ADMINISTRATOR'S GUIDE

Software 4.1.6 | March 2014 | 3725-46307-001 Rev B

Polycom® VVX® Expansion Module

Addendum to the Polycom UC Software 4.1.0 Administrator's Guide

6001 America Center Drive
San Jose, CA 95002
USA

Trademarks

Polycom®, the Polycom logo and the names and marks associated with Polycom products are trademarks and/or service marks of Polycom, Inc. and are registered and/or common law marks in the United States and various other countries. All other trademarks are property of their respective owners. No portion hereof may be reproduced or transmitted in any form or by any means, for any purpose other than the recipient's personal use, without the express written permission of Polycom.

End User License Agreement

By installing, copying, or otherwise using this product, you acknowledge that you have read, understand and agree to be bound by the terms and conditions of the End User License Agreement for this product.

Patent Information

The accompanying product may be protected by one or more U.S. and foreign patents and/or pending patent applications held by Polycom, Inc.

Open Source Software Used in this Product

This product may contain open source software. You may receive the open source software from Polycom up to three (3) years after the distribution date of the applicable product or software at a charge not greater than the cost to Polycom of shipping or distributing the software to you. To receive software information, as well as the open source software code used in this product, contact Polycom by email at OpenSourceVideo@polycom.com.

Disclaimer

While Polycom uses reasonable efforts to include accurate and up-to-date information in this document, Polycom makes no warranties or representations as to its accuracy. Polycom assumes no liability or responsibility for any typographical or other errors or omissions in the content of this document.

Limitation of Liability

Polycom and/or its respective suppliers make no representations about the suitability of the information contained in this document for any purpose. Information is provided "as is" without warranty of any kind and is subject to change without notice. The entire risk arising out of its use remains with the recipient. In no event shall Polycom and/or its respective suppliers be liable for any direct, consequential, incidental, special, punitive or other damages whatsoever (including without limitation, damages for loss of business profits, business interruption, or loss of business information), even if Polycom has been advised of the possibility of such damages.

Customer Feedback

We are striving to improve our documentation quality and we appreciate your feedback. Email your opinions and comments to DocumentationFeedback@polycom.com.

Visit the Polycom Support Center for End User License Agreements, software downloads, product documents, product licenses, troubleshooting tips, service requests, and more.

Contents

About This Guide

This Administrator's Guide Addendum uses a number of conventions that help you to understand information and perform tasks.

Conventions Used in this Guide

This guide contains terms, graphical elements, and a few typographic conventions. Familiarizing yourself with these terms, elements, and conventions will help you successfully perform tasks.

Information Elements

This guide may include any of the following icons to alert you to important information.

Icons Used in this Guide

Name	*Icon*	*Description*
Note		The Note icon highlights information of interest or important information needed to be successful in accomplishing a procedure or to understand a concept.
Administrator Tip		The Administrator Tip icon highlights techniques, shortcuts, or productivity related tips.
Caution		The Caution icon highlights information you need to know to avoid a hazard that could potentially impact device performance, application functionality, or successful feature configuration.
Warning		The Warning icon highlights an action you must perform (or avoid) to prevent issues that may cause you to lose information or your configuration setup, and/or affect phone or network performance.
Web Info		The Web Info icon highlights supplementary information available online such as documents or downloads on support.polycom.com or other locations.
Timesaver		The Timesaver icon highlights a faster or alternative method for accomplishing a method or operation.
Power Tip		The Power Tip icon highlights faster, alternative procedures for advanced administrators already familiar with the techniques being discussed.
Troubleshooting		The Troubleshooting icon highlights information that may help you solve a relevant problem or to refer you to other relevant troubleshooting resources.
Settings		The Settings icon highlights settings you may need to choose for a specific behavior, to enable a specific feature, or to access customization options.

Typographic Conventions

A few typographic conventions, listed next, are used in this guide to distinguish types of in-text information.

Typographic Conventions

Convention	*Description*
Bold	Highlights interface items such as menus, soft keys, file names, and directories. Also used to represent menu selections and text entry to the phone.
Italics	Used to emphasize text, to show example values or inputs, and to show titles of reference documents available from the Polycom Support Web site and other reference sites.
Blue Text	Used for cross references to other sections within this document and for hyperlinks to external sites and documents.
`Courier`	Used for code fragments and parameter names.

Writing Conventions

Convention	*Description*
<MACaddress>	Indicates that you must enter information specific to your installation, phone, or network. For example, when you see *<MACaddress>*, enter your phone's 12-digit MAC address. If you see *<installed-directory>*, enter the path to your installation directory.
>	Indicates that you need to select an item from a menu. For example, **Settings > Basic** indicates that you need to select **Basic** from the **Settings** menu.
`parameter.*`	Used for configuration parameters. If you see a parameter name in the form `parameter.*` , the text is referring to all parameters beginning with `parameter`.

What's in This Guide?

This partner solution guide is organized into six sections. The first section, *Getting Started*, introduces the Polycom® VVX® Expansion Module. The sections following show you how to configure and deploy the VVX Expansion Module. The final sections show you where to get help and you troubleshoot with a list of known issues and workarounds.

Get Started This section contains introductory information on the Polycom VVX Expansion Module.

Power the VVX Expansion Module This section provides information on the power process and values for Polycom VVX business media phones and expansion modules.

Configure VVX Expansion Module Features In this section, you’ll learn how to configure features available on for the VVX Expansion Module.

Configuration Parameters This section provides a list of parameters with descriptions.

Get Help In this section, you’ll find links to Polycom, partner, and third-party documents and web sites. In particular, you’ll find links to the Polycom Community, a number of discussion forums you can use to share ideas with your colleagues.

Troubleshoot This section lists troubleshooting problems and common solutions.

Get Started

The Polycom VVX Expansion Module (EM) extends the functionality of Polycom VVX business media phones and provides multifunctional line keys that you can configure as line registrations, Presence, Favorites, or Busy Lamp Field features. Polycom offers the expansion module with a paper display and a color LCD display. The expansion module with a paper display has 40 line keys; the Color expansion module has 28 line keys with three pages, for a total of 84 line keys. With the addition of the expansion module, you can configure a maximum of 34 registrations on each phone. Note that you cannot mix paper display and color display expansion modules.

This guide provides information on the new features in Unified Communications (UC) software 4.1.6 and shows you how to configure the Polycom VVX expansion modules. The following phones support the expansion modules running UC software 4.1.6:

- VVX 300, 310, 400, 410 business media phones
- VVX 500 and 600 business media phones

What's New?

This release of the VVX expansion modules includes a number of hardware features, listed next.

VVX Color Expansion Module

- Contains 28 line keys per page with a maximum of 84 line keys per module.
- Supports multiple modules attached to a VVX phone with a maximum of three modules and 252 additional line keys.
- Includes page buttons to navigate between pages on each attached module.
- Includes a color LCD display.

VVX Expansion Module

- Contains 40 line keys per module.
- Supports multiple modules attached to a VVX phone with a maximum of three modules and 120 additional line keys.

VVX Expansion Module Features

The following features are available on VVX Expansion Modules with UC software 4.1.6:

- Line key monitoring
- Flexible Line Key Assignments
- Line appearances
- Favorites
- Busy Lamp Field
- Lync contacts and Presence**
- Enhanced Feature Keys

- Display background images for VVX phones and expansion modules**
- Status information for connected modules
- Line key PDF for expansion module with paper display

**These features are only available on the VVX Color Expansion Module.

Get Help and Support Resources

This guide includes a Get Help section where you can find links to Polycom product and support sites and partner sites. You can also find information about The Polycom Community, which provides access to discussion forums you can use to discuss hardware, software, and partner solution topics with your colleagues. To register with the Polycom Community, you will need to create a Polycom online account.

The Polycom Community includes access to Polycom support personnel, as well as user-generated hardware, software, and partner solutions topics. You can view top blog posts and participate in threads on any number of recent topics.

Power the VVX Expansion Module

The Polycom VVX Expansion Modules are powered and signaled by VVX business media phones and require minimal setup. The expansion modules are powered by VVX phones using an auxiliary cable that connects the modules and phone. After you connect the module to a phone, the module is automatically configured to work with the phone. Note that you cannot connect paper display and color display expansion modules together on the same phone.

Note: Sufficient Power for VVX Expansion Module

Powering the VVX expansion modules depends on the VVX phone's power management system. If the phone does not have the power capabilities to support an expansion module, a message displays on the phone after the module is connected. See the section Power Values for more information.

To connect the VVX expansion module to your VVX phone:

» Connect an auxiliary cable from the AUX port on the phone to the AUX IN port on the expansion module.

The LED lights on the module's line keys flash red and green as the module starts up. After the first module is on, you can connect up to two additional modules to your VVX phone.

To connect multiple VVX expansion modules:

1. Connect an auxiliary cable from the AUX Out port on the first module connected to the phone to the AUX In port on the second module.
2. Connect an auxiliary cable from the AUX Out port on the second module to the AUX In port on the third module.

   The LED lights on the line keys light up for each connected module as the expansion modules start up.

After you connect the expansion modules to a VVX phone, you can view information about and check the status of the connected expansion modules on your VVX phone. Expansion modules are listed in the Status menu in the order each module is connected to the phone. For example, EM1 is the first expansion module connected to the VVX phone.

To view the status of an expansion module on the phone:

1. Select **Settings > Status**.
2. In the **Status** menu, select the module you want to view.

   Information for the expansion module—the type of module, software version, assembly revision, and serial number—displays.

Power Values

The table Phone Power Values outlines the power usage for each phone, as well as the power value sent in LLDP-MED.

Phone Power Values

Model	*Power Usage (Watts)*	*Power Value Sent in LLDP-MED Extended Power Via MDI TLV*
VVX 300	5.0	5000mW
VVX 310	5.0	5000mW
VVX 400	5.0	5000mW
VVX 410	5.0	5000mW
VVX 500	8.0	8000mW
VVX 600	8.0	8000mW

Web Info: Power Consumption on Polycom Phones

For more detailed information about power consumption on Polycom phones, see *Engineering Advisory 48152: Power Consumption on Polycom Phones*.

Note: Default Power Values

By default, the power values for VVX 300, 310, 400, 410, 500, and 600 are sent for the phone and the expansion module(s). The values are not adjusted when the expansion module(s) are detached from the phone.

Configure VVX Expansion Module Features

You can configure features for the VVX expansion module using the phone's interface, the Web Configuration Utility, or XML configuration files. Using the Web Configuration Utility, you can configure features and settings for your phone and expansion modules remotely on a per-phone basis. You can also assign lines to contacts, configure line functions, upload background images, and add or update contacts' profile pictures. Additionally, you can use Polycom's XML configuration files to configure multiple phones and expansion modules at one time.

Web Info: Polycom Configuration User Guides and Best Practices

For instructions on using the Web Configuration Utility, see the *Polycom Web Configuration Utility User's Guide*.
For information on mass provisioning, read *Provisioning with the Master Configuration File* Best Practices.

Web Info: Using the VVX Expansion Modules

You can read more information on using the VVX Expansion Modules and adding contacts on the modules in *Feature Profile: Using Polycom VVX Expansion Modules with Polycom VVX Business Media Phones*.

The following sections cover features you can configure for VVX phones and expansion modules:

- Set Display Backgrounds
- Assign Flexible Line Key Functions
- Customize Enhanced Feature Keys
- Configure Lync Presence
- Generate Configured Line Key Information

Set Display Backgrounds

You can set display backgrounds for your VVX phone and Color expansion module on your phone or by using the Web Configuration Utility or XML configuration files. You can display an image or a design for the background on VVX 300, 400, 500, and 600 phones.

The VVX phones display a default background picture. You can select your own background picture or design, or you can import a custom image. You can also select images from the Picture Frame on the VVX 500 and 600 phones (see Configuring the Digital Picture Frame in the *Polycom UC Software 4.1.0 Administrator's Guide*).

The table Setting Display Backgrounds explains the methods for setting background images and provides links to parameters and definitions in the section Configuration Parameters. Note that whereas an idle display image displays on a portion of the phone's screen (see Adding an Idle Display Image in the

Polycom UC Software 4.1.0 Administrator’s Guide), a background image displays on the entire screen, and the time, date, and line and soft key labels display over the backgrounds.

Web Info: Adding a Graphic Display Background

For detailed instructions on adding a display background to a VVX phone, see *Technical Bulletin 62470: Customizing the Display Background on Your Polycom VVX Business Media Phone*.

Setting Display Backgrounds

Central Provisioning Server ....template > parameter

Specify a background to display for your phone type .... **features.cfg >** bg.*

Web Configuration Utility

Specify which background to display by navigating to **Preferences > Background**.

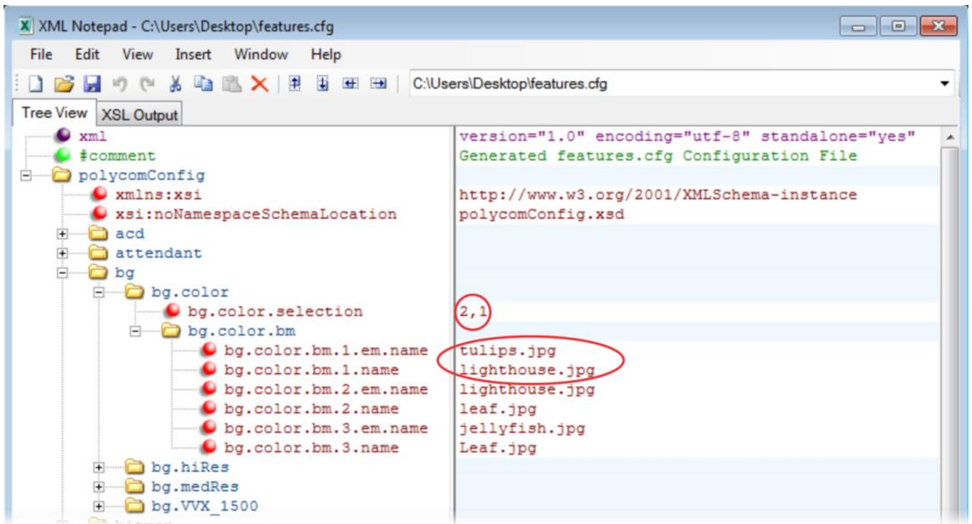
Local Phone User Interface

To select a background, on the phone, select **Settings > Basic > Preferences > Background > Select Background**.

On the VVX 500 and 600, you can save a Picture Frame image as the background by selecting **Save as Background** on the touch screen (see Configuring the Digital Picture Frame in the *Polycom UC Software 4.1.0 Administrator’s Guide*).

Example Graphic Display Background Configuration

This example configuration shows a background image applied to a VVX phone and expansion module. The default background in the features.cfg template file, specified in the `bg.color.selection` parameter, is set to `2,1`, where 2 enables background images and 1 selects the image. For example, 1= `bg.color.bm.1.em.name` and `bg.color.bm.1.name`. The phone displays the background image, in this case `lighthouse.jpg`, and the expansion module displays the `tulips.jpg`.



This example configuration results in the following graphic display background on the phone and expansion module's screens. Notice that line and soft key labels display over the background images.

Assign Flexible Line Key Functions

You can customize the function of a line key anywhere on the phone's screen and expansion modules. By default, functions are assigned to line keys in chronological order. This feature enables you to change that ordering and assign a line key function to any line key in any order anywhere on the phone's screen or expansion module. You can configure the following flexible line key functions:

- **Line Appearance (Registrations)** Allows a line extension or phone number to occupy multiple line keys on a single phone.
- **Favorites** Enables you to customize line, hard, and soft keys functions.
- **Busy Lamp Field (BLF)** Enables you to monitor and control the status and call activity of lines on remote phones.
- **Presence** Enables you to monitor the status of other remote users and phones.

You can configure line keys on the user interface, using the Web Configuration Utility, or using Polycom configuration files in XML format.

Settings: Configuring Flexible Line Key Assignments on Expansion Modules

To configure the Flexible Line Key assignment feature on the expansion modules, you must set the parameter `lineKey.reassignment.enabled` to `1` to enable the reassignment of line key functions. The default value for `lineKey.reassignment.enabled` is `0`, which does not allow any line key reassignments.

