service Activation Codes

Call_Return_Code                                "*69" ;
Blind_Transfer_Code                             "*98" ;
Call_Back_Act_Code                              "*66" ;
Call_Back_Deact_Code                            "*86" ;
Cfwd_All_Act_Code                               "*72" ;
Cfwd_All_Deact_Code                             "*73" ;
Cfwd_Busy_Act_Code                              "*90" ;
Cfwd_Busy_Deact_Code                            "*91" ;
Cfwd_No_Ans_Act_Code                            "*92" ;
Cfwd_No_Ans_Deact_Code                          "*93" ;
CW_Act_Code                                     "*56" ;
CW_Deact_Code                                   "*57" ;
CW_Per_Call_Act_Code                            "*71" ;
CW_Per_Call_Deact_Code                          "*70" ;
Block_CID_Act_Code                              "*67" ;
Block_CID_Deact_Code                            "*68" ;
Block_CID_Per_Call_Act_Code                     "*81" ;
Block_CID_Per_Call_Deact_Code                   "*82" ;
Block_ANC_Act_Code                              "*77" ;
Block_ANC_Deact_Code                            "*87" ;
DND_Act_Code                                    "*78" ;
DND_Deact_Code                                  "*79" ;
Secure_All_Call_Act_Code                        "*16" ;
Secure_No_Call_Act_Code                         "*17" ;
Secure_One_Call_Act_Code                        "*18" ;
Secure_One_Call_Deact_Code                      "*19" ;
Paging_Code                                     "*96" ;
Call_Park_Code                                  "*38" ;
Call_Pickup_Code                                "*36" ;
Call_UnPark_Code                                "*39" ;
Group_Call_Pickup_Code                          "*37" ;
Media_Loopback_Code                             "*03" ;
Referral_Services_Codes                         "" ;
Feature_Dial_Services_Codes                     "" ;

# *** Vertical Service Announcement Codes

Service_Annc_Base_Number                        "" ;
Service_Annc_Extension_Codes                    "" ;

# *** Outbound Call Codec Selection Codes

Prefer_G711u_Code                               "*017110" ;
Force_G711u_Code                                "*027110" ;
Prefer_G711a_Code                               "*017111" ;
Force_G711a_Code                                "*027111" ;
Prefer_G722_Code                                "*01722" ;
Force_G722_Code                                 "*02722" ;
Prefer_L16_Code                                 "*01016" ;
Force_L16_Code                                  "*02016" ;
Prefer_G726r32_Code                             "*0172632" ;
Force_G726r32_Code                              "*0272632" ;
```

```
Prefer_G729a_Code                                "*01729" ;
Force_G729a_Code                                 "*02729" ;

# *** Time

Time_Zone                                       "GMT-08:00" ;  # options: GMT-
12:00/GMT-11:00/GMT-10:00/GMT-09:00/GMT-08:00/GMT-07:00/GMT-06:00/GMT-05:00/
GMT-04:00/GMT-03:30/GMT-03:00/GMT-02:00/GMT-01:00/GMT/GMT+01:00/GMT+02:00/
GMT+03:00/GMT+03:30/GMT+04:00/GMT+05:00/GMT+05:30/GMT+05:45/GMT+06:00/
GMT+06:30/GMT+07:00/GMT+08:00/GMT+09:00/GMT+09:30/GMT+10:00/GMT+11:00/
GMT+12:00/GMT+13:00
Time_Offset_(HH/mm)                              "" ;
Ignore_DHCP_Time_Offset                          "Yes" ;
Daylight_Saving_Time_Rule                        "start=3/-1/7/2;end=10/-1/7/
2;save=1" ;
Daylight_Saving_Time_Enable                      "Yes" ;

# *** Language

SCCP_Dictionary_Server_Script_1                  "" ;
Language_Selection                               "English-US" ;

# *** Miscellaneous

DTMF_Playback_Level                              "-16" ;
DTMF_Playback_Length                             ".1" ;
Inband_DTMF_Boost                               "12dB" ;  # options: 0dB/3dB/
6dB/9dB/12dB/15dB/18dB
SCCP_Dictionary_Server_Script_2                  "" ;
SCCP_Language_Selection_2                        "" ;

# *** Locale Settings


# *** General

Subscribe_Expires                                "1800" ;
Subscribe_Retry_Interval                         "30" ;
Unit_1_Enable                                    "Yes" ;
Subscribe_Delay                                  "1" ;
Unit_2_Enable                                    "Yes" ;
Server_Type                                      "Broadsoft" ;  # options:
Broadsoft/SPA9000/Asterisk/RFC3265_4235/Sylantro
Test_Mode_Enable                                 "No" ;
Attendant_Console_Call_Pickup_Code               "*98" ;

# *** Unit 1

Unit_1_Key_1                                     "" ;
Unit_1_Key_2                                     "" ;
Unit_1_Key_3                                     "" ;
Unit_1_Key_4                                     "" ;
Unit_1_Key_5                                     "" ;
Unit_1_Key_6                                     "" ;
Unit_1_Key_7                                     "" ;
```

```
Unit_1_Key_8                                    "" ;
Unit_1_Key_9                                    "" ;
Unit_1_Key_10                                   "" ;
Unit_1_Key_11                                   "" ;
Unit_1_Key_12                                   "" ;
Unit_1_Key_13                                   "" ;
Unit_1_Key_14                                   "" ;
Unit_1_Key_15                                   "" ;
Unit_1_Key_16                                   "" ;
Unit_1_Key_17                                   "" ;
Unit_1_Key_18                                   "" ;
Unit_1_Key_19                                   "" ;
Unit_1_Key_20                                   "" ;
Unit_1_Key_21                                   "" ;
Unit_1_Key_22                                   "" ;
Unit_1_Key_23                                   "" ;
Unit_1_Key_24                                   "" ;
Unit_1_Key_25                                   "" ;
Unit_1_Key_26                                   "" ;
Unit_1_Key_27                                   "" ;
Unit_1_Key_28                                   "" ;
Unit_1_Key_29                                   "" ;
Unit_1_Key_30                                   "" ;
Unit_1_Key_31                                   "" ;
Unit_1_Key_32                                   "" ;

