Cisco BroadWorks

Device Management

Configuration Guide

Release 23.0

Document Version 5

Notification

The BroadSoft BroadWorks has been renamed to Cisco BroadWorks. Beginning in September 2018, you will begin to see the Cisco name and company logo, along with the new product name on the software, documentation, and packaging. During this transition process, you may see both BroadSoft and Cisco brands and former product names. These products meet the same high standards and quality that both BroadSoft and Cisco are known for in the industry.

Copyright Notice

Trademarks

Any product names mentioned in this document may be trademarks or registered trademarks of Cisco or their respective companies and are hereby acknowledged.

Document Revision History

Release	Version	Reason for Change	Date	Author
14.0	1	Created document for Release 14.sp6.	May 28, 2008	Alexis B. Deschamps
14.0	1	Completed first draft for Release 14.sp6.	July 7, 2008	Alexis B. Deschamps
14.0	1	Updated document following internal review for Release 14.sp6.	July 11, 2008	Alexis B. Deschamps
14.0	1	Added introduction and restructured flow of document for Release 14.sp6.	July 17, 2008	Jamie Palmer
14.0	1	Added information to the deployment models and migration sections for Release 14.sp6.	July 18, 2008	Alexis B. Deschamps
14.0	1	Updated document following official review for Release 14.sp6.	July 28, 2008	Alexis B. Deschamps
14.0	1	Edited and published document.	August 7, 2008	Andrea Fitzwilliam
15.0	1	Updated document for Release 15.0.	August 8, 2008	Alexis B. Deschamps
15.0	1	Edited and published document.	August 8, 2008	Patricia Renaud
15.0	2	Clarified the value to be entered for *deviceAccessAppServerClusterName*.	September 11, 2008	Yves Racine
15.0	2	Edited changes and published document.	September 12, 2008	Andrea Fitzwilliam
15.0	3	Updated the document as the creation of the device-type Polycom directory file has changed to automatically load a default template, change the remote file name to %BWDEVICEID%-directory.xml, and inherit the authentication configuration from the device type.	September 22, 2008	Yves Racine
15.0	3	Edited changes in document.	September 24, 2008	Andrea Fitzwilliam
15.0	3	Updated the document to add missing access profile standard and advanced options to sections *5.2.3 Standard Options* and *5.2.4 Advanced Options* for EV 67996.	September 25, 2008	Yves Racine
15.0	3	Edited and published document.	September 26, 2008	Patricia Renaud
15.0	4	Updated the document to correct errors in sections *5.2.3 Standard Options* and *5.2.4 Advanced Options*, indicating the formula for calculating the maximum number of events that the Provisioning Server can process in a schedule period (section *5.7 Generate File*). Renamed tags BWSCA-BARGE-IN-* to BWSCA-BRIDGING-* (section *5.13.2 Use of Tags in Templates*). Specified that the Profile Server FQDN should be resolved by the DNS using a fixed order (section *5.17.8.3 Add File Repository*).	September 29, 2008	Yves Racine
15.0	4	Edited changes and published document.	October 3, 2008	Andrea Fitzwilliam

Release	Version	Reason for Change	Date	Author
15.0	5	Updated the document to add a confirmation password text box for the device type add/modify pages and for the device add/modify pages at the group, service provider/enterprise, and system level.	October 3, 2008	Yves Racine
15.0	5	Edited changes and published document.	October 5, 2008	Patricia Renaud
15.0	6	Made a minor change in section *8.2.1.5 Enable Polycom Phone Services* for EV 66416.	October 30, 2008	Goska Auerbach
15.0	6	Updated sections *5.17.4 Network Access Lists*, *5.17.7 Network Server Configuration*, and *8.1.2 Network Server Configuration* for EV 69928.	November 5, 2008	Goska Auerbach
15.0	6	Edited changes and published document.	November 6, 2008	Andrea Fitzwilliam
15.0	7	Added dynamic tag %BWLINE-ENABLED-x%.	November 15, 2008	Yves Racine
15.0	7	Added section *8.2.1.5 Enable Polycom Phone Services* to indicate how the Polycom phone directories are migrated for EV 66535.	November 22, 2008	Yves Racine
15.0	7	Added dynamic tag %BWFQDEVICEID% for EV 69558.	November 24, 2008	Yves Racine
15.0	7	Edited changes and published document.	November 25, 2008	Andrea Fitzwilliam
15.0	8	Updated sections *5.2 Define Access Profiles*, *5.3 Define Configuration Profiles*, and *Appendix A: Generic Device/Identity Access Profile Definitions* for EV 87052.	December 4, 2008	Goska Auerbach
15.0	8	Updated section *8.1.1 Profile Server Configuration* for EV 84012.	December 9, 2008	Goska Auerbach
15.0	8	Removed the restriction on static file not being customizable.	January 8, 2009	Yves Racine
15.0	8	Edited changes and published document.	January 14, 2009	Andrea Fitzwilliam
16.0	1	Added system-defined tags for feature access codes added for Release 15.0 and Release 16.0.	March 2, 2009	Charles Leduc
16.0	1	Updated section *5.17.5.2 HTTP Configuration* for EV 66819.	May 4, 2009	Tony Pilote
16.0	1	Edited changes.	May 4, 2009	Patricia Renaud
16.0	1	Updated section *8.1.1 Profile Server Configuration* for EV 91883.	May 14, 2009	Roberta Boyle
16.0	1	Updated document for Release 16.0.	May 28, 2009	Yves Racine
16.0	1	Edited changes made for EVs 66819 and 91883.	June 3, 2009	Andrea Fitzwilliam
16.0	1	Edited changes and published document.	August 24, 2009	Andrea Fitzwilliam

Release	Version	Reason for Change	Date	Author
16.0	2	Added the tag *%BWDIALPLAN-OUTSIDE-ACCESS-CODE-[1-5]-x%* and introduced a wildcard symbol for device access Uniform Resource Identifier (URI). Updated document for EVs 99029 and 98029.	September 4, 2009	Yves Racine
16.0	2	Updated document following the usability enhancements.	September 18, 2009	Alexis Bazinet-Deschamps
16.0	2	Edited changes and published document.	September 23, 2009	Andrea Fitzwilliam
16.0	3	Updated section *5.13.2 Use of Tags in Templates* for EV 95007.	September 29, 2009	Tim Babin
16.0	3	Updated section *9.4 Common Problems* for EV 103347.	November 26, 2009	Roberta Boyle
16.0	3	Edited changes and published document.	January 28, 2010	Patricia Renaud
17.0	1	Updated document for Release 17.0.	March 9, 2010	Alexis Bazinet-Deschamps
17.0	1	Edited changes and published document.	April 13, 2010	Andrea Fitzwilliam
17.0	2	Updated section *5.11 Set Time Zone* for EV 101376.	April 19, 2010	Yves Racine
17.0	2	Edited changes and published document.	May 17, 2010	Andrea Fitzwilliam
17.0	3	Updated section *5.13 Use Tags and Tag Sets* for EV 104978.	June 2, 2010	Goska Auerbach
18.0	1	Added the following tags to section *5.13.2 Use of Tags in Templates*: %BWFAC-MONITORING-NEXT-CALL% tag for EV 121764 and %BWSHAREDLINE-BINARY%, %BWSHAREDLINE-BOOL%, %BWSHAREDLINE-ENABLED%, and %BWSHAREDLINE-SWITCH% for EV 110710.	November 16, 2010	Charles Leduc
18.0	1	Clarified overload control mechanism in section *5.17.5.5 Back-end Communication* for EV 110589.	December 1, 2010	Alexis Bazinet-Deschamps
18.0	1	Updated references to Xchange.	February 10, 2011	Goska Auerbach
18.0	1	Updated section *5.3 Define Configuration Profiles* for Release 18.0.	September 22, 2011	Réal Boivin
18.0	1	Updated screen captures throughout the document.	September 28, 2011	Réal Boivin
18.0	1	Updated section *5.17.6 Profile Server Configuration* for EV 142277. Updated section *10 Known Limitations* for EV 141997. Updated sections *5.17.6 Profile Server Configuration* and *8.1 Create New Device Type on Fresh Install* for EV 109762 and updated software version in section *5.17.5 Xtended Services Platform Configuration*.	October 6, 2011	Goska Auerbach

Release	Version	Reason for Change	Date	Author
18.0	1	Updated section *5.2.4 Advanced Options* with a description of the advanced options added in Release 18.0.	October 31, 2011	Goska Auerbach
18.0	1	Updated section *5.3.2.2 Device Management Options* for EV 152913.	November 01, 2011	Réal Boivin
18.0	1	Updated section *5.13 Use Tags and Tag Sets* for EV 156022.	December 13, 2011	Goska Auerbach
18.0	1	Added section *5.3 Define Configuration Profiles* for EV 156435.	January 4, 2012	Réal Boivin
18.0	1	Edited and published document.	January 25, 2012	Patricia Renaud
18.0	2	Updated section *5.5 Device Reset* to include information about email notifications.	March 29, 2012	Goska Auerbach
18.0	2	Edited changes and published document.	April 2, 2012	Jessica Boyle
19.0	1	Added new BWFAC tags in section *5.13.2 Use of Tags in Templates* for the Hunt Group busy feature. Updated section *5.13 Use Tags and Tag Sets* for EV 146324.	April 2, 2012	Martin Chabbert
19.0	1	Updated and renamed section *5.7.3 Automatic Rebuild Configuration* for EV 152314 – Device Management File Upload.	May 9, 2012	Jean-Francois Laporte
19.0	1	Edited changes to document.	May 22, 2012	Jessica Boyle
19.0	1	Updated section *5.5 Device Reset* for EV 149047	July 31, 2012	Réal Boivin
19.0	1	Edited changes to the document.	August 7, 2012	Jessica Boyle
19.0	1	Updated document for EV 152312 Device Management (DM) Line/Port Ordering Enhancement.	August 10, 2012	Réal Boivin
19.0	1	Updated section *5.17.5.4 DeviceManagementTFTP Application* for EV 157904.	August 10, 2012	Michel Guénette
19.0	1	Edited changes to document.	August 21, 2012	Jessica Boyle
19.0	1	Corrected definition of BWSERVERADDRESS in section *5.13 Use Tags and Tag Sets*.	August 21, 2012	Juliette D'Almeida
19.0	1	Updated section *5.11 Set Time Zone* for EV 172084.	September 20, 2012	Husam Al Sahli
19.0	1	Added new BW tags in section *5.13.2 Use of Tags in Templates* for the Device Feature Synchronization Enhancements feature.	October 30, 2012	Jean-Francois Laporte
19.0	1	Minor text modification to section *5.13.2 Use of Tags in Templates* for EV 172309.	October 30, 2012	Jean-Francois Laporte

Release	Version	Reason for Change	Date	Author
19.0	1	Updated section *5.17.8.3 Add File Repository* for EV 171923. Added an explanation on how to configure usage of a secure connection to access a WebDAV file repository.	October 31, 2012	Michel Guénette
19.0	1	Edited changes and published document.	November 16, 2012	Patricia Renaud
19.0	2	Updated section *5.13.2 Use of Tags in Templates* by adding a clarification to the tags %BWAUTHUSER-x% and %BWAUTHPASSWORD-x% for EV 142965.	November 28, 2012	Réal Boivin
20.0	1	Updated document for Release 20.0.	December 10, 2012	Jean-Francois Laporte
20.0	1	Updated section *5.13.2 Use of Tags in Templates* for EV 172309.	January 7, 2012	Jean-Francois Laporte
20.0	1	Updated section *5.17.5.2 HTTP Configuration* for EV 143355.	January 21, 2012	Pierre Drapeau
20.0	1	Repaired document.	January 21, 2013	Goska Auerbach
20.0	1	Updated section *5.13 Use Tags and Tag Sets* for EV 152315.	January 25, 2013	Réal Boivin
20.0	1	Updated section *5.12 Primary User* for EV 178161.	January 28, 2013	Réal Boivin
20.0	1	Added *Appendix C: Visual Device Management Support Provisioning* and updated section *5.2.4 Advanced Options* for EV 197032.	July 5, 2013	Réal Boivin
20.0	1	Added section *5.13.7.1 Tag Customization on Dynamic Per-Type Files* for EV 191589	August 13, 2013	Jean-Francois Laporte
20.0	1	Updated section *5.16.1 Dedicated Repository* for EV 192403.	August 14, 2013	Jean-Francois Laporte
20.0	1	Added section *5.9 Extended File Capture Mode* for EV 178321.	August 19, 2013	Jean-Francois Laporte
20.0	1	Updated section *5.14.5 Directory File* for EV 193561.	August 27, 2013	Jean-Francois Laporte
20.0	1	Updated section *5.13.3 Use of Tags in File Names* for EV 172309.	August 29, 2013	Jean-Francois Laporte
20.0	1	Updated section *4.2.3 Device Management Files* for EV 184164.	August 30, 2013	Jean-Francois Laporte
20.0	1	Edited changes and published document.	October 23, 2013	Patricia Renaud
21.0	1	Updated section *5.17.5.5 Back-end Communication*, *5.17.6.1 File Repository Web Application*, and *8.1.1 Profile Server Configuration* for EV 208740.	December 2, 2013	Renaud Beaudry-Magnan
21.0	1	Updated sections *5.13.2 Use of Tags in Templates* and *15.3.3 Tags Supporting PRIMARY Keyword* for EV 177502.	December 12, 2013	Réal Boivin

Release	Version	Reason for Change	Date	Author
21.0	1	Updated sections *5.3 Define Configuration Profiles*, *5.5 Device Reset*, *5.6 Customize Files*, and *5.7 Generate File* for EV 210524.	January 21, 2014	Réal Boivin
21.0	1	Updated section*5.13.2 Use of Tags in Templates* for EV 214890.	February 17, 2014	Réal Boivin
21.0	1	Updated section *5.13.2 Use of Tags in Templates* and deleted section *15.3.3 Tags Supporting PRIMARY Keyword* for EV 176149.	March 3, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.3.2.2 Device Management Options* for EV 219043.	March 17, 2014	Michel Guénette
21.0	1	Updated section *5.17.5.2 HTTP Configuration* for EV 209298.	March 20, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.11 Set Time Zone* for EV 214028.	April 2, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.3 Define Configuration Profiles* for EV 208481.	June 10, 2014	Réal Boivin
21.0	1	Updated section *5.2.4 Advanced Options* for EV 205504.	August 19, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.14.5 Directory File* for EV 220848.	August 22, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.17.6.1 File Repository Web Application* for EV 227181.	August 22, 2014	Jean-Francois Laporte
21.0	1	Updated section *5.3.2 Device Configuration Options Summary* for EV 223433.	August 25, 2014	Jean-Francois Laporte
21.0	1	Added section *7.1 Enable Server Thread Names in Logs*.	September 16, 2014	Jean-Francois Laporte
21.0	1	Added sections *5.5.1 Device Reset Pacing* and *5.7.3 Automatic Rebuild Configuration*. Updated sections *5.7 Generate File* and *7.8 Profile Server File Repository*.	October 8, 2014	Jean-Francois Laporte
21.0	1	Updated copyright notice and trademarks. Edited changes.	October 14, 2014	Joan Renaud
21.0	1	Removed note from section *4.2.3 Device Management Files* for EV 201827.	October 23, 2014	Joan Renaud
21.0	1	Updated BroadSoft and BroadWorks logos.	October 30, 2014	Joan Renaud
21.0	1	Added section *7.8 Profile Server File Repository* for EV 221286.	October 30, 2014	Michel Guénette
21.0	1	Edited changes and published document.	October 30, 2014	Joan Renaud
21.0	2	Updated section *5.2.4 Advanced Options* for EV 247827.	January 27, 2015	Jean-Francois Laporte
21.0	2	Updated document with rebranded server icons.	February 23, 2015	Joan Renaud

Release	Version	Reason for Change	Date	Author
21.0	2	Edited changes and published document.	March 13, 2015	Joan Renaud
21.0	3	Updated sections *5.13.3 Use of Tags in File Names* and *5.13.7 Customize Static Tags*. Added section *5.13.4 Use of Tags for Dynamic Per-Type Files* for PR-47210.	April 14, 2015	Jean-Francois Laporte
21.0	3	Restructured section *1 Summary of Changes*.	April 16, 2015	Goska Auerbach
21.0	3	Updated section *5.17.8.2 Device Management Settings* for EV 190903.	May 28, 2015	Charles Leduc
21.0	3	Corrected section numbering.	June 25, 2015	Goska Auerbach
22.0	1	Created Release 22.0 version of the document.	June 25, 2015	Goska Auerbach
22.0	1	Updated document for BW-2318 and BW-2319: added new *macMacInCert* element and changed the definition of *macFormatInNonRequestURI*. Updated figures accordingly.	June 29, 2015	Réal Boivin
22.0	1	Updated section *5.3.2.2.2 Device Type File Configuration Options* for PR-47986.	July 24, 2015	Jean-Francois Laporte
22.0	1	Updated section *5.13 Use Tags and Tag Sets* to add support for encrypted custom tags (in XS mode only).	May 18, 2016	Réal Boivin
22.0	1	Updated section *5.3.2.2.1 Device Type Configuration Options* for PR-51289.	June 15, 2016	Réal Boivin
22.0	1	Updated section *5.13.1 Overview* for PR-51814.	August 9, 2016	Jean-Francois Laporte
22.0	1	Edited changes and published document.	November 24, 2016	Joan Renaud
22.0	2	Added section *4.2.4 File Caching* and updated section *7.2 Enable File Caching* for PR-53073.	December 21, 2016	Jean-Francois Laporte
22.0	2	Edited changes and published document.	March 6, 2017	Jessica Boyle
23.0	1	Created Release 23.0 version of the document. Updated section *7.8 Profile Server File Repository* for PR-55099.	May 9, 2017	Gabriel Petrella
23.0	1	Updated section *5.6 Customize Files* for PR-55588.	30 August , 2017	Goska Auerbach
23.0	1	Updated section *5.17.5.2 HTTP Configuration* for PR-56504.	19 September 19, 2017	Goska Auerbach
23.0	1	Created *5.3.2.2.2.1 Multiple Authentication Support* for PR-56935.	November 9, 2017	Jean-Francois Laporte
23.0	1	Updated sections *5.15.3 Network Server Device File Location Lookup* and *5.17.5.4 DeviceManagementTFTP Application* for PR-56121.	December 20, 2017	Jean-Francois Laporte
23.0	1	Added *Appendix D: Authorization Based on Device Certificate CN Provisioning* for PR-56504.	January 16, 2018	Réal Boivin
23.0	1	Made editorial changes.	January 17, 2018	Goska Auerbach

Release	Version	Reason for Change	Date	Author
23.0	1	Added section *7.10 Good Practices and Constraints* and updated section *7.6 Increase Xtended Services Platform/Profile Server Thread Pool* for PR-57670.	January 31, 2018	Husam Al Sahli
23.0	1	Updated for PR-58579 to remove references to the Premier Partner program, which ended several years ago.	May 14, 2018	Goska Auerbach
23.0	1	Created section *6 Security Considerations* and updated various sections for Release 23.0 features impacting Device Management.	July 25, 2018	Jean-Francois Laporte
23.0	1	Updated section *6 Security Considerations* with more details. Made the section more aligned with the DM security presentation from Tech Connect.	August 21, 2018	Jean-Francois Laporte
23.0	1	Rebranded document for Cisco. Edited changes and published document.	September 21, 2018	Joan Renaud
23.0	2	Updated section *5.13.2 Use of Tags in Templates* for PR-59775.	October 4, 2018	Jean-Francois Laporte
23.0	2	Edited changes and published document.	October 4, 2018	Patricia Renaud
23.0	3	Added file cache configuration guidelines.	October 22, 2018	Martin Bernier
23.0	3	Edited changes and published document.	October 25, 2018	Patricia Renaud
23.0	4	Removed the obsolete container option: *bw.dms.resetTxPerSec* for PR-60394.	January 16, 2019	Michel Guénette
23.0	4	Updated to include new device profile type parameters introduced via BW-18223 and BW-2283 for PR-60557.	February 6, 2019	Jean-Francois Laporte
23.0	4	Edited changes and published document.	February 13, 2019	Jessica Boyle
23.0	5	Completed rebranding for Cisco and republished document.	March 18, 2019	Patricia Renaud

Table of Contents

Table of Figures

List of Tables

1 Summary of Changes

This section describes the changes to this document for each release and document version.

1.1 Changes for Release 23.0

Release 23.0, Document Version 5

This version of the document includes the following change:

- Completed rebranding for Cisco and republished document.

Release 23.0, Document Version 4

This version of the document includes the following changes:

- Removed mention of the obsolete container option: *bw.dms.resetTxPerSec* for PR-60394.
- Updated to include new device profile type parameters introduced via BW-18223 and BW-2283 for PR-60557.

Release 23.0, Document Version 3

This version of the document includes the following change:

Updated section *4.2.4 File Caching* for PR-45876.

Release 23.0, Document Version 2

This version of the document includes the following change:

- Updated section *5.13.2 Use of Tags in Templates* for PR-59775.

Release 23.0, Document Version 1

This version of the document includes the following changes:

- Updated section *7.8 Profile Server File Repository* for PR-55099.
- Updated section *5.6 Customize Files* for PR-55588.
- Updated section *5.17.5.2 HTTP Configuration* for PR-56504.
- Created section *5.3.2.2.2.1 Multiple Authentication Support* for PR-56935.
- Updated sections *5.15.3 Network Server Device File Location Lookup* and *5.17.5.4 DeviceManagementTFTP Application* for PR-56121.
- Added *Appendix D: Authorization Based on Device Certificate CN Provisioning* for PR-56504.
- Updated section *7.6 Increase Xtended Services Platform/Profile Server Thread Pool* for PR-57670.
- Created section *7.10 Good Practices and Constraints* for PR-57670.
- Updated for PR-58579 to remove references to the Premier Partner program, which ended several years ago.
- Created section *6 Security Considerations* and updated various sections for Release 23.0 features impacting Device Management.
- Updated section *6 Security Considerations* with more details. Made it more aligned with the DM security presentation from Tech Connect.

1.2 Changes for Release 22.0

Release 22.0, Document Version 2

This version of the document includes the following changes:

- Added section *4.2.4 File Caching* and updated section *7.2 Enable File Caching* for PR-53073.

Release 22.0, Document Version 1

This version of the document includes the following changes:

- Updated document for BW-2318 and BW-2319: added new *macMacInCert* element and changed the definition of *macFormatInNonRequestURI*. Updated impacted figures accordingly.
- Updated section *5.3.2.2.2 Device Type File Configuration Options* for PR-47986.
- Updated section *5.13 Use Tags and Tag Sets* for BW-9442: Option to Encrypt Custom Tag Values.
- Updated section *5.3.2.2.1 Device Type Configuration Options* for PR-51289.
- Updated section *5.13.1 Overview* for PR-51814.

1.3 Changes for Release 21.0

Release 21.0, Document Version 3

This version of the document includes the following changes:

- Updated sections *5.13.3 Use of Tags in File Names* and *5.13.7 Customize Static Tags*. Added section *5.13.4 Use of Tags for Dynamic Per-Type Files* for PR-47210.
- Updated section *5.17.8.2 Device Management Settings* for EV 190903.

Release 21.0, Document Version 2

This version of the document includes the following changes:

- Updated section *5.2.4 Advanced Options* for EV 247827.
- Updated document with rebranded server icons.

Release 21.0, Document Version 1

This version of the document includes the following changes:

- Updated sections *5.17.5.5 Back-end Communication*, *5.17.6.1 File Repository Web Application*, and *8.1.1 Profile Server Configuration* for EV 208740.
- Updated sections *5.13.2 Use of Tags in Templates* and *5.13.3 Tags Supporting PRIMARY Keywords* for EV 177502.
- Updated sections *5.3 Define Configuration Profiles*, *5.5 Device Reset*, *5.6 Customize Files*, and *5.7 Generate File* for EV 210524.
- Updated section *5.13.2 Use of Tags in Templates* and deleted section *5.13.3 Tags Supporting PRIMARY Keywords* for EV 176149.
- Updated section *5.3.2.2 Device Management Options* for EV 219043.
- Updated section *5.17.5.2 HTTP Configuration* for EV 209298.
- Updated section *5.11 Set Time Zone* for EV 214028.
- Updated section *5.3 Define Configuration Profiles* for EV 208481.
- Updated section *5.2.4 Advanced Options* for EV 205504.

- Updated section *5.14.5 Directory File* for EV 220848.
- Updated section *5.17.6.1 File Repository Web Application* for EV 227181.
- Updated section *5.3.2 Device Configuration Options Summary* for EV 223433.
- Added section *7.1 Enable Server Thread Names in Logs*.
- Added sections *5.5.1 Device Reset Pacing* and *5.7.3 Automatic Rebuild Configuration*.
- Updated sections *5.7 Generate File* and *7.8 Profile Server File Repository*.
- Added section *7.8 Profile Server File Repository* for EV 221286.

1.4 Changes for Release 20.0

Release 20.0, Document Version 1

This version of the document includes the following changes:

- Updated section *5.13.2 Use of Tags in Templates* for EV 172309.
- Updated section *5.17.5.2 HTTP Configuration* for EV 143355.
- Updated section *5.13 Use Tags and Tag Sets* for EV 152315.
- Updated section *5.12 Primary User* for EV 178161.
- Updated section *5.2.4 Advanced Options* and added *Appendix A: Generic Device/Identity Access Profile Definitions* for EV 197032.
- Created section *5.13.7.1 Tag Customization on Dynamic Per-Type Files* for EV 191589.
- Updated section *5.16.1 Dedicated Repository* for EV 192403.
- Created section *5.9 Extended File Capture Mode* for EV 178321.
- Updated section *5.14.5 Directory File* for EV 193561.
- Updated section *5.13.3 Tags Supporting PRIMARY Keywords* for EV 172309.
- Updated section *4.2.3 Device Management Files* for EV 184164.