The table Flexible Line Key Assignment lists the configuration parameters you need to configure to assign flexible line key functions.

Flexible Line Key Assignment

Central Provisioning Server	**template** > **parameter**
Enable flexible line key assignment	**reg-advanced.cfg** > lineKey.reassignment.enabled
Specify the line key category	**reg-advanced.cfg** > lineKey.x.category1
Specify the line key number (dependent on category)	**reg-advanced.cfg** > lineKey.x.index

You can apply flexible line keys to any line key function including line appearance, Favorites, BLF, and Presence. Line keys that you configure using this feature override the default line key assignments as well as any custom line key configurations you have made.

To use this feature, you need to specify the function of each line key on the phone by assigning a category (x) and an index (y) to each line key, both of which are explained in the table Line Key Parameters and the following example configuration.

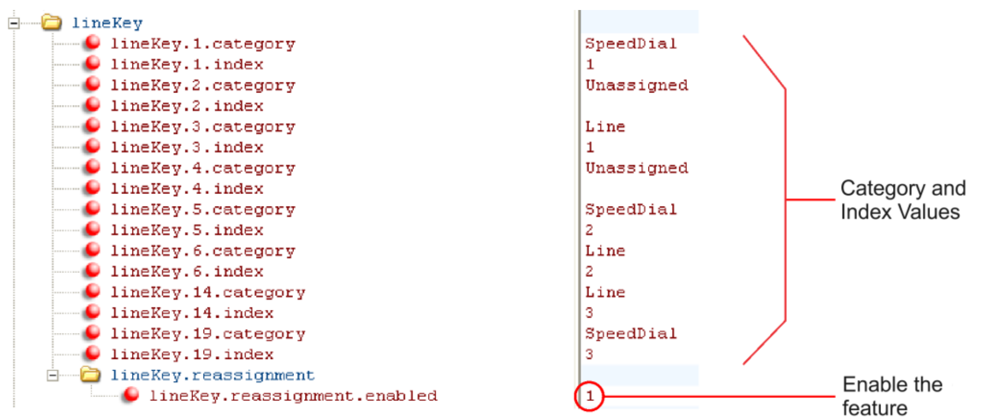
Specific conditions apply when you assign BLF or Presence to line keys. If you are assigning BLF or Presence to a line key, you need to assign that line key to index=0 to indicate automatic ordering. By default, BLF and Presence line keys are self-ordering, meaning that if you assign these features to multiple line keys, you can specify the line key number of the BLF or Presence line key but not the order in which they display. For example, you can assign a BLF line key to indexes 1, 3, and 5, but you cannot specify how the contacts are ordered and displayed on line keys 1, 3, and 5. In addition, to assign BLF and Presence to a line key, you need to assign a corresponding registration line. You can configure multiple line keys per registration if each line key has a corresponding `reg.x.lineKeys` parameter.

To enable flexible line key assignment, In the **features.cfg** template, set the `lineKey.reassignment.enabled` parameter to 1. Then assign each line key a category and an index. The category specifies the function of the line key and can include the following: Unassigned, Line, BLF, SpeedDial (Favorites), and Presence. Note that the Unassigned category leaves that line key blank.

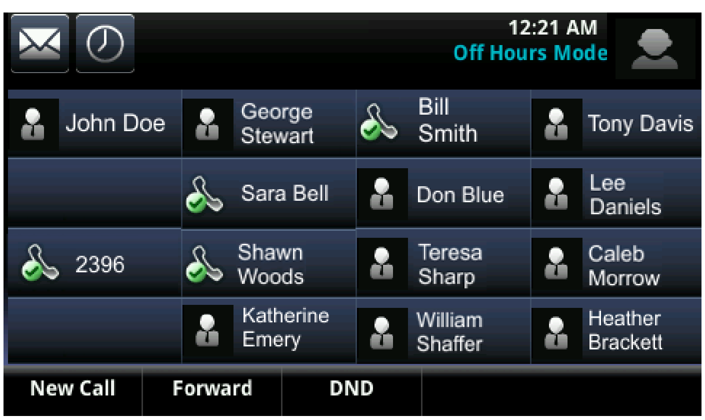
The index specifies the order in which the line keys display on the phone screen. Use the following figure to help you assign a category and an index to the line keys on your phone.

The following illustration shows an example Flexible Line Key assignment configuration in the features.cfg template file.

Example Flexible Line Key Assignment Configuration



This configuration displays on a VVX phone, as shown in the next figure.



Note: Line Keys are Numbered Sequentially

Line keys on VVX phones and expansion modules are numbered sequentially, and the line keys on your expansion module depends on how many lines your phone supports. For example, a VVX 600 phone supports 16 lines, numbered 1-16. The first line on an expansion module connected to a VVX 600 phone is line 17.

Enable Multiple Line Registrations

Polycom phones can have multiple registrations; each registration requires an address or phone number. All VVX phones support up to 48 registration line keys and up to 34 line registrations when connected to three expansion modules.

You can assign each registration to one or more line keys. Note that you can use a line key for only one registration. You can select which registration to use for outgoing calls. This feature is one of several features associated with Flexible Call Appearances.

Enabling Multiple Registrations

Central Provisioning Server ....**template** > **parameter**

Specify the local SIP signaling port and several optional SIP servers to register to. For each server, specify the registration period and the signaling failure behavior.

.... **sip-interop.cfg** > voIpProt.SIP.* and voIpProt.server.x.*

Specify a display name, a SIP address, an optional display label, an authentication user ID and password, the number of line keys to use, and an optional array of registration servers. The authentication user ID and password are optional and for security reasons can be omitted from the configuration files. The local flash parameters will be used instead. The optional array of servers and their parameters will override the servers specified in `<voIpProt.server/>` if non-Null.

.... **reg-basic.cfg, reg-advanced.cfg** > reg.x.*

Web Configuration Utility

Specify the local SIP signaling port and several optional SIP servers to register to.

Specify a display name, a SIP address, an optional display label, an authentication user ID and password, the number of line keys to use, and an optional array of registration servers. The authentication user ID and password are optional and for security reasons can be omitted from the configuration files. The local flash parameters are used instead. The optional array of servers will override the servers specified in `<server/>` if non-Null.

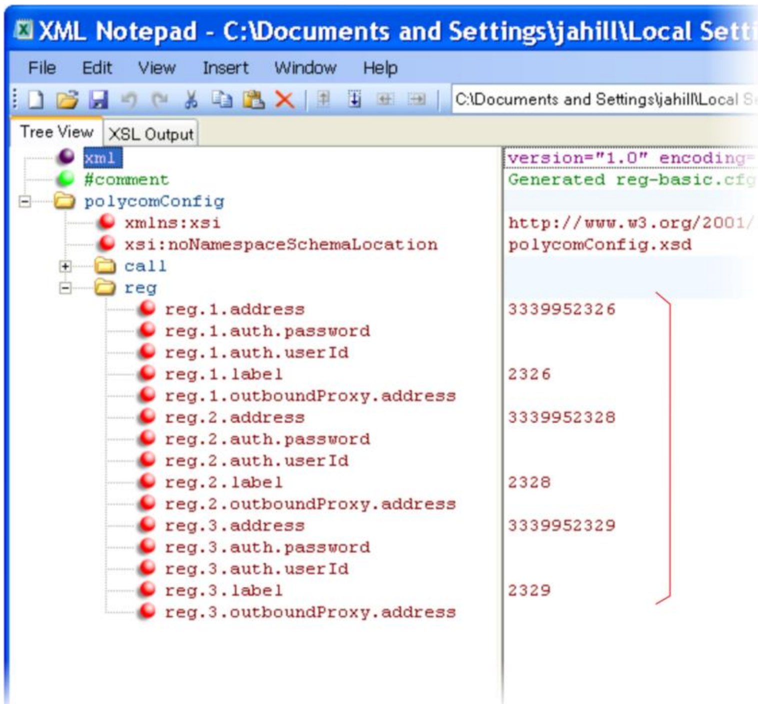
Configure multiple registrations by navigating to **Settings > Lines**.

Local Phone User Interface

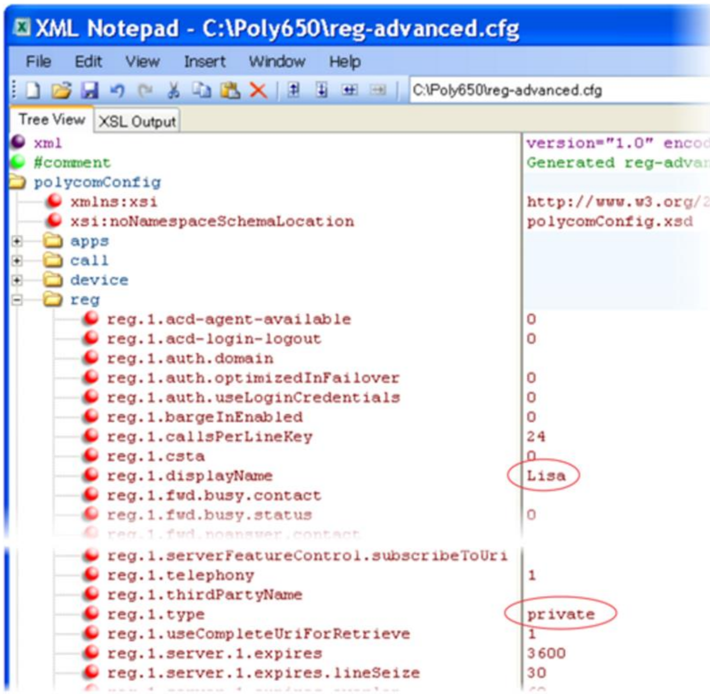
Use the **Call Server Configuration** and **Line Configuration** menu to specify the local SIP signaling port, a default SIP server to register to, and registration information for up to 12 registrations (depending on the phone model). These configuration menus contain a subset of all the parameters available in the configuration files.

Example Multiple Registration Configuration

The following figure shows an example configuration in the reg-basic.cfg template. Multiple line registrations and a label for each registration have been enabled for lines 1, 2, and 3.

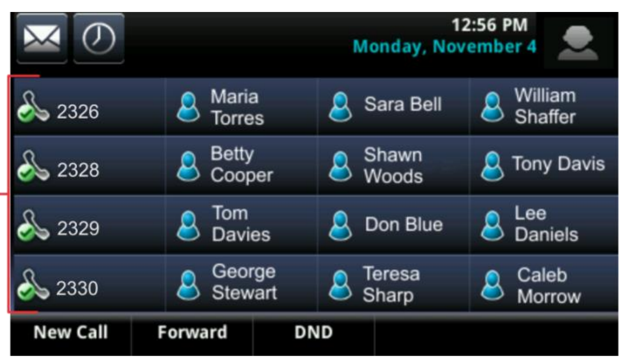


In the reg-advanced.cfg template shown next, when you make a call using line 1, the name you enter in `reg.1.displayname` displays as your caller ID, in this case *Lisa*. The parameter `reg.x.type` is set to the default `private`, which indicates that the registration uses standard call signaling.



This configuration results in the following registrations on a VVX 600 phone.

Registrations on Line Keys



Use the Favorites Feature

You can link entries in your local contact directory to favorites on the phone and modules. The Favorites feature enables you to place calls quickly using dedicated line keys. To set up favorites through the phone's contact directory, see Using the Local Contact Directory in *Polycom UC Software 4.1.0 Administrator's Guide*.

The Favorites' index range is 1 to 9999 for VVX phones.

On some call servers, enabling Presence for an active Favorites contact displays that contact's status on the favorite's line key label. For information on how to enable Lync Presence for contacts, see Configure Lync Presence.

Configuring the Favorites Feature

Central Provisioning Server .... template

Enter a favorites index number in the `<sd>x</sd>` element in the <*MAC address*>-directory.xml file to display a contact directory entry as a Favorites key on the phone. Favorites are assigned to unused line keys and to entries in the phone's Favorites list in numerical order.

The template contact directory file .... **000000000000-directory~.xml**

Local Phone User Interface

New directory entries are assigned to the Favorites Index in numerical order. To assign a Favorites index to a contact, navigate go to the **Contact Directory**, highlight the contact, press the **Edit** soft key, and specify a **Favorites Index**.

Power Tip: Quick Access to the Favorites List

To access the Favorites list quickly, press the phone's Up arrow key from the idle display.

The Favorites' configuration is explained briefly in the following table. To set up Favorites, use the table Local Directory Parameters for Setting Up Favorites Contacts, which identifies the parameters you need to set up your favorites.

Local Directory Parameters for Setting Up Favorites Contacts

Element	*Definition*	*Permitted Values*
fn	**First Name**	**UTF-8 encoded string of up to 40 bytes[1]**
The contact's first name.		
ln	**Last Name**	**UTF-8 encoded string of up to 40 bytes[1]**
The contact's last name.		
ct	**Contact**	**UTF-8 encoded string containing digits (the user part of a SIP URL) or a string that constitutes a valid SIP URL**
Used by the phone to address a remote party in the same way that a string of digits or a SIP URL are dialed manually by the user. This element is also used to associate incoming callers with a particular directory entry. The maximum field length is 128 characters. Note: This field cannot be Null or duplicated.		
sd	**Favorites Index**	**Null, 1 to 9999**
Associates a particular entry with a Favorites key for one-touch dialing or dialing from the Favorites menu.		

Element	*Definition*	*Permitted Values*
lb	**Label**	**UTF-8 encoded string of up to 40 bytes[1]**
The label for the contact. Note: The label of a contact directory item is by default the label attribute of the item. If the label attribute does not exist or is Null, then the first and last names form the label. A space is added between first and last names.		
pt	**Protocol**	**SIP, H323, or Unspecified**
The protocol to use when placing a call to this contact.		
rt	**Ring Tone**	**Null, 1 to 21**
When incoming calls match a directory entry, this field specifies the ringtone used.		
dc	**Divert Contact**	**UTF-8 encoded string containing digits (the user part of a SIP URL) or a string that constitutes a valid SIP URL**
The address to forward calls to if the Auto Divert feature is enabled.		
ad	**Auto Divert**	**0 or 1**
If set to 1, callers that match the directory entry are diverted to the address specified for the divert contact element. Note: If Auto Divert is enabled, it has precedence over Auto Reject.		
ar	**Auto Reject**	**0 or 1**
If set to 1, callers that match the directory entry specified for the Auto Reject element are rejected. Note: If Auto Divert is also enabled, it has precedence over Auto Reject.		
bw	**Buddy Watching**	**0 or 1**
If set to 1, this contact is added to the list of watched phones.		
bb	**Buddy Block**	**0 or 1**
If set to 1, this contact is blocked from watching this phone.		

[1] In some cases, this is less than 40 characters due to UTF-8’s variable bit length encoding.

Example Favorites Configuration

The first time you deploy and reboot the phones with UC software, a template contact directory file named 00000000000-directory~.xml is loaded to the provisioning server. You can edit and use this template file as a global contact directory for a group of phones, or you can create your own per-phone directory file.

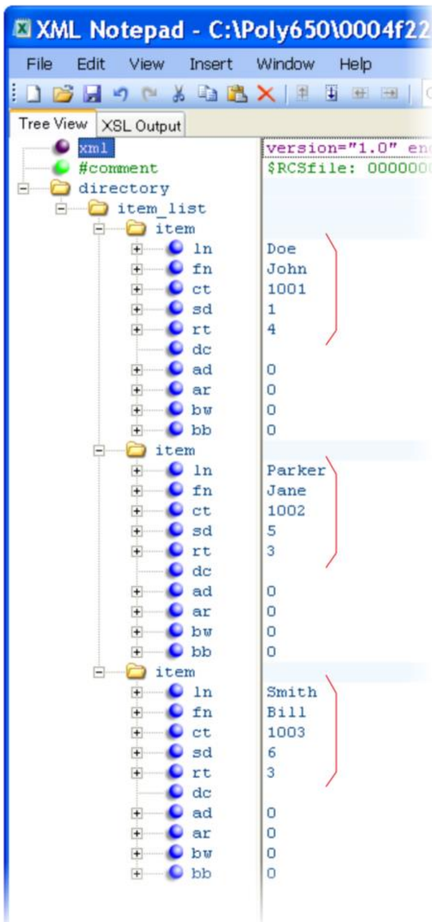
To create a global directory, locate the **00000000000-directory~.xml** template in your UC Software files and remove the tilde (~) from the file name. When you reboot the phone, the phone substitutes the global file with its own ***<MACaddress>*-directory.xml**, which is uploaded to the server. If you want to create a per-phone directory, replace ***<000000000000>*** in the global file name with the phone’s MAC address, for example, ***<MACaddress>*-directory.xml**.