# *** Unit 2

Unit_2_Key_1                                    "" ;
Unit_2_Key_2                                    "" ;
Unit_2_Key_3                                    "" ;
Unit_2_Key_4                                    "" ;
Unit_2_Key_5                                    "" ;
Unit_2_Key_6                                    "" ;
Unit_2_Key_7                                    "" ;
Unit_2_Key_8                                    "" ;
Unit_2_Key_9                                    "" ;
Unit_2_Key_10                                   "" ;
Unit_2_Key_11                                   "" ;
Unit_2_Key_12                                   "" ;
Unit_2_Key_13                                   "" ;
Unit_2_Key_14                                   "" ;
Unit_2_Key_15                                   "" ;
Unit_2_Key_16                                   "" ;
Unit_2_Key_17                                   "" ;
Unit_2_Key_18                                   "" ;
Unit_2_Key_19                                   "" ;
Unit_2_Key_20                                   "" ;
Unit_2_Key_21                                   "" ;
Unit_2_Key_22                                   "" ;
Unit_2_Key_23                                   "" ;
Unit_2_Key_24                                   "" ;
Unit_2_Key_25                                   "" ;
Unit_2_Key_26                                   "" ;
Unit_2_Key_27                                   "" ;
```

```
Unit_2_Key_28                                   "" ;
Unit_2_Key_29                                   "" ;
Unit_2_Key_30                                   "" ;
Unit_2_Key_31                                   "" ;
Unit_2_Key_32                                   "" ;
SPA525-SSID                                     "~" ;
SPA525-Encryption-type                          "~" ;
SPA525-Encryption-key                           "~" ;
SPA525-Encryption-key-code                      "" ;
Protect_IVR_FactoryReset                        "No" ;
```

Plain Text Sample Two

```
# ***
# *** Linksys SPA Series Configuration Parameters
# ***


# *** System Configuration

Restricted_Access_Domains          "" ;
Enable_Web_Server                  "Yes" ;
Web_Server_Port                    "80" ;
Enable_Web_Admin_Access            "Yes" ;
Admin_Passwd                       "" ;
User_Password                    ! "" ;

# *** Internet Connection Type

DHCP                             ! "Yes" ;
Static_IP                        ! "" ;
NetMask                          ! "" ;
Gateway                          ! "" ;

# *** Optional Network Configuration

HostName                         ! "" ;
Domain                           ! "" ;
Primary_DNS                      ! "" ;
Secondary_DNS                    ! "" ;
DNS_Server_Order                   "Manual" ;  # options: Manual/Manual,DHCP/
DHCP,Manual
DNS_Query_Mode                    "Parallel" ;  # options: Parallel/Sequential
Syslog_Server                      "" ;
Debug_Server                       "" ;
Debug_Level                        "0" ;  # options: 0/1/2/3
Primary_NTP_Server                 "" ;
Secondary_NTP_Server               "" ;

# *** Configuration Profile

Provision_Enable                   "Yes" ;
```

```
Resync_On_Reset                 "Yes" ;
Resync_Random_Delay             "2" ;
Resync_Periodic                 "3600" ;
Resync_Error_Retry_Delay        "3600" ;
Forced_Resync_Delay             "14400" ;
Resync_From_SIP                 "Yes" ;
Resync_After_Upgrade_Attempt    "Yes" ;
Resync_Trigger_1                "" ;
Resync_Trigger_2                "" ;
Profile_Rule                    "/spa$PSN.cfg" ;
Profile_Rule_B                  "" ;
Profile_Rule_C                  "" ;
Profile_Rule_D                  "" ;
Log_Resync_Request_Msg          "$PN $MAC -- Requesting resync $SCHEME://
$SERVIP:$PORT$PATH" ;
Log_Resync_Success_Msg          "$PN $MAC -- Successful resync $SCHEME://
$SERVIP:$PORT$PATH" ;
Log_Resync_Failure_Msg          "$PN $MAC -- Resync failed: $ERR" ;

# *** Firmware Upgrade

Upgrade_Enable                  "Yes" ;
Upgrade_Error_Retry_Delay       "3600" ;
Downgrade_Rev_Limit             "" ;
Upgrade_Rule                    "" ;
Log_Upgrade_Request_Msg         "$PN $MAC -- Requesting upgrade $SCHEME://
$SERVIP:$PORT$PATH" ;
Log_Upgrade_Success_Msg         "$PN $MAC -- Successful upgrade $SCHEME://
$SERVIP:$PORT$PATH -- $ERR" ;
Log_Upgrade_Failure_Msg         "$PN $MAC -- Upgrade failed: $ERR" ;

# *** General Purpose Parameters

GPP_A                           "" ;
GPP_B                           "" ;
GPP_C                           "" ;
GPP_D                           "" ;
GPP_E                           "" ;
GPP_F                           "" ;
GPP_G                           "" ;
GPP_H                           "" ;
GPP_I                           "" ;
GPP_J                           "" ;
GPP_K                           "" ;
GPP_L                           "" ;
GPP_M                           "" ;
GPP_N                           "" ;
GPP_O                           "" ;
GPP_P                           "" ;
GPP_SA                          "" ;
GPP_SB                          "" ;
GPP_SC                          "" ;
GPP_SD                          "" ;