Release 19.0, Document Version 2

This version of the document includes the following change:

- Updated section *5.13.2 Use of Tags in Templates* by adding a clarification to the tags %BWAUTHUSER-x% and %BWAUTHPASSWORD-x% for EV 142965.

1.5 Changes for Release 19.0

Release 19.0, Document Version 1

This version of the document includes the following changes:

- Corrected definition of BWSERVERADDRESS in section *5.13 Use Tags and Tag Sets*.
- Updated section *5.11 Set Time Zone* for EV 172084.
- Added new BWFAC tags in section *5.13.2 Use of Tags in Templates* for the Hunt Group busy feature.
- Added new BW tags in section *5.13.2 Use of Tags in Templates* and *5.13.3 Tags Supporting PRIMARY Keywords* for the Device Feature Synchronization Enhancements feature.

- Updated section *5.13 Use Tags and Tag Sets* for EV 146324.
- Made minor modification to section *5.13.2 Use of Tags in Templates* for EV 172309.
- Updated section *5.17.8.3 Add File Repository* for EV 171923. Added an explanation on how to configure usage of a secure connection to access a WebDAV file repository.
- Updated document for EV 152312 Device Management (DM) Line/Port Ordering Enhancement.
- Updated section *5.17.5.4 DeviceManagementTFTP Application* for EV 157904.
- Updated section *5.5 Device Reset* for EV 149047.
- Updated section *5.13 Use Tags and Tag Sets* for EV 146324.
- Added new BWFAC tags in section *5.13.2 Use of Tags in Templates* for the Hunt Group busy feature.
- Updated and renamed section *5.7.3 Automatic Rebuild Configuration* for EV 152314 – Device Management File Upload.

1.6 Changes for Release 18.0

Release 18.0, Document Version 2

This version of the document includes the following change:

- Updated section *5.5 Device Reset* to include and clarify information about email notifications, which used to be in a separate section.

Release 18.0, Document Version 1

This version of the document includes the following changes:

- Updated section *5.13 Use Tags and Tag Sets* for EV 110710 and EV 121764.
- Updated section *5.17.5.5 Back-end Communication* for EV 110589.
- Updated screen captures throughout the document.
- Updated section *5.17.6 Profile Server Configuration* for EV 142277.
- Updated section *10 Known Limitations* for EV 141997.
- Updated sections *5.17.6 Profile Server Configuration* and *8.1 Create New Device Type on Fresh Install* for EV 109762 and updated software version in section *5.17.5 Xtended Services Platform Configuration*.
- Updated section *5.3 Define Configuration Profiles* for Release 18.0.
- Updated section *5.17.6 Profile Server Configuration* for EV 142277.
- Updated section *10 Known Limitations* for EV 141997.
- Updated sections *5.17.6 Profile Server Configuration* and *8.1 Create New Device Type on Fresh Install* for EV 109762 and updated software version in section *5.17.5 Xtended Services Platform Configuration*.
- Updated section *5.2.4 Advanced Options* with a description of the advanced options added in Release 18.0.
- Updated section *5.3.2.2 Device Management Options* for EV 152913. Added more information about the *macFormatInNonRequestURI* option**.**
- Updated section *5.13 Use Tags and Tag Sets* for EV 156022.

- Added section *5.3 Define Configuration Profiles* for EV 156435.

1.7 Changes for Release 17.0

Release 17.0, Document Version 3

This version of the document includes the following change:

- Updated section *5.13 Use Tags and Tag Sets* for EV 104978.

Release 17.0, Document Version 2

This version of the document includes the following change:

- Updated section *5.11 Set Time Zone* for EV 101376.

Release 17.0, Document Version 1

This version of the document includes the following changes:

- Updated the screen captures with the new theme.
- Added new tags.
- Added device type import/export.

1.8 Changes for Release 16.0

Release 16.0, Document Version 3

This version of the document includes the following changes:

- Updated section *5.13.2 Use of Tags in Templates* for EV 95007.
- Updated section *9.4 Common Problems* for EV 103347.

Release 16.0, Document Version 2

This version of the document includes the following changes:

- Added the tag %BWDIALPLAN-OUTSIDE-ACCESS-CODE-[1-5]-x% and introduced a wildcard symbol for device access URI.
- Changed the screen captures to show the new usability enhancement.
- Updated document for EVs 99029 and 98029.

Release 16.0, Document Version 1

- This version of the document includes the following changes:
- Updated section *8.1.1 Profile Server Configuration* for EV 91883.
- Updated section *5.13.2 Use of Tags in Templates* and added tags for the new feature access codes (FACs) that were introduced for Release 15.0 and Release 16.0.
- Updated section *5.13.2 Use of Tags in Templates* and renamed the tags for the FACs associated with Shared Call Appearance.
- Added clarification to section *5.17.5.2 HTTP Configuration* for devices that do not support the 302 redirect message for EV 66819.
- Introduced the Trivial File Transfer Protocol (TFTP) support.
- Introduced the Network Server device file location lookup.
- Removed the device file type.

1.9 Changes for Release 15.0

Release 15.0, Document Version 8

This version of the document includes the following changes:

- Removed the restriction on the static file not being customizable.
- Updated sections *5.2 Define Access Profiles*, *5.3 Define Configuration Profiles*, and *Appendix A: Generic Device/Identity Access Profile Definitions* for EV 87052.
- Updated section *8.1.1 Profile Server Configuration* for EV 84012.

Release 15.0, Document Version 7

This version of the document includes the following changes:

- Added dynamic tag %BWFQDEVICEID%.
- Updated the remote file name of the device-type Polycom directory file to %BWFQDEVICEID%-directory.xml for EV 69558.
- Added dynamic tag %BWLINE-ENABLED-x%.
- Added section *8.2.1.5 Enable Polycom Phone Services* to indicate how the Polycom phone directories are migrated for EV 66535.

Release 15.0, Document Version 6

This version of the document includes the following changes:

- Made a minor change in section *8.2.1.5 Enable Polycom Phone Services* for EV 66416.
- Updated sections *5.17.4 Network Access Lists*, *5.17.7 Network Server Configuration*, and *8.1.2 Network Server Configuration* for EV 69928. Included information about adding the Xtended Services Platform IP address to the Network Server portal application programming interface (API) access control list.

Release 15.0, Document Version 5

This version of the document includes the following changes:

- Updated the document to add a confirmation password text box for the device type add/modify pages and for the device add/modify pages at the group, service provider/enterprise, and system level.

Release 15.0, Document Version 4

This version of the document includes the following changes:

- Updated the document to correct errors in sections *5.2.3 Standard Options* and *5.2.4 Advanced Options*.
- Indicated the formula for calculating the maximum number of events that the Provisioning Server can process in a schedule period (in section *5.7 Generate File*).
- Renamed tags BWSCA-BARGE-IN-* to BWSCA-BRIDGING-* (in section *5.13.2 Use of Tags in Templates*).
- Specified that the Profile Server fully qualified domain name (FQDN) should be resolved by the DNS using a fixed order (in section *5.17.8.3 Add File Repository*).

Release 15.0, Document Version 3

This version of the document includes the following changes:

- Updated the document to add missing access profile standard and advanced options to sections *5.2.3 Standard Options* and *5.2.4 Advanced Options* for EV 67996.
- Updated the document as the creation of the device-type Polycom directory file has changed to automatically load a default template, change the remote file name to %BWDEVICEID%-directory.xml, and inherit the authentication configuration from the device type.

Release 15.0, Document Version 2

This version of the document includes the following change:

- Clarified the value to be entered for *deviceAccessAppServerClusterName* parameter.

Release 15.0, Document Version 1

The activation references were removed throughout the document.

1.10 Changes for Release 14.0

Release 14.sp6, Document Version 1

The document was created for Release 14.sp6.

2 Glossary

This section describes the terms that are used frequently in this document:

Access URI – This is the Uniform Resource Identifier (URI) used by a device to access its files on Cisco BroadWorks, for example, http://xsp.broadworks.com/dms/polycom500/configuration.cfg.

Dynamic Tag – This is a pre-existing token that starts with "BW" and is delimited by two "%" characters. As a token reserved by Cisco BroadWorks, a dynamic tag is replaced with a dynamic value depending on the context of a template generation. For example, the dynamic tag %BWMACADDRESS% is replaced with the Media Access Control (MAC) address of the device for which the files are generated.

Dynamic Template – This is a file that contains static and/or dynamic tags in the file itself or in the file name. Tags are replaced by their associated values when the template generation occurs before the file is sent to the file repository.

Custom Tag – See *Static Tag*. If a device type is configured to allow custom tags, individual static tags can be customized at the device profile type level, at the group level, and at the device profile level, and additional tags can be added one by one at the device profile type level, the group level, and the device profile level.

File Repository – The data store where generated templates are stocked until devices request them. For example, the file repository was a File Transfer Protocol (FTP) server in legacy deployments. The Profile Server is also a file repository as any WebDAV-compliant server would be.

Legacy Template – This is a file that was created prior to the introduction of the new Device Management feature or that was provisioned using the legacy interfaces. This type of file should slowly disappear as more templates are converted to be used under the new Device Management functionality.

Customization Mechanism – This is the process of replacing something at a more specific level than what was previously defined. For example, a template that is global to all users in the system can be customized for a specific group with a different template.

Polycom Phone Directory – This is a Device Management feature that is used to synchronize a phone directory across Polycom devices and Cisco BroadWorks.

Static Tag – This is a custom token delimited by two "%" characters that is replaced with a fixed value. For example, the user might create a static tag called %MY_STATIC_TAG% associated with a value of MY_VALUE. The tag is resolved to its associated value in a template or in a template file name when a template generation occurs.

Static Template – This is a file that does not contain tags either in the file or in the file name. It is uploaded to the file repository without modification when the template generation process occurs.

Template Generation – This is the process of resolving all tags in a template, replacing the tags with the values, and uploading the resulting file to the file repository. The template generation is also used to refer to the process of uploading static files although the tag resolving phase is skipped in this case.

3 Introduction

In a hosted communications service offering, it is critical to have successful integration of the access device at the customer premises with the hosted services in the operator's network. In the customer network, access devices (phones, analog terminal adapters, soft clients, and integrated access devices) must be configured to point to the service provider, to bind to the correct line and the service profile, and to enable a set of local features that interact properly with the services being delivered in the network. Similarly, hosted applications in the operator's network must be provisioned to recognize authorized access devices and deliver services in a manner that is compatible with the capabilities of each device. Even after devices have been successfully deployed, operators face an ongoing maintenance challenge, as access devices may need to have their firmware upgraded to enable new features or patched to address any issues discovered in the field.

The process of configuring access devices and provisioning Application Servers can involve a number of steps that must be done in concert with the addition of new users to the network. A large part of the budget for deploying a new communications service offering can be spent designing and automating this process. The rollout of new firmware can also involve a number of coordinated steps that must be automated to minimize the opportunity for human error. If this investment in automation is not done up front, then it is lost through high operational expenses and lost revenue caused by a customer activation and support process that does not scale to meet customer expectations.

Cisco BroadWorks provides a number of integrated Device Management features that are designed to address this challenge. By providing a single integrated provisioning API for managing users, services, and devices, Cisco BroadWorks reduces the number of steps involved and the number of systems to integrate into the network. The result is faster time to market, lower operating expenses, and better customer satisfaction. Once Device Management has been set up, configured, and integrated into the flow-through provisioning of the offering, it hides the complexity of installing and deploying new devices, and ultimately accelerates turning up new users in the network.

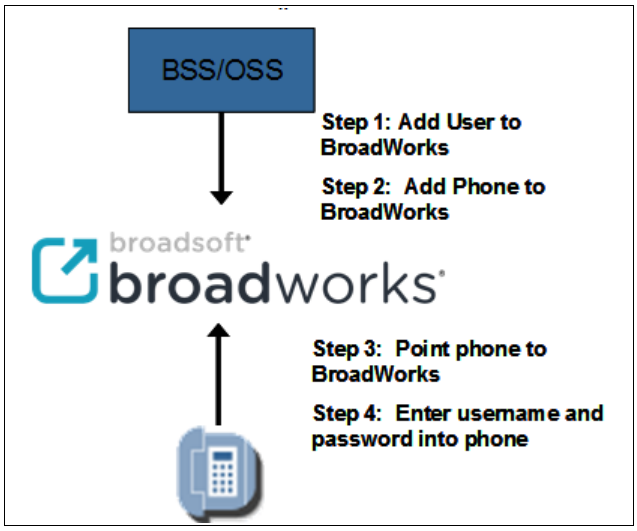


Figure 1 Simplified Deployment Model

Note that in this document, partner configuration guides are referred to as device configuration guides.

4 Understanding Device Management on Cisco BroadWorks

4.1 Overview

Cisco BroadWorks Device Management is a comprehensive solution for simplifying the integration, deployment, and maintenance of access devices in the operator's network. It can be used in both IP Multimedia Subsystem (IMS) and stand-alone next generation network architectures.

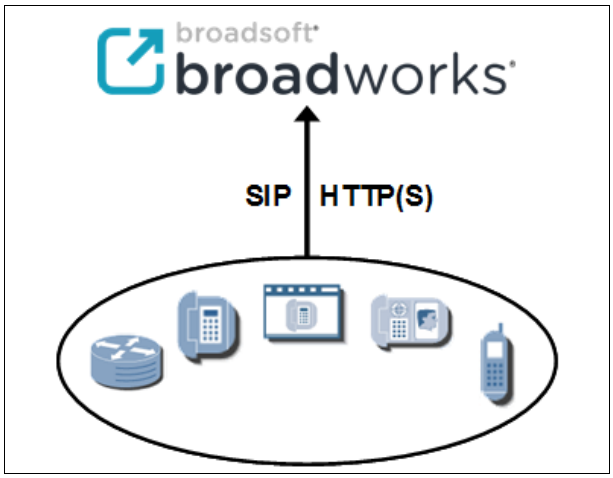


Figure 2 Cisco BroadWorks Device Management Overview

The key areas of functionality provided by Device Management on Cisco BroadWorks are:

- Access profiles
- Configuration profiles
- Service integration
- Resource management
- Inventory management

The following subsections provide an overview of each of these areas of functionality.

4.1.1 Access Profiles

Before an access device can connect to Cisco BroadWorks, a corresponding “access profile” must be defined for that device. An access profile specifies the signaling and media capabilities of the device. This allows Cisco BroadWorks to tailor service delivery to match the specific capabilities of each device in the network. For instance, one device may support dynamic registration, while another may need to have its contact address provisioned statically. Another device may support multiple call appearances, while yet another may require waiting calls to be managed in the network. By defining unique access profiles for each of these device types, Cisco BroadWorks can adjust the way it signals to each device accordingly. The access profile also defines the number of “ports” or unique line addresses that the device supports. As ports are assigned to line addresses, Cisco BroadWorks keeps track of which port is allocated and which port is free. This helps operators manage not only the inventory of devices in the network, but also the number of ports that are in use as well.

Figure 3 Attributes of Access Profiles

The Cisco Interoperability Program allows an access vendor to validate their access device with the Cisco BroadWorks service suite. As vendors complete the Cisco Interoperability Program, they collaborate with Cisco to publish a device configuration guide, which describes the device capabilities and how it integrates with Cisco BroadWorks. Part of the configuration guide describes the exact access profile that should be created on Cisco BroadWorks to properly define the signaling and media capabilities of the device.

4.1.2 Configuration Profiles

To simplify deployment, Cisco BroadWorks allows “configuration profiles” to be defined for each device it is managing. A configuration profile defines all the attributes and settings required for the device to connect to the network and deliver service. Configuration profiles are optional. If Cisco BroadWorks is not responsible for the configuration profile of the device, then this part of Device Management can be disabled.

When enabled, Cisco BroadWorks uses the configuration profile to generate configuration files. Configuration files must be stored on a configuration file repository and made accessible to the devices over the access network. Cisco BroadWorks can be configured to deposit configuration files on an external configuration file repository or Cisco BroadWorks can be configured to use an integrated configuration file repository. When using an external repository, the Cisco BroadWorks Application Server uses either FTP or Hypertext Transfer Protocol (HTTP) to deposit files on the repository. This is shown in *Figure 4*.

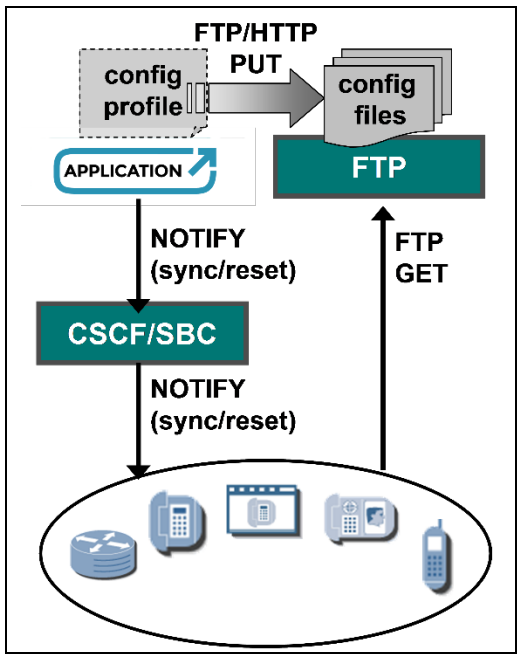


Figure 4 Integration with External Configuration Repository

When Cisco BroadWorks is configured to use the integrated configuration repository, the Profile Server (PS) and the Xtended Services Platform (Xsp) must be deployed. The Cisco BroadWorks Application Server uses HTTP to deposit all generated configuration files on to the Profile Server. The Profile Server provides geographically redundant, highly available, scalable storage for all generated configuration files. Access devices request files through the Xtended Services Platform, either through HTTP or HTTPS GET requests or TFTP RRQ (Read Request). This is shown in *Figure 5*.

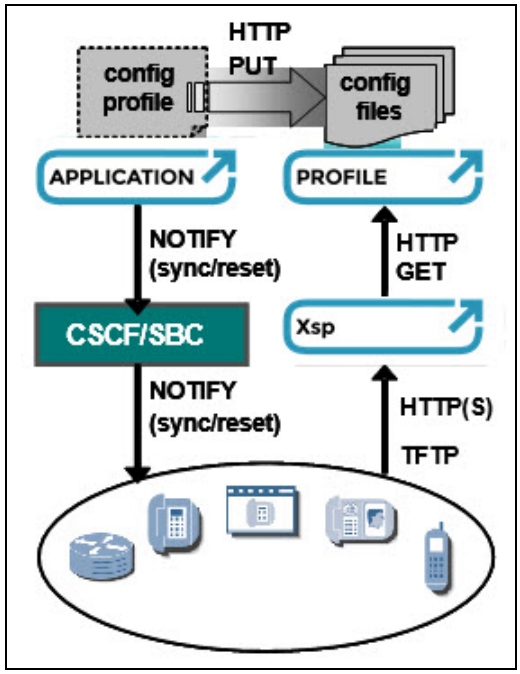


Figure 5 Integrated Cisco BroadWorks Configuration Repository

Note that the integrated configuration repository through the Profile Server and the secure access through the Xtended Services Platform are based on HTTP and HTTPS transports. Digest authentication is used over HTTP to securely identify the requesting device without transmitting credentials in the clear. Secure Sockets Layer (SSL) is (optionally) used over HTTPS for device requests when confidentiality of the transmitted configuration files is required. This should be employed whenever there is a risk of man-in-the-middle attacks on the access network. Trivial File Transfer Protocol (TFTP) is supported but it does not provide any security.

IMPORTANT NOTE: Because of the nature or the TFTP, which does not permit any login information to be sent, the only authentication that can be applied to a file is the MAC-based authentication mode. The MAC address should be sent in an "HTTP Request-URI".

The FTP protocol is not supported. If a device only supports FTP for configuration file download, then an FTP front end would need to be integrated. This would have to be performed by a system integrator. This is not part of the turnkey solution.

4.1.3 Service Integration

One of the most powerful features of Device Management on Cisco BroadWorks is the ability to easily integrate Cisco BroadWorks user services with features on the access device. This is most applicable in a hosted PBX application offering, where advanced business services, such as Shared Call Appearances and Busy Lamp Field (BLF), require attribute values to be set in both Cisco BroadWorks and the access device before the service operates properly. For instance, the Busy Lamp Field service on Cisco BroadWorks requires the use of the Session Initiation Protocol (SIP) dialog event package. To access the service, the device must SUBSCRIBE to a specific SIP URI representing the specific user's Busy Lamp Field state. This SIP URI must be provisioned on both the user's service profile as well as any device they wish to use to access the service. When using Device Management, the provisioning system only needs to set this value once on the user's service profile. This triggers Cisco BroadWorks to update any corresponding device configuration files and deposit them on the configuration file repository. If the changes are to take effect immediately, the operator can initiate a remote reboot of the affected access devices.

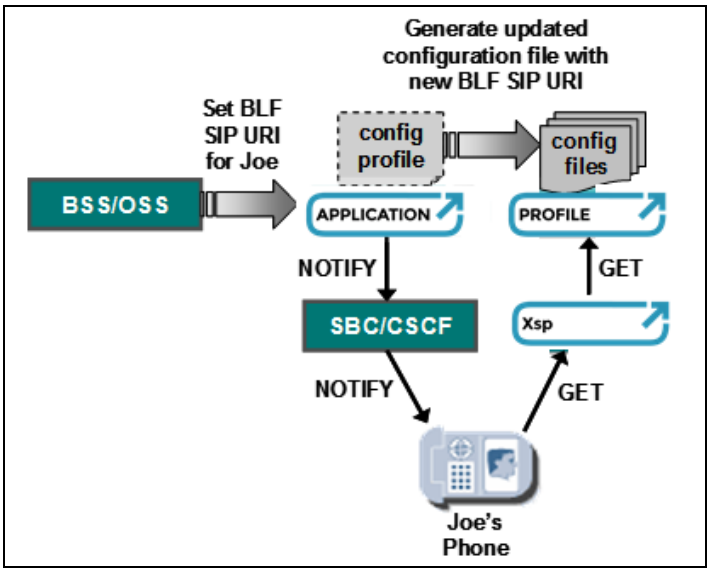


Figure 6 Busy Lamp Field Service Integration

Cisco BroadWorks supports a long list of service attributes that can be integrated with device profile settings, including, but not limited to the following:

- Shared Call Appearances
- Busy Lamp Field
- Any service using a feature access code
- Voice Portal and Voice Mail
- Language and Locale

4.1.4 Resource Management

In addition to configuration files, an access device can require one or more resource files before it can deliver service. Resource files can be bitmaps, audio files, and contact directory files. However, the most common type of resource that must be managed is the firmware files that represent the current version of software embedded in the access device. Device Management provides methods and procedures that can be used to manage which device uses which version of a resource file. This allows operators to easily control which version of firmware is deployed in the network, and to easily roll out new versions of firmware in a controlled and predictable manner. Specific group resources can be deployed to subsets of users in the network by customizing resources at the corresponding group level. This allows operators to present specific backgrounds or logos to the liquid crystal display (LCD) on Internet Protocol (IP) phones or provide custom ring tones tailored to specific customer requests.

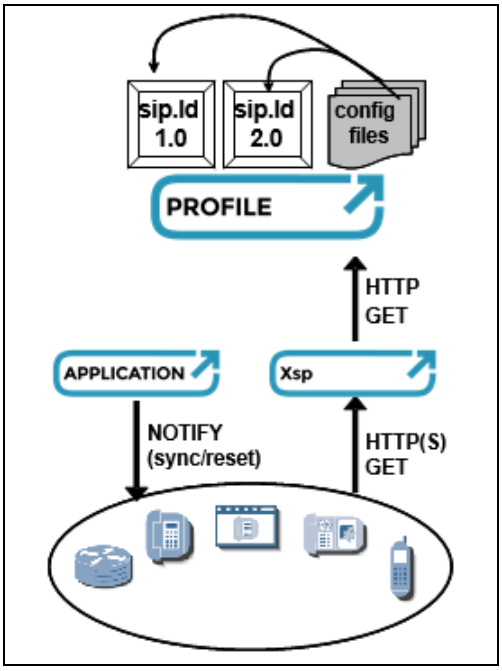


Figure 7 Managing Firmware Resources

4.1.5 Inventory Management

Cisco BroadWorks maintains a list of all devices that are provisioned in the network. This is integrated into the same database that manages all users, lines, and services in the network. This means Cisco BroadWorks can easily track relationships between devices, the ports that are free, the ports that are in use, and the corresponding users who are associated with each port on the device. This type of information is invaluable when tracking the state of devices in the network and troubleshooting problems on the access network. Cisco BroadWorks also provides basic inventory management reporting tools that can be used either by themselves or integrated with a broader inventory management system.

4.2 Key Concepts

To use Device Management, it is important to first understand a few key concepts and how they apply to the overall Cisco BroadWorks system. This section describes the Cisco BroadWorks data model as it relates to Device Management, and then describes the key concepts that must be understood to be able to deploy and use Cisco BroadWorks Device Management in a hosted service offering.

4.2.1 Data Model

Cisco BroadWorks uses three key concepts for delivering services and managing devices:

- Device Profile Type
- Device Profile
- User

All of these concepts are modeled directly in the Cisco BroadWorks Application Server database.