On each subsequent reboot, the phone looks for its own ***<MACaddress>*-directory.xml** then looks for the global directory. Contact directories stored locally on the phone can override the ***<MACaddress>*-directory.xml** on the server depending on your server configuration. The phone always looks for a local directory or ***<MACaddress>*-directory.xml** before looking for the global directory.

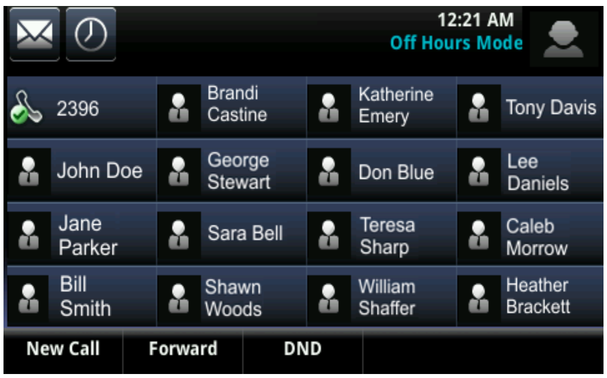
For more information on how to use the template directory file **000000000000-directory~.xml**, see Using the Local Contact Directory in *Polycom UC Software 4.1.0 Administrator's Guide*.

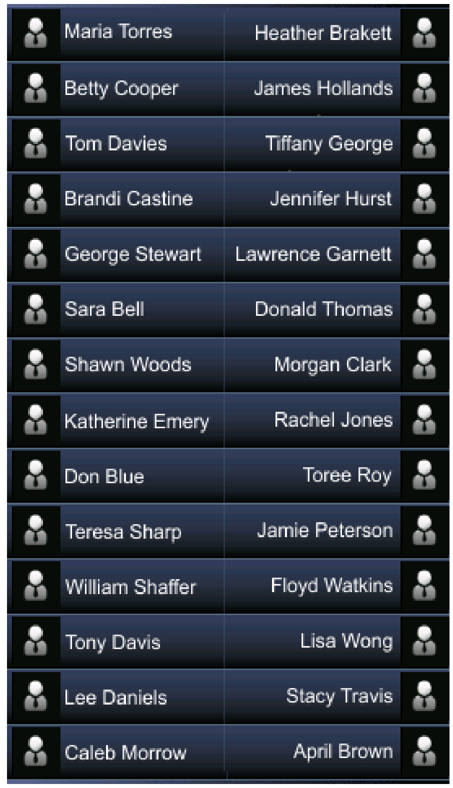
After you have renamed the directory file as a per-phone directory, enter a number in the **Favorites <sd>** field to display a contact directory entry as a favorite on the phone. Favorite entries automatically display on unused line keys on the phone and modules, and the favorites are assigned in numerical order.

The example local contact directory file shown next is saved with the phone's MAC address and shows the contact *John Doe* with extension number *1001* as favorite entry 1 on the phone.



This configuration results in the following favorites keys on the phone and module.





Use Busy Lamp Field Feature

Use the Busy Lamp Field (BLF) feature to monitor the status of lines on remote phones, display remote party information, and answer incoming calls to remote phones on your VVX phone and expansion modules.

The BLF feature must be supported by a call server, and the specific functions vary with the call server you use. Consult with your SIP server partner or Polycom channel partner to find out how to configure BLF. Note that BLF is not available with Polycom phones registered with Microsoft Lync Server.

The BLF feature offers the following functions:

- Visual and audible indications when monitored BLF lines have incoming calls, calls on hold, or are busy
- Caller ID of incoming calls to remote monitored phones
- Pickup soft key that you can press to answer incoming calls to monitored resources
- A list of monitored contacts for a maximum of 47 contacts with configured line key labels
- Configurable key functions
- Ability to disable spontaneous call appearances from incoming calls on monitored lines

The following call servers support BLF:

- Back-to-Back2 User Agent (B2BUA) architecture
  - ➢ Metaswitch Metasphere call feature server (CFS)

- Asterisk® v1.6 or later
- BroadSoft® BroadWorks

- Proxy architecture
  - Avaya® SipX Enterprise Communications Server (ECS)
  - eZuce openUC™

  These proxy architectures may support the full range of statically configured BLF features. However, they do not provide configuration control through their web management console.

The following call servers can support this feature, depending on the call server software variation and deployment:

- Proxy architecture
  - OpenSIPS (formerly OpenSER)
  - Repro ReSIProcate

  These proxy architectures or any other proxy server that allows the phone end-to-end communications with the monitored phone can be supported. However, these solutions have not been specifically tested by Polycom nor does Polycom guarantee their full interoperability.

Settings: Use BLF With TCPpreferred Transport

Use this feature with TCPpreferred transport (see <server/>).

The table Configuring the Busy Lamp Field lists the parameters you need to set BLF. You can configure the following functions for the BLF feature:

- Multiple BLF lines
- Monitoring of remote phones in active, ringing, and idle states

  When BLF is enabled and you are monitoring a remote user, a BLF line key icon displays on the phone's screen.
- The display of line key labels, call appearances, and caller ID information
- One-touch call park and retrieve and one-touch directed call pickup
- The type of monitored resource as normal or automata and the default actions of key presses

  As the resource type, enter `normal` if the monitored resource type is a phone and `automata` if the monitored resource type is, for example, a call orbit. If you select `normal`, pressing the BLF line key places an active call on hold before dialing the selected BLF phone. If you select `automata`, pressing the BLF line key immediately transfers active calls to that resource. To learn how to configure a park orbit and for examples, see Customize Enhanced Feature Keys.

Note that how you manage calls on BLF lines depends on the state of your phone—whether it is in the idle, active, or alerting state.

Web Info: Managing Monitored Lines

For information on how to manage calls to monitored phones, see the section Handling Remote Calls on Attendant Phones in *Technical Bulletin 62475: Using Statically Configured Busy Lamp Field with Polycom® SoundPoint IP Phones*.

Configuring the Busy Lamp Field Feature

Central Provisioning Server	template > parameter
Specify an index number for the BLF resource	**features.cfg** > attendant.reg
Specify the ringtone to play when a BLF dialog is in the offering state	**features.cfg** > attendant.ringType
Specify the SIP URI of the call server resource list	**features.cfg** > attendant.uri
Specify how call appearances and remote party caller ID display on the attendant phone	**features.cfg** > attendant.behaviours.display.*
Specify the address of the monitored resource, a label for the resource, and the type of resource	**features.cfg** > attendant.resourceList.*

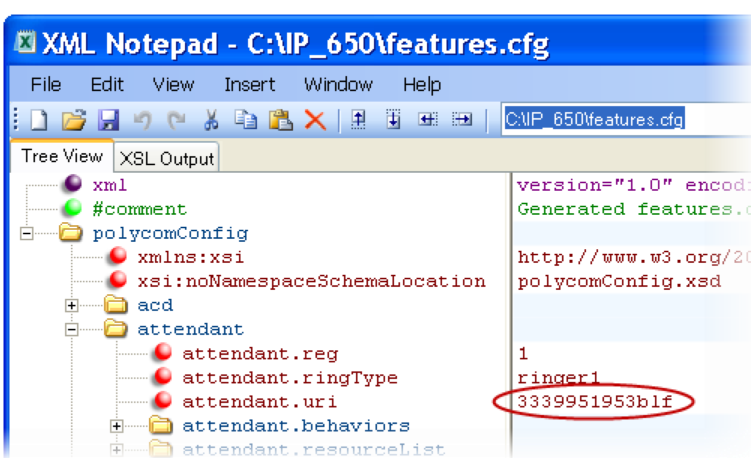
Example BLF Configuration

Typically, call servers support one of two methods of BLF configuration:

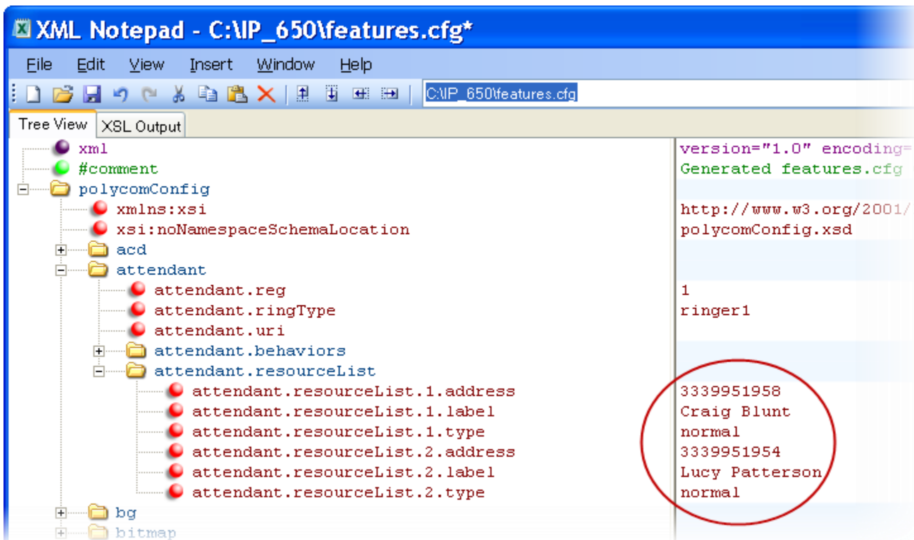
- Subscribing to a BLF resource list that is set up on your call server
- Entering BLF resources to a configuration file; the call server then directs the requests to those BLF resources

If you are unsure of which method to use, consult your SIP server partner or Polycom Channel partner. This section shows you how to set up BLF using both methods.

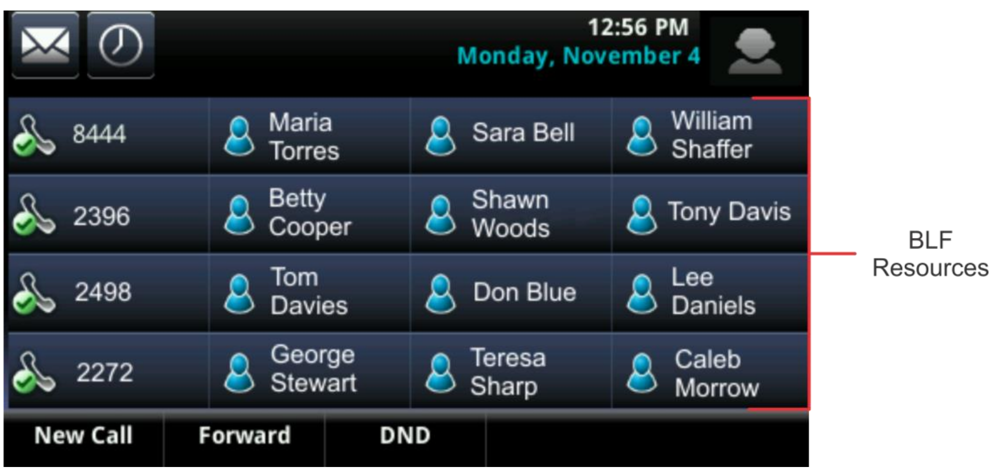
To subscribe to a BLF list on a call server, you need to access the call server and set up a list of monitored resources. The call server provides you with an address for that BLF resource list. To subscribe to that list, enter the address and any other information specific to your call server in the **`attendant.uri`** field located in the features.cfg template file, as shown next.



To specify BLF resources in the configuration file, open the **features.cfg** template file and enter the address (phone number) of the BLF resource you want to monitor, the label to display beside the line key on the phone, and the type of resource you are monitoring. Your call server must support static BLF in order to configure BLF using the static method. In the following example, the phone is monitoring *Craig Blunt* and *Lucy Patterson*.

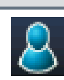


Both configuration methods result in the following BLF contacts, called BLF resources, that display on the phone and beside line keys on the expansion module.



The following table illustrates the BLF key icons.

BLF Line Key Icons

States	*Line Icons*
Line monitoring is active	

States	*Line Icons*
Monitored line is busy	
Monitored line is ringing	

Configure Lync Presence

The Lync presence feature enables you to monitor the status of other remote users and phones. By adding remote users to your Buddy List, you can monitor changes in the status of remote users in real time or you can monitor remote users as Favorites on the VVX phone and expansion module. The table Using the Presence Feature lists the parameters you can configure. Note that other phone users can block you from monitoring their phones.

Note: Lync Not Supported on VVX Expansion Modules with the Paper Display

The VVX Expansion Modules with paper displays do not support Lync, and you cannot configure Lync features to work on the expansion modules with paper displays. You can only configure VVX Color expansion modules to work with Lync.

For more information about the Lync presence feature, see *Feature Profile 84538: Using Polycom VVX Business Media Phones with Microsoft Lync Server 2013*.

Configuring the Lync Presence Feature

Central Provisioning Server ....template > parameter

Specify the line/registration number used to send SUBSCRIBE for presence .... **features.cfg** > pres.reg

Specify if the MyStatus and Buddies soft keys display on the Home screen .... **features.cfg** > pres.idleSoftkeys

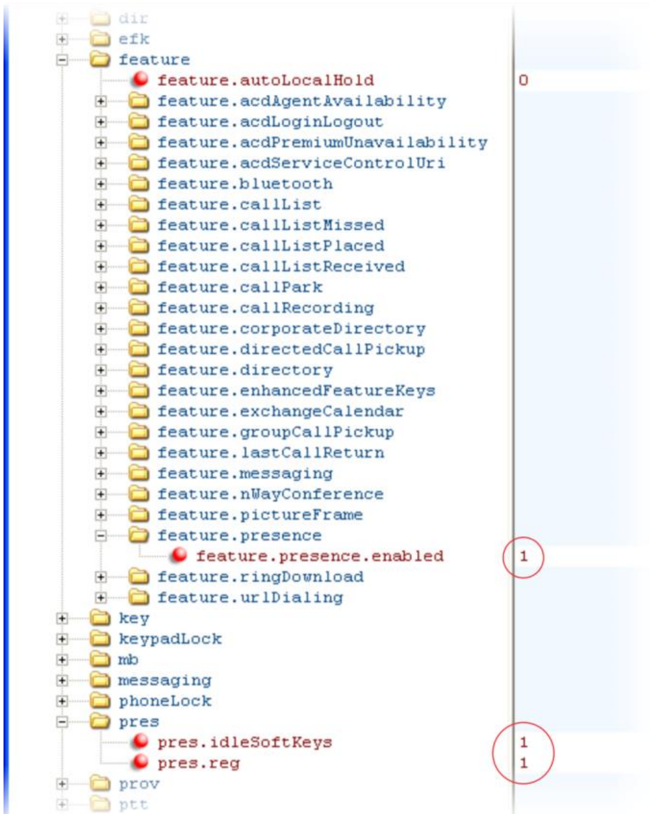
Turn the presence feature on or off .... **features.cfg** > feature.presence.enabled

Local Phone User Interface

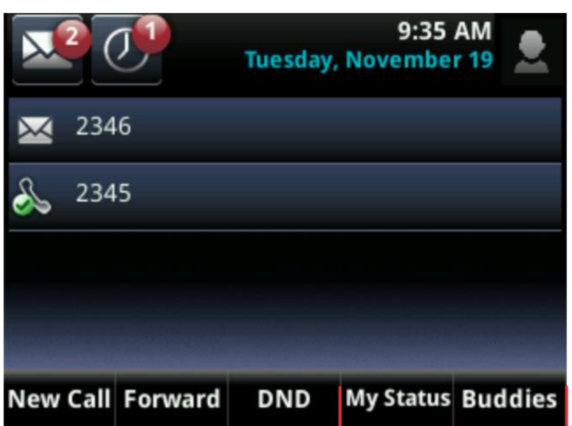
You can edit the directory contents. The *Buddy Watch* and *Buddy Block* fields control the buddy behavior of contacts.