# *** SIP Parameters
```

```
Max_Forward                         "70" ;
Max_Redirection                     "5" ;
Max_Auth                            "2" ;
SIP_User_Agent_Name                 "$VERSION" ;
SIP_Server_Name                     "$VERSION" ;
SIP_Accept_Language                 "" ;
DTMF_Relay_MIME_Type                "application/dtmf-relay" ;
Hook_Flash_MIME_Type                "application/hook-flash" ;
Remove_Last_Reg                     "No" ;
Use_Compact_Header                  "No" ;

# *** SIP Timer Values (sec)

SIP_T1                              ".5" ;
SIP_T2                              "4" ;
SIP_T4                              "5" ;
SIP_Timer_B                         "32" ;
SIP_Timer_F                         "32" ;
SIP_Timer_H                         "32" ;
SIP_Timer_D                         "32" ;
SIP_Timer_J                         "32" ;
INVITE_Expires                      "240" ;
ReINVITE_Expires                    "30" ;
Reg_Min_Expires                     "1" ;
Reg_Max_Expires                     "7200" ;
Reg_Retry_Intvl                     "30" ;
Reg_Retry_Long_Intvl                "1200" ;

# *** Response Status Code Handling

SIT1_RSC                            "" ;
SIT2_RSC                            "" ;
SIT3_RSC                            "" ;
SIT4_RSC                            "" ;
Try_Backup_RSC                      "" ;
Retry_Reg_RSC                       "" ;

# *** RTP Parameters

RTP_Port_Min                        "16384" ;
RTP_Port_Max                        "16482" ;
RTP_Packet_Size                     "0.030" ;
Max_RTP_ICMP_Err                    "0" ;
RTCP_Tx_Interval                    "0" ;

# *** SDP Payload Types

NSE_Dynamic_Payload                 "100" ;
AVT_Dynamic_Payload                 "101" ;
G726r16_Dynamic_Payload             "98" ;
G726r24_Dynamic_Payload             "97" ;
G726r40_Dynamic_Payload             "96" ;
G729b_Dynamic_Payload               "99" ;
NSE_Codec_Name                      "NSE" ;
```

```
AVT_Codec_Name                  "telephone-event" ;
G711u_Codec_Name                "PCMU" ;
G711a_Codec_Name                "PCMA" ;
G726r16_Codec_Name              "G726-16" ;
G726r24_Codec_Name              "G726-24" ;
G726r32_Codec_Name              "G726-32" ;
G726r40_Codec_Name              "G726-40" ;
G729a_Codec_Name                "G729a" ;
G729b_Codec_Name                "G729ab" ;
G723_Codec_Name                 "G723" ;

# *** NAT Support Parameters

Handle_VIA_received             "No" ;
Handle_VIA_rport                "No" ;
Insert_VIA_received             "No" ;
Insert_VIA_rport                "No" ;
Substitute_VIA_Addr             "No" ;
Send_Resp_To_Src_Port           "No" ;
STUN_Enable                     "No" ;
STUN_Test_Enable                "No" ;
STUN_Server                     "" ;
EXT_IP                          "" ;
EXT_RTP_Port_Min                "" ;
NAT_Keep_Alive_Intvl            "15" ;

# ***

Line_Enable[1]                  "Yes" ;
SAS_Enable[1]                   "No" ;
MOH_Server[1]                   "" ;
SAS_DLG_Refresh_Intvl[1]        "30" ;
NAT_Mapping_Enable[1]           "No" ;
SAS_Inbound_RTP_Sink[1]         "" ;
SIP_Port[1]                     "5060" ;
NAT_Keep_Alive_Enable[1]        "No" ;
EXT_SIP_Port[1]                 "" ;
NAT_Keep_Alive_Msg[1]           "$NOTIFY" ;
SIP_TOS/DiffServ_Value[1]       "0x68" ;
NAT_Keep_Alive_Dest[1]          "$PROXY" ;
RTP_TOS/DiffServ_Value[1]       "0xb8" ;
SIP_Debug_Option[1]             "none" ;  # options: none/1-line/1-line
excl. OPT/1-line excl. NTFY/1-line excl. REG/1-line excl. OPT|NTFY|REG/full/
full excl. OPT/full excl. NTFY/full excl. REG/full excl. OPT|NTFY|REG
Network_Jitter_Level[1]         "high" ;  # options: low/medium/high/very
high
SIP_100REL_Enable[1]            "No" ;
Blind_Attn-Xfer_Enable[1]       "No" ;
SIP_Proxy-Require[1]            "" ;
Auth_Resync-Reboot[1]           "Yes" ;
SIP_Remote-Party-ID[1]          "No" ;

# *** Proxy and Registration

Proxy[1]                        "" ;
```

```
Use_Outbound_Proxy[1]              "No" ;
Outbound_Proxy[1]                  "" ;
Use_OB_Proxy_In_Dialog[1]          "Yes" ;
Register[1]                        "Yes" ;
Make_Call_Without_Reg[1]           "No" ;
Register_Expires[1]                "3600" ;
Ans_Call_Without_Reg[1]            "No" ;
Use_DNS_SRV[1]                     "No" ;
DNS_SRV_Auto_Prefix[1]             "No" ;
Proxy_Fallback_Intvl[1]            "3600" ;

# *** Subscriber Information

Display_Name[1]                    "" ;
User_ID[1]                         "" ;
Password[1]                        "" ;
Use_Auth_ID[1]                     "No" ;
Auth_ID[1]                         "" ;
Mini_Certificate[1]                "" ;
SRTP_Private_Key[1]                "" ;