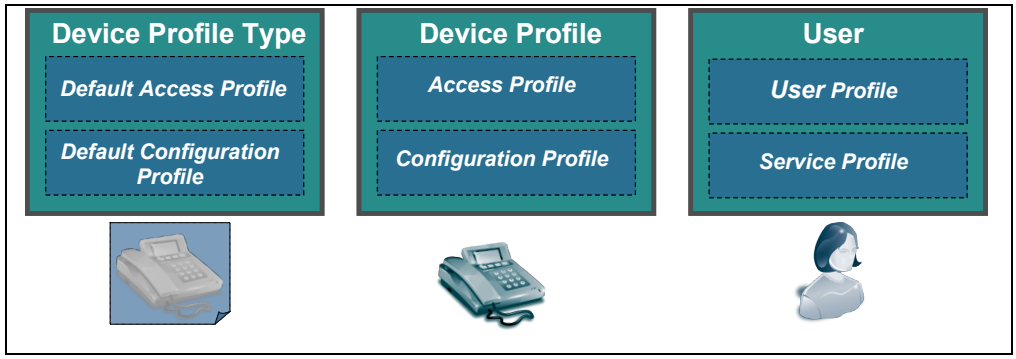


Figure 8 Key Data Model Concepts

Device Profile Type

The device profile type is the foundation for Device Management. Conceptually, the device profile type is a “device template”. It allows the operator to define default access and configuration settings for all devices of a given make, model, and application in the network, and then easily reuse those settings for a given group of devices. Whenever a new type of device must be introduced into the network, the operator must define a new device profile type to model the characteristics of that device.

Device Profile

When a new device is added to the network, a new device profile must be created on Cisco BroadWorks to “manage” that device. The device profile models the actual instance of a device in the network. Every device profile must be created from a given device profile type. This gives the device profile a predefined set of default settings that are consistent with other devices of the same type in the network.

User

A device profile maintains a list of “ports”. The maximum number of ports is set at the device profile type. Each “port” on the device profile can be mapped to one “line/port address” on a user in Cisco BroadWorks. This association between ports on devices and line/ports on users creates an important relationship between devices and users. For some device types, a device may only have one user associated with it at a time. For other device types, many users may need to be associated with it.

Figure 9 shows one user per phone device relationship.

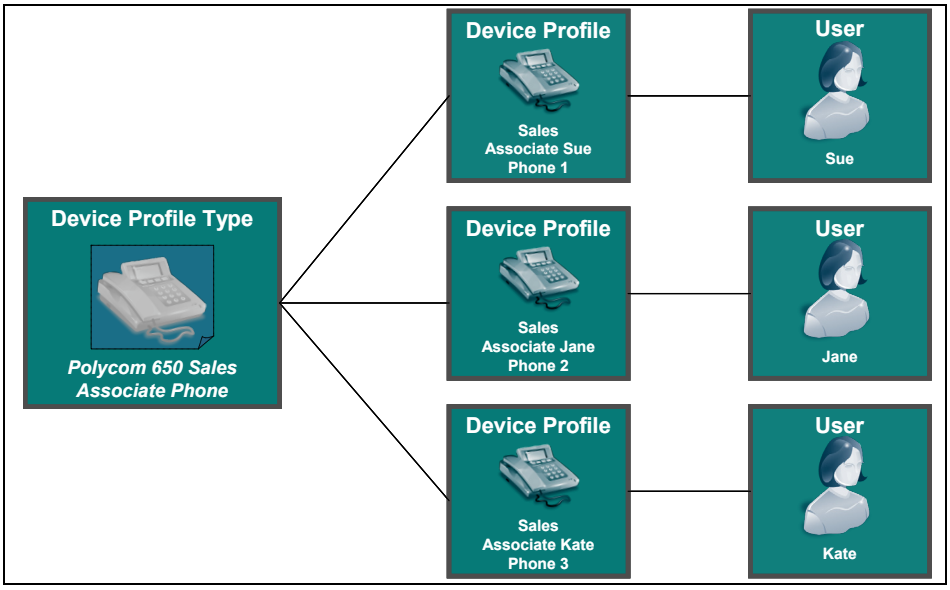


Figure 9 One User per Phone Device Relationship

Figure 10 shows multiple users per phone device relationship.

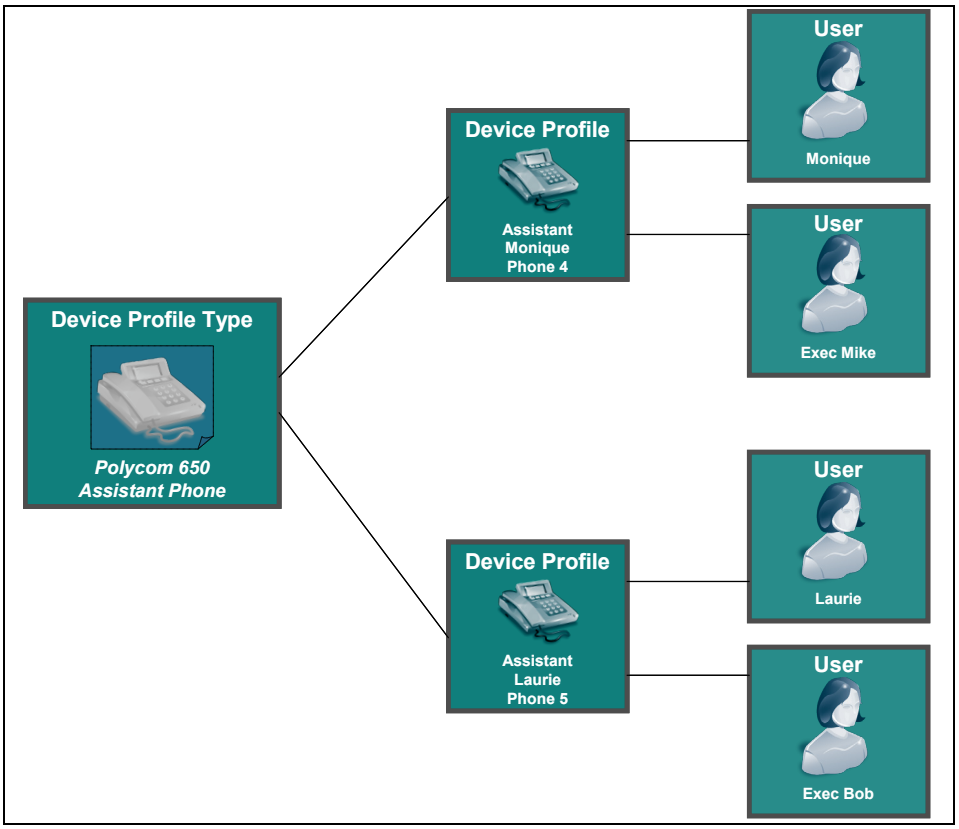


Figure 10 Multiple Users per Phone Device Relationship

4.2.2 Device Profiles and Device Profile Types

Device profiles represent the devices themselves. Device profile types are the templates for device profiles. In a typical deployment, there are usually only a handful of device profile types and hundreds of thousands of device profiles (as many as one per user in the system).

Creating device profile types is a crucial step in the initial planning and deployment of a Cisco BroadWorks-based service offering. Device profile types should be defined in conjunction with the services being offered to the users. For every "type" of user, there should be a corresponding "type" of device that is defined to integrate with their specific service set. Defining the device profile types requires a detailed understanding of the overall solution being offered, how many lines are allowed on each phone, what services are enabled, and so on.

Properly defining the device profile type also requires intimate technical knowledge of the device itself. Cisco BroadWorks Device Management divides device profile type definition into two primary steps:

- Defining the default access profile attributes
- Defining the default configuration profile attributes

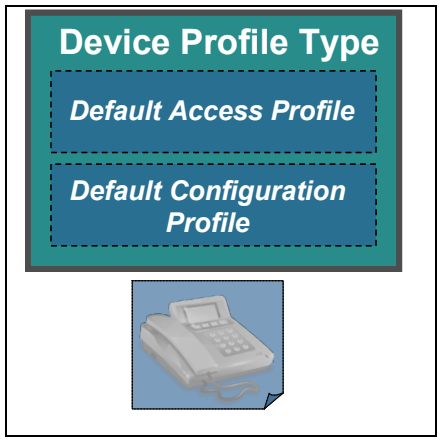


Figure 11 Device Profile Type Model

The default access profile consists of attributes relating to the signaling and media integration with Cisco BroadWorks. For instance, how the device registers and addresses with the network, whether it supports early media, whether it supports authentication, and so on are all attributes of the access profile of a device profile type. These attributes tell Cisco BroadWorks how to interact with device profiles of this type. Another important part of the access profile is the maximum number of ports available on the device. This attribute allows Cisco BroadWorks to control the number of users who can be associated with a given device. This is important for inventory management and troubleshooting procedures.

The default configuration profile consists of attributes relating to how the device is configured, the protocol it uses to download its configuration, the structure and organization of the device configuration files, and what type of authentication it uses to access them.

When a new device profile is created from a device profile type, it inherits a representation of the access and configuration profiles defined at the type level.

To facilitate the definition of device profile types, the Cisco Interoperability Program is continuously publishing new device configuration guides for new devices that passed interoperability testing. These device configuration guides help operators define the device profile types on Cisco BroadWorks.

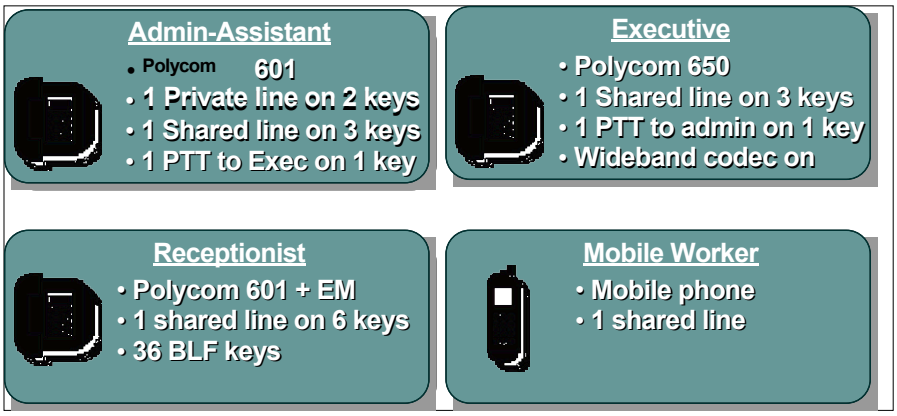


Figure 12 Example Device Profile Types

Cisco has defined "Customer Premises Equipment (CPE) Kits" for certain devices. These kits include predefined device profile types that match user types defined in the Cisco Quick Start Program.

4.2.3 Device Management Files

If Cisco BroadWorks is being used to manage the configuration and resources files of a device, then it is important to first understand exactly how a specific device expects these file to be organized on the server as well as the order in which they are downloaded. This dictates how Device Management files should be created and associated with the device profile type.

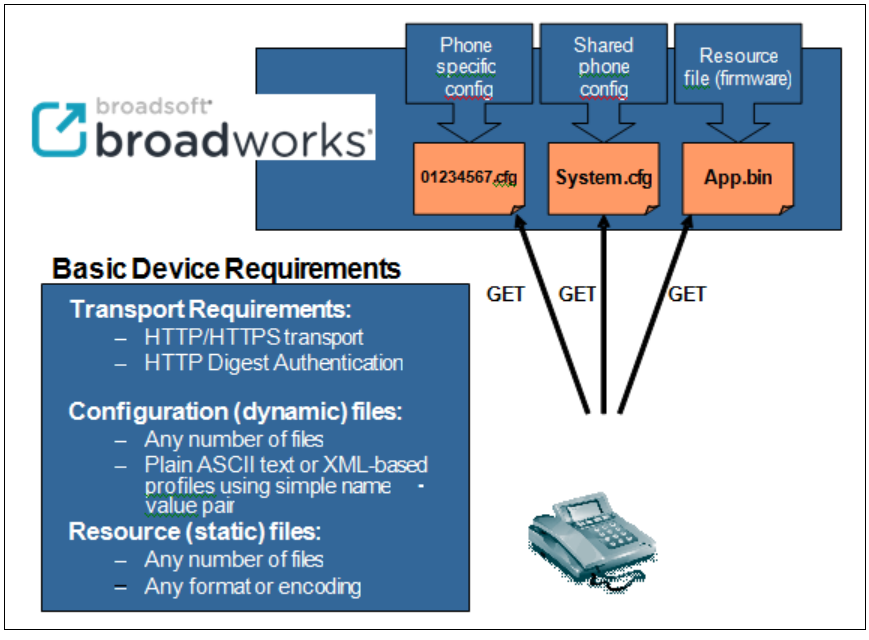


Figure 13 Device Configuration Interface Requirements

Every device is different; however, the typical pattern for most devices is as follows:

- One or more shared phone configuration files
- One or more phone-specific configuration files
- One or more shared resources files

To be compatible with the built-in configuration repository on Cisco BroadWorks, a device must support HTTP, HTTPS, or TFTP as a file transport. For protocols HTTP and HTTPS, the device must also be able to respond to an HTTP digest challenge for authenticating requests for configuration files and other resources. Cisco BroadWorks uses a simple text preprocessing engine to support service integration. This means the device configuration files must be formatted in some readable ASCII-encoded document format, such as text or eXtensible Markup Language (XML). Cisco BroadWorks puts no limit on the number of configuration or resource files that the device downloads.

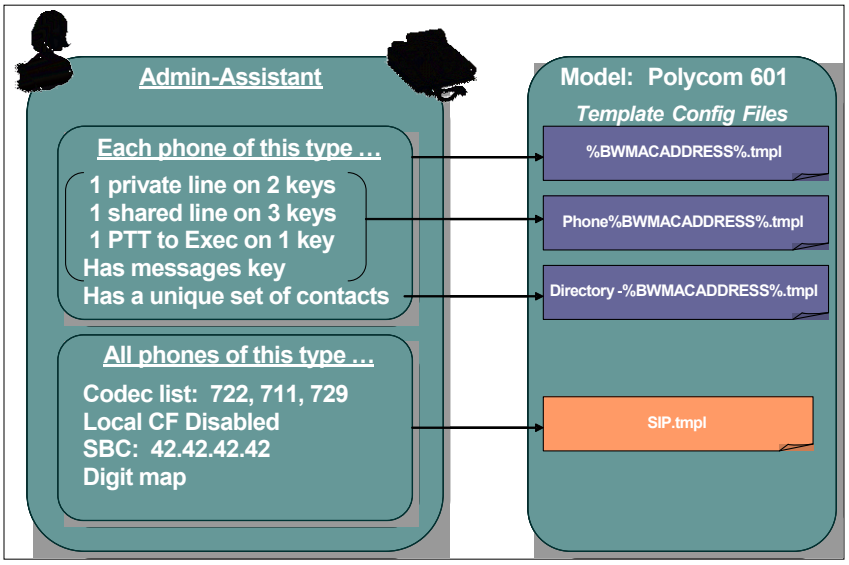


Figure 14 Specifying Template Configuration Files

When defining the configuration profile of a device profile type, the operator must specify one or more template configuration files. The template configuration files contain the default configuration that all devices of that type use. Template configuration files are formatted in the native configuration format of the device, that is, template configuration files are specific to a make and model of a device.

There are two types of template configuration files.

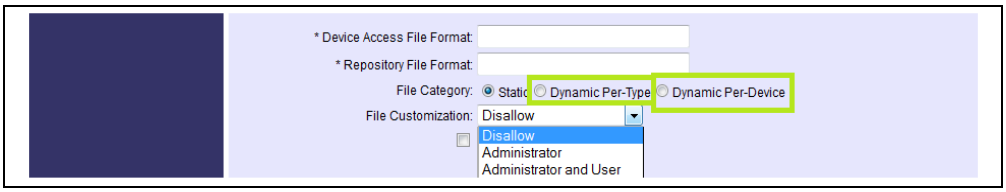


Figure 15 Two Types of Template Configuration Files

- Device Profile Type Template Files – These configuration template files have a one-to-one relationship with the device profile type. There is only one file generated for each device profile level template file. These template files do not generate configuration files every time a device profile is added. On the web portal, these template files are called “Dynamic Per-Type”.
- Device Profile Template Files – These configuration template files have a one-to-one relationship with the device profile. There is one file generated for each device profile in the system. Every time a new device profile is created from a device profile type, a new device profile file is generated for each device profile template. Therefore, for a given device profile template file there may be hundreds of thousands of configuration files generated. On the web portal, these template files are called “Dynamic Per-Device”.

Template configuration files are uploaded to the Cisco BroadWorks Application Server. Each Application Server keeps its own templates, that is, the templates for the users provisioned on a given Application Server are stored locally on disk. The Application Server preprocesses the template configuration files to generate the actual configuration files that the access device downloads. These files are then deposited on the corresponding configuration repository. By default, device profile types use the Profile Server as the configuration repository.

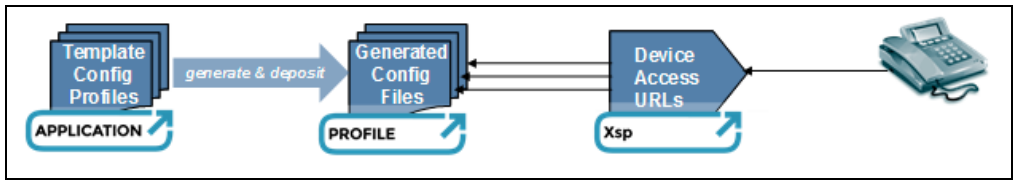


Figure 16 Template and Generated Configuration Files

All template configuration files are defined by the system administrator at the device profile type level. These represent the “default configuration template files” for the device profile type. For a number of reasons, it can be useful for these templates to be “customized” for a set of devices in the network. For instance, by default a device profile type may define a device to have two private lines and one shared line, but for a few enterprise customers, this configuration may need to be modified slightly to include one private line and two shared lines; otherwise, the configuration is the same. Instead of creating a completely new device profile type to meet the requirements of one or two customers, Cisco BroadWorks allows template configuration files to be customized at both the group level and the device profile level. The system administrator has control over which template files can be customized. Files can be customized by the administrator or by the administrator and the user. In the latter case, the user credentials are sufficient to perform a file customization from the Open Client Interface (OCI).

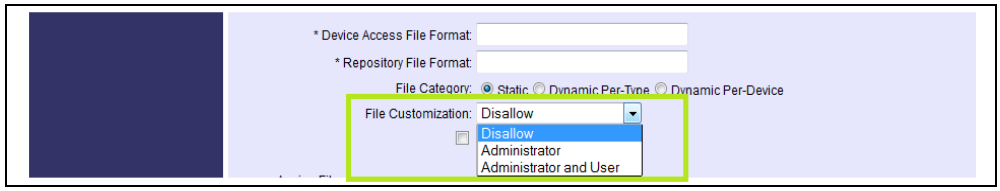


Figure 17 File Customization

Cisco BroadWorks also supports the concept of “static files”. Static files are used for managing resource files such as firmware, bitmap logos, and ringback wave files associated with the device profile type.

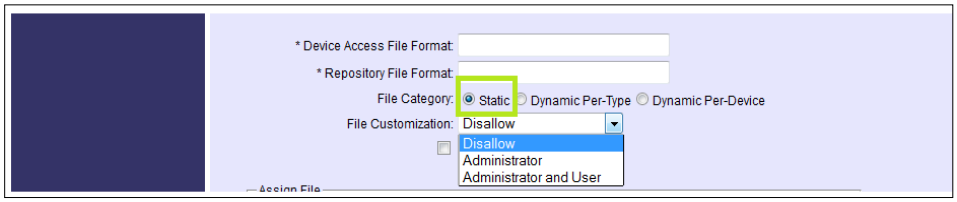


Figure 18 Static Resource Files

4.2.4 File Caching

File caching allows the Xtended Services Platform to keep a copy of the file locally, thereby removing the need to download the file from the Profile Server.

This functionality can be enabled via the CommPilot web portal at the device type level (see the following figure) or through the command line interface (CLI) under the *CLI/System/DeviceType/SIP/Files>* level.

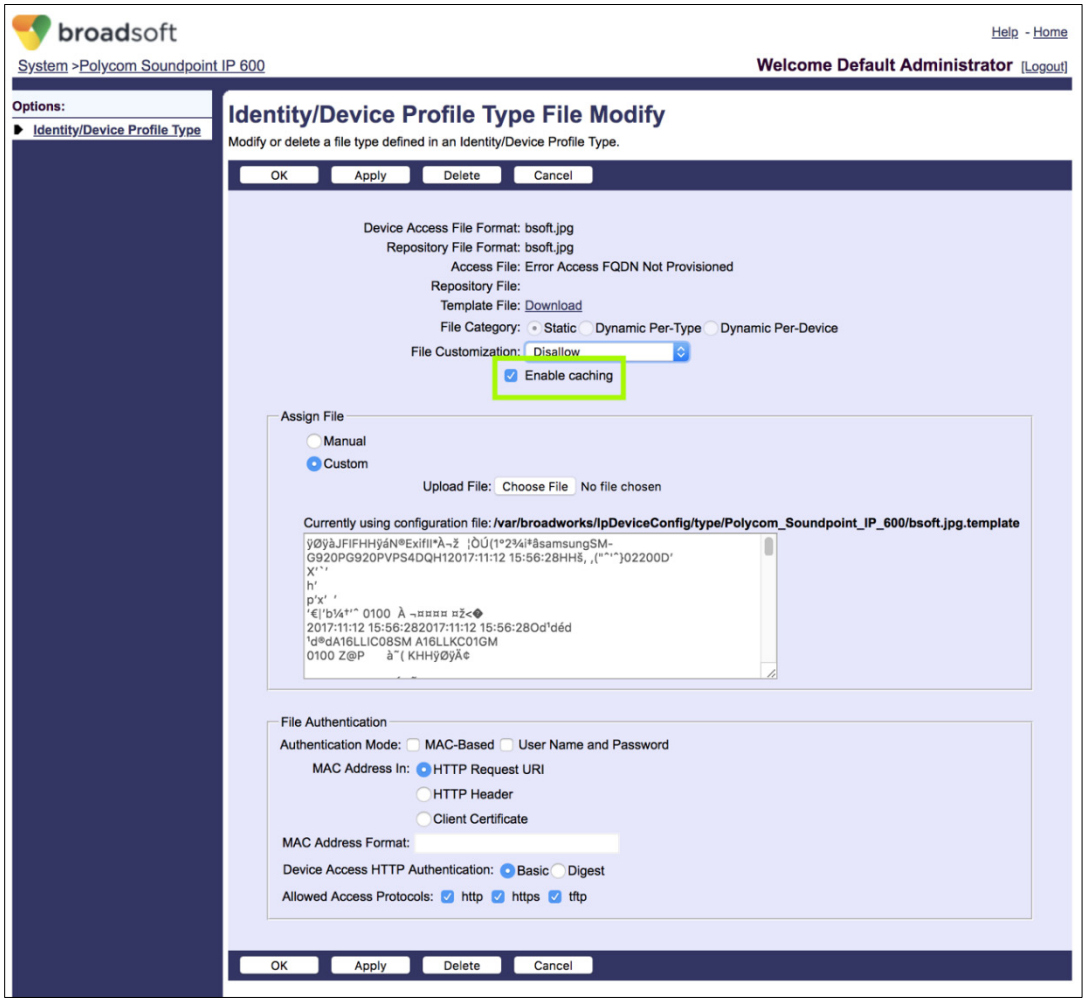


Figure 19 Enabling Caching on Static or Dynamic Per-Type File

File caching should be enabled for static and dynamic per-type files. This is enabled on a per-file basis on the Application Server.

Note that the exported DTAFs retain the caching property set at the time of the export.

The Xtended Services Platform allows configuration of some parameters of the caching mechanism, such as *timeToLiveInHours*, *entryCapacity*, and *byteCapacityInMBytes*. This can be done at the CLI level *XSP_CLI/Applications/BroadworksDms/Cache>*. The cache is always active; its use is solely determined by the file-level cache settings detailed above. The intent of the cache is to optimize delivery of the selected file to the device by serving the file directly from the Xtended Services Platform disk when possible. The added use of disk resources on the Xtended Services Platform reduces network occupancy to the Profile Server, reduces the HTTP load on the Profile Server (CPU, Memory).

Ideally, the available disk on the Xtended Service Platform would be large enough to hold all cacheable files, especially large firmware files. The *entryCapacity* and *byteCapacityInMBytes* serve as upper bounds meant to protect against excessive disk usage. These should set according to the reserved/planned disk capacity set aside for caching. Focus should be placed on the *byteCapacityInMBytes* in that regard. The *entryCapacity* protects against holding excessive numbers of small files.

The *timeToLiveInHours* settings is mainly directed at cache cleanup and eliminating files that are seldom requested. This is generally not required. Note that cache content is actively verified against the Profile Server and that no out-of-date file is served from cache. Cache content therefore does not require particular cleanup.

4.2.5 Tags and Tag Sets

Service integration on Cisco BroadWorks is based on the concept of “Tags”. Tags are variables that can be embedded in the configuration template files. When Cisco BroadWorks generates a configuration file from a configuration template, the tags are replaced with actual values. Tags are delimited with a beginning and ending % sign.

Tags come in two varieties:

- Dynamic Built-in Tags – These tags are predefined by Cisco. The value of each built-in tag is dynamically evaluated based on the context of the device profile. Depending on the services assigned to users of the device and the values of various service attributes, a built-in tag for one device evaluates differently than a built-in tag for another device. Built-in tags provide rich service integration with Cisco BroadWorks. There are over 100 built-in tags defined in the system. All built-in tags are prefixed with “BW”.
- Static Tags – These tags are defined by the operator when building the device profile type. The value of each static tag is assigned by the operator.

To simplify the management of static tags, the operator can define “Tag Sets” at the system level. Tag sets allow the operator to define a list of static tags that can be re-used across device profile types.

There is one “System Default Tag Set” that is initially empty. The operator can populate the System Default Tag Set with tags that are common to all device types such as domain names, domain name system (DNS) servers, session border controller addresses, and so on.

The operator can define any number of additional tag sets. For every device profile type, the administrator can indicate if the “System Default Tag Set” applies and select one additional tag set to be applied to the device profile type.

Tag sets created at the system level can also be assigned at the service provider level, the group level, and the device profile level.

Individual static tags can be customized at the device profile type level, at the service provider level, at the group level, and at the device profile level. Additional tags can be added one by one at the device profile type level, the service provider level, the group level, and the device profile level.