Example Presence Configuration

In the following illustration, the presence feature is enabled in `feature.presence.enabled`. The MyStatus and Buddies soft keys both display on the phone's home screen when you enable the `pres.idleSoftkeys` parameter. The `pres.reg` parameter uses the address of phone line 1 for the presence feature.

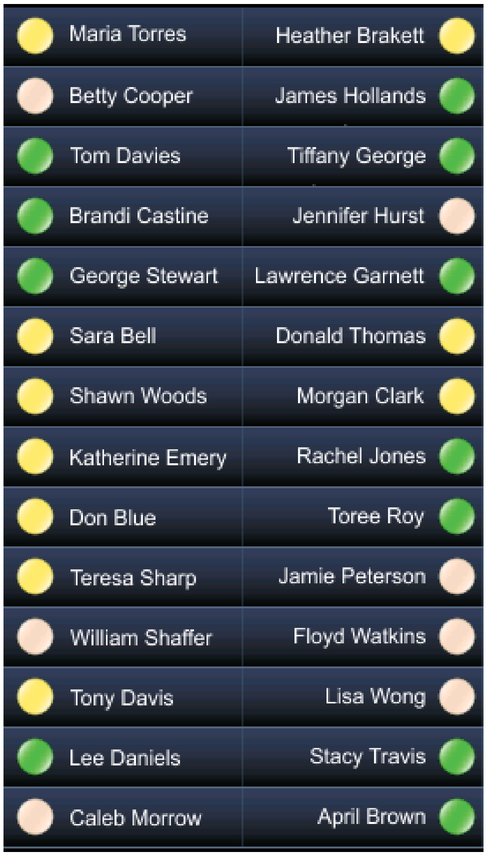


This configuration enables the presence feature and displays the MyStatus and Buddies soft keys on the phone, as shown next. When you press the Buddies soft key, contacts you have entered to your Buddy List display.



Presence Soft Keys

The following figure shows Lync Presence contacts on the Color expansion module.



The following table shows the Lync presence icons that display on the VVX phone and expansion module.

Lync Presence Icons

Icons	*Description*
	Available
	Busy, In a Call
	Away, Be Right Back, Inactive
	Do Not Disturb
	Offline
	Unknown

Customize Enhanced Feature Keys

The Enhanced Feature Keys (EFK) feature enables you to customize the functions of any line keys, soft keys, and hard keys on VVX phones and expansion modules. You can use EFK to assign frequently used functions to line keys, soft keys, and hard keys or to create menu shortcuts to frequently used phone settings on VVX 300, 310, 400, 410, 500, and 600 phones running UC software 4.1.6 or later.

See the table Configuring Enhanced Feature Keys for parameters you can configure and a brief explanation of how to use the contact directory to configure line keys. Enhanced feature key functionality is implemented using star code sequences (for example, *69) and SIP messaging. Star code sequences that define EFK functions are written as macros that you apply to line and soft keys.

The rules for configuring EFK for line keys, soft keys, and hard keys are different. Before using EFK, you are advised to become familiar with the macro language and parameters shown in the <efk/> section. For more information on configuring enhanced feature keys and using macros, see Understanding Macro Definitions in the *Polycom UC Software 4.1.0 Administrator's Guide*.

Web Info: Using Enhanced Feature Keys

For instructions and details on how to use Enhanced Feature Keys, refer to *Technical Bulletin 42250: Using Enhanced Feature Keys and Configurable Soft Keys on SoundPoint IP, SoundStation IP, and VVX 1500 Phones*.

Note that you can include the configuration file changes and the Enhanced Feature Key definitions together in one configuration file. Polycom recommends creating a new configuration file to make configuration changes.

Configuring Enhanced Feature Keys

Central Provisioning Server	template > parameter
Specify at least two calls per line key	**reg-basic.cfg** > reg.x.callsPerLineKey
Enable or disable Enhanced Feature Keys	**features.cfg** > feature.enhancedFeatureKeys.enabled
Specify the EFK List parameters	**features.cfg** > efk.efklist.x.*
Specify the EFK Prompts	**features.cfg** > efk.efkprompt.x.*
Because line keys and their functions are linked to fields in the contact directory file - **000000000000-directory.xml** (global) or **<MACaddress>-directory.xml** (per phone) - you need to match the contact field (ct) in the directory file to the macro name field (mname) in the configuration file that contains the EFK parameters. When you enter macro names to the contact field (ct) in the directory file, add the '!' prefix to the macro name. For more detailed information on using the contact directory, see Using the Local Contact Directory in *Polycom UC Software 4.1.0 Administrator's Guide*	**000000000000-directory~.xml**

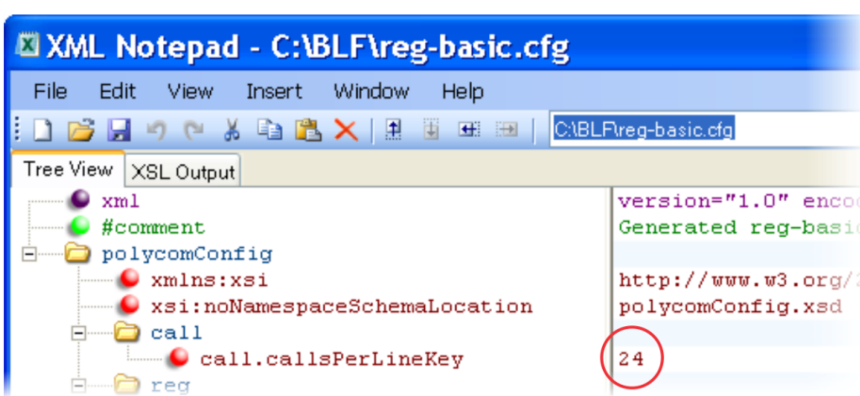
Guidelines for Configuring Enhanced Feature Keys

Read the following guidelines to learn how to configure EFK efficiently:

- Activation of EFK functions requires valid macro construction.
- All failures are logged at level 4 (minor).
- If two macros have the same name, the first one is used and the subsequent ones are ignored.
- A sequence of characters prefixed with an exclamation point (!) are parsed as a macro name. The exception is the favorites reference, which starts with ! and contains digits only.
- A sequence of characters prefixed with a caret (^) is the action string.
- ! and ^ macro prefixes cannot be mixed in the same macro line.
- The sequence of characters must be prefixed by either ! or ^ so it is processed as an enhanced feature key. All macro references and action strings added to the local directory contact field must be prefixed by either ! or ^.
- Action strings used in soft key definitions do not need to be prefixed by ^. However, the ! prefix must be used if macros or favorites are referenced.
- A sequence of macro names in the same macro is supported (for example, `!m1!m2`).
- A sequence of favorites references is supported (for example, `!1!2`).
- A sequence of macro names and favorites references is supported (for example, `!m1!2!m2`).
- Macro names that display in the local contact directory must follow the format `!<macro name>`, where `<macro name>` must match an `<elklist>` mname entry. The maximum macro length is 100 characters.
- A sequence of macros is supported but cannot be mixed with other action types.
- Action strings that appear in the local contact directory must follow the format `^<action string>`. Action strings can reference other macros or Favorites' indexes. Protection against recursive macro calls exists (the enhanced feature keys fails once you reach 50 macro substitutions).

Example Enhanced Feature Keys Configuration

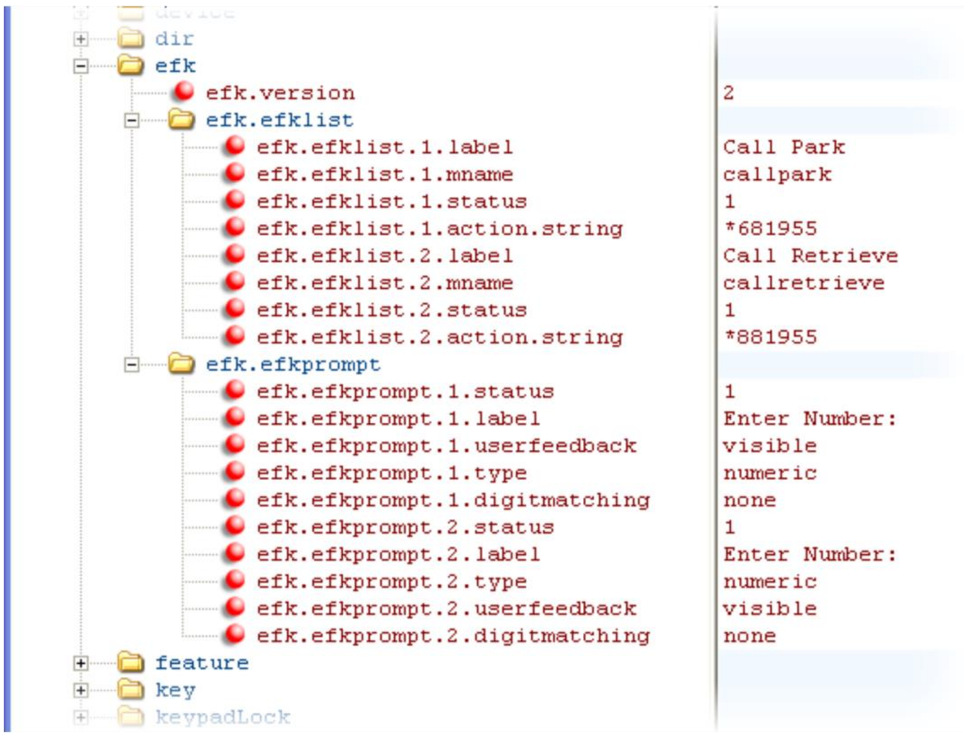
The following illustration shows the default value of 24 calls per line key. Ensure that you specify at least two calls per line key.



Enable enhanced feature keys in the features.cfg template file, as shown next.

In the following illustration, the EFK parameters are located in the features.cfg template file. In the `efk.efklist.x.*` parameters, line key 1 has been assigned a `Call Park` address (1955) and line key 2 a `Call Retrieve` function. The parameter `action.string` shows you the macro definition for these two functions. In addition, `status` is enabled and a label has been specified to display next to the line key. The entry in the `mname` parameter corresponds to the `contact (ct)` field in the contact directory.

In the `efk.prompt.*` parameters, `status` has been enabled. The label on the user prompt is defined as `Enter Number:` and this prompt displays on the phone screen. The `type` parameter is set to `numeric` to allow only numbers, and because `userfeedback` is specified as `visible`, you are able to see the numbers you enter into the prompt.



For a complete list of internal key functions for enhanced feature keys, see Internal Key Functions in the *Polycom UC Software 4.1.0 Administrator's Guide*.

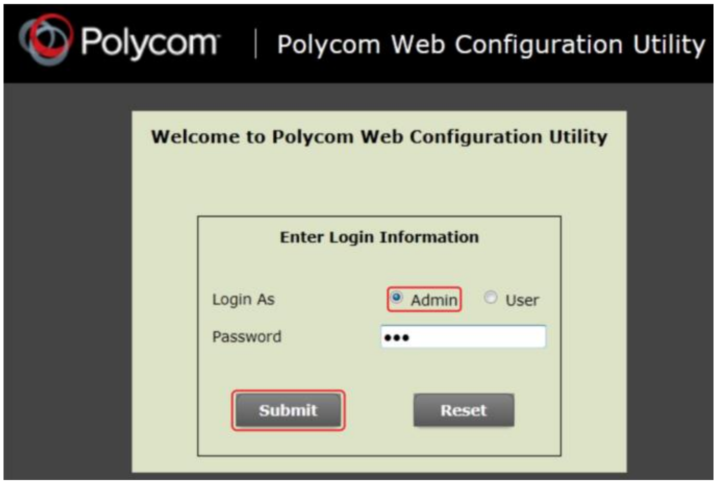
Generate Configured Line Key Information

Using the Web Configuration Utility, you can generate and download a PDF file with the configured line key information for each expansion module with a paper display connected to your VVX phone. The generated PDF enables you to print line key information for line keys on your expansion modules and insert the PDF as a directory card on your modules.

To generate and download the line key information PDF using the Web Configuration Utility:

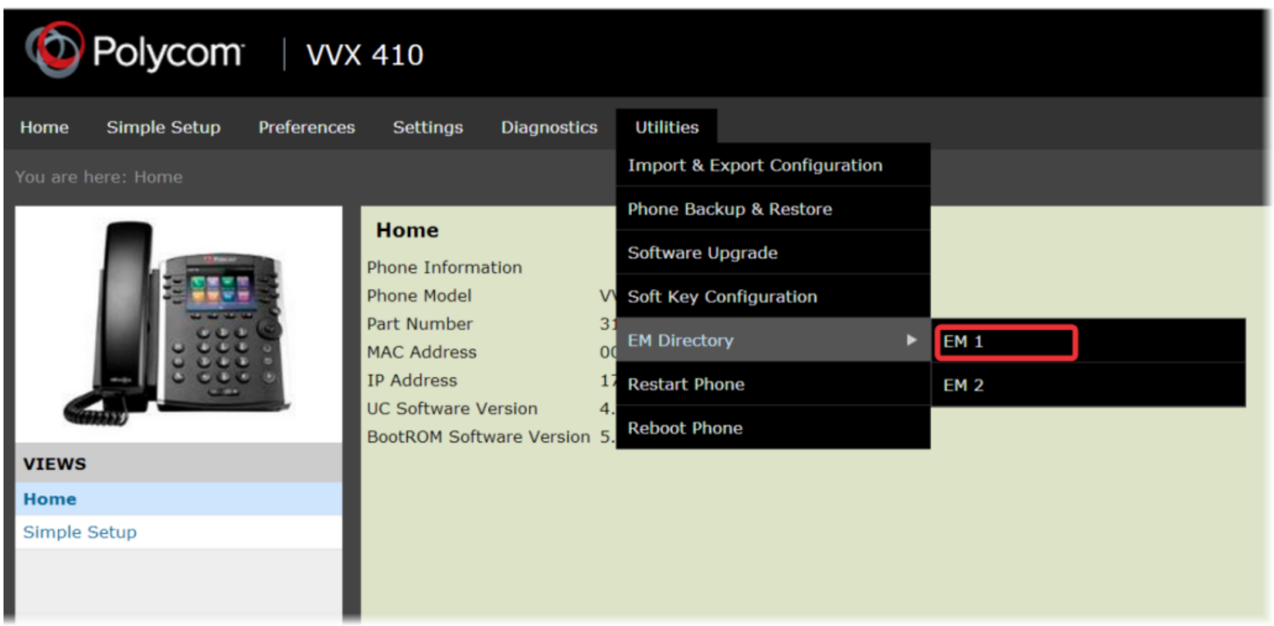
1 In your Internet browser, enter your phone's IP address into your browser's address bar.

2 Log in as an Admin, enter the default password, and select **Submit**, as shown next.



3 Select **Utilities > EM Directory**.

4 Select the expansion module you want to generate a PDF for. For example, EM1 is chosen in the following figure.



5 In the confirmation dialog, select **Yes** to download the PDF for the configured lines for your expansion module.

6 Select **Save > Open**.

The PDF with the configured line key information for your expansion module displays.

After you download the PDF with configured line key information for your expansion module, you can print the PDF and insert the PDF as the directory card for the expansion module.

Configuration Parameters

This reference section shows the UC software configuration parameters used to configure the features and functions for VVX phones and expansion modules running UC software 4.1.6 or later. The following information is helpful if you need a detailed description of a particular configuration parameter or want to see the default or permitted values for that parameter. See the *Polycom UC Software 4.1.0 Administrators's Guide* for a full list of parameters used to configure Polycom VVX business media phones. See the *Polycom UC Software 4.1.6 Release Notes* for a list of parameters added for UC Software 4.1.6.

Note: Configuration Parameters Not Runtime Configurable

Configuration parameters for UC software 4.1.6 are not runtime configurable, and the VVX phones and modules need to be rebooted after any configuration changes.

<attendant>

The Busy Lamp Field (BLF)/attendant console feature enhances support for phone-based monitoring. In the following table, x is the monitored user number.