# *** Supplementary Service Subscription

Call_Waiting_Serv[1]               "Yes" ;
Block_CID_Serv[1]                  "Yes" ;
Block_ANC_Serv[1]                  "Yes" ;
Dist_Ring_Serv[1]                  "Yes" ;
Cfwd_All_Serv[1]                   "Yes" ;
Cfwd_Busy_Serv[1]                  "Yes" ;
Cfwd_No_Ans_Serv[1]                "Yes" ;
Cfwd_Sel_Serv[1]                   "Yes" ;
Cfwd_Last_Serv[1]                  "Yes" ;
Block_Last_Serv[1]                 "Yes" ;
Accept_Last_Serv[1]                "Yes" ;
DND_Serv[1]                        "Yes" ;
CID_Serv[1]                        "Yes" ;
CWCID_Serv[1]                      "Yes" ;
Call_Return_Serv[1]                "Yes" ;
Call_Back_Serv[1]                  "Yes" ;
Three_Way_Call_Serv[1]             "Yes" ;
Three_Way_Conf_Serv[1]             "Yes" ;
Attn_Transfer_Serv[1]              "Yes" ;
Unattn_Transfer_Serv[1]            "Yes" ;
MWI_Serv[1]                        "Yes" ;
VMWI_Serv[1]                       "Yes" ;
Speed_Dial_Serv[1]                 "Yes" ;
Secure_Call_Serv[1]                "Yes" ;
Referral_Serv[1]                   "Yes" ;
Feature_Dial_Serv[1]               "Yes" ;

# *** Audio Configuration

Preferred_Codec[1]                 "G711u" ;  # options: G711u/G711a/G726-16/
G726-24/G726-32/G726-40/G729a/G723
Silence_Supp_Enable[1]             "No" ;
```

```
Use_Pref_Codec_Only[1]              "No" ;
Echo_Canc_Enable[1]                 "Yes" ;
G729a_Enable[1]                     "Yes" ;
Echo_Canc_Adapt_Enable[1]           "Yes" ;
G723_Enable[1]                      "Yes" ;
Echo_Supp_Enable[1]                 "Yes" ;
G726-16_Enable[1]                   "Yes" ;
FAX_CED_Detect_Enable[1]            "Yes" ;
G726-24_Enable[1]                   "Yes" ;
FAX_CNG_Detect_Enable[1]            "Yes" ;
G726-32_Enable[1]                   "Yes" ;
FAX_Passthru_Codec[1]               "G711u" ;  # options: G711u/G711a
G726-40_Enable[1]                   "Yes" ;
FAX_Codec_Symmetric[1]              "Yes" ;
DTMF_Tx_Method[1]                   "Auto" ;  # options: InBand/AVT/INFO/Auto
FAX_Passthru_Method[1]              "NSE" ;  # options: None/NSE/ReINVITE
Hook_Flash_Tx_Method[1]             "None" ;  # options: None/AVT/INFO
FAX_Process_NSE[1]                  "Yes" ;
Release_Unused_Codec[1]             "Yes" ;

# *** Dial Plan

Dial_Plan[1]                        "(*xx|[3469]11|0|00|[2-9]xxxxxx|1xxx[2-
9]xxxxxxS0|xxxxxxxxxxx.)" ;
Enable_IP_Dialing[1]                "No" ;

# *** FXS Port Polarity Configuration

Idle_Polarity[1]                    "Forward" ;  # options: Forward/Reverse
Caller_Conn_Polarity[1]             "Forward" ;  # options: Forward/Reverse
Callee_Conn_Polarity[1]             "Forward" ;  # options: Forward/Reverse

# *** Call Forward Settings

Cfwd_All_Dest[1]                  ! "" ;
Cfwd_Busy_Dest[1]                 ! "" ;
Cfwd_No_Ans_Dest[1]               ! "" ;
Cfwd_No_Ans_Delay[1]              ! "20" ;

# *** Selective Call Forward Settings

Cfwd_Sel1_Caller[1]               ! "" ;
Cfwd_Sel1_Dest[1]                 ! "" ;
Cfwd_Sel2_Caller[1]               ! "" ;
Cfwd_Sel2_Dest[1]                 ! "" ;
Cfwd_Sel3_Caller[1]               ! "" ;
Cfwd_Sel3_Dest[1]                 ! "" ;
Cfwd_Sel4_Caller[1]               ! "" ;
Cfwd_Sel4_Dest[1]                 ! "" ;
Cfwd_Sel5_Caller[1]               ! "" ;
Cfwd_Sel5_Dest[1]                 ! "" ;
Cfwd_Sel6_Caller[1]               ! "" ;
Cfwd_Sel6_Dest[1]                 ! "" ;
Cfwd_Sel7_Caller[1]               ! "" ;
Cfwd_Sel7_Dest[1]                 ! "" ;
```

```
Cfwd_Sel8_Caller[1]             ! "" ;
Cfwd_Sel8_Dest[1]               ! "" ;
Cfwd_Last_Caller[1]             ! "" ;
Cfwd_Last_Dest[1]               ! "" ;
Block_Last_Caller[1]            ! "" ;
Accept_Last_Caller[1]           ! "" ;

# *** Speed Dial Settings

Speed_Dial_2[1]                 ! "" ;
Speed_Dial_3[1]                 ! "" ;
Speed_Dial_4[1]                 ! "" ;
Speed_Dial_5[1]                 ! "" ;
Speed_Dial_6[1]                 ! "" ;
Speed_Dial_7[1]                 ! "" ;
Speed_Dial_8[1]                 ! "" ;
Speed_Dial_9[1]                 ! "" ;