4.2.6 Phone Services

Device Management provides the framework for a new class of user services called Phone Services. Phone Services are end-user services designed to expose specific capabilities of a specific make and model of a phone to the end user. The intent is to provide a tightly integrated end-user experience on Cisco BroadWorks for a specific phone.

When defining a device profile type, it is possible to specify whether the device profile type supports one or more Phone Services. When a Phone Service is enabled on the device profile type, it manages the creation of any template configuration files required for the Phone Service. This eliminates some of the complexity regarding defining the template configuration files for some advanced service integration.

5 Configure Device Management on Cisco BroadWorks

5.1 Create Device Profile Types

Only system administrator level access can add new device profile types. This can be done through the CLI or through the Open Client Interface-Provisioning (OCI-P). However, since this is a task that requires a number of steps with a lot of detail, and it is only done infrequently, most often the system administrator logs in and creates device profile type through the web portal. Access to device profile type management functions is done by navigating to the *Resources* page at the system level, and then clicking on **Identity/Device Profile Types**. This is shown in *Figure 20*.

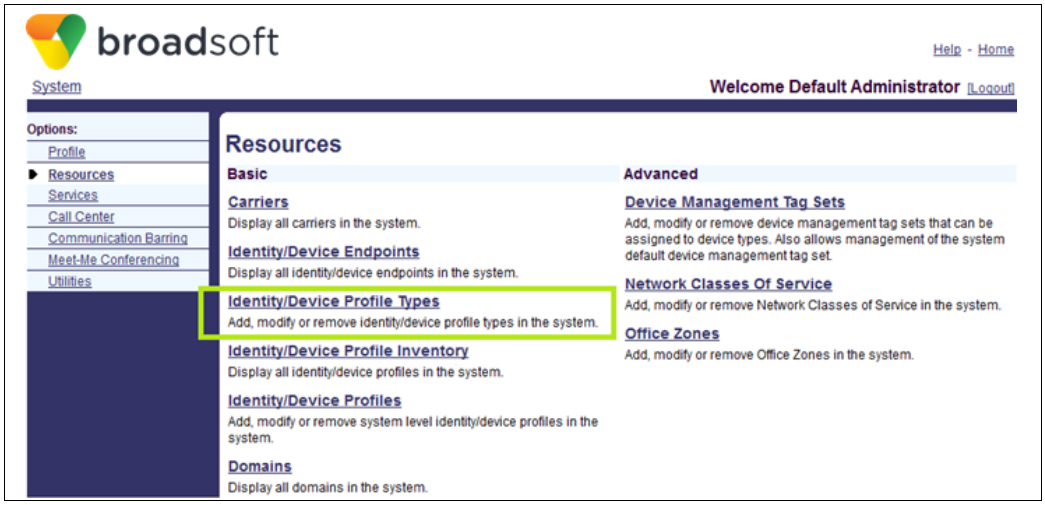


Figure 20 Identity/Device Profile Types Navigation

The administrator has the ability to add, modify, delete, and make device profile types obsolete. Device profiles types can only be deleted when all references to the device profile type are removed. All references to a device profile type are removed when no users are associated with any device profile of the device profile type.

The administrator can also make device profile type obsolete. Obsolete device profile types are retained within Cisco BroadWorks to allow users associated with a device profile of the obsolete device profile type to continue to use the device profile. However, the device profile type is removed from the list of available device profile types and no new device profiles can be created using the obsolete device profile type.

NOTE: In AS mode, the Reseller level also allows the management of device types.

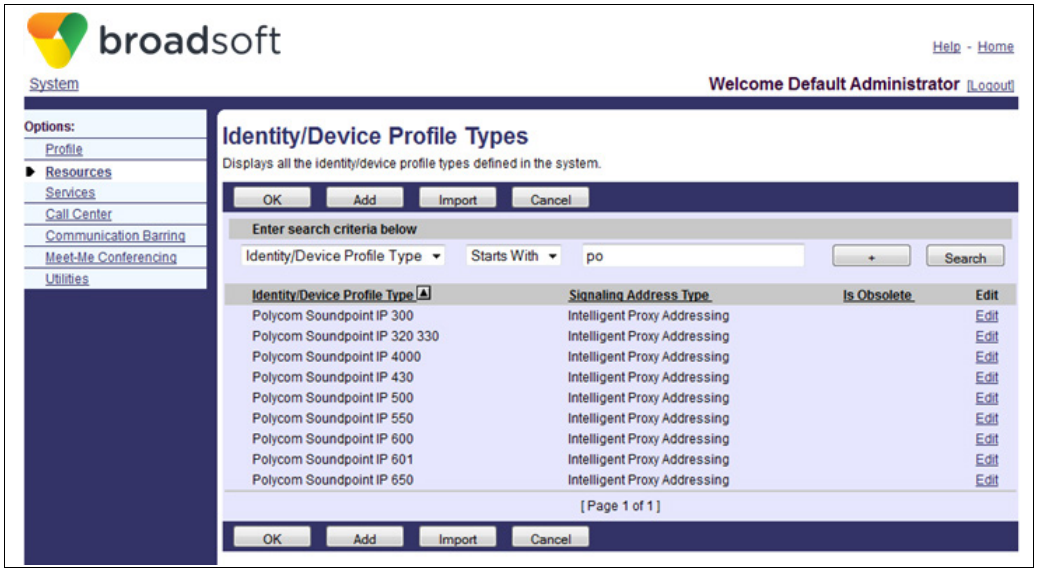


Figure 21 Managing Device Profile Types

System administrators can only manage device profile type for SIP devices. All Media Gateway Control Protocol (MGCP) device profile types are pre-populated in the database and require Cisco support to add or modify the MGCP identity/device profile type.

Creating the device profile type involves two primary steps:

- Defining the access profile – For managing aspects related to the signaling and media interoperability with Cisco BroadWorks.
- Defining the configuration profile – For managing aspects related to the configuration of the device.

Each of these steps is defined in the sections that follow.

5.2 Define Access Profiles

5.2.1 Overview

When adding a new device profile type to the system, the first step is to define the access profile. Every device profile type must have a well-defined access profile before it can be used. The most important part of the access profile is the “Signaling Address Type”; this is the only required option.

The rest of the access profile is encapsulated in the “Standard Options” and the “Advanced Options” blocks of the new Device Profile Type dialog, as shown in *Figure 22*.

This section describes all the attributes of the access profile. To help facilitate setting the access profile attributes, the Cisco Interoperability Program is continuously publishing new device configuration guides for new devices that passed interoperability testing. To understand which attributes should be used for your device, see the corresponding device configuration guide.

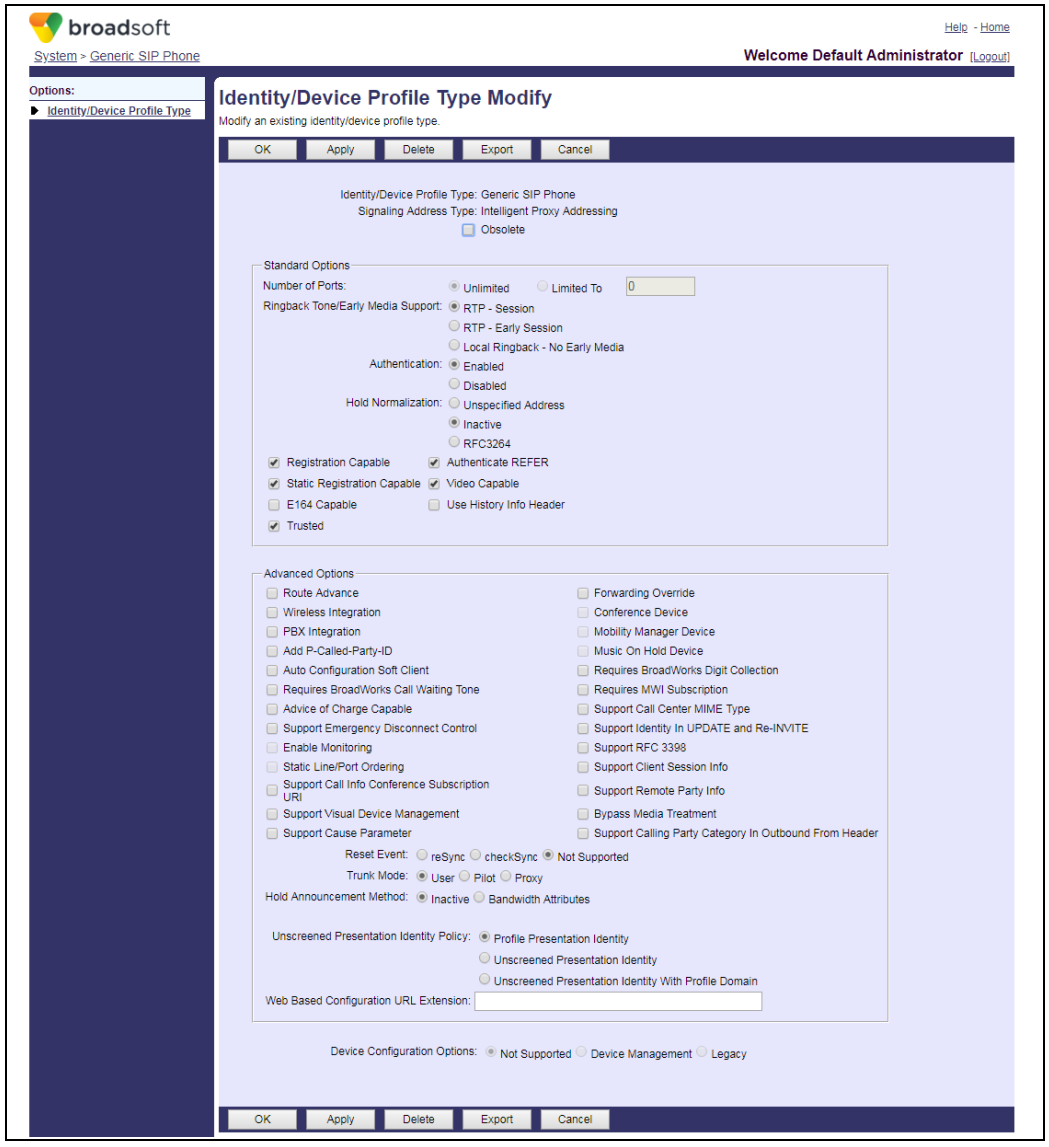


Figure 22 Access Profile

5.2.2 Signaling Address Types

Device profile types are associated with a signaling address type. Signaling address types define the fundamental behavior of an identity/device profile type. There are two fundamental parameters for a device profile type:

- Intelligent/Non-intelligent
  - Intelligent profile types support devices that perform their own call control, including services such as Call Transfer and Three-Way Calling. They also allow devices to place or receive multiple calls. Devices associated with an intelligent profile type create a SIP dialog per call and can use the REFER method for advanced Call Control services. They can also perform mixing locally within the device or use network-based mixing on Cisco BroadWorks to provide Conferencing services.

- Non-intelligent profile types support devices that use the Cisco BroadWorks INFO message for flash hook processing and call control. Services such as Call Transfer and Three-Way Calling are provided by Cisco BroadWorks and are controlled by the device via flash hooks sent via the INFO method to Cisco BroadWorks. The mixing of multiple calls for the user is handled through a single connection (SIP dialog) to the device. Devices associated with an unintelligent profile type may only place or receive two simultaneous calls and must use the Cisco BroadWorks INFO message for flash hook processing to support the second call, as only one SIP dialog is created for all simultaneous calls. In addition, these devices may not use the REFER method.

■ Device Addressing/Proxy Address (Device Domain/Proxy Domain)

- Device addressing profile types support devices that use the device's addressing space rather than the addressing space of the Application Server. Devices associated with a device addressing profile type use the device's IP address or hostname to populate the host portion of the *From* header for INVITE requests and the *To* header for REGISTER requests. These devices typically do not register as the device address identifies both the endpoint and the location of the endpoint. Device addressing profile types are also referred to as device domain profile types.
- Proxy addressing profile types support devices that use the addressing space of the Application Server. Devices associated with a proxy addressing profile type use a logical address of record (AoR) consisting of both a user portion and a host portion. The user portion corresponds to the line/port (identity in IMS mode) and the host portion corresponds to the line/port (identity) domain in the user addressing on Cisco BroadWorks. The device must populate the *From* header user and host portions for INVITE requests and the *To* header user and host portions for REGISTER requests with the line/port (identity) and domain configured in Cisco BroadWorks. Proxy addressing profile types are also referred to as proxy domain profile types.

Signaling address types consist of four base profiles. These base profiles are defined within Cisco BroadWorks and cannot be changed by the administrator. The four signaling address types are:

■ Non-intelligent Device Addressing

■ Non-intelligent Proxy Addressing

■ Intelligent Device Addressing

■ Intelligent Proxy Addressing

These signaling types provide the base behavior required to support all combinations of profile types within Device Management.

5.2.3 Standard Options

Several policies are available on the device profile type. The following is a list of standard profile policies that can be used when creating/modifying a device profile type.

■ Number of Ports

Number of Ports defines the number of users who can be associated with a profile instance of a device profile type.

- Unlimited – This option may be used when the device does not impose a hard limit on the number of ports.

- Limited To – Specifying the number of ports is useful when creating a device profile type for a fixed-port device such as an analog gateway.

NOTE: The Number of Ports cannot be modified on an existing device profile type.

- Ringback Tone/Early Media Support

  Cisco BroadWorks uses this setting to determine Session Description Protocol (SDP) handling for initial INVITE messages sent to the device in scenarios such as Simultaneous Ringing and Shared Call Appearance to prevent multiple devices from streaming early media to the remote end. Depending on the system-level parameter *broadworksHoldingSDPMethod,* configured via the Application Server CLI under *Interface/SIP* level, Cisco BroadWorks sends, in these scenarios, either "HOLD" or "modified address SDP" to non-primary devices that send early media.

  An early media session is established if an offer/answer exchange occurs before a SIP dialog is confirmed (with a 200 response). Early media is typically carried out within a regular session, but it can also be carried out in an early session (*RFC 3959/3960*). This explains the different values that can be assigned to the *ringbackToneEarlyMediaSupport* device option. Following is a brief description of the different assignable values:

  - Local Ringback – No Early Media – The device cannot provide early media.
  - RTP – Session – The device uses a regular session to establish early media. The originating endpoint generates the offer in the INVITE request and the terminating endpoint generates an answer in the 18x response. Alternatively, the originating endpoint does not generate the offer in the INVITE request. It is the terminating endpoint that generates the offer in the 18x response while the originating endpoint generates the answer in a PRACK request.
  - RTP – Early Session – The device uses an early session to establish early media. Regardless of the regular session offer/answer exchange, the terminating endpoint generates an early session offer (that is, SDP message body with disposition-type="early-session") in the 18x response. The originating endpoint generates the early-session answer in the PRACK. Early media is carried out in this early session, which is brought down upon answer or release. The regular session is negotiated in parallel and established once the terminating endpoint provides the 200 response.

- Authentication

  Authentication defines whether requests for a device are authenticated. The device option can be set to either "enabled", "disabled", or "enabled with web portal credentials".

  - Enabled – Requests from a device associated with this device profile type are challenged if the Authentication service is assigned to the user. The authentication is performed using the Authentication service credentials.
  - Disabled – Requests from a device associated with this device profile type are not challenged even if the Authentication service is assigned to the user. However, this setting does not prevent the Application Server from answering an authentication challenge from the device if the credentials are available within the assigned Authentication service.

- Enabled With Web Portal Credentials – Requests from a device associated with this device profile type are challenged if the Authentication service is assigned to the user. The authentication is performed using the subscriber's OCI login credentials. This setting applies specifically to Auto Configuration Soft Clients.

- Registration Capable

  Registration Capable defines whether a profile instance of a device profile type is allowed to register with Cisco BroadWorks. If this policy is not enabled on an identity/device profile, REGISTERs received from a device associated with this profile type are rejected with a *403 Forbidden*.

- Static Registration Capable

  Static Registration Capable defines whether a profile instance of an identity/device profile type allows a static location to be entered for each user associated with this profile type. If this policy is enabled on a device profile type, an optional contact entry is available on each user associated with a profile of this device profile type that can be populated with the location of the user.

- E.164 Capable

  E.164 Capable defines whether E.164 addressing is used in the SIP signaling for a profile instance of an identity/device profile type. If this policy is not enabled on a device profile, the user portion of the *From* header sent out to the device contains the normalized phone number of the calling party (without the + and country code) including the national prefix if configured. If the policy is enabled, the user portion of the *From* header contains the E.164 number of the calling party. This policy applies only to calls from outside the group.

- Trusted

  Trusted defines whether the calling party information is sent in the SIP signaling over the access interface for calls with restricted calling line identity. If this policy is not enabled, the calling party information is restricted and sent as anonymous@anonymous.invalid in the *From* header for calls with calling line identity restriction. If this policy is enabled, the calling party information is included in the *From* header unless the *encryptFromHeader* parameter in the *Interface/SIP* level of the CLI is enabled. If *encryptFromHeader* is enabled, the *From* header is populated with anonymous@anonymous.invalid for the calling party identity. If this policy is enabled, the calling party identity is also sent in the appropriate privacy header based on the *privacyVersion* parameter in the *Interface/SIP* level of the CLI. Based on the *privacyVersion* parameter, the calling party identity may be included in the *P-Asserted-Identity* or *Remote-Party-Identity* header with the appropriate restriction indicators.

- Authenticate REFER

  Authenticate REFER identifies whether SIP REFER requests from the device are authenticated. If this policy is disabled, REFER requests are not authenticated. If this policy is enabled and the Authentication service is assigned, all REFER requests are authenticated.

- RFC 3264 Hold

RFC 3264 Hold defines whether the 3264 Hold mechanism is used in the SIP signaling. If the policy is enabled, hold requests sent by Cisco BroadWorks include the
a attribute, a=inactive. Hold requests received by Cisco BroadWorks set the *a* attribute to "a=inactive" in the *200 OK* response and leave the connection address (c line) unchanged. If this policy is not enabled, Cisco BroadWorks uses a hold connection address (c line) format of 0.0.0.0 for hold.

- Video Capable

  Video Capable defines whether a profile instance of a device profile type can be configured as a Video Add-on device. If this policy is enabled, profiles of this profile type can be configured as Video Add-on devices. These profiles show up on the drop-down list on the web for Video Add-on devices and are accepted via the Open Client Interface (OCI). This policy also must be enabled for Cisco BroadWorks to send video offer in the SDP for initial INVITEs to the device. If this policy is disabled, Cisco BroadWorks does not send video offer in initial INVITEs to the device.

- Use History-Info header

  The *Use History-Info* header option determines whether Cisco BroadWorks inserts a *History-Info* header (policy enabled) or a *Diversion* header (policy disabled) in the INVITE message when propagating redirection information to the device (on the access side).

NOTE: When PBX Integration is selected and Network Server synchronization is enabled, new Line/Ports created under this new DeviceType are synchronized to the Network Server (NS) stating they support the OrigRedirect policy. If the PBX Integration attribute value is changed after the Device Type has been created, modification or creation of Line/Ports will use the new attribute value, but the value for existing Line/Ports will not be synchronized to the Network Server. The existing Line/Ports must be set manually on the Network Server.

5.2.4 Advanced Options

The following is a list of advanced profile policies that can be used when creating or modifying a device profile type.

- Route Advance

  Route Advance defines whether a profile instance of a device profile type handles 3xx responses from the device as a call forward or a contact advance. If this policy is enabled on an identity/device profile, 3xx responses to INVITEs sent to the device associated with this device profile type are treated as contact advances. Cisco BroadWorks uses *RFC 3261* processing and routes the INVITE based on the contact entries specified in the *Contact* header of the 3xx response. Cisco BroadWorks attempts all entries in the 3xx response until it receives a 100 Trying, 18x, or 200 OK response to the INVITE. No additional processing is performed on the contact entries when this policy is enabled. If this policy is disabled, Cisco BroadWorks only accepts the first contact entry in the *Contact* header. Cisco BroadWorks treats the entry as a call forward destination and executes loop detection, outgoing calling plan screening (as applicable), and other call forward processing. Cisco BroadWorks ignores any *Diversion* or *History-Info* header entries in the 3xx responses and includes its own *Diversion* or *History-Info* header entry for the forward attempt. If the forward attempt is unsuccessful, the other contact entries, if any, in the *Contact* header of the 3xx response are discarded.

- Wireless Integration

Wireless Integration defines whether a profile instance of a device profile type allows the called party information to be derived from the *P-Called-Party-ID* header, *Diversion* header, or *Call-History* header rather than the Request-URI. If this policy is enabled, the called party is derived from the *P-Called-Party-ID* header if present or the oldest (bottom-most) *Diversion or History-Info* header entry if a *Diversion or History-Info* header is present.

NOTE: The *P-Called-Party-ID* header takes precedence over the *Diversion* and *History-Info* headers for the called-party determination.

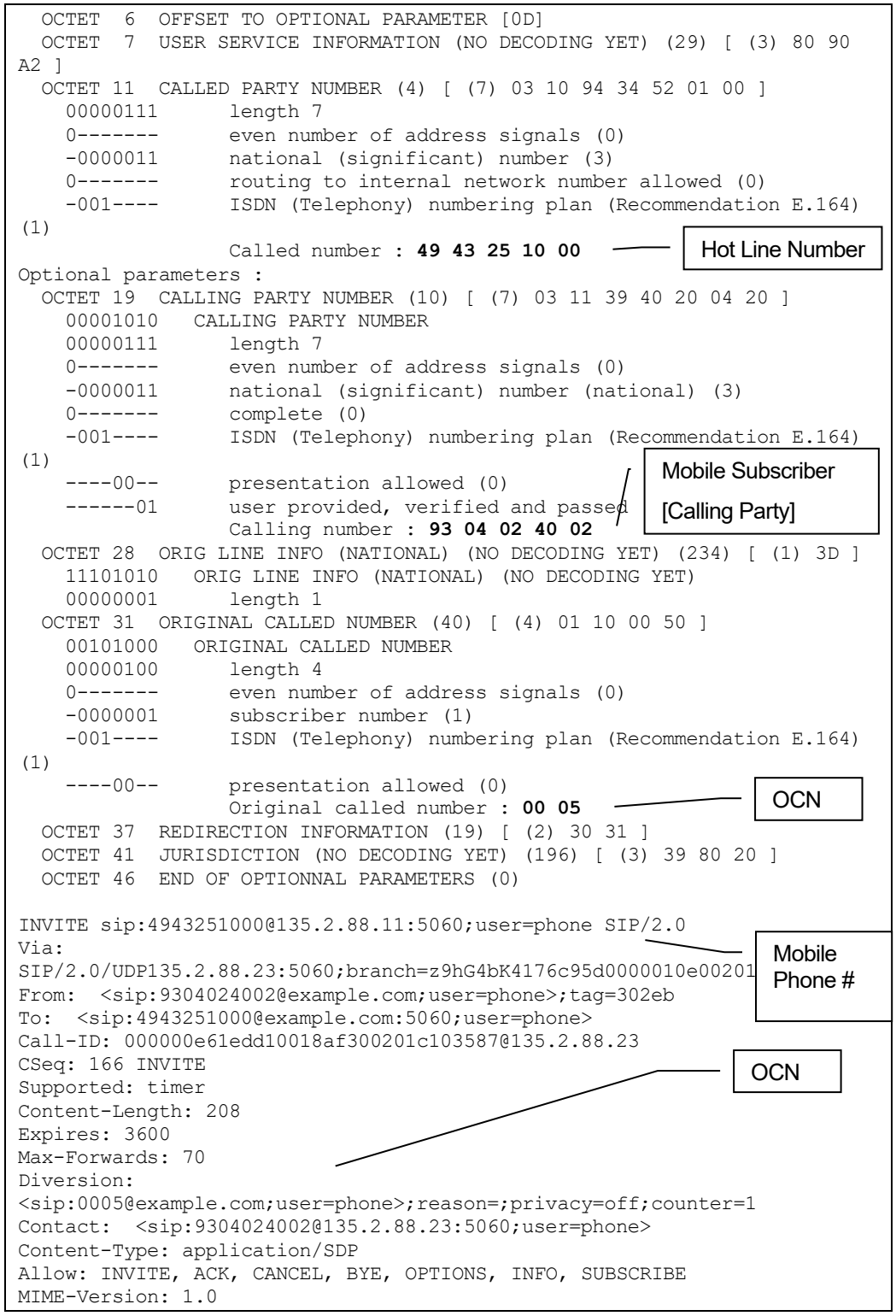
The primary use of this policy is to interwork devices that connect to a wireless Mobile Switching Center (MSC) to facilitate originations from the MSC to Cisco BroadWorks. For a mobile phone to access Cisco BroadWorks services, the mobile phone typically has the "hotline" or auxiliary line feature active in its mobile profile, with a destination directory number hosted by the Cisco BroadWorks Application Server. The mobile softswitch or gateway receives the call from the serving MSC and places the original called number (OCN), if provided by the MSC, into the *P-Called-Party-ID*, *Diversion*, or *History-Info* header of the resulting SIP INVITE. The content of the SIP INVITE for addressing information is as follows:

- **Request-URI**: Destination dn@BroadWorks Application Server address or mobile user domain
- **From**: Mobile Identification Number (MIN) or Mobile Station ISDN Number (MSISDN), which is MSISDN@gateway/softswitch address or mobile user domain
- **To**: Destination dn@BroadWorks Application Server address or mobile user domain
- **Diversion**: Original Called Number (OCN)@BroadWorks Application Server address or mobile user domain

This policy allows Cisco BroadWorks to accept mobile originations via mobile gateways or softswitches and obtain the original called number (typically obtained from the *Request-URI*) from the *P-Called-Party-ID*, or bottom-most, earliest (oldest) *Diversion* or *History-Info* header entry. For originations from subscribers using devices associated with profiles of this profile type, the called number is obtained from the *P-Called-Party-ID* header, if present, then the bottom-most (oldest) *Diversion* or *History-Info* header entry, if present, and then the Request-URI. If multiple *Diversion* or *History-Info* header entries are present, Cisco BroadWorks selects the oldest or last entry to obtain the called number. The oldest entry is regarded as the original called number and normally there should not be multiple headers since this is a mobile origination.