Attendant/Busy Lamp Field Parameters

Parameter	*Permitted Values*	*Default*
attendant.reg[1]	**positive integer**	**1**
The index of the registration that will be used to send a SUBSCRIBE to the list SIP URI specified in `attendant.uri`. For example, `attendant.reg = 2` means the second registration will be used.		
attendant.ringType	**default, ringer1 to ringer24**	**ringer1**
The ringtone to play when a BLF dialog is in the offering state.		
attendant.uri[1]	**string**	**Null**
The list SIP URI on the server. If this is just a user part, the URI is constructed with the server hostname/IP. Note: If this parameter is set, the individually addressed users configured by `attendant.resourceList` and `attendant.behaviors` are ignored		
attendant.behaviors.display.spontaneousCallAppearances.normal[1] **Normal**	**0 or 1**	**1**
attendant.behaviors.display.spontaneousCallAppearances.automata[1] **Automatic**	**0 or 1**	**0**
If 1, the normal or automatic call appearance is spontaneously presented to the attendant when calls are alerting on a monitored resource (and a ring tone is played). If 0, the call appearance is not spontaneously presented to the attendant. The information displayed after a press-and-hold of a resource's line key is unchanged by this parameter.		

Parameter	*Permitted Values*	*Default*
attendant.behaviors.display.remoteCallerID.normal[1] **Normal** **attendant.behaviors.display.remoteCallerID.automata**[1] **Automatic**	**0 or 1**	**1**
If 1, normal and automatic remote party caller ID information is presented to the attendant. If 0, the string `unknown` will be substituted for both name and number information.		
attendant.resourceList.x.address	**string that constitutes a valid SIP URI (sip:6416@polycom.com) or contains the user part of a SIP URI (6416)**	**Null**
The user referenced by `attendant.reg=""` will subscribe to this URI for dialog. If a user part is present, the phone will subscribe to a sip URI constructed from the user part and domain of the user referenced by `attendant.reg`.		
attendant.resourceList.x.callAddress[1]	**string**	**Null**
If the BLF call server is not at the same address as the BLF presence server, calls will be sent to this address instead of the address specified by `attendant.resourceList.x.address`.		
attendant.resourceList.x.label	**UTF-8 encoded string**	**Null**
The text label displays adjacent to the associated line key. If set to Null, the label will be derived from the user part of `attendant.resourceList.x.address`.		
attendant.resourceList.x.proceedingIsRecipent[1]	**0 or 1**	**0**
A flag to determine whether pressing the associated line key for the monitored user will pick up the call.		
attendant.resourceList.x.type	**normal or automata**	**normal**
The type of resource being monitored and the default action to perform when pressing the line key adjacent to monitored user x. If `normal`, the default action is to initiate a call if the user is idle or busy and to perform a directed call pickup if the user is ringing. Any active calls are first placed on hold. If `automata`, the default action when is to perform a park/blind transfer of any currently active call. If there is no active call and the monitored user is ringing/busy, an attempt to perform a directed call pickup/park retrieval is made.		

[1] Change causes phone to restart or reboot.

bg/>

This section defines the backgrounds you can display on the VVX phones and expansion modules.

Background Parameters

Parameter	*Permitted Values*	*Default*
bg.color.selection	**w,x**	**1,1**
Set the background. Specify which type of background (w) and index (x) for that type is selected on reboot. The default selection is 2,1 the first solid background. Use w=1 and x=1 (1,1) to select the built-in image. Use w=2 and x= 1 to 4 to select one of the four `solid` backgrounds. Use w=3 and x= 1 to 6 to select one of the six background `bm` images		
bg.color.bm.x.name **Phone screen background image file** **bg.color.bm.x.em.name** **Expansion module (EM) background image file**	**URL or file path of a BMP or JPEG image** **URL or file path of a BMP or JPEG image**	**built-in value of Thistle**
The name of the image file (including extension). The six (x: 1 to 6) default screen and expansion module (EM) background images are: x=1: `Leaf.jpg` and `LeafEM.jpg` x=2: `Sailboat.jpg` and `SailboatEM.jpg` x=3: `Beach.jpg` and `BeachEM.jpg` x=4: `Palm.jpg` and `PalmEM.jpg` x=5 `Jellyfish.jpg` and `JellyfishEM.jpg` x=6 `Mountain.jpg` and `MountainEM.jpg` Note: If the file is missing or unavailable, the built-in default solid pattern is displayed.		

<lineKey/>

The flexible line key assignment feature uses the `<lineKey/>` parameter.

Line Key Parameters

Parameter	*Permitted Values*	*Default*
lineKey.x.category[1]	**BLF, Line, SpeedDial, Presence, or Unassigned**	**Unassigned**
Defines categories you can assigned to line key x where x defines the location of a physical line button. For example, VXX 600 + 3 LCD EMS = 16 + 252 = 268 lines **BLF** or **Presence** `lineKey.x.index` can only be set to 0, which automatically assigns line keys to contacts. **Line** `lineKey.x.index` contains the registration index, from 1 to 34, but automatic assignment is not supported. **Speed Dial** `lineKey.x.index` contains the favorites’ index, ranging from 1 to 9999, but automatic assignment is not supported. **Unassigned** Nothing can be assigned to the line.		

Parameter	*Permitted Values*	*Default*
lineKey.reassignment.enabled	**0, 1**	**0**
Enables the line key reassignment feature—Flexible Line Key.		
lineKey.x.index	**0, 1, 2**	**0**
Specifies which index is used to pick up the line key x, within the specified category, and assigned to the line key. Depends on the `lineKey.x.category` parameter.		

<efk/>

Use the following three tables to configure the enhanced feature key feature on your phone.

Enhanced Feature Key (EFK) Parameters

Parameter Name	*Permitted Values*	*Default*
efk.version	**2 (1 for SIP 3.0 and earlier)**	**2**
The version of the EFK elements. For SIP 3.0.x or earlier, 1 is the only supported version. For SIP 3.1 and later, 2 is the only supported version. If this parameter is Null, the EFK feature s disabled. This parameter is not required if there are no `efk.efklist` entries.		

The EFK list parameters are outlined in the following table.

Enhanced Feature Key (EFK) List Parameters

Parameter Name	*Permitted Values*	*Default*
efk.efklist.x.action.string		
The action string contains a macro definition of the action that the feature key will perform. If EFK is enabled, this parameter must have a value (it cannot be Null). For a list of macro definitions and example macro strings, see Understanding Macro Definitions in the *Polycom UC Software 4.1.0 Administrator's Guide*.		
efk.efklist.x.label	**string**	**Null**
The text string that will be used as a label on any user text entry screens during EFK operation. If Null, the Null string is used. Note: If the label does not fit on the screen, the text is shortened and ellipses (…) is appended.		
efk.efklist.x.mname		**expanded_macro**
The unique identifier used by the favorites configuration to reference the enhanced feature key entry. Cannot start with a digit. Note that this parameter must have a value, it cannot be Null.		
efk.efklist.x.status	**0 or 1**	**0**
If 0 or Null, key x is disabled. If 1, the key is enabled.		

Parameter Name	*Permitted Values*	*Default*
efk.efklist.x.type		**invite**
The SIP method to be performed. If set to `invite`, the action required is performed using the SIP INVITE method. Note: This parameter is included for backward compatibility. Do not use if possible. If `efk.x.action.string` contains types, this parameter is ignored. If Null, the default of INVITE is used.		

The EFK prompt parameters are listed in the following table.

Enhanced Feature Key (EFK) Prompt Parameters

Parameter Name	*Permitted Values*	*Default*
efk.efkprompt.x.label[1]	**string**	**Null**
The prompt text that is presented to the user on the user prompt screen. If Null, no prompt displays. Note: If the label does not fit on the screen, the label is shortened and an ellipses (…) is appended.		
efk.efkprompt.x.status[1]	**0 or 1**	**0**
If 0, key x is disabled. If 1, the key is enabled. This parameter must have a value; it cannot be Null. Note: If a macro attempts to use a prompt that is disabled or invalid, the macro execution will fail.		
efk.efkprompt.x.type[1]	**numeric or text**	**text**
The type of characters entered by the user. If set to `numeric`, the characters are interpreted as numbers. If set to `text`, the characters are interpreted as letters. If Null, `numeric` is used. If this parameter has an invalid value, this prompt, and all parameters depending on this prompt, are invalid. Note: A mix of `numeric` and `text` is not supported.		
efk.efkprompt.x.userfeedback[1]	**visible or masked**	**visible**
The user input feedback method. If set to `visible`, the text is visible. If set to `masked`, the text displays as asterisk characters (*), which can be used to mask password fields. If Null, visible is used. If this parameter has an invalid value, this prompt, and all parameters depending on this prompt, are invalid.		

[1] Change causes phone to restart or reboot.

<feature/>

The `<feature/>` parameter controls the activation or deactivation of a feature at run time. See the *Polycom UC Software 4.1.0 Administrators’s Guide* for a full list of <feature/> parameters.

Feature Activation/Deactivation Parameters

Parameter	*Permitted Values*	*Default*
feature.enhancedFeatureKeys.enabled	**0 or 1**	**0**
If 0, the enhanced feature keys feature is disabled. If 1, the feature is enabled.		

Parameter	Permitted Values	Default
feature.presence.enabled[1]	**0 or 1**	**0**
If 0, the presence feature—including buddy managements and user status—is disabled. If 1, the presence feature is enabled with the buddy and status options.		

[1] Change causes phone to restart or reboot.

<pres/>

The parameter `pres.reg` is the line number used to send SUBSCRIBE. If this parameter is missing, the phone uses the primary line to send SUBSCRIBE.

Presence Parameters

Parameter	Permitted Values	Default
pres.idleSoftkeys	**0 or 1**	**1**
If 0, the MyStat and Buddies presence idle soft keys do not display. If 1, the soft keys display.		
pres.idleTimeout.offHours.enabled	**0 or 1**	**1**
If 0, the off hours idle timeout feature is disabled. If 1, the feature is enabled.		
pres.idleTimeout.offHours.period	**1 to 600**	**15**
The number of minutes to wait while the phone is idle during off hours before showing the Away presence status.		
pres.idleTimeout.officeHours.enabled	**0 or 1**	**1**
If 0, the office hours idle timeout feature is disabled. If 1, the feature is enabled.		
pres.idleTimeout.officeHours.period	**1 to 600**	**15**
The number of minutes to wait while the phone is idle during office hours before showing the Away presence status.		
pres.reg	**1 to 34**	**1**
The valid line/registration number that is used for presence. This registration sends a SUBSCRIBE for presence. If the value is not a valid registration, this parameter is ignored.		

<reg/>

In the following tables, x is the registration number. For VVX 300, 310, 400, 410, 500, and 600 phones with three connected expansion modules, x=1–34.

Tables Registration Parameters and Registration Server Parameters show the registration parameters and the server registration parameters.

Registration Parameters

Parameter	*Permitted Values*	*Default*
reg.x.acd-login-logout **reg.x.acd-agent-available**	**0 or 1** **0 or 1**	**0** **0**
If both ACD login/logout and agent available are set to 1 for registration x, the ACD feature will be enabled for that registration.		
reg.x.address	**string address**	**Null**
The user part (for example, 1002) or the user and the host part (for example, 1002@polycom.com) of the registration SIP URI or the H.323 ID/extension.		
reg.x.applyServerDigitMapLocally	**0 or 1**	**0**
If 1 and `reg.x.server.y.specialInterop` is set to `lync2010`, the phone uses the dial plan from the Microsoft Lync Server. Any dialed number will apply the dial plan locally. If 0, the dial plan from the Microsoft Lync Server is not used.		
reg.x.auth.domain	**string**	**Null**
The domain of the authorization server that is used to check the user names and passwords.		
reg.x.auth.optimizedInFailover	**0 or 1**	**0**
The destination of the first new SIP request when failover occurs. If 0, the SIP request is sent to the server with the highest priority in the server list. If 1, the SIP request is sent to the server which sent the proxy authentication request.		
reg.x.auth.password	**string**	**Null**
The password to be used for authentication challenges for this registration. If the password is non-Null, it will override the password entered into the Authentication submenu on the Settings menu of the phone.		
reg.x.auth.userId	**string**	**Null**
User ID to be used for authentication challenges for this registration. If the User ID is non-Null, it will override the user parameter entered into the Authentication submenu on the Settings menu of the phone.		
reg.x.auth.useLoginCredentials	**0 or 1**	**0**
If 0, login credentials are not used for authentication to the server on registration x. If 1, login credentials are used for authentication to the server.		
reg.x.bargeInEnabled	**0 or 1**	**0**
If 0, barge-in is disabled for line x. If 1, barge-in is enabled (remote users of shared call appearances can interrupt or barge in to active calls).		
reg.x.callsPerLineKey[1]	**1 to 4, 1 to 8, 1 to 24**	**24 (for VVX phones)** **8 (for all other phones)**
Set the maximum number of concurrent calls for a single registration x. This parameter applies to all line keys using registration x. If registration x is a shared line, an active call counts as a call appearance on all phones sharing that registration. This parameter overrides `call.callsPerLineKey`.		
reg.x.csta	**0 or 1**	**0**
If 0, the uaCSTA (User Agent Computer Supported Telecommunications Applications) feature is disabled. If 1, uaCSTA is enabled (overrides the global parameter `voIpProt.SIP.csta`.		

Parameter	*Permitted Values*	*Default*
reg.x.dialPlanName	**String**	**Null**
If `reg.x.server.y.specialInterop` is set to `lync2010`, the dial plan name from the Microsoft Lync Server is stored here. Each registration has its own name for this dial plan. Note: Do not change this parameter if set by Microsoft Lync.		
reg.x.displayName	**UTF-8 encoded string**	**Null**
The display name used in SIP signaling and/or the H.323 alias used as the default caller ID.		
reg.x.filterReflectedBlaDialogs	**0 or 1**	**1**
If 0, bridged line appearance NOTIFY messages (dialog state change) will not be ignored. If 1, the messages will be ignored.		
reg.x.fwd.busy.contact	**string**	**Null**
The forward-to contact for calls forwarded due to busy status. If Null, the contact specified by `divert.x.contact` will be used.		
reg.x.fwd.busy.status	**0 or 1**	**0**
If 0, incoming calls that receive a busy signal will not be forwarded. If 1, busy calls are forwarded to the contact specified by `reg.x.fwd.busy.contact`.		
reg.x.fwd.noanswer.contact	**string**	**Null**
The forward-to contact used for calls forwarded due to no answer. If Null, the contact specified by `divert.x.contact` will be used.		
reg.x.fwd.noanswer.ringCount	**0 to 65535**	**0**
The number of seconds the phone should ring for before the call is forwarded because of no answer. Note: The maximum value accepted by some call servers is 20.		
reg.x.fwd.noanswer.status	**0 or 1**	**0**
If 0, calls are not forwarded if there is no answer. If 1, calls are forwarded to the contact specified by `reg.x.noanswer.contact` after ringing for the length of time specified by `reg.x.fwd.noanswer.ringCount`.		
reg.x.ice.turn.callAdmissionControl.enabled	**0 or 1**	**0**
If 0, call admission control is disabled. If 1, call admission control is enabled for calls using the Microsoft Lync 2010 Server.		
reg.x.label	**UTF-8 encoded string**	**Null**
The text label that displays next to the line key for registration x. If Null, the user part of `reg.x.address` is used.		
reg.x.lcs	**0 or 1**	**0**
If 0, the Microsoft Live Communications Server (LSC) is not supported for registration x. If 1, LSC is supported.		
reg.x.lineKeys	**1 to max**	**1**
The number of line keys to use for a single registration. The maximum number of line keys you can use per registration depends on your phone model.		