# *** Supplementary Service Settings

CW_Setting[1]                   ! "Yes" ;
Block_CID_Setting[1]            ! "No" ;
Block_ANC_Setting[1]            ! "No" ;
DND_Setting[1]                  ! "No" ;
CID_Setting[1]                  ! "Yes" ;
CWCID_Setting[1]                ! "Yes" ;
Dist_Ring_Setting[1]            ! "Yes" ;
Secure_Call_Setting[1]            "No" ;

# *** Distinctive Ring Settings

Ring1_Caller[1]                 ! "" ;
Ring2_Caller[1]                 ! "" ;
Ring3_Caller[1]                 ! "" ;
Ring4_Caller[1]                 ! "" ;
Ring5_Caller[1]                 ! "" ;
Ring6_Caller[1]                 ! "" ;
Ring7_Caller[1]                 ! "" ;
Ring8_Caller[1]                 ! "" ;

# *** Ring Settings

Default_Ring[1]                 ! "1" ;  # options: 1/2/3/4/5/6/7/8
Default_CWT[1]                  ! "1" ;  # options: 1/2/3/4/5/6/7/8
Hold_Reminder_Ring[1]           ! "8" ;  # options: 1/2/3/4/5/6/7/8/none
Call_Back_Ring[1]               ! "7" ;  # options: 1/2/3/4/5/6/7/8
Cfwd_Ring_Splash_Len[1]         ! "0" ;
Cblk_Ring_Splash_Len[1]         ! "0" ;
VMWI_Ring_Splash_Len[1]         ! ".5" ;
VMWI_Ring_Policy[1]               "New VM Available" ;  # options: New VM
Available/New VM Becomes Available/New VM Arrives
Ring_On_No_New_VM[1]              "No" ;

# ***
```

```
Line_Enable[2]                    "Yes" ;
SAS_Enable[2]                     "No" ;
MOH_Server[2]                     "" ;
SAS_DLG_Refresh_Intvl[2]          "30" ;
NAT_Mapping_Enable[2]             "No" ;
SAS_Inbound_RTP_Sink[2]           "" ;
SIP_Port[2]                       "5061" ;
NAT_Keep_Alive_Enable[2]          "No" ;
EXT_SIP_Port[2]                   "" ;
NAT_Keep_Alive_Msg[2]             "$NOTIFY" ;
SIP_TOS/DiffServ_Value[2]         "0x68" ;
NAT_Keep_Alive_Dest[2]            "$PROXY" ;
RTP_TOS/DiffServ_Value[2]         "0xb8" ;
SIP_Debug_Option[2]               "none" ;  # options: none/1-line/1-line
excl. OPT/1-line excl. NTFY/1-line excl. REG/1-line excl. OPT|NTFY|REG/full/
full excl. OPT/full excl. NTFY/full excl. REG/full excl. OPT|NTFY|REG
Network_Jitter_Level[2]           "high" ;  # options: low/medium/high/very
high
SIP_100REL_Enable[2]              "No" ;
Blind_Attn-Xfer_Enable[2]         "No" ;
SIP_Proxy-Require[2]              "" ;
Auth_Resync-Reboot[2]             "Yes" ;
SIP_Remote-Party-ID[2]            "No" ;

# *** Proxy and Registration

Proxy[2]                          "" ;
Use_Outbound_Proxy[2]             "No" ;
Outbound_Proxy[2]                 "" ;
Use_OB_Proxy_In_Dialog[2]         "Yes" ;
Register[2]                       "Yes" ;
Make_Call_Without_Reg[2]          "No" ;
Register_Expires[2]               "3600" ;
Ans_Call_Without_Reg[2]           "No" ;
Use_DNS_SRV[2]                    "No" ;
DNS_SRV_Auto_Prefix[2]            "No" ;
Proxy_Fallback_Intvl[2]           "3600" ;

# *** Subscriber Information

Display_Name[2]                   "" ;
User_ID[2]                        "" ;
Password[2]                       "" ;
Use_Auth_ID[2]                    "No" ;
Auth_ID[2]                        "" ;
Mini_Certificate[2]               "" ;
SRTP_Private_Key[2]               "" ;

# *** Supplementary Service Subscription

Call_Waiting_Serv[2]              "Yes" ;
Block_CID_Serv[2]                 "Yes" ;
Block_ANC_Serv[2]                 "Yes" ;
Dist_Ring_Serv[2]                 "Yes" ;
Cfwd_All_Serv[2]                  "Yes" ;
```

```
Cfwd_Busy_Serv[2]                "Yes" ;
Cfwd_No_Ans_Serv[2]              "Yes" ;
Cfwd_Sel_Serv[2]                 "Yes" ;
Cfwd_Last_Serv[2]                "Yes" ;
Block_Last_Serv[2]               "Yes" ;
Accept_Last_Serv[2]              "Yes" ;
DND_Serv[2]                      "Yes" ;
CID_Serv[2]                      "Yes" ;
CWCID_Serv[2]                    "Yes" ;
Call_Return_Serv[2]              "Yes" ;
Call_Back_Serv[2]                "Yes" ;
Three_Way_Call_Serv[2]           "Yes" ;
Three_Way_Conf_Serv[2]           "Yes" ;
Attn_Transfer_Serv[2]            "Yes" ;
Unattn_Transfer_Serv[2]          "Yes" ;
MWI_Serv[2]                      "Yes" ;
VMWI_Serv[2]                     "Yes" ;
Speed_Dial_Serv[2]               "Yes" ;
Secure_Call_Serv[2]              "Yes" ;
Referral_Serv[2]                 "Yes" ;
Feature_Dial_Serv[2]             "Yes" ;