The following is a sample Integrated Services User Part Initial Address Message (ISUP IAM) and resulting INVITE.

```
IAM (Initial Address Message) (1) [ (47) 00 20 00 0A 03 06 0D 03 80 90 A2
07 03 10 94 34 52 01 00 0A 07 03 11 39 40 20 04 20 EA 01 3D 28 04 01 10
00 50 13 02 30 31 C4 03 39 80 20 00 ]
Mandatory fixed parameters:
  OCTET  0  NATURE OF CONNECTION IND. (6) [ (1) 00 ]
  OCTET  1  FORWARD CALL IND. (7) [ (2) 20 00 ]
  OCTET  3  CALLING PARTY'S CATEGORY (9) [ (1) 0A ]
  OCTET  4  OFFSET TO FIRST MANDATORY VARIABLE PARAMETER [03]
  OCTET  5  OFFSET TO SECOND MANDATORY VARIABLE PARAMETER [06]
```



- PBX Integration

  Private Branch Exchange (PBX) Integration defines whether a profile instance of a device profile type provides special handling on Cisco BroadWorks for the *Diversion* or *History-Info* header in received INVITE and 3xx responses, and for the *Referred-By* header in REFER requests. If this policy is enabled, INVITEs that contain a *Diversion* or *History-Info* header where the top-most header entry is associated with a device with this profile type, the INVITE is treated as a call forward on behalf of the subscriber identified in the top-most Diversion of *History-Info* header entry and processing occurs such that the call looks as if it were forwarded on Cisco BroadWorks by the subscriber. Additionally, when a 3xx response is received and contains a *Diversion* or *History-Info* header where the top-most header entry is associated with a device with this profile type, the 3xx is treated as a call forward on behalf of the subscriber identified in the top-most Diversion or *History-Info* header entry and processing occurs such that the call looks as if it were forwarded on Cisco BroadWorks by the subscriber. And last, when a REFER request is received and contains a *Referred-By* header that is associated with a device with this profile type, the REFER is treated as a call transfer on behalf of the subscriber identified in the *Referred-By* header and processing occurs such that the call looks as if it were transferred on Cisco BroadWorks by the subscriber.

  The primary use of this policy is to interwork Cisco BroadWorks with IP PBXs or devices that connect to legacy PBXs. This policy provides PBX redirection and diversion support on the access-side interface for PBX devices connected to Cisco BroadWorks. Redirections on the access-side interface can occur both within and outside an existing dialog. In particular, this policy supports scenarios in which calls are originated within the PBX device and then diverted to Cisco BroadWorks. This functionality is also added to handle intelligent PBXs with hosted unified communications service that redirects calls to minimize trunking capacity by removing the PBX from the signaling paths for these scenarios. This is common when users on the PBX have their Local Call Forwarding service configured to go to a voice portal number that is hosted on Cisco BroadWorks.

- Add P-Called-Party-ID

  Add P-Called-Party-ID controls whether or not the Application Server should add the *P-Called-Party-ID* header to terminating requests. This control applies to all SIP devices, including trunking and regular devices.

  When this option is set to “false”, the Application Server does not insert a new *P-Called-Party-ID* header in the outgoing request. The Application Server may still proxy a *P-Called-Party-ID* header, if the header proxy policy permits this.

  When this option is set to “true”, the Application Server inserts a new *P-Called-Party-ID* header in the outgoing request message. The content of the header reflects the terminating user identity that is reached. This identity can be the user’s main address or one of their alternate addresses. If the Application Server has received a *P-Called-Party-ID* header in the incoming request, the request is discarded and the newly built one is sent along with the outgoing termination request. Note that the content of the *P-Called-Party-ID* header is normalized using the *sendE164* device option.

  The following example shows a scenario where the Application Server adds the *P-Called-Party-ID* header. This is an INVITE going to a terminating device when the *Add P-Called-Party-ID* option is set to “true”. The content of the header reflects the terminating user’s address that is called.

Terminating user attributes:

```
Main Address: +13004006000@as.net
Alternate Address: +15146986000@as.net
Registered Location: 3004006000@device.net
sendE164 device option is enabled.
```

Incoming INVITE (to main address):

```
INVITE sip:+13004006000@as.net SIP/2.0
From:”NetworkUser”<sip:+12409884321@network.com;user=phone>;tag=3
To:<sip:+13004006000@as.net>
```

Outgoing INVITE:

```
INVITE sip:3004006000@device.net SIP/2.0
From:”NetworkUser”<sip:+12409884321@network.com;user=phone>;tag=3
To:<sip:+13004006000@as.net>
P-Called-Party-ID: <sip:+13004006000@as.net>
```

This device option is used in both stand-alone and IMS deployments.

- Auto Configuration Soft Client

  Auto Configuration Soft Client defines whether a profile instance of an identity/device profile type is a SIP-enabled client supporting auto-configuration user profile information obtained via the OCI interface and based solely on the user's Cisco BroadWorks user ID and password. The policy has no execution impact; however, it is required as an attribute on the device type reported in an OCI-P command.

- Requires Cisco BroadWorks Call Waiting Tone

  The *Requires BroadWorks Call Waiting Tone* option defines whether a profile instance of a device profile type requires Cisco BroadWorks to play a call waiting tone. If the policy is enabled, it indicates that the called device is incapable of playing the call waiting tone and requires an interactive voice response (IVR) connection to be created to the Media Server to play the call waiting tone. This policy applies to non-intelligent device profiles with the *Requires BroadWorks Digit* option enabled. If the device does not support BroadWorks INFO requests to play call waiting tone, this policy must be enabled for Cisco BroadWorks to generate the call waiting tone on behalf of the device.

- Advice Of Charge Capable

  The *Advice Of Charge Capable* option defines whether a profile instance of an identity/device profile type is capable of handling advice of charge information. Advice of charge information is the result of the calculation of a communication charge at a given time based on the applicable tariff information. This information is typically carried in an *application/vnd.etsi.aoc+xml* body. If this device option is not enabled, the Application Server avoids sending the *application/vnd.etsi.aoc+xml* body toward the access device. Otherwise, the advice of charge body is sent as expected.

- Support Emergency Disconnect Control

  Support Emergency Disconnect Control specifies whether the device supports the ability to put the emergency call on hold when the handset is put on hook. When the option is enabled, the SIP device should know when an emergency call is made after digits are dialed. This is required in particular to address the situation when an emergency originator hangs up and the emergency operator needs to alert or recall the originator. This option also indicates whether the device is capable of sending a re-INVITE on disconnect for emergency calls.

- Enable Monitoring

  Enable Monitoring defines whether a profile instance of a device profile type is monitored by Cisco BroadWorks CPE monitoring. If this policy is enabled, Cisco BroadWorks periodically pings the device via a SIP OPTIONS request to determine whether the device is responsive to SIP requests. Caution should be used when enabling this feature, as potentially large amounts of traffic can be generated if the polling interval is set too low. The polling interval is configurable via CLI on a per identity/device profile type basis under the *System/Device/Monitor/AccessDevice* level.

- Static Line/Port Ordering

  By default, Device Management uses a dynamic line concept for provisioning IP devices. If the first line on an IP phone is unassigned, the remaining lines are migrated up a line position on the device.

  Static Line/Port Ordering is used to determine whether a device uses static line/port ordering or not. It can be enabled when the device type is created; however, by default, it is disabled.

  When enabled, it allows Device Management to assign a user to a port on a device and to always keep it assigned to that port unless specifically changed by the administrator.

  For more information on this concept, see the *Device Management LinePort Ordering Enhancement Feature Description* [6].

- Support Call-Info Conference Subscription URI

  This option specifies whether the device type supports the ability to use the Call-Info conference subscription URI to subscribe to the conference event package.

- Support Visual Device Management

  When enabled, this device option indicates that the device type is set up to support the Leonid Visual Device Management Portal Provisioning.

  The following device types supported by Cisco BroadWorks are supported by the Loki Portals Visual Device Management; therefore, the option is enabled by default for these (whereas all others device types have this option not selected):

  - Aastra 39i
  - Aastra 55i
  - Aastra 57i
  - Polycom SoundPoint IP 3xx
  - Polycom SoundPoint IP 450
  - Polycom SoundPoint IP 550, 560
  - Polycom SoundPoint IP 650, 670
  - Cisco SPA 525(x)
  - Cisco SPA 509G
  - Cisco SPA 508G
  - Cisco SPA 504G
  - Cisco SPA 502G

- Cisco SPA 501G

For more information on using this option, see *Appendix C: Visual Device Management Support Provisioning* and the *Visual Device Management Support Feature Description* [7].

- Support Cause Parameter

This option specifies whether the cause parameter is supported. When not supported, the mapping between the *History-Info* header and *Diversion* header acts in accordance to the existing configuration. Otherwise, the cause URI parameter in *History-Info* as specified in *RFC 4458* is supported, which enables the mapping between the *History-Info* header and *Diversion* header in accordance to *RFC 6044*.

- Forwarding Override

Forwarding Override defines whether a profile instance of an identity/device profile type includes signaling information to short circuit or inhibit forwarding on a remote system. If this policy is enabled, a *Diversion* or *History-Info* header determined by the *Use History-Info Header* policy setting, is included in the INVITEs sent to the device. The *Diversion* header, if included, contains a counter set to the value of the *maxForwardingHops* parameter in the *Interface/SIP* level of the CLI. The *History-Info* header, if included, contains an index in the Request-URI with maxForwardingHops + 1 levels.

The primary use of this policy is to interwork devices that connect to a local exchange switch to facilitate terminations to the subscriber's phone on the local exchange switch. This policy prevents a local exchange switch from forwarding a call to an alternate number other than the subscriber's phone by relying on the switch to bypass the Forwarding service on the switch and short-circuit the call directly to the subscriber's phone.

The *maxForwardingHops* value in the CLI should be set equal to the maximum number of forwards allowed by the local exchange switch when enabling this policy.

Figure 23 shows an example of the outgoing INVITE sent toward the device with the Forwarding Override policy in the identity/device profile.

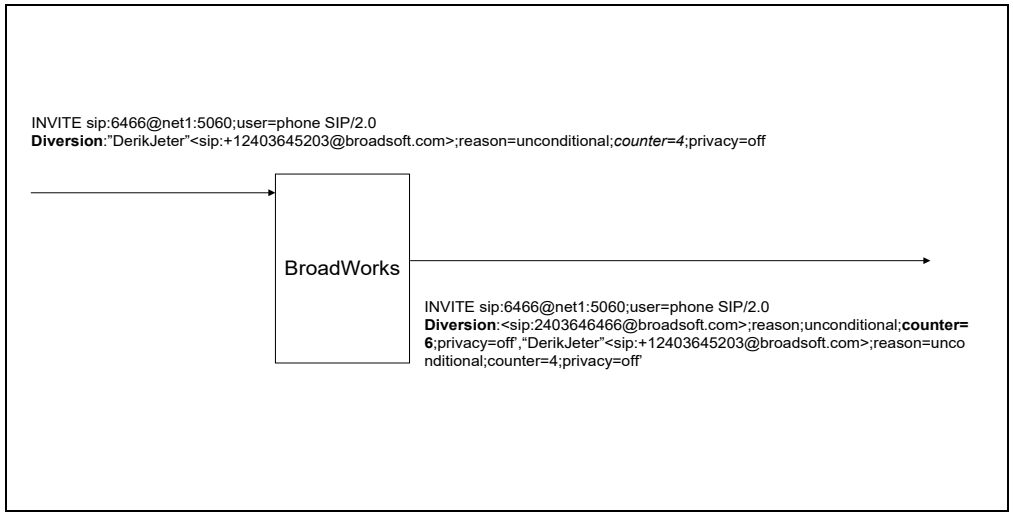


Figure 23 Example of Forwarding Counter Override

- Conference Device

  Conference Device defines whether a profile instance of a device profile type can be configured as an instant conferencing device. If this policy is enabled, profiles of this profile type can be configured as conference devices. These profiles show up in the CLI at the *System/Device/InstantConf* level and are also accepted via the OCI interface.

  NOTE: The Conference Device setting cannot be modified on an existing device profile type.

- Mobility Manager Device

  The *Mobility Manager Device* option defines whether a profile instance of a device profile type can be configured as a Mobility Manager Device. If this policy is enabled, then profiles of this profile type can be configured as Mobility Manager Devices.

  NOTE: The Mobility Manager Device setting cannot be modified on an existing identity/profile type.

- Music On Hold Device

  Music On Hold Device defines whether a profile instance of a device profile type can be configured as a Music On Hold device. If this policy is enabled, profiles of this profile type can be configured as Music On Hold devices. These profiles show up in the drop-down list on the web and via the OCI interface for the external devices on the Music On Device service.

  NOTE: The Music On Hold setting cannot be modified on an existing identity/profile type.

- Requires Cisco BroadWorks Digit Collection

  The In-Call Service Activation service allows detection of a mid-call “flash” digit string. In addition to requiring the In-Call Service Activation service for In-Call Service Activation, the call must be made to or from a non-intelligent (that is, Cisco BroadWorks-controlled) device that has the *Requires BroadWorks Digit Collection* device option; otherwise, the media path is not monitored for the configured digit sequence.

  The *Requires BroadWorks Digit* device option is important as it helps to avoid monitoring calls that do not require it. For example, a user can have the Shared Call Appearance service with a *Requires BroadWorks Digit* device and a SIP phone. If a call is made with the SIP phone, chances are it will handle a flash itself (if it is an “intelligent” device). Another example is if the other phone is off a Media Gateway Control Protocol (MGCP) gateway that reports flash events to the Application Server. In these cases, there is no need to monitor the media path.

  Nothing specifically prevents an MGCP device from using In-Call Service Activation, but since these devices support hook flash events as part of the MGCP, In-Call Service Activation is typically used with non-intelligent SIP devices. In-Call Service Activation does not monitor the speech path for intelligent devices since these already handle multiple calls and thus have no need for the service.

- Requires MWI Subscription

  The *Requires MWI Subscription* option defines the subscription usage for the phone Message Waiting Indication. When the policy is enabled and the subscriber has the Voice Messaging User service assigned, the NOTIFY message on *Message Waiting Indication* state is sent only if the phone has subscribed to the message-summary package.

- Support Call Center MIME Type

  The *Support Call Center MIME Type* option determines whether the agent's device supports the Call Center MIME type. The MIME type allows Cisco BroadWorks to send additional call information related to Call Center calls to an agent's SIP phone. When this option is enabled, the INVITE message for the Call Center or Dialed Number Identification Service (DNIS) call received by a Call Center agent carries the Call Center MIME.

  The MIME type contains the following information:

  - Wait Time: Is the amount of time (in seconds) that the caller has spent in a Call Center queue waiting for an agent.
  - Call Center User ID: This is the Cisco BroadWorks user ID configured against the Call Center.
  - Call Center Name: This is the name configured against the Call Center or the DNIS.
  - Number of Calls in Queue: The number of calls remaining in the Call Center queue except the call being offered to the agent. The call being offered to the agent is not counted while reporting this number.
  - Longest Wait Time: The wait time (in seconds) of the call present longest in the Call Center queue. All calls, except the call being offered to the agent, are considered when calculating the Longest Wait Time. These include bounced and re-ordered calls as well.

- Support Identity UPDATE and Re-INVITE

  This option specifies whether the identity/device profile type supports identity information in UPDATE and Re-INVITE messages.

- Support RFC 3398

  This option specifies whether *RFC 3398* is supported. When not supported, the Application Server allows remote to local ringback transitions. Otherwise, the Application Server does not allow remote to local ringback transitions unless it determines that the remote side is another Cisco BroadWorks server.

- Support Client Session Info

  This option specifies whether the device type supports the ability to communicate client session information.

- Support Remote Party Info

  This option specifies whether the device type supports the ability to process the *X-BroadWorks-Remote-Party-Info* header (when available), which can be used to perform contact lookup.

- Bypass Media Treatment

  This option specifies whether the bypass media treatment is supported. When not supported, the handling of media treatment acts in accordance to the existing configuration, including any default or configured media treatment. Otherwise, when supported, originating calls that encounter an internal or external error condition do not provide any media treatment. In addition, they proxy or generate the default or configurable SIP response/reason handling output.

- Support Calling Party Category In Outbound From Header

  This option is an advanced option that applies to trusted terminating access devices. If the terminating device type does not have the *Trusted* device option selected, then the *Support Calling Party Category In Outbound From Header* selection is ignored.

  The setting of *Support Calling Party Category In Outbound From Header* controls whether the CPC headers are included in the outbound SIP *From* header to trusted terminating access devices.

  – When set to “false”, the CPC parameters are not included in the outbound SIP From header to trusted access devices.

  – When set to “true”, the existing CPC policies may include the CPC parameters in the outbound SIP From header.

- Reset Event

  The Reset Event policies are only applicable when the Allow CPE Configuration policy is enabled. When this policy is enabled, administrators on Cisco BroadWorks can remotely reset devices. The Reset Event policy determines which type of NOTIFY event is sent to the device. Cisco BroadWorks resets/reboots remote devices via a NOTIFY request with an event type of either reSync or checkSync. The setting of the Reset Event policy determines which event type is used for the reset. These fields should only be included for Cisco BroadWorks-validated devices. For configuration details on Enhanced IP Device Configuration support with the identity/profile types, see the applicable partner configuration guide.

- Trunk Mode

  Trunk Mode is used to route the outgoing requests. It replaces the *Use Business Trunking Contact* option. It is introduced to add a new mode of operation to the existing two modes to improve interoperability.

  The Trunk Mode can take one of the following three values:

  – *User* – This is equivalent to setting the *Use Business Trunking Contact* option to “Off”. In the *User* mode, there is no parent-child relationship between the trunk user and the trunk group pilot user.

  – *Pilot* – This is equivalent to setting the *Use Business Trunking Contact* option to “On”. In the *Pilot* mode, there is a parent-child relationship between the trunk group user (child) and the trunk group pilot user (parent). In that mode, the trunk group users partially use the identity of the pilot user to populate the identification information in termination signaling.

  – *Proxy* – The *Proxy* mode is the new operating mode. In the *Proxy* mode, there is a parent-child relationship between the trunk user (child) and the trunk group pilot user (parent).

  The outgoing Request-URI content is added based on the value of *Trunk Mode*.

- Unscreened Presentation Identity Policy

  Unscreened Presentation Identity Policy controls the presentation identity of the originator. It can take the values of “Use Profile Presentation Identity”, “Use Unscreened Presentation Identity”, and “Use Unscreened Presentation Identity with Profile Domain”.

  – If *Unscreened Presentation Identity Policy* is set to “Use Profile Presentation Identity”, the presentation identity is set to the profile presentation identity of Cisco BroadWorks user.

  – If *Unscreened Presentation Identity Policy* is set to “Use Unscreened Presentation Identity with Profile Domain”, the Application Server identifies whether the user part of From URI of the original INVITE request is a phone number. The user part is considered to be a phone number if one of the following conditions is met:

    The user parameter in the From URI is set to “user=phone”.

    -OR-

    The user consists of digits with an optional a “+” prefix, and the host portion of the From URI is a valid domain name or a valid IP address of the Application Server.

    If the user part is identified as a phone number, the presentation name and presentation number are set to the display name of the *From* header and the phone number present in the From URI of the original INVITE request. The host portion is set to same host of the profile presentation identity. The presentation identity includes a user=phone parameter. Otherwise, the presentation identity is set in the same way as if *Unscreened Presentation Identity Policy* were set to “Use Unscreened Presentation Identity”.

    The following examples show how the presentation identity is set. The “broadworks” and “10.0.40.49” are the domain and IP address of the Application Server.

Received From	Presentation Identity
"PhoneA" <sip:2405559001@example>	"PhoneA" <sip:2405559001@example>
"PhoneA" <sip:unavailable@broadworks>	"PhoneA" <sip:unavailable@broadworks>
"PhoneA" <sip:2405559001@130.94.149.229>	"PhoneA" <sip:2405559001@130.94.149.229>
"PhoneA" <sip:2405559001@example;user=phone>	"PhoneA" <sip:2405559001@10.0.40.49;user=phone>
"PhoneA" <sip:2405559001@broadworks>	
"PhoneA" <sip:2405559001@10.0.40.49>	
"PhoneA" <sip:2405559001@broadworks;user=phone>	
"PhoneA" <tel:+12405559001>	

  – If *Unscreened Presentation Identity Policy* is set to “Use Unscreened Presentation Identity”, the presentation identity is obtained from the *From* header field of the original INVITE request. If the original *From* URI includes a user=phone parameter, the user=phone URI parameter is preserved as well.

Note that if the user part is considered as a phone number and the host portion of the From URI is a valid domain name or a valid IP address of the Application Server, the host portion is set to same host of the profile presentation identity. The presentation identity includes a user=phone parameter.

The following examples show how the presentation identity is set. The "broadworks" and "10.0.40.49" are the domain and IP address of the Application Server.

Received From	Presentation Identity
"PhoneA" <sip:2405559001@example>	"PhoneA" <sip:2405559001@example>
"PhoneA" <sip:unavailable@broadworks>	"PhoneA" <sip:unavailable@broadworks>
"PhoneA" <sip:2405559001@130.94.149.229>	"PhoneA" <sip:2405559001@130.94.149.229>
"PhoneA" <sip:2405559001@example;user=phone>	"PhoneA" <sip:2405559001@example;user=phone>
"PhoneA" <sip:2405559001@broadworks;user=phone>	"PhoneA" <sip:2405559001@10.0.40.49;user=phone>

Note that this attribute does not apply to emergency calls or trunk group calls. The setting of the device attribute is ignored if the *enableTS29163Compliance* SIP parameter is set to "false".

- Web-based Configuration URL Extension

  Web-based Configuration URL Extension identifies the extension, if any, to the IP address necessary to access the web page for a profile instance of an identity/device profile type that provides web-based configuration. The Cisco BroadWorks web portal provides a link via the *Device Profile Configuration* page for accessing the device's GUI. This link defaults to *http://<device-address>*, where <device-address> is an IP address or a domain name identified via device registration or static configuration. If a specific extension is required in addition to the address, it must be supplied here.

  Examples:

```
Web Based Configuration URL: http:<device-address>/Admin
Web Based Configuration URL Extension: /Admin

Web Based Configuration URL: http:<device-address>:8085
Web Based Configuration URL Extension: :8085
```

- Allow Auto-Configuration

  Auto-Configuration in Device Management allows devices to automatically configure a device profile without operator intervention. Auto-configured device profiles have a generated name and are associated to an auto-generated endpoint. This enables devices to make calls as soon as they retrieve their profile files.

- No Calling Party Category In Form

  Applies to trusted terminating device types. If the device profile type does not have *Trusted* selected, *No Calling Party Category In From* selection is ignored.

  In IMS mode, the Application Server receives call termination requests where the incoming SIP INVITE PAI header may include CPC parameters. The setting of *No Calling Party Category In From* for the terminating device type controls if the CPC header is not included in the outbound SIP FROM header.

When *No Calling Party Category In From* is set to “true”, the CPC parameters are not included in the outbound SIP FROM header.

When *No Calling Party Category In From* is set to “false”, the CPC parameters are allowed to be included in the outbound SIP FROM header.

The *No Calling Party Category In From* does not control the CPC parameters received within the FROM header.

5.3 Define Configuration Profiles

5.3.1 Overview

Prior to Release 14.sp6, Cisco BroadWorks supported a feature called Enhanced IP Device Configuration. Device Management builds on this functionality by adding advanced configuration profile management capabilities.

When adding a new device profile type to the system, the system administrator must decide which level of configuration management is supported. The Device Provisioning presentation is as follows:

- Initially, only the *Device Configuration Options* setting is shown, along with the “Legacy”, “Device Management”, and “Not Supported” options.
- When the “Not Supported” option is selected, no other parameters appear. “Not Supported” is the default option (as shown in *Figure 22)*.
- When the “Legacy” option is selected, the *Legacy Configuration* settings appear (as shown in *Figure 25)*.
- When the “Device Management” option is selected, the *Device Management* settings appear (as shown in *Figure 25)*.