Parameter	*Permitted Values*	*Default*
reg.x.lisdisclaimer	**string, 0 to 256 characters**	**Null**
Sets the value of the location policy disclaimer. For example, the disclaimer may be "Warning: If you do not provide a location, emergency services may be delayed in reaching your location should you need to call for help." This parameter is set by in-band provisioning when the phone is registered to Microsoft Lync Server 2010.		
reg.x.lync.autoProvisionCertLocation	**0 to 6**	**6**
If 0, the certificate download is disabled. If non-0, the certificate corresponding to the index of the appropriate `sec.TLS.customCaCert.X` is downloaded.		
reg.x.musicOnHold.uri	**a SIP URI**	**Null**
A URI that provides the media stream to play for the remote party on hold. If present and not Null, this parameter overrides `voIpProt.SIP.musicOnHold.uri`.		
reg.x.outboundProxy.address	**dotted-decimal IP address or hostname**	**Null**
The IP address or hostname of the SIP server to which the phone sends all requests.		
reg.x.outboundProxy.failOver.failBack.mode	**newRequests DNSTTL registration duration**	**newRequests**
The mode for failover failback (overrides `reg.x.server.y.failOver.failBack.mode`). - `newRequests` All new requests are forwarded first to the primary server regardless of the last used server. - `DNSTTL` The phone tries the primary server again after a timeout equal to the DNS TTL configured for the server that the phone is registered to. - `registration` The phone tries the primary server again when the registration renewal signaling begins. - `duration` The phone tries the primary server again after the time specified by `reg.x.outboundProxy.failOver.failBack.timeout` expires.		
reg.x.outboundProxy.failOver.failBack.timeout	**0, 60 to 65535**	**3600**
The time to wait (in seconds) before failback occurs (overrides `reg.x.server.y.failOver.failBack.timeout`). If the failback mode is set to Duration, the phone waits this long after connecting to the current working server before selecting the primary server again. If 0, the phone will not failback until a failover event occurs with the current server.		
reg.x.outboundProxy.failOver.failRegistrationOn	**0 or 1**	**0**
When set to 1, and the `reRegisterOn` parameter is enabled, the phone will silently invalidate an existing registration (if it exists), at the point of failing over. When set to 0, and the `reRegisterOn` parameter is enabled, existing registrations will remain active. This means that the phone will attempt failback without first attempting to register with the primary server to determine if it has recovered. Note that `reg.x.outboundProxy.failOver.RegisterOn` must be enabled.		
reg.x.outboundProxy.failOver.onlySignalWithRegistered	**0 or 1**	**1**
When set to 1, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, no signaling is accepted from or sent to a server that has failed until failback is attempted or failover occurs. If the phone attempts to send signaling associated with an existing call via an unregistered server (for example, to resume or hold a call), the call will end. No SIP messages will be sent to the unregistered server. When set to 0, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, signaling will be accepted from and sent to a server that has failed (even though failback hasn't been attempted or failover hasn't occurred).		

Parameter	*Permitted Values*	*Default*
reg.x.outboundProxy.failOver.reRegisterOn	**0 or 1**	**0**
This parameters overrides `reg.x.server.y.failOver.failBack.RegisterOn`. When set to 1, the phone will attempt to register with (or via, for the outbound proxy scenario) the secondary server. If the registration succeeds (a 200 OK response with valid expires), signaling will proceed with the secondary server. When set to 0, the phone won't attempt to register with the secondary server, since the phone will assume that the primary and secondary servers share registration information.		
reg.x.outboundProxy.port	**1 to 65535**	**0**
The port of the SIP server to which the phone sends all requests.		
reg.x.outboundProxy.transport	**DNSnaptr, TCPpreferred, UDPOnly, TLS, TCPOnly**	**DNSnaptr**
The transport method the phone uses to communicate with the SIP server. - `Null` or `DNSnaptr` If `reg.x.outboundProxy.address` is a hostname and `reg.x.outboundProxy.port` is 0 or Null, do NAPTR and then SRV lookups to try to discover the transport, ports and servers, as per RFC 3263. If `reg.x.outboundProxy.address` is an IP address, or a port is given, then UDP is used. - `TCPpreferred` TCP is the preferred transport, UDP is used if TCP fails. - `UDPOnly` Only UDP will be used. - `TLS` If TLS fails, transport fails. Leave port field empty (will default to `5061`) or set to `5061`. - `TCPOnly` Only TCP will be used.		
reg.x.proxyRequire	**string**	**Null**
The string that needs to be entered in the Proxy-Require header. If Null, no Proxy-Require will be sent.		
reg.x.ringType	**default, ringer1 to ringer24**	**ringer2**
The ringer to be used for calls received by this registration. The default is the first non-silent ringer.		
reg.x.ringType.privateLine	**default, ringer1 to ringer24**	**default**
The ringer to be used for calls received by a private line connected to Microsoft Lync Server 2010.		
reg.x.serverAutoDiscovery	**0 or 1**	**1**
Determines whether or not to discover the server address automatically. This parameter is used with Microsoft Lync Server 2010.		
reg.x.serverFeatureControl.cf[1]	**0 or 1**	**0**
If 0, server-based call forwarding is not enabled; this is the old behavior. If 1, server-based call forwarding is enabled. This parameter overrides `voIpProt.SIP.serverFeatureControl.cf`.		
reg.x.serverFeatureControl.dnd[1]	**0 or 1**	**0**
If 0, server-based do-not-disturb (DND) is not enabled. If 1, server-based DND is enabled and the call server has control of DND. This parameter overrides `voIpProt.SIP.serverFeatureControl.dnd`.		

Parameter	*Permitted Values*	*Default*
reg.x.serverFeatureControl.localProcessing.cf	**0 or 1**	**1**
If 0 and `reg.x.serverFeatureControl.cf` is set to 1, the phone will not perform local call-forward behavior. If set to 1, the phone will perform local call-forward behavior on all calls received. This parameter overrides `voIpProt.SIP.serverFeatureControl.localProcessing.cf`.		
reg.x.serverFeatureControl.localProcessing.dnd	**0 or 1**	**1**
If 0 and `reg.x.serverFeatureControl.dnd` is set to 1, the phone will not perform local DND call behavior. If set to 1, the phone will perform local DND call behavior on all calls received. This parameter overrides `voIpProt.SIP.serverFeatureControl.localProcessing.dnd`.		
reg.x.serverFeatureControl.signalingMethod	**string**	**serviceMsForwardContact**
Controls the method used to perform call-forwarding requests to the server.		
reg.x.server.y.registerRetry.maxTimeout		**180 seconds**
The maximum period of time in seconds that you want the phone to try registering with the server.		
reg.x.srtp.enable[1]	**0 or 1**	**1**
If 0, the registration always declines SRTP offers. If 1, the registration accepts SRTP offers.		
reg.x.srtp.offer[1]	**0 or 1**	**0**
If 1, the registration includes a secure media stream description along with the usual non-secure media description in the SDP of a SIP INVITE. This parameter applies to the registration initiating (offering) a phone call. If 0, no secure media stream is included in SDP of a SIP invite.		
reg.x.srtp.require[1]	**0 or 1**	**0**
If 0, secure media streams are not required. If 1, the registration is allowed to use only secure media streams. Any offered SIP INVITEs must include a secure media description in the SDP or the call will be rejected. For outgoing calls, only a secure media stream description is included in the SDP of the SIP INVITE, meaning that the nonsecure media description is not included. If this parameter set to 1, `reg.x.srtp.offer` will also be set to 1, regardless of the value in the configuration file.		
reg.x.srtp.simplifiedBestEffort	**0 or 1**	**0**
If 0, no SRTP is supported. If 1, negotiation of SRTP compliant with Microsoft Session Description Protocol Version 2.0 Extensions is supported. This parameter overrides `sec.srtp.simplifiedBestEffort`.		
reg.x.strictLineSeize	**0 or 1**	**0**
If 1, the phone is forced to wait for 200 OK on registration x when receiving a TRYING notify. If 0, the old behavior is used. This parameter overrides `voIpProt.SIP.strictLineSeize` for registration x.		
reg.x.tcpFastFailover	**0 or 1**	**0**
If 1, failover occurs based on the values of `reg.x.server.y.retryMaxCount` and `voIpProt.server.x.retryTimeOut`. If 0, the old behavior is used.		
reg.x.telephony	**0 or 1**	**1**
If 0, telephony calls are not enabled on this registration (use this value if the registration is used with Microsoft Office Communications Server 2007 R2 or Microsoft Lync 2010). If 1, telephony calls are enabled on this registration.		

Parameter	Permitted Values	Default
reg.x.thirdPartyName	**string address**	**Null**
This field must match the `reg.x.address` value of the registration that makes up the part of a bridged line appearance (BLA). It must be Null in all other cases.		
reg.x.type	**private or shared**	**private**
If set to private, use standard call signaling. If set to shared, augment call signaling with call state subscriptions and notifications and use access control for outgoing calls.		
reg.x.useCompleteUriForRetrieve	**0 or 1**	**1**
This parameters overrides `voipPort.SIP.useCompleteUriForRetrieve`. If set to 1, the target URI in BLF signaling will use the complete address as provided in the xml dialog document. If set to 0, only the user portion of the XML dialog document is used and the current registrar's domain is appended to create the full target URI.		

[1] Change causes phone to restart or reboot.

You can list multiple registration servers for fault tolerance. The server registration parameters are listed in the following table. You can list four servers by using y=1 to 4. If the `reg.x.server.y.address` is not Null, all of the parameters in the following table override the parameters specified in `voIpProt.server.*`.

Registration Server Parameters

Parameter	Permitted Values	Default
reg.x.server.H323.y.address	**dotted-decimal IP address or hostname**	**Null**
Address of the H.323 gatekeeper.		
reg.x.server.H323.y.port	**0 to 65535**	**0**
Port to be used for H.323 signaling. If set to Null, 1719 (H.323 RAS signaling) is used.		
reg.x.server.H323.y.expires	**positive integer**	**3600**
Desired registration period.		
reg.x.server.y.address	**dotted-decimal IP address or hostname**	**Null**
The IP address or hostname of a SIP server that accepts registrations. If not Null, all of the parameters in this table will override the parameters specified in `voIpProt.server.*`.		
reg.x.server.y.expires	**positive integer, minimum 10**	**3600**
The phone's requested registration period in seconds. Note: The period negotiated with the server may be different. The phone will attempt to re-register at the beginning of the overlap period. For example, if `expires=300` and `overlap=5`, the phone will re-register after 295 seconds (300–5).		

Parameter	*Permitted Values*	*Default*
reg.x.server.y.expires.lineSeize	**0 to 65535**	**30**
Requested line-seize subscription period.		
reg.x.server.y.expires.overlap	**5 to 65535**	**60**
The number of seconds before the expiration time returned by server x at which the phone should try to re-register. The phone will try to re-register at half the expiration time returned by the server if the server value is less than the configured overlap value.		
reg.x.server.y.failOver.failBack.mode	**newRequests, DNSTTL, registration, duration**	**newRequests**
The mode for failover failback (this parameter overrides `voIpProt.server.x.failOver.failBack.mode`): - `newRequests` All new requests are forwarded first to the primary server regardless of the last used server. - `DNSTTL` The phone tries the primary server again after a timeout equal to the DNS TTL configured for the server to which the phone is registered. - `registration` The phone tries the primary server again when the registration renewal signaling begins. - `duration` The phone tries the primary server again after the time specified by `reg.x.server.y.failOver.failBack.timeout`.		
reg.x.server.y.failOver.failBack.timeout	**0, 60 to 65535**	**3600**
The time to wait (in seconds) before failback occurs (overrides `voIpProt.server.x.failOver.failBack.timeout`).If the failback mode is set to Duration, the phone waits this long after connecting to the current working server before selecting the primary server again. If 0, the phone will not failback until a failover event occurs with the current server.		
reg.x.server.y.failOver.failRegistrationOn	**0 or 1**	**0**
When set to 1, and the `reRegisterOn` parameter is enabled, the phone will silently invalidate an existing registration (if it exists), at the point of failing over. When set to 0, and the `reRegisterOn` parameter is enabled, existing registrations will remain active. This means that the phone will attempt failback without first attempting to register with the primary server to determine if it has recovered.		
reg.x.server.y.failOver.onlySignalWithRegistered	**0 or 1**	**1**
When set to 1, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, no signaling is accepted from or sent to a server that has failed until failback is attempted or failover occurs. If the phone attempts to send signaling associated with an existing call via an unregistered server (for example, to resume or hold a call), the call will end. No SIP messages will be sent to the unregistered server. When set to 0, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, signaling will be accepted from and sent to a server that has failed (even though failback hasn't been attempted or failover hasn't occurred).		
reg.x.server.y.failOver.reRegisterOn	**0 or 1**	**0**
This parameter overrides the `voIpProt.server.x.failOver.reRegisterOn`. When set to 1, the phone will attempt to register with (or via, for the outbound proxy scenario) the secondary server. If the registration succeeds (a 200 OK response with valid expires), signaling will proceed with the secondary server. When set to 0, the phone won't attempt to register with the secondary server, since the phone will assume that the primary and secondary servers share registration information.		
reg.x.server.y.lcs	**0 or 1**	**0**
If 0, the Microsoft Live Communications Server (LSC) is not supported. If 1, LCS is supported for registration x.		

Parameter	*Permitted Values*	*Default*
reg.x.server.y.useOutboundProxy	**0 or 1**	**1**
Specify whether or not to use the outbound proxy specified in `reg.x.outboundProxy.address` for server x. This parameter overrides `voIpProt.server.x.useOutboundProxy` for registration x.		
reg.x.server.y.port	**0, 1 to 65535**	**Null**
The port of the SIP server that specifies registrations. If 0, the port used depends on `reg.x.server.y.transport`.		
reg.x.server.y.register	**0 or 1**	**1**
If 0, calls can be routed to an outbound proxy without registration. See voIpProt.server.x.register. For more information, see *Technical Bulletin 5844: SIP Server Fallback Enhancements on Polycom Phones*.		
reg.x.server.y.registerRetry.baseTimeOut	**10 - 120**	**60**
The base time period to wait before a registration retry. Used in conjunction with `reg.x.server.y.registerRetry.maxTimeOut` to determine how long to wait. The algorithm is defined in RFC 5626.		
reg.x.server.y.registerRetry.maxTimeOut	**60 - 1800**	**60**
The maximum time period to wait before a registration retry. Used in conjunction with `reg.x.server.y.registerRetry.baseTimeOut` to determine how long to wait. The algorithm is defined in RFC 5626.		
reg.x.server.y.retryMaxCount	**0 to 20**	**3**
If set to 0, 3 is used. The number of retries that will be attempted before moving to the next available server.		
reg.x.server.y.retryTimeOut	**0 to 65535**	**0**
The amount of time (in milliseconds) to wait between retries. If 0, use standard RFC 3261 signaling retry behavior.		
reg.x.server.y.transport	**DNSnaptr, TCPpreferred, UDPOnly, TLS, TCPOnly**	**DNSnaptr**
The transport method the phone uses to communicate with the SIP server. - `Null` or `DNSnaptr` If `reg.x.server.y.address` is a hostname and `reg.x.server.y.port` is 0 or Null, do NAPTR then SRV lookups to try to discover the transport, ports, and servers, as per RFC 3263. If `reg.x.server.y.address` is an IP address, or a port is given, then UDP is used. - `TCPpreferred` TCP is the preferred transport; UDP is used if TCP fails. - `UDPOnly` Only UDP will be used. - `TLS` If TLS fails, transport fails. Leave port field empty (will default to `5061`) or set to `5061`. - `TCPOnly` Only TCP will be used.		

<voIpProt/>

You must set up the call server and DTMF signaling parameters. This parameter includes the following configuration parameters:

- <server/>
- <SIP/>

<server/>

This configuration parameter is defined as follows.