# *** Audio Configuration

Preferred_Codec[2]               "G711u" ;  # options: G711u/G711a/G726-16/
G726-24/G726-32/G726-40/G729a/G723
Silence_Supp_Enable[2]           "No" ;
Use_Pref_Codec_Only[2]           "No" ;
Echo_Canc_Enable[2]              "Yes" ;
G729a_Enable[2]                  "Yes" ;
Echo_Canc_Adapt_Enable[2]        "Yes" ;
G723_Enable[2]                   "Yes" ;
Echo_Supp_Enable[2]              "Yes" ;
G726-16_Enable[2]                "Yes" ;
FAX_CED_Detect_Enable[2]         "Yes" ;
G726-24_Enable[2]                "Yes" ;
FAX_CNG_Detect_Enable[2]         "Yes" ;
G726-32_Enable[2]                "Yes" ;
FAX_Passthru_Codec[2]            "G711u" ;  # options: G711u/G711a
G726-40_Enable[2]                "Yes" ;
FAX_Codec_Symmetric[2]           "Yes" ;
DTMF_Tx_Method[2]                "Auto" ;  # options: InBand/AVT/INFO/Auto
FAX_Passthru_Method[2]           "NSE" ;  # options: None/NSE/ReINVITE
Hook_Flash_Tx_Method[2]          "None" ;  # options: None/AVT/INFO
FAX_Process_NSE[2]               "Yes" ;
Release_Unused_Codec[2]          "Yes" ;

# *** Dial Plan

Dial_Plan[2]                     "(*xx|[3469]11|0|00|[2-9]xxxxxx|1xxx[2-
9]xxxxxxS0|xxxxxxxxxxx.)" ;
Enable_IP_Dialing[2]             "No" ;

# *** FXS Port Polarity Configuration
```

```
Idle_Polarity[2]                  "Forward" ;  # options: Forward/Reverse
Caller_Conn_Polarity[2]           "Forward" ;  # options: Forward/Reverse
Callee_Conn_Polarity[2]           "Forward" ;  # options: Forward/Reverse

# *** Call Forward Settings

Cfwd_All_Dest[2]                ! "" ;
Cfwd_Busy_Dest[2]               ! "" ;
Cfwd_No_Ans_Dest[2]             ! "" ;
Cfwd_No_Ans_Delay[2]            ! "20" ;

# *** Selective Call Forward Settings

Cfwd_Sel1_Caller[2]             ! "" ;
Cfwd_Sel1_Dest[2]               ! "" ;
Cfwd_Sel2_Caller[2]             ! "" ;
Cfwd_Sel2_Dest[2]               ! "" ;
Cfwd_Sel3_Caller[2]             ! "" ;
Cfwd_Sel3_Dest[2]               ! "" ;
Cfwd_Sel4_Caller[2]             ! "" ;
Cfwd_Sel4_Dest[2]               ! "" ;
Cfwd_Sel5_Caller[2]             ! "" ;
Cfwd_Sel5_Dest[2]               ! "" ;
Cfwd_Sel6_Caller[2]             ! "" ;
Cfwd_Sel6_Dest[2]               ! "" ;
Cfwd_Sel7_Caller[2]             ! "" ;
Cfwd_Sel7_Dest[2]               ! "" ;
Cfwd_Sel8_Caller[2]             ! "" ;
Cfwd_Last_Caller[2]             ! "" ;Cfwd_Sel8_Dest[2]               ! "" ;
Cfwd_Last_Dest[2]               ! "" ;
Block_Last_Caller[2]            ! "" ;
Accept_Last_Caller[2]           ! "" ;

# *** Speed Dial Settings

Speed_Dial_2[2]                 ! "" ;
Speed_Dial_3[2]                 ! "" ;
Speed_Dial_4[2]                 ! "" ;
Speed_Dial_5[2]                 ! "" ;
Speed_Dial_6[2]                 ! "" ;
Speed_Dial_7[2]                 ! "" ;
Speed_Dial_8[2]                 ! "" ;
Speed_Dial_9[2]                 ! "" ;

# *** Supplementary Service Settings

CW_Setting[2]                   ! "Yes" ;
Block_CID_Setting[2]            ! "No" ;
Block_ANC_Setting[2]            ! "No" ;
DND_Setting[2]                  ! "No" ;
CID_Setting[2]                  ! "Yes" ;
CWCID_Setting[2]                ! "Yes" ;
Dist_Ring_Setting[2]            ! "Yes" ;
Secure_Call_Setting[2]            "No" ;
```

```
# *** Distinctive Ring Settings

Ring1_Caller[2]                 ! "" ;
Ring2_Caller[2]                 ! "" ;
Ring3_Caller[2]                 ! "" ;
Ring4_Caller[2]                 ! "" ;
Ring5_Caller[2]                 ! "" ;
Ring6_Caller[2]                 ! "" ;
Ring7_Caller[2]                 ! "" ;
Ring8_Caller[2]                 ! "" ;

# *** Ring Settings

Default_Ring[2]                 ! "1" ;  # options: 1/2/3/4/5/6/7/8
Default_CWT[2]                  ! "1" ;  # options: 1/2/3/4/5/6/7/8
Hold_Reminder_Ring[2]           ! "8" ;  # options: 1/2/3/4/5/6/7/8/none
Call_Back_Ring[2]               ! "7" ;  # options: 1/2/3/4/5/6/7/8
Cfwd_Ring_Splash_Len[2]         ! "0" ;
Cblk_Ring_Splash_Len[2]         ! "0" ;
VMWI_Ring_Splash_Len[2]         ! ".5" ;
VMWI_Ring_Policy[2]               "New VM Available" ;  # options: New VM
Available/New VM Becomes Available/New VM Arrives
Ring_On_No_New_VM[2]              "No" ;