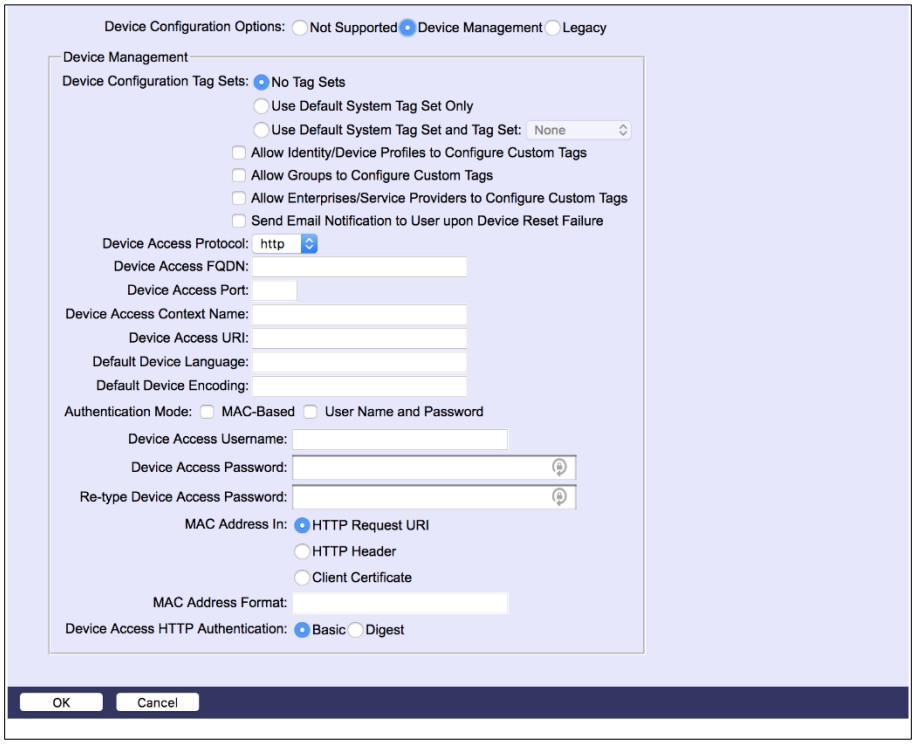


Figure 24 Setting Configuration Profile Attributes for Device Management Configuration Option

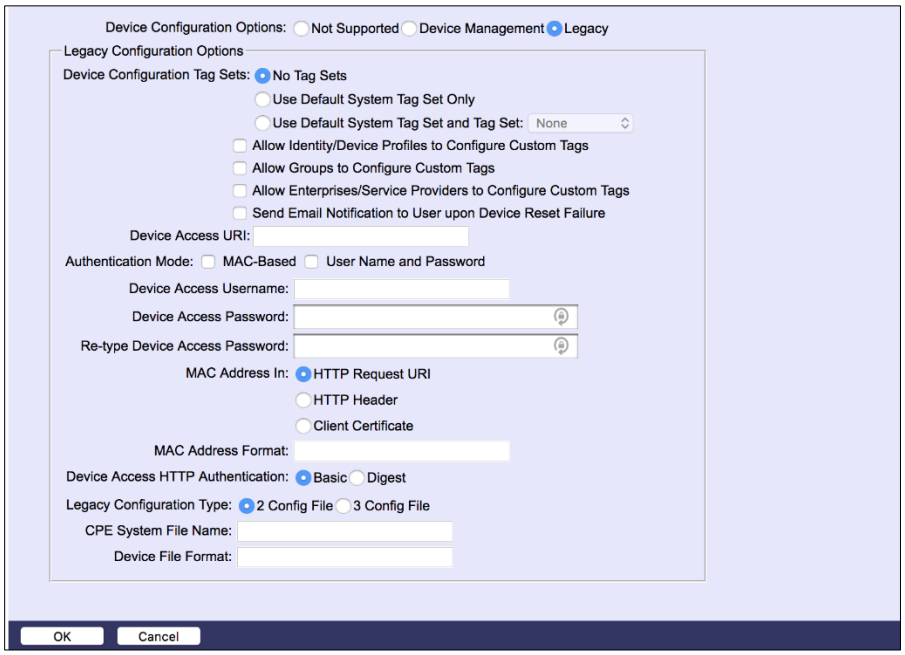


Figure 25 Setting Configuration Profile Attributes for Legacy Configuration Option

5.3.2 Device Configuration Options Summary

Several policies are also available for the identity/profile type for auto-configuration and device integration. These options should only be used when creating a device profile type for a specific device model. The following subsections describe the device configuration/device integration policies that can be used when creating a device profile type. Note that the device configuration/device integration policies cannot be modified on an existing device profile type.

5.3.2.1 Legacy Configurations Options

- Allow CPE Configuration

  Allow CPE configuration defines whether a profile instance of an identity/device profile type has integrated device configuration capability with the Cisco BroadWorks Device Management capability. If this policy is enabled, Cisco BroadWorks provisioning auto-populates the device configuration files with the subscriber line and SIP information. These files are then obtained by the device via applicable device mechanism.

  Configuration Type/CPE System file Name/Device File Format

  – The Configuration Type, CPE System File Name, and Device File Format policies are only applicable when the Allow CPE Configuration policy is enabled. The configuration type, the CPE system file name, and device file format are required to be entered when the Allow CPE Configuration policy is enabled. These fields should only be included for Cisco BroadWorks-validated devices. For configuration details on enhanced IP device configuration support with the identity/profile types, see the applicable partner configuration guide.

- Legacy File Management

  For detailed information about legacy migration and legacy template support, see *Appendix B: Legacy Support and Migration Paths*. A legacy file refers to a file that has been provisioned prior to the introduction of the new Device Management. Although it is still technically possible to provision a legacy file with the new Device Management, the ability to do this might be deprecated in a future release.

NOTE 1: The recommend usage is through the new interface described in section *5.3.2.2 Device Management Options*.

NOTE 2: To replace a legacy file with a new file, you can simply create a new template with the same name as the legacy file.

Legacy files were (and still are) provisioned when the device type is created. When a new device type is created, specify the system file and device file formats.

From the CLI:

```
AS_CLI/System/DeviceType/SIP>add […]
```

(Parameters are omitted for the purpose of clarity. Type "help add" for the full listing.)

… where:

- **<deviceConfigurationOption>** must be set to "Legacy".
- **<LegacyConfigType>** indicates the number of configuration files for this device type. Legacy device types support two or three files.
- **<systemFileName>** specifies the system file name as requested by the device on the front end. For example, a Polycom device might use *PolyCom600System.cfg*.
- **<deviceFileFormat>** specifies the device file name as requested on the front end. For example, a Polycom device might use *0004f2173767.cfg*.
- **<tagMode>** specifies which static tag sets to use when resolving tags for this device. For information on how to create the static tags, see section *5.13.6 Create Static Tags*.
- **<allowDeviceProfileCustomTagSet>** determines whether or not new static tags can be defined or customized at the profile level.
- **<allowGroupCustomTagSet>** indicates whether or not new static tags can be defined or customized at the group level.
- **<sendEmailNotifUponResetFailure>** indicates whether a notification email is sent when the device cannot be reset.
- **<tagSet>** is the static tag set that is available for this device type in addition to the system tag set.

After the device type has been created, legacy files must be manually copied on the Application Server under */var/broadworks/IpDeviceConfig/* with the following formats:

- Configuration templates, per device type, when using the legacy 2/3-file device configuration management files.

```
/var/broadworks/IpDeviceConfig/BW_SYSTEM_DEFAULT_<Device Type Name>.template
/var/broadworks/IpDeviceConfig/BW_DEFAULT_<Device Type Name>.template
```

```
/var/broadworks/IpDeviceConfig/BASE_FILE_TP2_<Device Type Name>.template
```

- Configuration templates, per group and per device type, when using the legacy 2/3-file device configuration management files.

```
/var/broadworks/IpDeviceConfig/<group UID path>/Group_<Group ID>_<Device Type Name>.template
```

- Configuration templates, per-group device profile, when using the legacy 2/3-file device configuration management files.

```
/var/broadworks/IpDeviceConfig/<group UID path>/Custom_<Group ID>_<Device Type Name>_<Device Profile Name>.template
```

5.3.2.2 Device Management Options

5.3.2.2.1 Device Type Configuration Options

The feature allows files to be provisioned for an existing device type. The Application Server must first know that a device type supports this capability. This is accomplished from the CLI when the device type is created:

```
AS_CLI/System/DeviceType/SIP>add […]
```

… where:

- **<deviceConfigurationOption>** must be set to “deviceManagement”.
- **<resetEvent>** indicates whether the device supports a SIP NOTIFY to be notified as to when to synchronize its files with a new version available on Cisco BroadWorks. The options are:
  - Send no notification
  - Send a NOTIFY resync event (Event:resync)
  - Send a NOTIFY checkSync event (Event: checkSync)
- **<deviceAccessProtocol>** determines the transfer protocol used by the device to fetch its files. HTTP, HTTPS, and TFTP are used when the deployment includes an Xtended Services Platform (Xsp) and a Profile Server (PS).
- **<tagMode>** specifies which static tag sets to use when resolving tags for this device. For information on how to create the static tags, see section *5.13.6 Create Static Tags*.
- **<allowDeviceProfileCustomTagSet>** determines whether or not new static tags can be defined or customized at the profile level.
- **<allowGroupCustomTagSet>** indicates whether or not new static tags can be defined or customized at the group level.
- **<allowSPCustomTagSet>** indicates whether or not new static tags can be defined or customized at the Service Provider/Enterprise level.
- **<sendEmailNotifUponResetFailure>** indicates whether a notification email is sent when the device cannot be reset.

- **<useHttpDigestAuthentication>** protects the file access by a digest challenge if this flag is set to "true", when using the HTTP or HTTPS protocols. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (for more information, see section *5.3.2.2.2 Device Type File Configuration Options*).
- **<macBasedFileAuthentication>** authenticates the file access when the provisioned MAC and the MAC coming from the access device are identical, when using the HTTP or HTTPS protocols. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (see section *5.3.2.2.2 Device Type File Configuration Options*).
- **<userNamePasswordFileAuthentication>** authenticates the file access by a basic challenge if this flag is set to "true", when using the HTTP or HTTPS protocols. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (for more information, see section *5.3.2.2.2 Device Type File Configuration Options*).
- **<macInNonRequestURI>** indicates that the MAC address used in the MAC-based authentication is not embedded in the URL but instead passed in a custom HTTP header or extracted from the client certificate CN. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (for more information, see section *5.3.2.2.2 Device Type File Configuration Options*).
- **<macInCert>** controls whether media access control (MAC) in certificate is enabled. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (for more information, see section *5.3.2.2.2 Device Type File Configuration Options*).
- **<tagSet>** is the static tag set that is available for this device type in addition to the system tag set.
- **<deviceAccessFQDN>** represents the fully qualified domain name (FQDN) of the Xtended Services Platform used by the device to fetch its files.
- **<deviceAccessPortNumber>** is the port number on the Xtended Services Platform used by the device to fetch its files.
- **<deviceAccessContextName>** is the context name given to the BroadworksDms web application when it was deployed on the Xtended Services Platform (for more information, see section *5.17.5.3 BroadworksDms Web Application*).
- **<deviceAccessURI>** is a token used to ensure the uniqueness of the URL for each device type. It typically contains the device type name itself, for example, "Polycom_Soundpoint_IP_500". To determine exactly where in the URI this token is used, see section *5.17.5.1 URI Scheme*.
- **<defaultDeviceLanguage>** contains the default device language.
- **<defaultDeviceEncoding>** contains the default device encoding.
- **<UserName>** contains the user name.

- **<macFormatInNonRequestURI>** gives the format used to read the MAC address of the device from the HTTP header or client certificate CN. This option currently applies to the creation of the Polycom Phone Services directory file only (for more information, see section *5.14 Phone Services*). For all other files, it is not used. The option is instead on a per-file basis at the Device Type Files level (for more information, see section *5.3.2.2.2 Device Type File Configuration Options*).

Figure 26 shows the same task from the web portal.

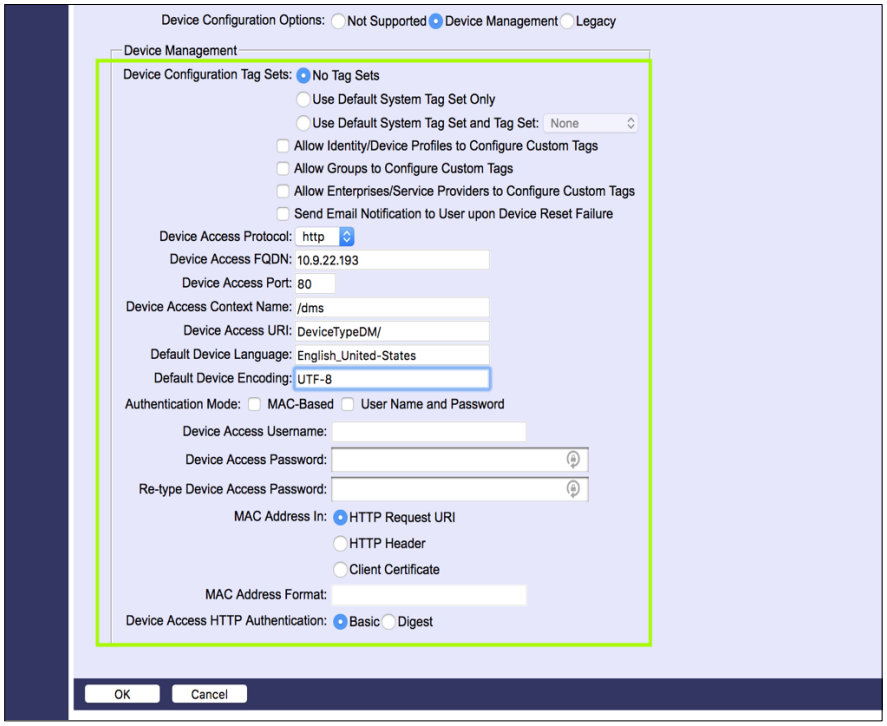


Figure 26 Creating New Device Type with Support for Device Management

5.3.2.2.2 *Device Type File Configuration Options*

Once the device type has been created with settings that allow the provisioning of files, creating a new file can be accomplished from the CLI.

```
AS_CLI/System/DeviceType/SIP/Files>add <deviceTypeName> <fileFormat>
<remoteFileFormat> <fileCategory> <allowFileCustomization> <fileSource>
<uploadFilePath> <useHttpDigestAuthentication>
<macBasedFileAuthentication> <userNamePasswordFileAuthentication>
<macInNonRequestURI> <macInCert> <allowHttp><allowHttps><allowTftp><
enableCaching ><macFormatInNonRequestURI>
```

… where:

- **<deviceTypeName>** is the device type for which the file is added.
- **<fileFormat>** represents the file name known by the device. For example, if a device requires a file called *polycomSystemFile.cfg*, the name requested by a device must be in this field.

- **<remoteFileFormat>** stands for the file name as it appears on the file repository. It can be different or identical to the fileFormat if required. When using a Profile Server, the name can be different because the device never directly accesses the file. When using an FTP server, the name must be identical to the file format. For example, Polycom can reuse the name *polycomSystemFile.cfg* set for the file format or it can use a different name on the repository. This parameter becomes useful when the concept of tags is introduced (for more information, see section *5.13.5 Ensure Unique File Repository Names using Remote File Format*).
- **<fileCategory>** indicates the way the file is treated by the Device Management. Values can be static, dynGroup, or dynProfile.

  A static file is a file that does not have to be processed before it can be sent to a device. It does not contain tags. A static file is usually shared among many devices. A good example would be the firmware file, although a static file does not need to be binary.

  - A dynamic per-type file has tags that are resolved for a given group.
  - A dynamic per-profile file has tags that are resolved for a given device profile.
- **<allowFileCustomization>** specifies whether this file can be customized at the group level or at the profile level. For example, a system file with this flag set to "true" can be customized by a single group because they must have a different configuration. Setting this flag to "false" allows certain control over what a group or a specific profile can change.
- **<fileSource>** provides a mechanism for the Application Server to fetch the file from a local directory on the Application Server. If this setting is set to "custom", the uploadFilePath location is used to fetch the file, which is then uploaded to the Profile Server automatically. When the manual mode is selected, there is no template associated to the device type file and the file can be uploaded to the Profile Server.
- **<uploadFilePath>** contains the local path where the file to be provisioned is currently stored when the custom setting is specified for the fileSource.
- **<useHttpDigestAuthentication>** indicates whether to protect the file with an HTTP digest challenge.
- **<macBasedFileAuthentication>** indicates whether to protect the file by ensuring that the MAC address matches part of the file name or is found within the headers of the HTTP request, or extracted from the client certificate CN.
- **<macInCert>** controls whether media access control (MAC) in certificate is enabled.
- **<userNamePasswordFileAuthentication>** indicates whether to protect the file with an HTTP basic challenge.
- **<macInNonRequestURI>** indicates that the MAC address must be located from an *HTTP* header in the request instead of as part of the requested file name.
- **<allowHttp>** when set to "true", specifies that the HTTP protocol is allowed for file download, otherwise is forbidden (default is "true").
- **<allowHttps>** when set to "true", specifies that the HTTPS protocol is allowed for file download, otherwise is forbidden (default is "true").
- **<allowTftp>** when set to "true", specifies that the TFTP protocol is allowed for file download, otherwise is forbidden (default is "true").
- **<enableCaching>** enables file caching for the specified file on the Xtended Services Platform.

- **<macFormatInNonRequestURI>** specifies a regular expression including the *HTTP* header and the format where the MAC address can be extracted either from the HTTP header or specifies the format used to extract the MAC address from the client certificate.

  This field should be populated with the header name followed by a separator and a regular expression.

  Device file authentication using a MAC address can be based on the HTTP header of the HTTP request coming from the device. The MAC HTTP header authentication is performed by looking at the HTTP header configured for the device profile file. The *macFormatInNonRequestURI* setting must specify the header name followed by a separator and a regular expression.

  Example:

```
macFormatInNonRequestURI=macaddress:.*\(([0-9a-fA-F]{12})\)
```

  …. where the header is “macaddress”, the separator is “:”, and the regular expression to look for is a 12-character string with each character being either a digit “0” through “9”, or a lowercase or uppercase letter between “A” through “F”.

NOTE: The parentheses in the regular expression are mandatory.

Additional examples are provided in the following table. In scenario 3, the parentheses look superfluous; however, they are important as they define a group. For more information, see https://docs.oracle.com/javase/tutorial/essential/regex/groups.html.

If the parentheses are not provided in scenario 2, the expression does not work because without the group to indicate where to extract the MAC address from the matching regex, an incorrect MAC address is returned.

Scenario	Header	Value	Example Regex to Extract MAC
1	User-Agent	Panasonic_KX-TGP550T04/12.02 (000000000000)	User-Agent:.*\(([0-9a-fA-F]{12})\).*
2	Mac-Header	Sony Model 00001111000000000000	Mac-Header:.*[0-9]{8}([0-9a-fA-F]{12})
3	Deviceid	000000000000	Deviceid:([0-9a-fA-F]{12})
4	User-Agent	User-Agent: AUDC-IPPhone/2.0.2.62 (420HD; 000000000000)	User-Agent:.*((([0-9A-Fa-f]{2}[-:.]){5}[0-9A-Fa-f]{2})|(([0-9A-Fa-f]{4}[-:.]){2}[0-9A-Fa-f]{4})|(([0-9A-Fa-f]){12})).*
5	User-Agent	User-Agent: yealink SIP-T26P 10.0.0.100 00:00:00:00:00:00	User-Agent:.*((([0-9A-Fa-f]{2}[-:.]){5}[0-9A-Fa-f]{2})|(([0-9A-Fa-f]{4}[-:.]){2}[0-9A-Fa-f]{4})|(([0-9A-Fa-f]){12})).*

Figure 27 Example of File Authentication HTTP Header Format

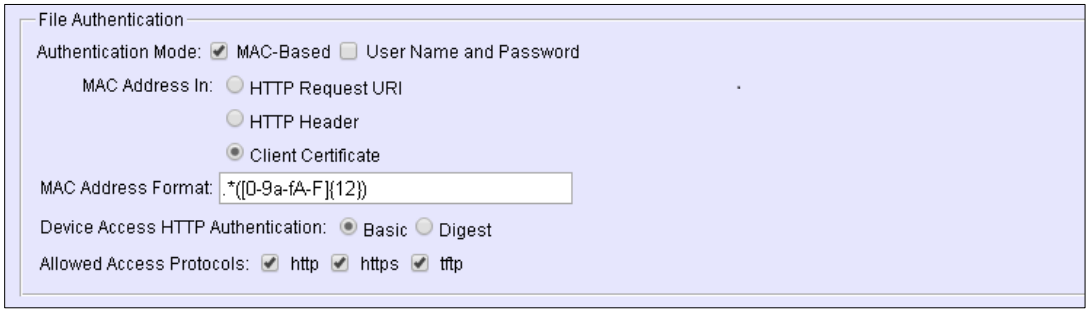


Figure 28 Example of File Authentication MAC address from Client Certificate CN

Figure 29 shows the task of adding a new file to a device type from the web.

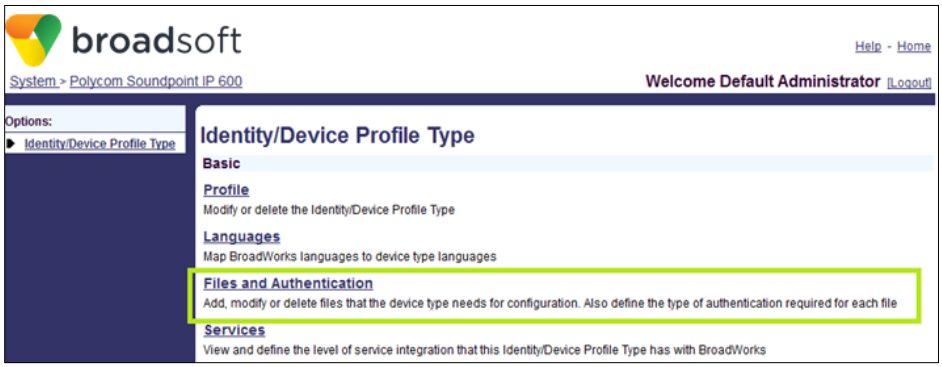


Figure 29 Adding New File to Device Type

… where:

- **<Device Access File Format>** maps to the “fileFormat” configured from the CLI.
- **<Repository File Format>** maps to the “remoteFileFormat” configured from the CLI.
- **<Manual, Custom>**
  - When set to “custom”, the uploaded file is automatically transferred from the Web Server to the Application Server.

NOTE: Template files preview (shown in the text field below *Currently Using File:* in *Figure 29*) only displays the first 128 KB of the template file. This applies to all web pages displaying a preview of device profile file templates (using the OCI transaction *SystemFileGetContentRequest*).

  - When set to “manual”, there is no template associated to the device type file. The file can be uploaded to the Profile Server.

- Per device type. The file-type token is chosen by the administrator when creating a new file.

```
/var/broadworks/IpDeviceConfig/type/<Device Type Name>/<Access File
Format>.template
```

- For file customization, per group, and per device type (see section *5.6 Customize Files*). The file-type token is chosen by the administrator when creating a new file.

```
/var/broadworks/IpDeviceConfig/<group UID path>/<Device Type Name>/Group_<Group
ID>_<Access File Format>.template
```

- For file customization, per group device profile (see section *5.6 Customize Files*). The file-type token is chosen by the administrator when creating a new file.

```
/var/broadworks/IpDeviceConfig/<group UID path>/<Device Type Name>/Custom_<Group
ID>_<Device Profile Name>_<Access File Format>.template
```

5.3.2.2.2.1 Multiple Authentication Support

When both MAC-based and user name and password authentication modes are enabled, both modes must be configured and provisioned. This means that for MAC-based authentication, a MAC address must be provisioned on the server; and if fetching the MAC address from a client certificate, a proper expression must be provisioned as well.

The authentication will be effectively performed for both modes.

Although multiple-authentication mode is supported, it is recommended to only use a single authentication mode.

NOTE 1: Template files are replicated to other hosts in the Application Server cluster.

NOTE 2: Per-profile files cannot be retrieved using the system credentials. Custom credentials or the MAC address defined at the profile level must be used for per-profile files.

5.4 Device Type Import/Export

A device type and its templates can be exported to a file. This allows an administrator who has many Application Server clusters to define the device type only once. After configuring the device type on the first cluster, the administrator can export it to a file and then import in back into another Application Server cluster.

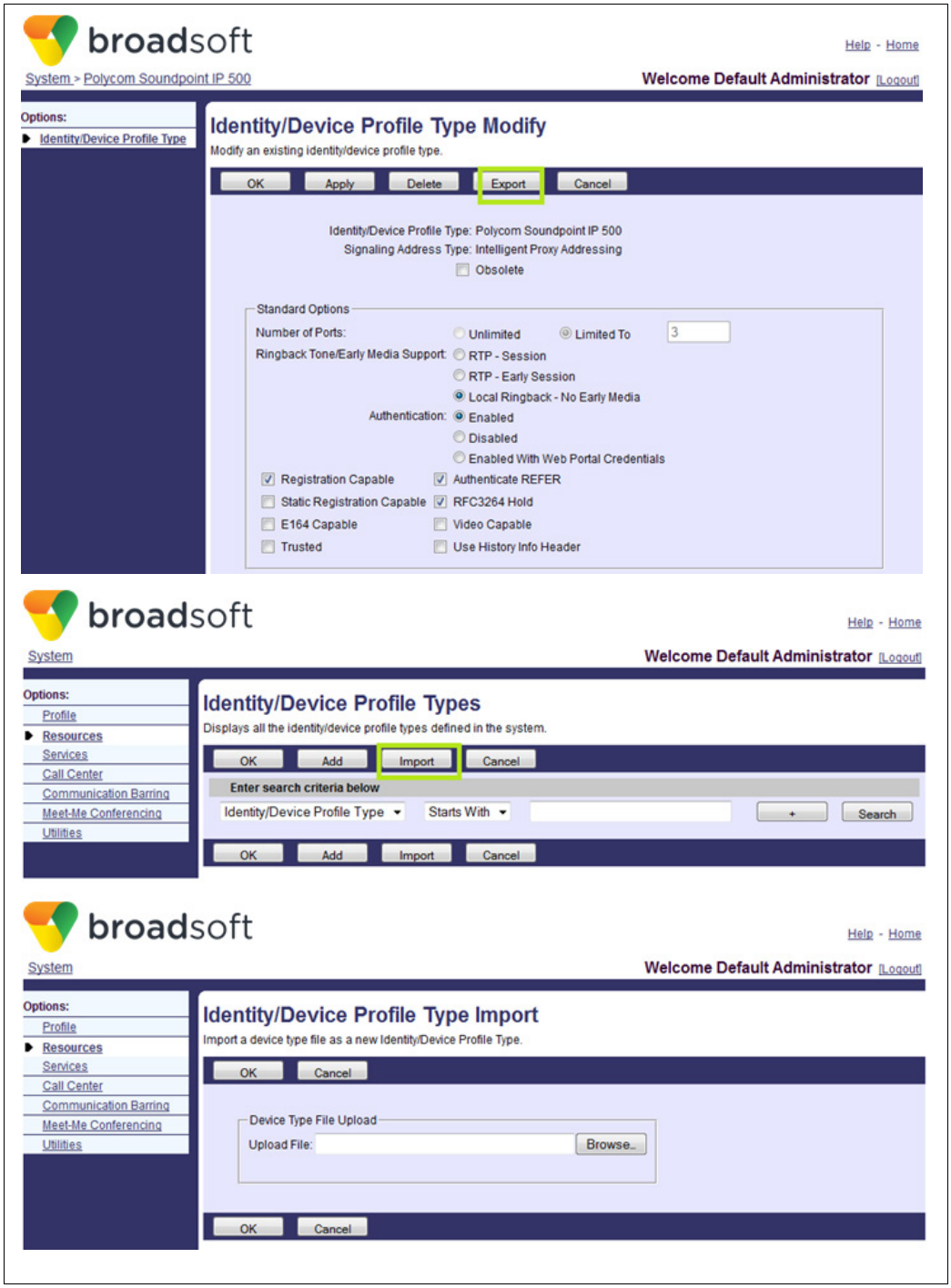


Figure 30 Export and Import of Device Type

5.5 Device Reset

If a device type supports the reset functionality (see the *resetEvent* parameter in section *5.3.2.2 Device Management Options*) and once the files have been generated on the file repository, a reset command can be sent to the device to force a synchronization of its files. Most of the time, the reset must be initiated by the administrator manually as shown in *Table 1*.