VoIP Server Parameters

Parameter	*Permitted Values*	*Default*
voIpProt.server.dhcp.available[1]	**0 or 1**	**0**
If 0, do not check with the DHCP server for the SIP server IP address. If 1, check with the server for the IP address.		
voIpProt.server.dhcp.option[1]	**128 to 254**	**128**
The option to request from the DHCP server if `voIpProt.server.dhcp.available`= 1. Note: If `reg.x.server.y.address` is non-Null, it takes precedence even if the DHCP server is available.		
voIpProt.server.dhcp.type[1]	**0 or 1**	**0**
Type to request from the DHCP server if `voIpProt.server.dhcp.available` is set to 1. If this parameter is set to 0, IP request address. If set to 1, request string		
voIpProt.server.x.address	**dotted- decimal IP address or hostname**	**Null**
The IP address or hostname and port of a SIP server that accepts registrations. Multiple servers can be listed starting with x=1 to 4 for fault tolerance.		
voIpProt.server.x.port	**0, 1 to 65535**	**0**
The port of the server that specifies registrations. If 0, the port used depends on `voIpProt.server.x.transport.`		
voIpProt.server.x.registerRetry.baseTimeOut	**10 to 120**	**60**
The base time period to wait before a registration retry. Used in conjunction with `voIpProt.server.x.registerRetry.maxTimeOut` to determine how long to wait. The algorithm is defined in RFC 5626. If both parameters `voIpProt.server.x.registerRetry.baseTimeOut` and `reg.x.server.y.registerRetry.baseTimeOut` are set, the value of `reg.x.server.y.registerRetry.baseTimeOut` takes precedence.		
voIpProt.server.x.registerRetry.maxTimeOut	**60 to 1800**	**60**
The maximum time period to wait before a registration retry. Used in conjunction with `voIpProt.server.x.registerRetry.maxTimeOut` to determine how long to wait. The algorithm is defined in RFC 5626. If both parameters `voIpProt.server.x.registerRetry.maxTimeOut` and `reg.x.server.y.registerRetry.maxTimeOut` are set, the value of `reg.x.server.y.registerRetry.maxTimeOut` takes precedence.		

Parameter	*Permitted Values*	*Default*
voIpProt.server.x.transport	**DNSnaptr, TCPpreferred, UDPOnly, TLS, TCPOnly**	**DNSnaptr**
The transport method the phone uses to communicate with the SIP server. - Null or DNSnaptr If voIpProt.server.x.address is a hostname and voIpProt.server.x.port is 0 or Null, do NAPTR then SRV lookups to try to discover the transport, ports, and servers, as per RFC 3263. If voIpProt.server.x.address is an IP address, or a port is given, then UDP is used. - TCPpreferred TCP is the preferred transport; UDP is used if TCP fails. - UDPOnly Only UDP will be used. - TLS If TLS fails, transport fails. Leave port field empty (will default to 5061) or set to 5061. - TCPOnly Only TCP will be used.		
voIpProt.server.x.protocol.SIP	**0 or 1**	**1**
If 1, server is a SIP proxy/registrar. Note: if set to 0, and the server is confirmed to be a SIP server, then the value is assumed to be 1.		
voIpProt.server.x.expires	**positive integer, minimum 10**	**3600**
The phone's requested registration period in seconds. Note: The period negotiated with the server may be different. The phone will attempt to re-register at the beginning of the overlap period. For example, if expires=300 and overlap=5, the phone will re-register after 295 seconds (300–5).		
voIpProt.server.x.expires.overlap	**5 to 65535**	**60**
The number of seconds before the expiration time returned by server x at which the phone should try to re-register. The phone will try to re-register at half the expiration time returned by the server if the server value is less than the configured overlap value.		
voIpProt.server.x.expires.lineSeize	**positive integer, minimum 0 was 10**	**30**
Requested line-seize subscription period.		
voIpProt.server.x.failOver.failBack.mode	**newRequests, DNSTTL, registration, duration**	**newRequest s**
The mode for failover failback: - newRequests All new requests are forwarded first to the primary server regardless of the last used server. - DNSTTL The phone tries the primary server again after a timeout equal to the DNS TTL configured for the server that the phone is registered to. - registration The phone tries the primary server again when the registration renewal signaling begins. - duration The phone tries the primary server again after the time specified by voIpProt.server.x.failOver.failBack.timeout.		
voIpProt.server.x.failOver.failBack.timeout	**0, 60 to 65535**	**3600**
If voIpProt.server.x.failOver.failBack.mode is set to duration, this is the time in seconds after failing over to the current working server before the primary server is again selected as the first server to forward new requests to. Values between 1 and 59 will result in a timeout of 60, and 0 means do not fail back until a failover event occurs with the current server.		

Parameter	*Permitted Values*	*Default*
voIpProt.server.x.failOver.failRegistrationOn	**0 or 1**	**0**
When set to 1, and the `reRegisterOn` parameter is enabled, the phone will silently invalidate an existing registration (if it exists), at the point of failing over. When set to 0, and the `reRegisterOn` parameter is enabled, existing registrations will remain active. This means that the phone will attempt failback without first attempting to register with the primary server to determine if it has recovered.		
voIpProt.server.x.failOver.onlySignalWithRegistered	**0 or 1**	**1**
When set to 1, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, no signaling is accepted from or sent to a server that has failed until failback is attempted or failover occurs. If the phone attempts to send signaling associated with an existing call via an unregistered server (for example, to resume or hold a call), the call will end. No SIP messages will be sent to the unregistered server. When set to 0, and the `reRegisterOn` and `failRegistrationOn` parameters are enabled, signaling will be accepted from and sent to a server that has failed (even though failback hasn't been attempted or failover hasn't occurred).		
voIpProt.server.x.failOver.reRegisterOn	**0 or 1**	**0**
When set to 1, the phone will attempt to register with (or via, for the outbound proxy scenario) the secondary server. If the registration succeeds (a 200 OK response with valid expires), signaling will proceed with the secondary server. When set to 0, the phone won't attempt to register with the second.		
voIpProt.server.x.lcs	**0 or 1**	**0**
If 0, the Microsoft Live Communications Server (LSC) is not supported. If 1, LCS is supported for registration x. This parameter overrides `voIpProt.SIP.lcs`.		
voIpProt.server.x.register	**0 or 1**	**1**
If 0, calls can be routed to an outbound proxy without registration. See `reg.x.server.y.register`. For more information, see *Technical Bulletin 5844: SIP Server Fallback Enhancements on Polycom Phones*.		
voIpProt.server.x.retryTimeOut	**0 to 65535**	**0**
The amount of time (in milliseconds) to wait between retries. If 0, use standard RFC 3261 signaling retry behavior.		
voIpProt.server.x.retryMaxCount	**0 to 20**	**3**
If set to 0, 3 is used. The number of retries that will be attempted before moving to the next available server.		
voIpProt.server.x.specialInterop	**standard, ocs2007r2, lcs2005, lync2010**	**standard**
Specify whether this registration should support Microsoft Office Communications Server 2007 R2 (ocs2007r2), Microsoft Live Communications Server 2005 (lcs2005), or Microsoft Lync 2010 (lync2010).		
voIpProt.server.x.useOutboundProxy	**0 or 1**	**1**
Specify whether or not to use the outbound proxy specified in `voIpProt.SIP.outboundProxy.address` for server x.		
voIpProt.server.H323.x.address	**dotted-decimal IP address or hostname**	**Null**
Address of the H.323 gatekeeper. Note: Only one H.323 gatekeeper per phone is supported; if more than one is configured, only the first is used.		

Parameter	*Permitted Values*	*Default*
voIpProt.server.H323.x.port	**0 to 65535**	**1719**
Port to be used for H.323 signaling. Note: The H.323 gatekeeper RAS signaling uses UDP, while the H.225/245 signaling uses TCP.		
voIpProt.server.H323.x.expires	**positive integer**	**3600**
Desired registration period.		

[1] Change causes phone to restart or reboot.

<SIP/>

This configuration parameter is defined as follows:

Session Initiation Protocol (SIP) Parameters

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.acd.signalingMethod[1]	**0 or 1**	**0**
If set to 0, the SIP-B signaling is supported. (This is the older ACD functionality.) If set to 1, the feature synchronization signaling is supported. (This is the new ACD functionality.)		
voIpProt.SIP.alertInfo.x.class	**see the list of ring classes in <rt/> in the *Polycom UC Software 4.1.0 Administrator's Guide***	**default**
Alert Info fields from INVITE requests will be compared against as many of these parameters as are specified (x=1, 2, ..., N) and if a match is found, the behavior described in the corresponding ring class is applied.		
voIpProt.SIP.alertInfo.x.value	**string**	**Null**
A string to match the Alert Info header in the incoming INVITE.		
voIpProt.SIP.allowTransferOnProceeding	**0, 1, 2**	**1**
If set to 0, a transfer is not allowed during the proceeding state of a consultation call. If set to 1, a transfer can be completed during the proceeding state of a consultation call. If set to 2, phones will accept an INVITE with replaces for a dialog in early state. This is needed when using transfer on proceeding with a proxying call server such as openSIPS, reSIProcate or SipXecs.		
voIpProt.SIP.authOptimizedInFailover	**0 or 1**	**0**
If set to 1, when failover occurs, the first new SIP request is sent to the server that sent the proxy authentication request. If set to 0, when failover occurs, the first new SIP request is sent to the server with the highest priority in the server list. If `reg.x.auth.optimizedInFailover` set to 0, this parameter is checked. If `voIpProt.SIP.authOptimizedInFailover` is 0, this feature is disabled. If both parameters are set, the value of `reg.x.auth.optimizedInFailover` takes precedence.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.CID.sourcePreference	**ASCII string up to 120-characters long**	**Null**
The priority order for the sources of caller ID information. The headers can be in any order. If Null, caller ID information comes from P-Asserted-Identity, Remote-Party-ID, and From in that order. The values `From,P-Asserted-Identity,Remote-Party-ID`, and `P-Asserted-Identity,From,Remote-Party-ID` are also valid.		
voIpProt.SIP.compliance.RFC3261.validate.contentLanguage	**0 or 1**	**1**
If set to 1, validation of the SIP header content language is enabled. If set to 0, validation is disabled.		
voIpProt.SIP.compliance.RFC3261.validate.contentLength	**0 or 1**	**1**
If set to 1, validation of the SIP header content length is enabled.		
voIpProt.SIP.compliance.RFC3261.validate.uriScheme	**0 or 1**	**1**
If set to 1, validation of the SIP header URI scheme is enabled. If set to 0, validation is disabled.		
voIpProt.SIP.conference.address	**ASCII string up to 128 characters long**	**Null**
If Null, conferences are set up on the phone locally. If set to some value, conferences are set up by the server using the conferencing agent specified by this address. Acceptable values depend on the conferencing server implementation policy.		
voIpProt.SIP.conference.parallelRefer	**0 or 1**	**0**
If 1, a parallel REFER is sent to the call server. Note: This parameter must be set for Siemens Openscape Centralized Conferencing.		
voIpProt.SIP.connectionReuse.useAlias	**0 or 1**	**0**
If set to 0, the alias parameter is not added to the via header. If set to 1, the phone uses the connection reuse draft, which introduces an alias.		
voIpProt.SIP.csta	**0 or 1**	**0**
If 0, the uaCSTA (User Agent Computer Supported Telecommunications Applications) feature is disabled. If 1, uaCSTA is enabled. (If `reg.x.csta` is set, it will override this parameter.)		
voIpProt.SIP.dialog.strictXLineID	**0 or 1**	**0**
If 0, the phone will not look for x-line-id (call appearance indec) in a SIP INVITE message, if one is not present. Instead, when it receives INVITE, the phone will generate the call appearance locally and pass that information to other parties involved in the call.		
voIpProt.SIP.dialog.usePvalue	**0 or 1**	**0**
If set to 0, phone uses a `pval` field name in Dialog. This obeys the draft-ietf-sipping-dialog-package-06.txt draft. If set to 1, the phone uses a field name of `pvalue`.		
voIpProt.SIP.dialog.useSDP	**0 or 1**	**0**
If set to 0, a new dialog event package draft is used (no SDP in dialog body). If set to 1, for backward compatibility, use this setting to send SDP in the dialog body.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.dtmfViaSignaling.rfc2976[1]	**0 or 1**	**0**
If set to 1, DTMF digit information is sent in RFC2976 SIP INFO packets during a call. If set to 0, no DTMF digit information is sent.		
voIpProt.SIP.enable[1]	**0 or 1**	**1**
A flag to determine whether the SIP protocol is used for call routing, dial plan, DTMF, and URL dialing. If set to 1, the SIP protocol is used.		
voIpProt.SIP.failoverOn503Response	**0 or 1**	**1**
A flag to determine whether or not to trigger a failover if the phone receives a 503 response.		
voIpProt.SIP.header.diversion.enable[1]	**0 or 1**	**0**
If set to 1, the diversion header is displayed if received. If set to 0, the diversion header is not displayed.		
voIpProt.SIP.header.diversion.list.useFirst[1]	**0 or 1**	**1**
If set to 1, the first diversion header is displayed. If set to 0, the last diversion header is displayed.		
voIpProt.SIP.header.warning.codes.accept	**comma separated list**	**Null**
Specify a list of accepted warning codes. If set to Null, all codes are accepted. Only codes between 300 and 399 are supported. For example, if you want to accept only codes 325 to 330: `voIpProt.SIP.header.warning.codes.accept=325,326,327,328,329,330`. Text is shown in the appropriate language.		
voIpProt.SIP.header.warning.enable	**0 or 1**	**0**
If set to 1, the warning header is displayed if received. If set to 0, the warning header is not displayed.		
voIpProt.SIP.IM.autoAnswerDelay	**0 to 40, seconds**	**10**
The time interval from receipt of the instant message invitation to automatically accepting the invitation.		
voIpProt.SIP.keepalive.sessionTimers	**0 or 1**	**0**
If set to 1, the session timer is enabled. If set to 0, the session timer is disabled, and the phone does not declare "timer" in "Support" header in an INVITE. The phone still responds to a re-INVITE or UPDATE. The phone does not try to re-INVITE or UPDATE even if the remote end point asks for it.		
voIpProt.SIP.lcs	**0 or 1**	**0**
If 0, the Microsoft Live Communications Server (LCS) is not supported. If 1, LCS is supported. This parameter can set for a specific registration using `reg.x.lcs`.		
voIpProt.SIP.lineSeize.retries	**3 to 10**	**10**
Controls the number of times the phone retries to notify when attempting to seize a line (BLA).		
voIpProt.SIP.local.port[1]	**0 to 65535**	**5060**
The local port for sending and receiving SIP signaling packets. If set to 0, 5060 is used for the local port but is not advertised in the SIP signaling. If set to some other value, that value is used for the local port and it is advertised in the SIP signaling.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.ms-forking	**0 or 1**	**0**
If set to 0, support for MS-forking is disabled. If set to 1, support for MS-forking is enabled and the phone rejects all Instant Message INVITEs. This parameter applies when installing Microsoft Live Communications Server. Note that if any end point registered to the same account has MS-forking disabled, all other end points default back to non-forking mode. Windows Messenger does not use MS-forking so be aware of this behavior if one of the end points is using Windows Messenger.		
voIpProt.SIP.mtls.enable	**0 or 1**	**1**
If 0, Mutual TLS is disabled. If 1, Mutual TLS is enabled. Used in conjunction with Microsoft Lync 2010.		
voIpProt.SIP.musicOnHold.uri	**a SIP URI**	**Null**
A URI that provides the media stream to play for the remote party on hold. This parameter is used if `reg.x.musicOnHold.uri` is Null. Note: The SIP URI parameter transport is supported when configured with the values of `UDP`, `TCP`, or `TLS`.		
voIpProt.SIP.outboundProxy.address	**dotted-decimal IP address or hostname**	**Null**
The IP address or hostname of the SIP server to which the phone sends all requests.		
voIpProt.SIP.outboundProxy.port	**0 to 65535**	**0**
The port of the SIP server to which the phone sends all requests.		
voIpProt.SIP.outboundProxy.failOver.failBack.mode	**newRequests, DNSTTL, registration, duration,**	**newRequests**
The mode for failover failback (overrides `voIpProt.server.x.failOver.failBack.mode`). - `newRequests` All new requests are forwarded first to the primary server regardless of the last used server. - `DNSTTL` The phone tries the primary server again after a timeout equal to the DNS TTL configured for the server that the phone is registered to. - `registration` The phone tries the primary server again when the registration renewal signaling begins. - `duration` The phone tries the primary server again after the time specified by `reg.x.outboundProxy.failOver.failBack.timeout` expires.		
voIpProt.SIP.outboundProxy.failOver.failBack.timeout	**0, 60 to 65535**	**3600**
The time to wait (in seconds) before failback occurs (overrides `voIpProt.server.x.failOver.failBack.timeout`).If the failback mode is set to Duration, the phone waits this long after connecting to the current working server before selecting the primary server again. If 0, the phone does not failback until a failover event occurs with the current server.		
voIpProt.SIP.outboundProxy.failOver.failRegistrationOn	**0 or 1**	**0**
When set to 1, and the `reRegisterOn` parameter is enabled, the phone silently invalidates an existing registration (if it exists), at the point of failing over. When set to 0, and the `reRegisterOn` parameter is enabled, existing registrations remains active. This means that the phone attempts a failback without first attempting to register with the primary server to determine if it has recovered. Note that `voIpProt.SIP.outboundProxy.failOver.RegisterOn` must be enabled.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.outboundProxy.failOver.onlySignalWithRegistered	**0 or 1**	**1**
When set to 1, and the reRegisterOn and failRegistrationOn parameters are enabled, no signaling is accepted from or sent to a server that has failed until failback is attempted or failover occurs. If the phone attempts to send signaling associated with an existing call via an unregistered server (for example, to resume or hold a call), the call ends. No SIP messages are sent to the unregistered server. When set to 0, and the reRegisterOn and failRegistrationOn parameters are enabled, signaling is accepted from and sent to a server that has failed (even though failback hasn't been attempted or failover hasn't occurred). This parameter overrides voIpProt.server.x.failOver.onlySignalWithRegistered.		
voIpProt.SIP.outboundProxy.failOver.reRegisterOn	**0 or 1**	**0**
This parameter overrides the voIpProt.server.x.failOver.reRegisterOn. When set to 1, the phone attempts to register with (or via, for the outbound proxy scenario) the secondary server. If the registration succeeds (a 200 OK response with valid expires), signaling proceeds with the secondary server. When set to 0, the phone won't attempt to register with the secondary server, since the phone assumes that the primary and secondary servers share registration information.		
voIpProt.SIP.outboundProxy.transport	**DNSnaptr, TCPpreferred, UDPOnly, TLS, TCPOnly**	**DNSnaptr**
The transport method the phone uses to communicate with the SIP server. - Null or DNSnaptr if reg.x.outboundProxy.address is a hostname and reg.x.outboundProxy.port is 0 or Null, do NAPTR and then SRV lookups to try to discover the transport, ports and servers, as per RFC 3263. If reg.x.outboundProxy.address is an IP address, or a port is given, then UDP is used. - TCPpreferred TCP is the preferred transport, UDP is used if TCP fails. - UDPOnly Only UDP is used. - TLS If TLS fails, transport fails. Leave port field empty (defaults to 5061) or set to 5061. - TCPOnly Only TCP is used.		
voIpProt.SIP.pingInterval	**0 to 3600**	**0**
The number in seconds to send ping message. This feature is disabled by default.		
voIpProt.SIP.pingMethod	**PING, OPTIONS**	**PING**
The ping method used.		
voIpProt.SIP.presence.nortelShortMode[1]	**0 or 1**	**0**
Different headers sent in SUBSCRIBE when used for presence on an Avaya (Nortel) server. Support is indicated by adding a header Accept-Encoding: x-nortel-short. A PUBLISH is sent to indicate the status of the phone.		
voIpProt.SIP.requestValidation.digest.realm[1]	**A valid string**	**PolycomSPIP**
Determines the string used for Realm.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.requestValidation.x.method[1]	**Null, source, digest, both, all**	**Null**
If Null, no validation is made. Otherwise this sets the type of validation performed for the request: - **source** Ensure request is received from an IP address of a server belonging to the set of target registration servers. - **digest** Challenge requests with digest authentication using the local credentials for the associated registration (line). - **both** or **all** Apply both of the methods.		
voIpProt.SIP.requestValidation.x.request[1]	**INVITE, ACK , BYE, REGISTER, CANCEL, OPTIONS, INFO, MESSAGE, SUBSCRIBE, NOTIFY, REFER, PRACK, UPDATE**	**Null**
Sets the name of the method for which validation is applied. Note: Intensive request validation may have a negative performance impact due to the additional signaling required in some cases.		
voIpProt.SIP.requestValidation.x.request.y.event[1]	**A valid string**	**Null**
Determines which events specified with the Event header should be validated; applicable only when `voIpProt.SIP.requestValidation.x.request` is set to `SUBSCRIBE` or `NOTIFY`. If set to Null, all events are validated.		
voIpProt.SIP.requestURI.E164.addGlobalPrefix	**0 or 1**	**0**
If set to 1, + global prefix is added to the E.164 user parts in SIP: URIs.		
voIpProt.SIP.sendCompactHdrs	**0 or 1**	**0**
If set to 0, SIP header names generated by the phone use the long form)for example, `From`). If set to 1, SIP header names generated by the phone use the short form (for example, `f`).		
voIpProt.SIP.serverFeatureControl.cf[1]	**0 or 1**	**0**
If set to 1, server-based call forwarding is enabled. The call server has control of call forwarding. If set to 0, server-based call forwarding is not enabled.		
voIpProt.SIP.serverFeatureControl.dnd[1]	**0 or 1**	**0**
If set to 1, server-based DND is enabled. The call server has control of DND. If set to 0, server-based DND is not enabled.		
voIpProt.SIP.serverFeatureControl.missedCalls[1]	**0 or 1**	**0**
If set to 1, server-based missed calls is enabled. The call server has control of missed calls. If set to 0, server-based missed calls is not enabled.		
voIpProt.SIP.serverFeatureControl.localProcessing.cf	**0 or 1**	**1**
If set to 0 and `voIpProt.SIP.serverFeatureControl.cf` is set to 1, the phone will not perform local call-forward behavior. If set to 1, the phone performs local call-forward behavior on all calls received.		