# *** Call Progress Tones

Dial_Tone                         "350@-19,440@-19;10(*/0/1+2)" ;
Second_Dial_Tone                  "420@-19,520@-19;10(*/0/1+2)" ;
Outside_Dial_Tone                 "420@-16;10(*/0/1)" ;
Prompt_Tone                       "520@-19,620@-19;10(*/0/1+2)" ;
Busy_Tone                         "480@-19,620@-19;10(.5/.5/1+2)" ;
Reorder_Tone                      "480@-19,620@-19;10(.25/.25/1+2)" ;
Off_Hook_Warning_Tone             "480@-10,620@0;10(.125/.125/1+2)" ;
Ring_Back_Tone                    "440@-19,480@-19;*(2/4/1+2)" ;
Confirm_Tone                      "600@-16;1(.25/.25/1)" ;
SIT1_Tone                        "985@-16,1428@-16,1777@-16;20(.380/0/1,.380/
0/2,.380/0/3,0/4/0)" ;
SIT2_Tone                        "914@-16,1371@-16,1777@-16;20(.274/0/1,.274/
0/2,.380/0/3,0/4/0)" ;
SIT3_Tone                        "914@-16,1371@-16,1777@-16;20(.380/0/1,.380/
0/2,.380/0/3,0/4/0)" ;
SIT4_Tone                        "985@-16,1371@-16,1777@-16;20(.380/0/1,.274/
0/2,.380/0/3,0/4/0)" ;
MWI_Dial_Tone                    "350@-19,440@-19;2(.1/.1/1+2);10(*/0/1+2)" ;
Cfwd_Dial_Tone                   "350@-19,440@-19;2(.2/.2/1+2);10(*/0/1+2)" ;
Holding_Tone                      "600@-19;*(.1/.1/1,.1/.1/1,.1/9.5/1)" ;
Conference_Tone                   "350@-19;20(.1/.1/1,.1/9.7/1)" ;
Secure_Call_Indication_Tone       "397@-19,507@-19;15(0/2/0,.2/.1/1,.1/2.1/
2)" ;

# *** Distinctive Ring Patterns

Ring1_Cadence                     "60(2/4)" ;
Ring2_Cadence                     "60(.3/.2,1/.2,.3/4)" ;
Ring3_Cadence                     "60(.8/.4,.8/4)" ;
```

```
Ring4_Cadence                       "60(.4/.2,.3/.2,.8/4)" ;
Ring5_Cadence                       "60(.2/.2,.2/.2,.2/.2,1/4)" ;
Ring6_Cadence                       "60(.2/.4,.2/.4,.2/4)" ;
Ring7_Cadence                       "60(.4/.2,.4/.2,.4/4)" ;
Ring8_Cadence                       "60(0.25/9.75)" ;

# *** Distinctive Call Waiting Tone Patterns

CWT1_Cadence                        "30(.3/9.7)" ;
CWT2_Cadence                        "30(.1/.1, .1/9.7)" ;
CWT3_Cadence                        "30(.1/.1, .3/.1, .1/9.3)" ;
CWT4_Cadence                        "30(.1/.1,.1/.1,.1/9.5)" ;
CWT5_Cadence                        "30(.3/.1,.1/.1,.3/9.1)" ;
CWT6_Cadence                        "30(.1/.1,.3/.2,.3/9.1)" ;
CWT7_Cadence                        "30(.3/.1,.3/.1,.1/9.1)" ;
CWT8_Cadence                        "2.3(.3/2)" ;

# *** Distinctive Ring/CWT Pattern Names

Ring1_Name                          "Bellcore-r1" ;
Ring2_Name                          "Bellcore-r2" ;
Ring3_Name                          "Bellcore-r3" ;
Ring4_Name                          "Bellcore-r4" ;
Ring5_Name                          "Bellcore-r5" ;
Ring6_Name                          "Bellcore-r6" ;
Ring7_Name                          "Bellcore-r7" ;
Ring8_Name                          "Bellcore-r8" ;

# *** Ring and Call Waiting Tone Spec

Ring_Waveform                       "Sinusoid" ;  # options: Sinusoid/Trapezoid
Ring_Frequency                      "25" ;
Ring_Voltage                        "70" ;
CWT_Frequency                       "440@-10" ;

# *** Control Timer Values (sec)

Hook_Flash_Timer_Min                ".1" ;
Hook_Flash_Timer_Max                ".9" ;
Callee_On_Hook_Delay                "0" ;
Reorder_Delay                       "5" ;
Call_Back_Expires                   "1800" ;
Call_Back_Retry_Intvl               "30" ;
Call_Back_Delay                     ".5" ;
VMWI_Refresh_Intvl                  "30" ;
Interdigit_Long_Timer               "10" ;
Interdigit_Short_Timer              "3" ;
CPC_Delay                           "2" ;
CPC_Duration                        "0" ;

# *** Vertical Service Activation Codes

Call_Return_Code                    "*69" ;
Blind_Transfer_Code                 "*98" ;
Call_Back_Act_Code                  "*66" ;
```