Table 1 Automatic versus Manual Reset Events

Event	Type of Reset
User authentication is changed	Automatic if accomplished by the user; otherwise manual.
User information is modified	Automatic if accomplished by the user; otherwise manual.
For all other events	Administrator must manually request a reset (see as follows).

NOTE: A reset event is not generated if the device is not associated with a *fileRepos*. In addition, it is possibly further restricted to a WebDAV *fileRepos* if the *bw.dms.ignoreFtpFileRepos* container option has been enabled.

A reset event is queued with other DMS events. It can be processed immediately or in within an hour, however, there is no way to determine exactly when the event will be processed.

5.5.1 Device Reset Pacing

It allows greater control on device reset pacing, controlling delays between resets sent to devices. It controls the time the system waits between two consecutive resets sent to devices. In other words, it controls the pacing of device resets. It allows the administrator to set long delays for a slower system, which was not possible with the container option used before.

This is controlled through a system parameter *AS_CLI/System/Device/IpDeviceMgmt/minTimeBetweenResetMilliseconds.*

5.5.2 Reset Procedure

Perform a reset from the Application Server CLI.

```
AS_CLI/System/Device/IpDeviceMgmt>reset deviceType system <deviceType>
AS_CLI/System/Device/IpDeviceMgmt>reset deviceType group <svId> <groupId>
<deviceType>
AS_CLI/System/Device/IpDeviceMgmt>reset device >spId> <groupId>
<deviceName>
AS_CLI/System/Device/IpDeviceMgmt>reset all
```

… where:

- **<deviceType>** is the name of the device type to be reset.
- **<svId>** is the name of the service provider.
- **<groupId>** contains the name of the group.
- **<deviceName>** represents the name of the device.

From the web:

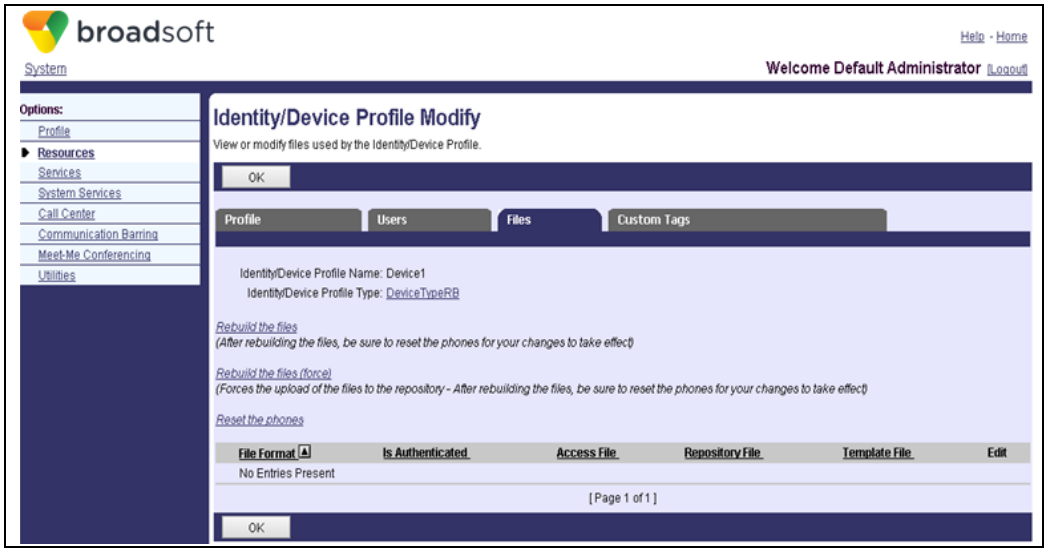


Figure 31 Reset Device from Web

5.5.3 Force Rebuild Procedure

It is possible to trigger a device configuration files rebuild FORCE at the web portal. Only system administrators can see this link. When a force rebuild command is issued, the templates are uploaded to the file repository even if the content has not changed since the last rebuild was performed.

5.5.4 Email Notifications

When a device reset fails, an email is sent to the service provider or group administrator provided that the device type supports email notifications (see the *sendEmailNotifUponResetFailure* parameter in section *5.3.2.2 Device Management Options*). At the service provider and group levels, an email address must be configured as specified in the following subsections.

5.5.4.1 Service Provider Notification

An email notification is sent to the *Contact E-mail* address (and not the *Support E-mail* address) listed on the *Service Provider → Profile and Enterprise → Profile* page. This email address is also used when the group does not have an email address provided.

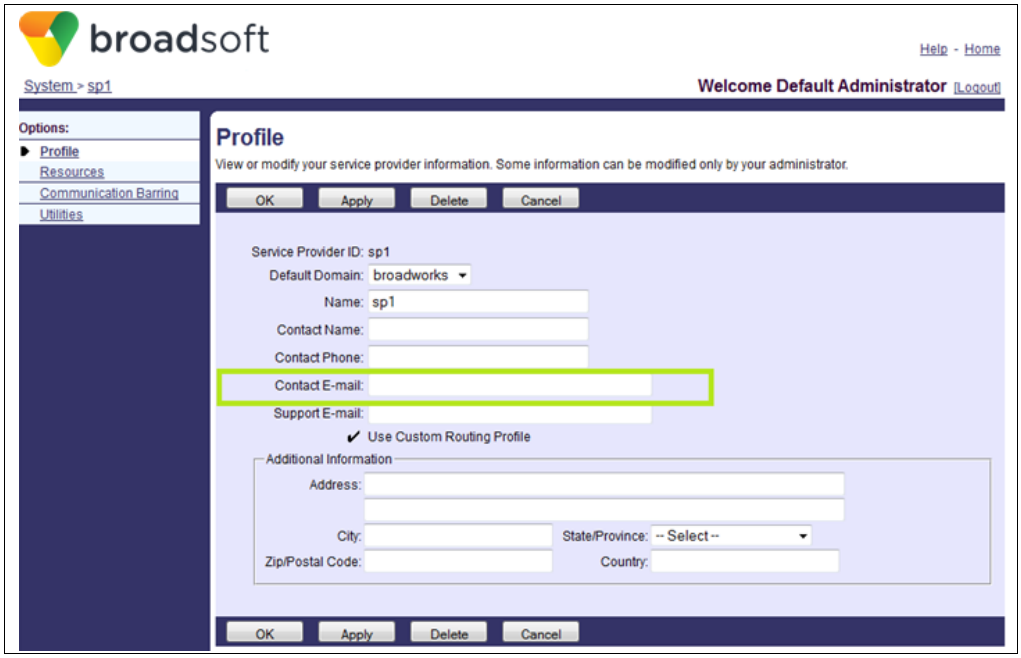


Figure 32 Email Notifications for Service Providers

NOTE: The *Enterprise → Profile* page is identical to the *Service Provider → Profile* page (shown in *Figure 32*.)

5.5.4.2 Group Administrator Notification

The entire group has only one email address. The group email address is taken from the *Contact E-mail* address on the *Group Profile* page. If multiple group administrators should be contacted, it is recommended to set up wizard to forward an email to each administrator. If a group contact email does not exist, the email is sent to the service provider's contact address.

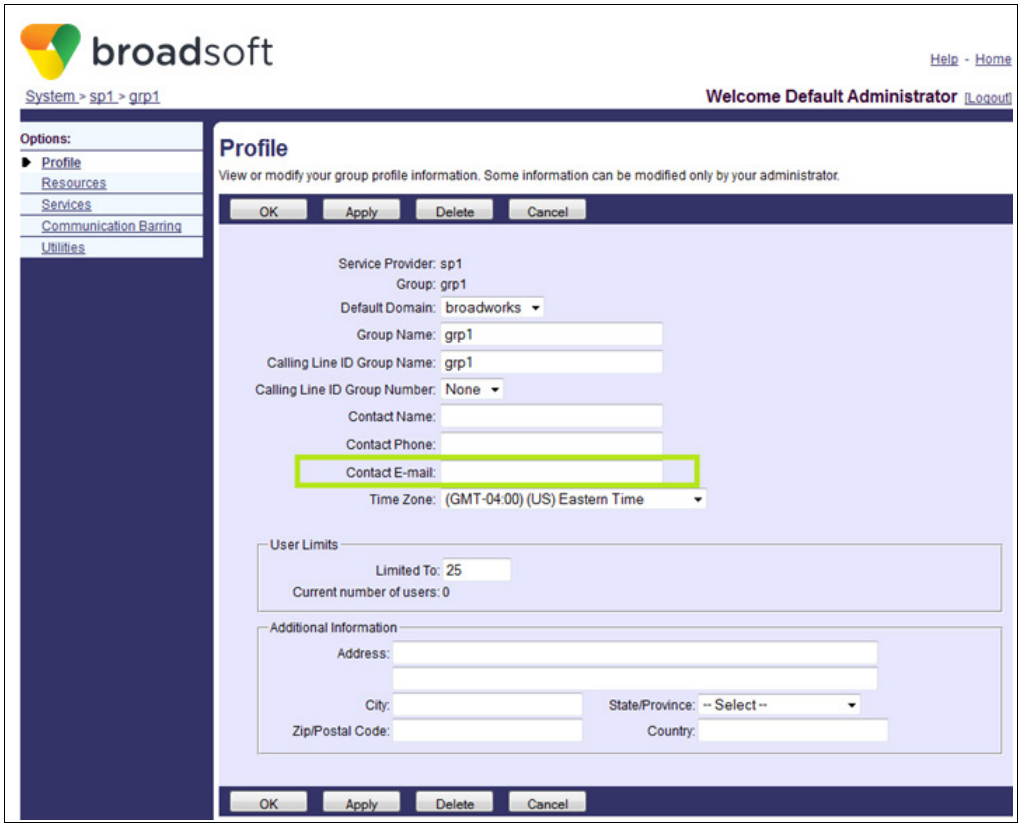


Figure 33 Group – Profile Page

5.5.5 System Provider Notification

If an operation is performed that requires system provider attention, a Simple Network Management Protocol (SNMP) alarm is generated. If the operation was performed by a service provider, an alarm is sent to the system provider and an email is sent to the service provider or group administrator, depending on who issued the operation.

5.6 Customize Files

Once files for a specific device type have been defined, it may be convenient to have a different version of the file at different "levels". This is accomplished through a customization mechanism available from the web or the CLI interfaces as shown in *Figure 34* and *Figure 35*.

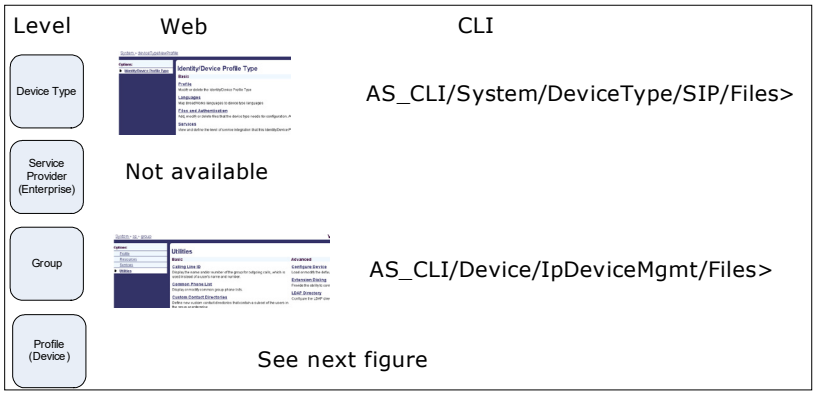


Figure 34 Device Files Customization Hierarchy

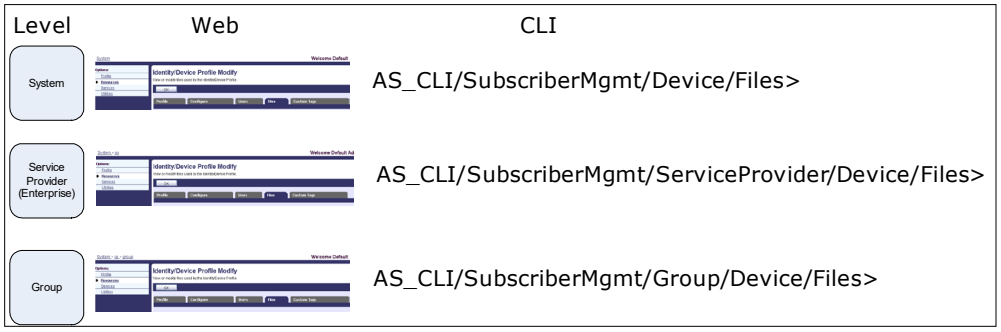


Figure 35 Profile Files Customization Hierarchy

As shown in *Figure 34* and *Figure 35*, files can only be created at the device-type level. Once a file is created, it can be customized at the group level or the profile level. Customizing a file at the profile level is done at the same level where the profile exists (system, service provider, or group).

For the customization mechanism to be available, the *<allowFileCustomization>* parameter must have been set to "true" when the file was added to a device type (see section *5.3.2.2 Device Management Options*). For example, BigCorp Inc. has 1000 identical Polycom phones that share a common configuration. For this purpose, a configuration file was created at the device-type level. However, for tax reasons, BigCorp Inc. decided to split a quarter of the company into a distinct entity called MiniCorp Inc. They require 250 of the phones to display a different branding using a specially crafted configuration file. To customize the original system file for these 250 devices, they can go to the CLI:

```
AS_CLI/SubscriberMgmt/Group/Device/Files>set <svcProviderId> <groupId>
<deviceName> <fileFormat> <fileSource> <configurationFileName>
```

… where:

- **<svcProviderId>** is the service provider name of the group.

- **<groupId>** is the group name for the 250 users.
- **<deviceName>** is the name of the device profile assigned to the group.
- **<fileFormat>** represents the file name known by the device. For example, if a device requires a file called *polycomSystemFile.cfg*, the name requested by a device must be in this field.
- **<fileSource>** can be default, manual, or custom. A default setting indicates that the parent level file is used (in this case, it would be the original system file provisioned for the 1,000 users). A manual setting means that the file is customized but that it is uploaded manually to the file repository in the proper directory. Custom means that the file is picked up locally using the path from the next parameter. It is then automatically pushed to the file repository by the Device Management.
- **<configurationFileName>** is the local path on the server indicating where to pick up the file that customizes the service provider file.

Imagine that a small group of vocal vice presidents would love to have the MiniCorp Inc. branding back on their phones. They require the original configuration file to be put back on their 10 devices, which are part of the rebranded service provider. This can be done via the CLI for each of their profiles.

```
AS_CLI/SubscriberMgmt/Device/Files>set <deviceName> <fileFormat>
<fileSource> <configurationFileName>
```

… where:

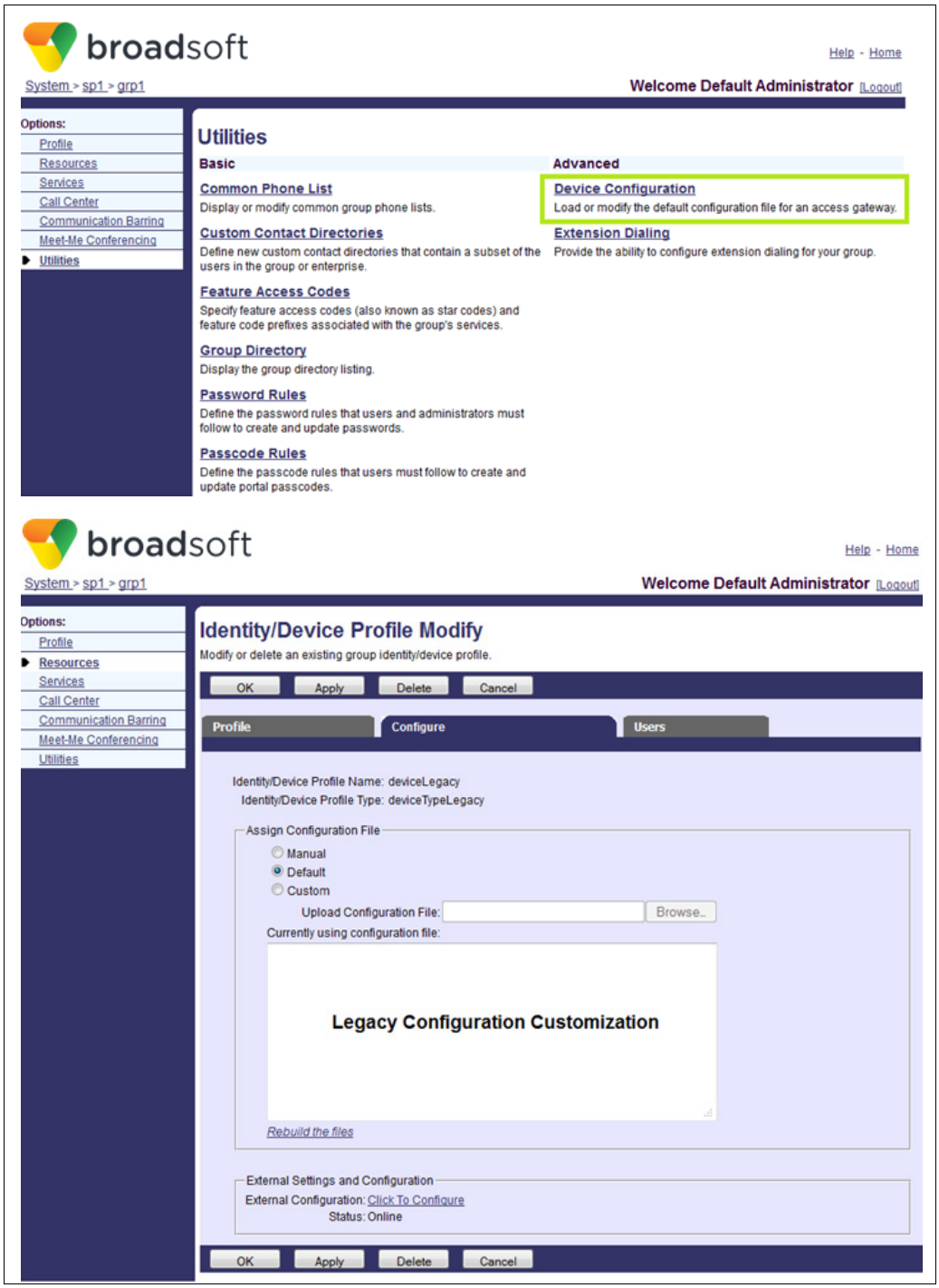
- **<deviceName>** is the name of the device profile assigned with this user.
- **<fileFormat>** represents the file name known by the device. For example, if a device requires a file called *polycomSystemFile.cfg*, the name requested by a device must be in this field.
- **<fileSource>** (see the <*fileSource*> parameter above).
- **<configurationFileName>** is the local path on the server indicating where to pick up the file that customizes the group file.

When using the CLI and adding new template files at the system level, or when customizing a template file at the service provider, group, or device profile levels, the source template file shall be located under the following directories (or under a subdirectory located under the following directories):

- /var/broadworks/ocifiles/
- /bw/broadworks/ocifiles/
- /var/broadworks/IpDeviceConfig/tmp/
- custom directory

To specify a custom directory, the container option bw.dms.customFileDirectory shall be set to the directory path where the file is located. This container option provides the ability to specify a supported directory path for local file inclusion in DeviceManagement filemodify transactions. If defined, the supported directories include the container-specified directory and default directory paths. If undefined, only the default directory paths previously listed are supported.

From the web, a file customization at the group level for a legacy or a normal file would look similar to *Figure 36*.



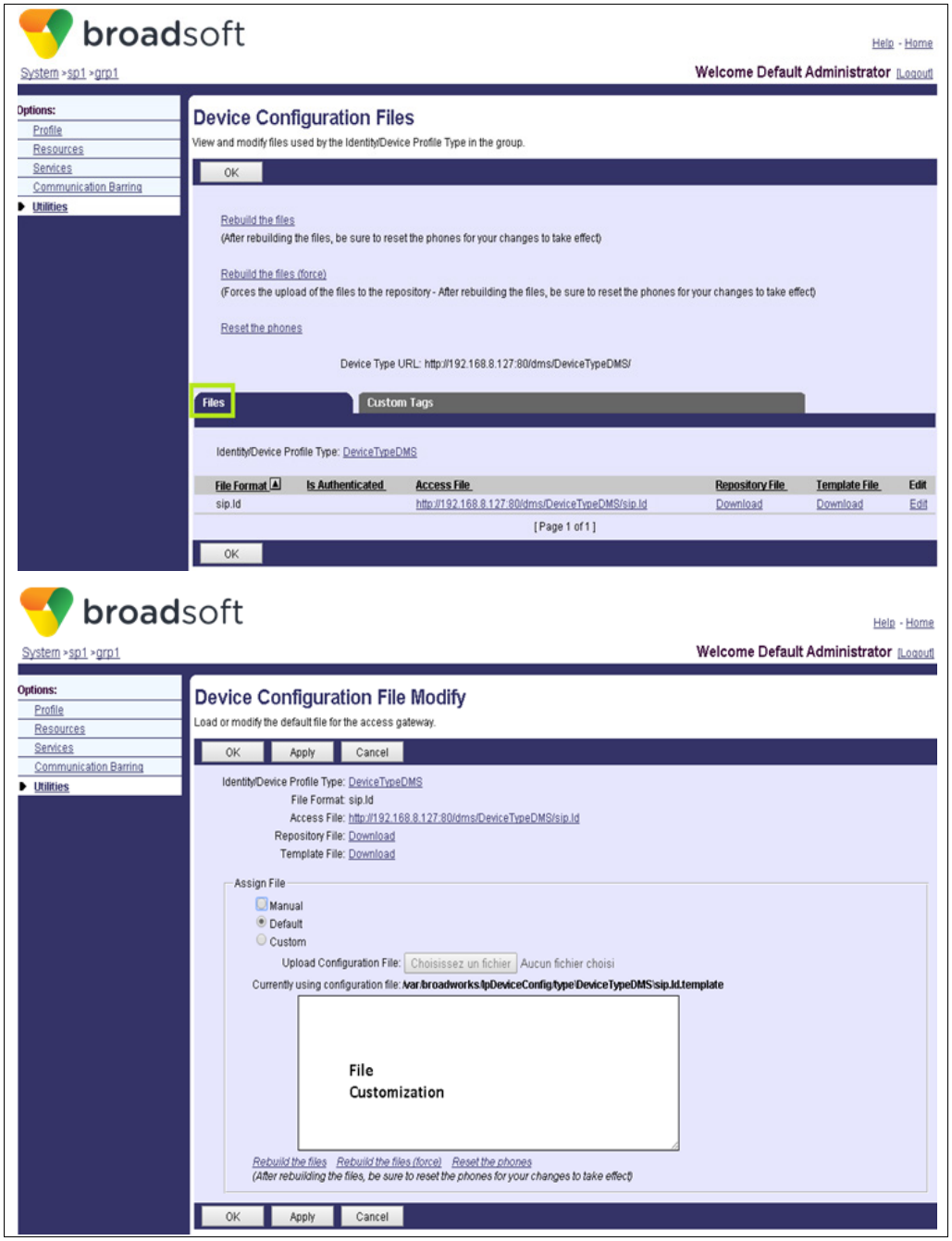


Figure 36 Customizing Legacy and Normal Files at Group Level

NOTE: System and service provider devices cannot customize their configuration templates.

5.7 Generate File

When a new template is provisioned or a configuration change affects it, the Application Server regenerates the associated file and pushes it on the file repository. Sometimes you may want to force a rebuild. Perhaps you accidentally deleted a file on the Profile Server or you changed to a new FTP server. The following shows how to manually force a file generation, that is, force the Application Server to resolve the template tags with the actual values, and upload the file to the file repository.

From the CLI:

```
AS_CLI/System/Device/IpDeviceMgmt>rebuilddefaultfile all
AS_CLI/System/Device/IpDeviceMgmt>rebuilddefaultfile system <deviceType>
AS_CLI/System/Device/IpDeviceMgmt>rebuilddefaultfile group
<serviceProvider> <group> <deviceType>
```

As you can see from the last two commands, you can refine the files to a subset based on the service provider, the group, and the device type. This avoids a costly rebuild of all the files for all the device types.