Parameter	*Permitted Values*	*Default*
voIpProt.SIP.serverFeatureControl.localProcessing.dnd	**0 or 1**	**1**
If set to 0 and `voIpProt.SIP.serverFeatureControl.dnd` is set to 1, the phone will not perform local DND call behavior. If set to 1, the phone performs local DND call behavior on all calls received.		
voIpProt.SIP.specialEvent.checkSync.alwaysReboot[1]	**0 or 1**	**0**
If set to 1, always reboot when a NOTIFY message is received from the server with event equal to check-sync. If set to 0, only reboot if any of the files listed in <*MAC-address*>.cfg have changed on the FTP server when a NOTIFY message is received from the server with event equal to check-sync.		
voIpProt.SIP.specialEvent.lineSeize.nonStandard[1]	**0 or 1**	**1**
If set to 1, process a 200 OK response for a line-seize event SUBSCRIBE as though a line-seize NOTIFY with Subscription State: active header had been received. This speeds up processing.		
voIpProt.SIP.strictLineSeize	**0 or 1**	**0**
If set to 1, The phone is forced to wait for a 200 OK response when receiving a TRYING notify. If set to 0, dial prompt is provided immediately when you attempt to seize a shared line without waiting for a successful OK from the call server.		
voIpProt.SIP.strictReplacesHeader	**0 or 1**	**1**
This parameter applies only to directed call pick-up attempts initiated against monitored BLF resources. If set to 1, the phone requires call-id, to-tag, and from-tag to perform a directed call-pickup when `call.directedCallPickupMethod` is configured as `native`. If set to 0, call pick-up requires a call-id only.		
voIpProt.SIP.strictUserValidation	**0 or 1**	**0**
If set to 1, the phone is forced to match the user portion of signaling exactly. If set to 0, the phone uses the first registration if the user part does not match any registration.		
voIpProt.SIP. supportFor100rel	**0 or 1**	**1**
If set to 1, the phone advertises support for reliable provisional responses in its offers and responses. If set to 0, the phone does not offer 100rel and rejects offers requiring 100rel.		
voIpProt.SIP.tcpFastFailover	**0 or 1**	**0**
If set to 1, failover occurs based on the values of `reg.x.server.y.retryMaxCount` and `voIpProt.server.x.retryTimeOut`. If 0, a full 32-second RFC-compliant timeout is used. See reg.x.tcpFastFailover.		
voIpProt.SIP.tlsDsk.enable	**0 or 1**	**0**
If 0, TLS DSK is disabled. If 1, TLS DSK is enabled. For more information, see Session Initiation Protocol (SIP) Authentication Extensions Protocol Overview on Microsoft Developer Network.		
voIpProt.SIP.turnOffNonSecureTransport[1]	**0 or 1**	**0**
If set to 1, stop listening to port 5060 when using AS-SIP enabled.		
voIpProt.SIP.use486forReject	**0 or 1**	**0**
If set to 1 and the phone is indicating a ringing inbound call appearance, the phone transmits a 486 response to the received INVITE when the Reject soft key is pressed. If set to 0, no 486 response is transmitted.		

Parameter	*Permitted Values*	*Default*
voipPort.SIP.useCompleteUriForRetrieve	**0 or 1**	**1**
If set to 1, the target URI in BLF signaling uses the complete address as provided in the XML dialog document. If set to 0, only the user portion of the XML dialog document is used and the current registrar's domain is appended to create the full target URI.		
voipPort.SIP.useLocalTargetUriforLegacyPickup	**0 or 1**	**1**
If set to 1, BLF signaling uses the address as provided in the local target URI in the XML dialog document with additional rules based on `voipPort.SIP.useCompleteUriForRetrieve`. If set to 0, the local target URI is not considered and instead the identity attribute is used with additional rules based on `voipPort.SIP.useCompleteUriForRetrieve`.		
voIpProt.SIP.useContactInReferTo	**0 or 1**	**0**
If set to 0, the "To URI" is used in the REFER. If set to 1, the "Contact URI" is used in the REFER.		
voIpProt.SIP.useRFC2543hold	**0 or 1**	**0**
If set to 0, use SDP media direction parameters (such as a=sendonly) per RFC 3264 when initiating a call. Otherwise use the obsolete c=0.0.0.0 RFC2543 technique. In either case, the phone processes incoming hold signaling in either format. Note: `voIpProt.SIP.useRFC2543hold` is effective only when the call is initiated.		
voIpProt.SIP.useSendonlyHold	**0 or 1**	**1**
If set to 1, the phone sends a reinvite with a stream mode parameter of `sendonly` when a call is put on hold. This is the same as the previous behavior. If set to 0, the phone sends a reinvite with a stream mode parameter of `inactive` when a call is put on hold. NOTE: The phone ignores the value of this parameter if set to 1 when the parameter `voIpProt.SIP.useRFC2543hold` is also set to 1 (default is 0).		

[1] Change causes phone to restart or reboot.

Get Help

This section provides a list of Polycom documents referred to in this guide as well as partner resources you can use. For more information on using and configuring Polycom phones, view the following resources on Polycom Voice Support.

- To update Polycom phones with the latest UC software, see the Latest Polycom® UC Software Release page on the Polycom Voice Support Web site.
- For details on how to provision your Polycom phones with the latest UC software, see the *Polycom UC Software 4.1.0 Administrators's Guide*.
- For information on using the VVX Expansion Module, see the *Feature Profile: Using Polycom VVX Expansion Modules with Polycom VVX Business Media Phones*.
- For more detailed information about power consumption on Polycom phones, see *Engineering Advisory 48152: Power Consumption on Polycom Phones*.

If you are looking for help or technical support for your Polycom phones, the following types of documents are available on the Business Media Phones page on the Polycom Voice Support site:

- Quick Start Guides, which show you how to assemble your phone.
- Quick User Guides, which describe basic phone features.
- User Guides, which describe both basic and advanced phone features.

Polycom and Partner Resources

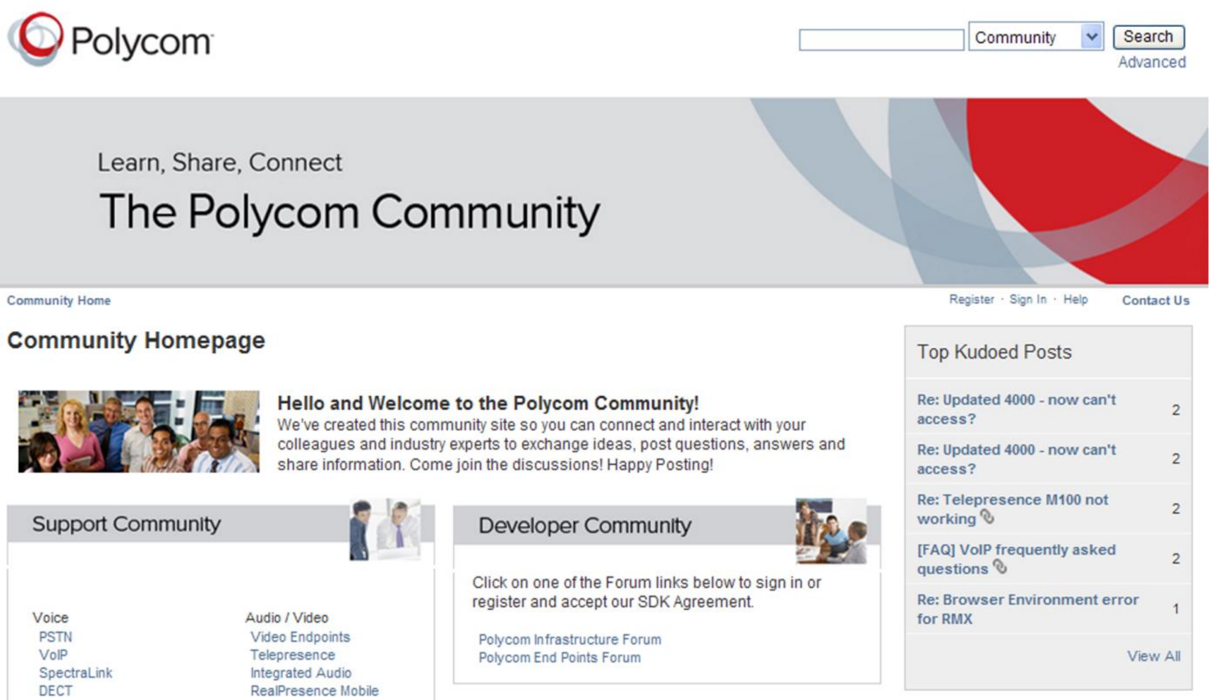
For more information about installing, configuring, and administering Polycom products, refer to Documents and Downloads at Polycom Support.

To find all Polycom partner solutions, see Polycom Global Strategic Partner Solutions.

For more information on solution with this Polycom partner, see the partner site at Polycom Global Strategic Partner Solutions.

The Polycom Community

The Polycom Community gives you access to the latest developer and support information. Participate in discussion forums to share ideas and solve problems with your colleagues. To register with the Polycom Community, simply create a Polycom online account. When logged in, you can access Polycom support personnel and participate in developer and support forums to find the latest information on hardware, software, and partner solutions topics.



Troubleshoot

This section shows you some tools and techniques for identifying issues and troubleshooting Polycom phones and expansion modules running Polycom UC software.

Use the following list as a guide to testing hardware and resolving issues, problems, or common difficulties you may encounter while deploying this solution.

To view warnings or hardware diagnostics:

» Do one of the following:

- For Warning Messages, select **Settings > Status > Diagnostics > Warnings**
- For Diagnostics, select **Settings > Status > Diagnostics > Test Hardware**, and select one of the following options:
  - **Keypad Diagnostics** Tests the line keys, hard keys, and page keys on the phone and expansion module.
  - **LED Diagnostics** Tests the LED lights on the phone's hard keys and expansion module's line keys.
  - **Display Diagnostics** Tests the display screen on the phone and Color expansion module.
  - **Brightness Diagnostics** Tests the brightness levels of the display screen on the phone and Color expansion modules.
- For Line Key Information, select **Settings > Status > Line Key Info** and press a line key to view information for that line key.

Flexible line key assignments do not display or display in an incorrect order on the Color expansion module.

If the flexible line key assignments you configured for your module are not displaying or display incorrectly on the EM, you need to detach the auxiliary cable connecting the first EM to the VVX phone. Wait a moment, then reattach the cable to the phone. Note that if you reconnect the cable to the phone immediately after disconnection, the software can fail to account for the number of connected EMs.