```
Call_Back_Deact_Code              "*86" ;
Cfwd_All_Act_Code                 "*72" ;
Cfwd_All_Deact_Code               "*73" ;
Cfwd_Busy_Act_Code                "*90" ;
Cfwd_Busy_Deact_Code              "*91" ;
Cfwd_No_Ans_Act_Code              "*92" ;
Cfwd_No_Ans_Deact_Code            "*93" ;
Cfwd_Last_Act_Code                "*63" ;
Cfwd_Last_Deact_Code              "*83" ;
Block_Last_Act_Code               "*60" ;
Block_Last_Deact_Code             "*80" ;
Accept_Last_Act_Code              "*64" ;
Accept_Last_Deact_Code            "*84" ;
CW_Act_Code                       "*56" ;
CW_Deact_Code                     "*57" ;
CW_Per_Call_Act_Code              "*71" ;
CW_Per_Call_Deact_Code            "*70" ;
Block_CID_Act_Code                "*67" ;
Block_CID_Deact_Code              "*68" ;
Block_CID_Per_Call_Act_Code       "*81" ;
Block_CID_Per_Call_Deact_Code     "*82" ;
Block_ANC_Act_Code                "*77" ;
Block_ANC_Deact_Code              "*87" ;
DND_Act_Code                      "*78" ;
DND_Deact_Code                    "*79" ;
CID_Act_Code                      "*65" ;
CID_Deact_Code                    "*85" ;
CWCID_Act_Code                    "*25" ;
CWCID_Deact_Code                  "*45" ;
Dist_Ring_Act_Code                "*26" ;
Dist_Ring_Deact_Code              "*46" ;
Speed_Dial_Act_Code               "*74" ;
Secure_All_Call_Act_Code          "*16" ;
Secure_No_Call_Act_Code           "*17" ;
Secure_One_Call_Act_Code          "*18" ;
Secure_One_Call_Deact_Code        "*19" ;
Referral_Services_Codes           "" ;
Feature_Dial_Services_Codes       "" ;

# *** Outbound Call Codec Selection Codes

Prefer_G711u_Code                 "*017110" ;
Force_G711u_Code                  "*027110" ;
Prefer_G711a_Code                 "*017111" ;
Force_G711a_Code                  "*027111" ;
Prefer_G723_Code                  "*01723" ;
Force_G723_Code                   "*02723" ;
Prefer_G726r16_Code               "*0172616" ;
Force_G726r16_Code                "*0272616" ;
Prefer_G726r24_Code               "*0172624" ;
Force_G726r24_Code                "*0272624" ;
Prefer_G726r32_Code               "*0172632" ;
Force_G726r32_Code                "*0272632" ;
Prefer_G726r40_Code               "*0172640" ;
Force_G726r40_Code                "*0272640" ;
```

```
Prefer_G729a_Code                  "*01729" ;
Force_G729a_Code                   "*02729" ;

# *** Miscellaneous

Set_Local_Date_(mm/dd)             "" ;
Set_Local_Time_(HH/mm)             "" ;
Time_Zone                          "GMT-07:00" ;  # options: GMT-12:00/GMT-
11:00/GMT-10:00/GMT-09:00/GMT-08:00/GMT-07:00/GMT-06:00/GMT-05:00/GMT-04:00/
GMT-03:30/GMT-03:00/GMT-02:00/GMT-01:00/GMT/GMT+01:00/GMT+02:00/GMT+03:00/
GMT+03:30/GMT+04:00/GMT+05:00/GMT+05:30/GMT+05:45/GMT+06:00/GMT+06:30/
GMT+07:00/GMT+08:00/GMT+09:00/GMT+09:30/GMT+10:00/GMT+11:00/GMT+12:00/
GMT+13:00
FXS_Port_Impedance                 "600" ;  # options: 600/900/600+2.16uF/
900+2.16uF/270+750||150nF/220+820||120nF/220+820||115nF/370+620||310nF
FXS_Port_Input_Gain                "-3" ;
FXS_Port_Output_Gain               "-3" ;
DTMF_Playback_Level                "-16" ;
DTMF_Playback_Length               ".1" ;
Detect_ABCD                        "Yes" ;
Playback_ABCD                      "Yes" ;
Caller_ID_Method                   "Bellcore(N.Amer,China)" ;  # options:
Bellcore(N.Amer,China)/DTMF(Finland,Sweden)/DTMF(Denmark)/ETSI DTMF/ETSI
DTMF With PR/ETSI DTMF After Ring/ETSI FSK/ETSI FSK With PR(UK)
FXS_Port_Power_Limit               "3" ;  # options: 1/2/3/4/5/6/7/8
Protect_IVR_FactoryReset           "No" ;
```

B

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)
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

C

Where to Go From Here

Cisco provides a wide range of resources to help you and your customer obtain the full benefits of the Cisco Small Business IP Telephony Device.

Product Resources

Resource	Location
Technical Documentation	Voice System (SPA400): www.cisco.com/en/US/products/ps10030/tsd_products_support_series_home.html Voice Gateways/Analog Telephone Adapters: www.cisco.com/en/US/products/ps10024/tsd_products_support_series_home.html IP Phones: www.cisco.com/en/US/products/ps10033/tsd_products_support_series_home.html
Firmware Downloads	Go to tools.cisco.com/support/downloads, and enter the model number in the Software Search box.
Cisco Community Central > Small Business Support Community	www.myciscocommunity.com/community/smallbizsupport/voiceandconferencing
Phone Support	www.cisco.com/en/US/support/tsd_cisco_small_business_support_center_contacts.html
Warranty and End User License Agreement	www.cisco.com/go/warranty

Resource	Location
Open Source License Notices	www.cisco.com/go/osln
Regulatory Compliance and Safety Information	See the Technical Documentation pages listed above.
Cisco Partner Central site for Small Business	www.cisco.com/web/partners/sell/smb
Cisco Small Business Home	www.cisco.com/smb