Table 2 Automatic versus Manual Generation of Template Files

Event Type	Potential Impacted Files	Event Generation
User change	Device files for all the devices for which the user owns a port	Automatic
Group change	Device files for all the devices for which users of the group own a port	Automatic
Service provider or enterprise change	Device files for all the devices for which users of the service provider (or enterprise) own a port	Automatic
Device profile change	Device files for the device profile being changed	Automatic
Device type change	Device files for all the devices of this type	Automatic
System default tag set changes	All device files in the system	Manual
Common tag set changes	All device files in the system	Manual

NOTE: It is possible to configure the Application Server to always push files to the file repository, even if the contents of the file have not changed since it was last pushed. This is accomplished through the system parameter *AS_CLI/System/Device/IpDeviceMgmt/alwaysPushFilesOnRebuild*.

5.7.1 Force Rebuild from CLI

The configuration files are pushed to the Profile Server when the configuration file has been modified. To bypass this, a “force” attribute can be used with the *rebuildDefaultFile* command. When present, the templates are uploaded to the repository even if the content has not changed since the last rebuild was performed. For example, to rebuild all files including the ones that were not changed, use the following CLI command:

- *AS_CLI/System/Device/IpDeviceMgmt>rebuildDefaultFile all force*

A force rebuild of the files for a particular device profile is also possible. Use the following commands:

For a device profile defined at the system level (Device1 as example):

- *AS_CLI/SubscriberMgmt/Device/rebuildDefaultFile device1 force*

For a device profile defined at the service provider level (Device 2 as example):

- *AS_CLI/SubscriberMgmt/ServiceProvider/Device/rebuildDefaultFile sp1 Device2 force*

For a device profile defined at the group level (Device3 as example):

- *AS_CLI/SubscriberMgmt/Group/Device/rebuildDefaultFile sp1 grp1 Device3 force*

5.7.2 Force Rebuild from Web Portal

The web portal also provides the capability to issue a rebuild force of the configuration template files by providing a link to trigger a device rebuild FORCE. This functionality is only available to system administrators.

The following will display the web pages from which the functionality can be obtained on the web portal.

System Identity/Device Profile Type:

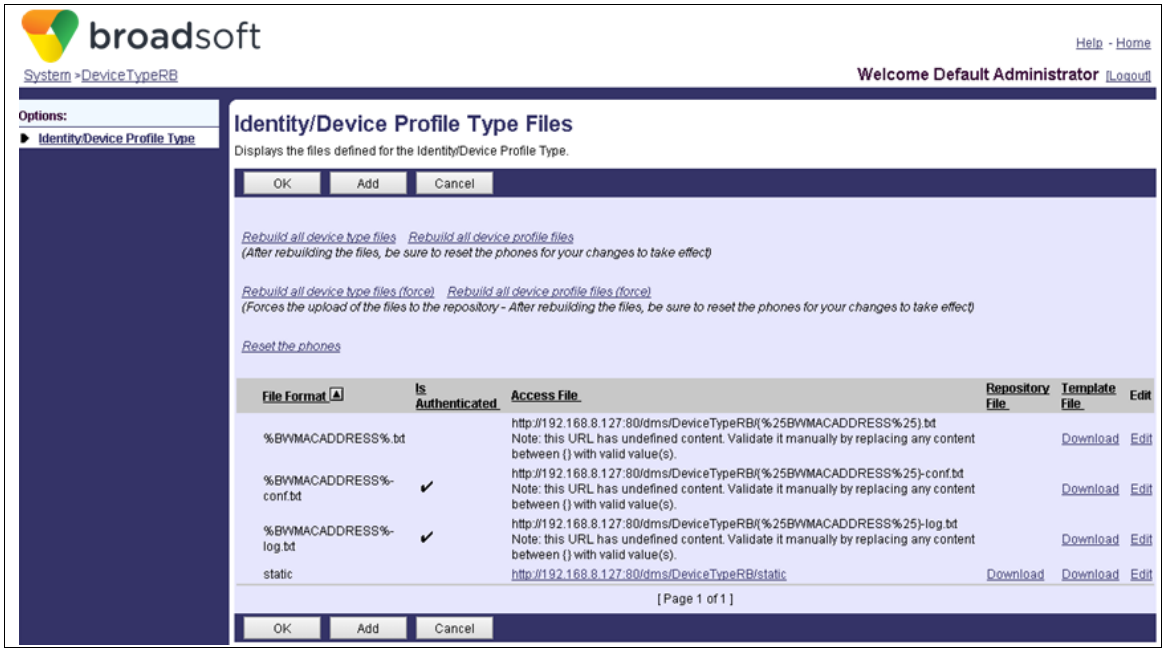


Figure 37 System Identity/Device Profile Type Files Web Page

The page has two links called *Rebuild all device type files (force)* and *Rebuild all device profile files (force)*. These two links allow the system administrator to force a rebuild of the configuration files even if the file contents have not changed.

System Identity/Device Profile Modify:

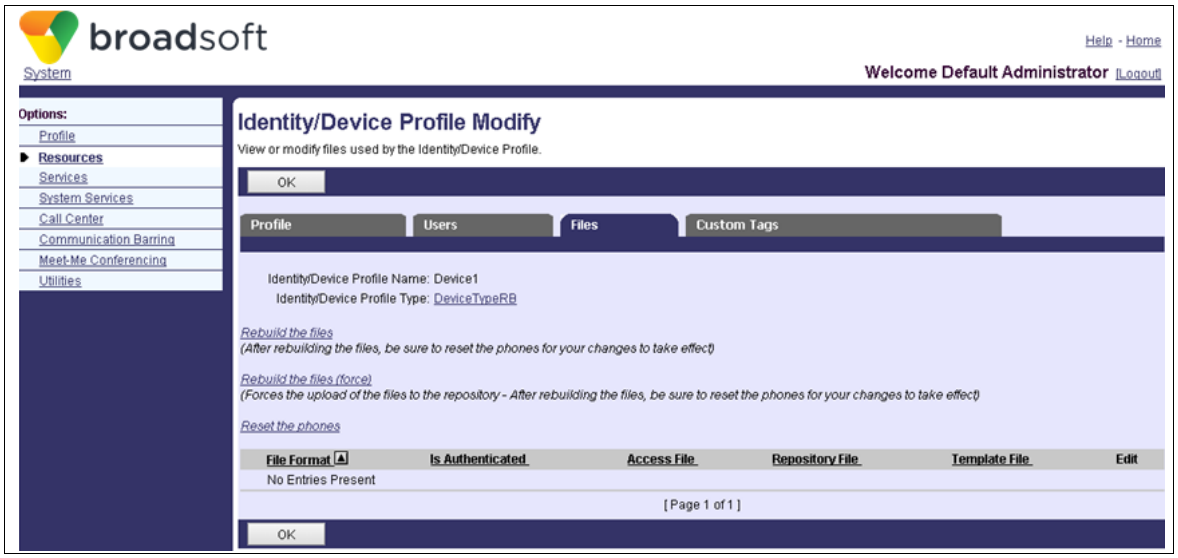


Figure 38 System Identity/Device Profile Modify Web Page (Files Tab)

The page has a link called *Rebuild the files (force)*. This link allows a system administrator to force a rebuild of the device configuration files even if the file contents have not changed.

System Identity/Device Profile Modify:

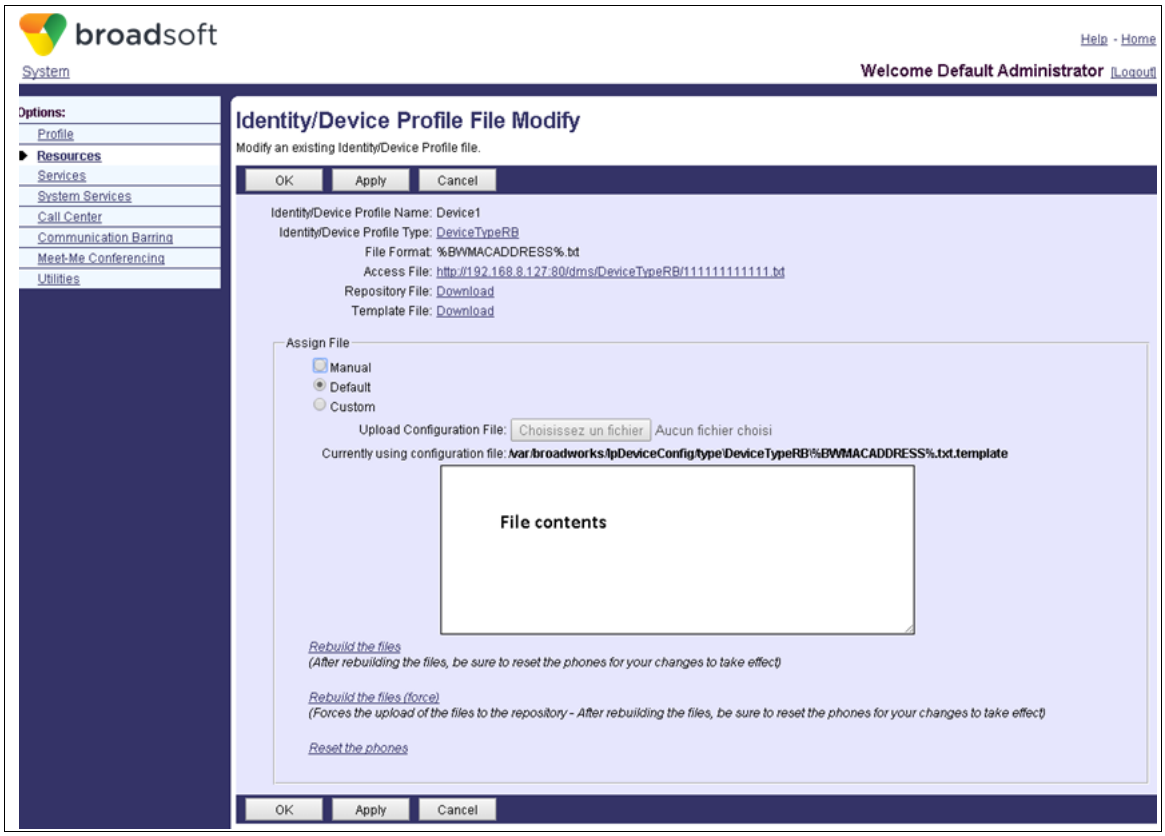


Figure 39 System Identity/Device Profile Modify Web Page (Edit File)

The page has a link called *Rebuild the files (force)*. This link allows a system administrator to force a rebuild of the device configuration files even if the file contents have not changed.

Service Provider Identity/Device Profile Modify:

Figure 40 Service Provider Identity/Device Profile Modify Web Page (Files Tab)

The page has a link called *Rebuild the files (force)*. It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link.

Service Provider Identity/Device Profile Modify:

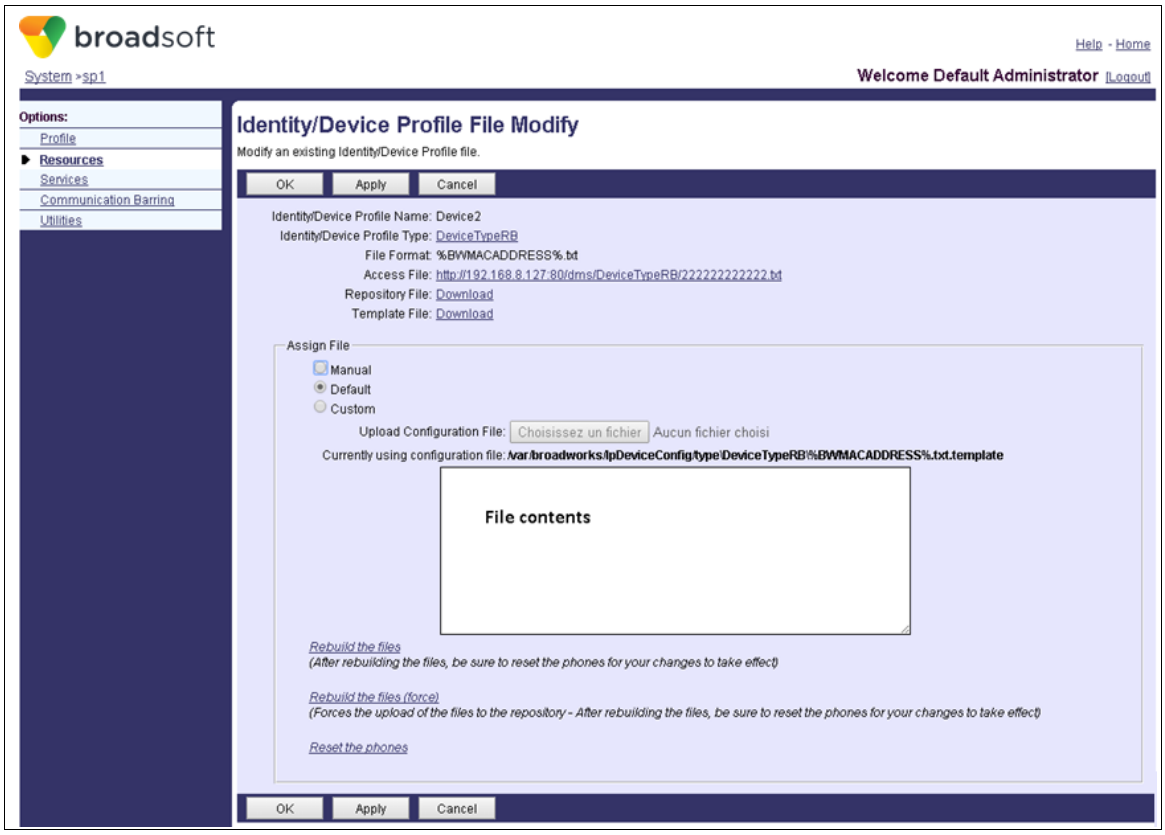


Figure 41 Service Provider Identity/Device Profile Modify Web Page (Edit File)

The page has a link called *Rebuild the files (force)*. It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link.

Service Provider Device Configuration Files:

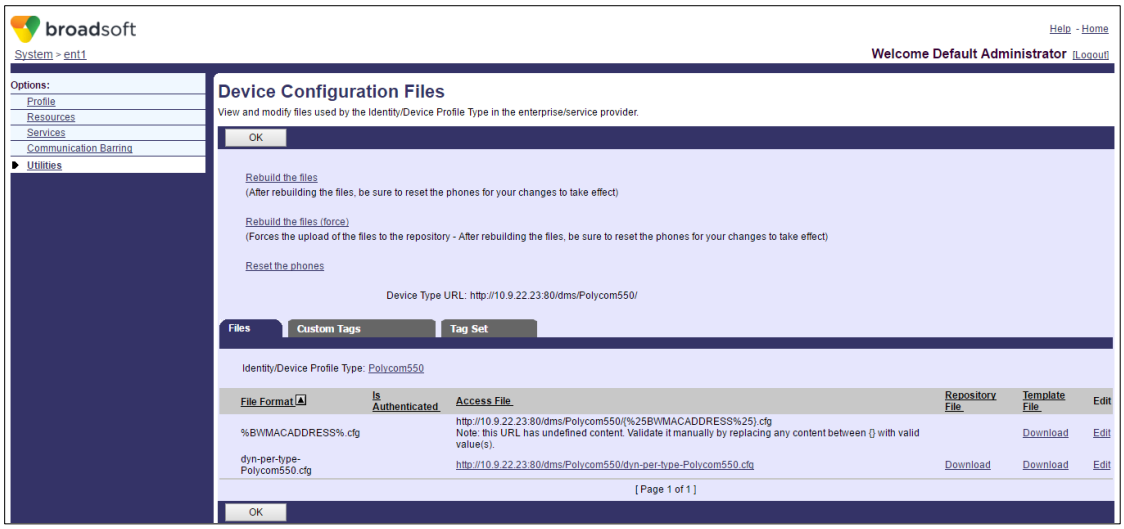


Figure 42 Service Provider Device Configuration Files Web Page

The page has a link called *Rebuild the files (force)*. It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link. Note that the same force functionality is also available with the other two tabs part of this web page (Custom Tags and Tag Set).

Group Identity/Device Profile Modify:

Figure 43 Group Identity/Device Profile Modify Web Page (Files Tab)

The page has a link called *Rebuild the files (force)*. It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link.

Group Identity/Device Profile Modify:

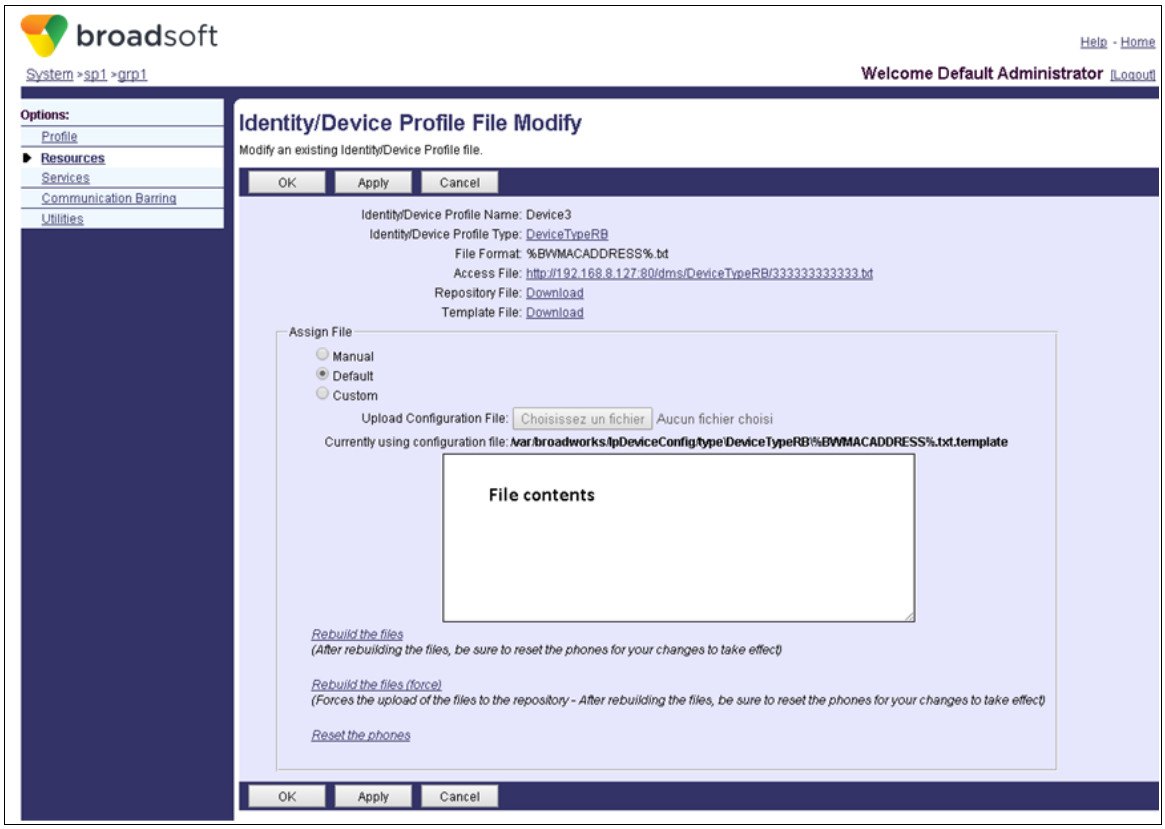


Figure 44 Group Identity/Device Profile Modify Web Page (Edit File)

The page has a link called *Rebuild the files (force)*. It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link.

Group Device Configuration Files:

Figure 45 Group Device Configuration Web Page

The page has a link called *Rebuild the files (force).* It allows to force a rebuild of the device configuration files even if the file contents have not changed. Only system administrators can see this link. Note that the same force functionality is also available with the other two tabs that are part of this web page (Custom Tags and Tag Set).

5.7.3 Automatic Rebuild Configuration

The CLI allows a system administrator to disable the generation of automatic Device Management events via an automatic rebuild configuration list.

This is controlled through the CLI level *AS_CLI/System/Device/IpDeviceMgmt/AutoRebuildConfig.*

The automatic rebuild configuration list is statically populated and no entries can be added or removed from the table. Automatic generation of Device Management events can only be enabled or disabled. On a new install or an upgrade, it is populated with OCI request prefixes.

The usage of the list is controlled with *AS_CLI/System/Device/IpDeviceMgmt /enableAutoRebuildConfig* system parameter flag. If the flag is set to “true”, the list is used. Otherwise, the list is not used and the requests trigger rebuilds events automatically. Changes made to this parameter do not require a system restart.

5.8 Upload File to File Repository

Cisco BroadWorks can be configured to support the upload of any file to the file repository. For example, when a Polycom device boots, a log is generated and the device tries to HTTP PUT this file to Cisco BroadWorks. For security reasons, uploads are refused by default.

To enable the support for a given file, three steps are required. First, a file must be of File Category; Dynamic Per-Device. Second, the file must be allowed for upload (check box). Third, one of the authentication methods must be chosen.

From the web:

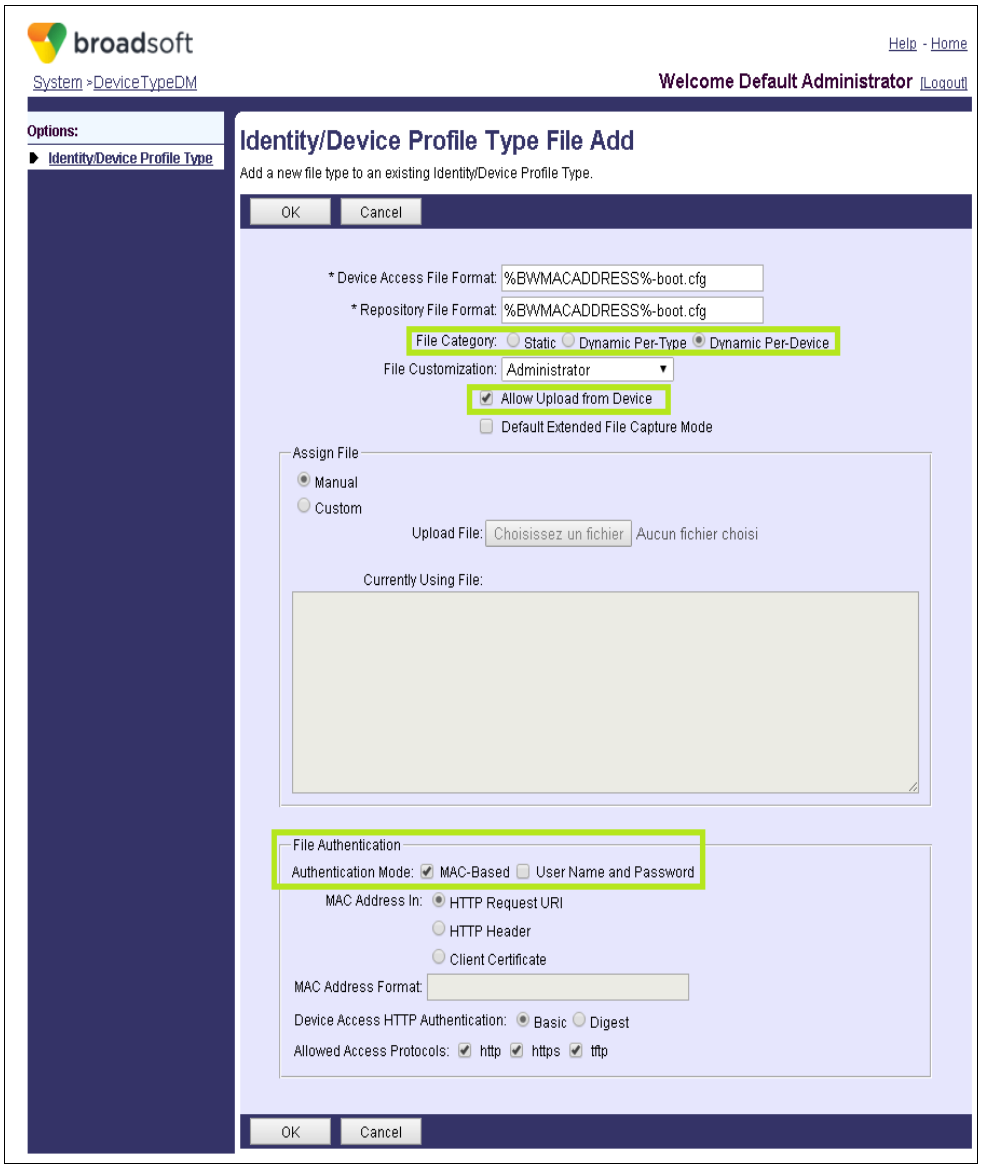


Figure 46 Allowing a File to be Uploaded to File Repository

5.9 Extended File Capture Mode

This functionality allows a number of uploads to be kept for a given file (that can be uploaded) for a device under Device Management. Each device type file stored under Device Management has a property to allow more than one version of a file to be uploaded from a device. The number of files to be kept is configurable on the Profile Server and can vary between two to 100 files; this configuration is global for the system.

The configuration required for the extended file capture mode on a given device type file is as follows:

1) Deploy the extended file repository web application on a Profile Server.

2) Provision the extended file repository on the Application Server under CLI level *AS_CLI/System/Device/FileRepos*.

```
AS_CLI/System/Device/FileRepos> add 192.168.13.46_EXT webdav
192.168.13.46 false true
…Done
AS_CLI/System/Device/FileRepos> get
             Name    Protocol  Root Directory
Extended File Capture Support

=========================================================================
192.168.13.46 webdav                                                  /
false
192.168.13.46_EXT  webdav                   /extendedDir
true
```

3) Provision the user for the file repository.

```
AS_CLI/System/Device/FileRepos/Users> add 192.168.13.46_EXT admin
put,delete,get
Initial Password:
Re-type Initial Password:
...Done
```

4) Associate the extended file repository to the device type.

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> set Polycom-550_0
extendedCaptureFileReposName 192.168.13.46_EXT
...Done
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> get
Device Type File Repository Name Extended Capture File Repository Name
Directory

=========================================================================
==
Polycom-550_0             192.168.13.46
192.168.13.46_EXT                       Polycom-550_0
```

5) For the device type (either when creating or modifying it), make sure that the file has the desired value for new instances of the file being created.

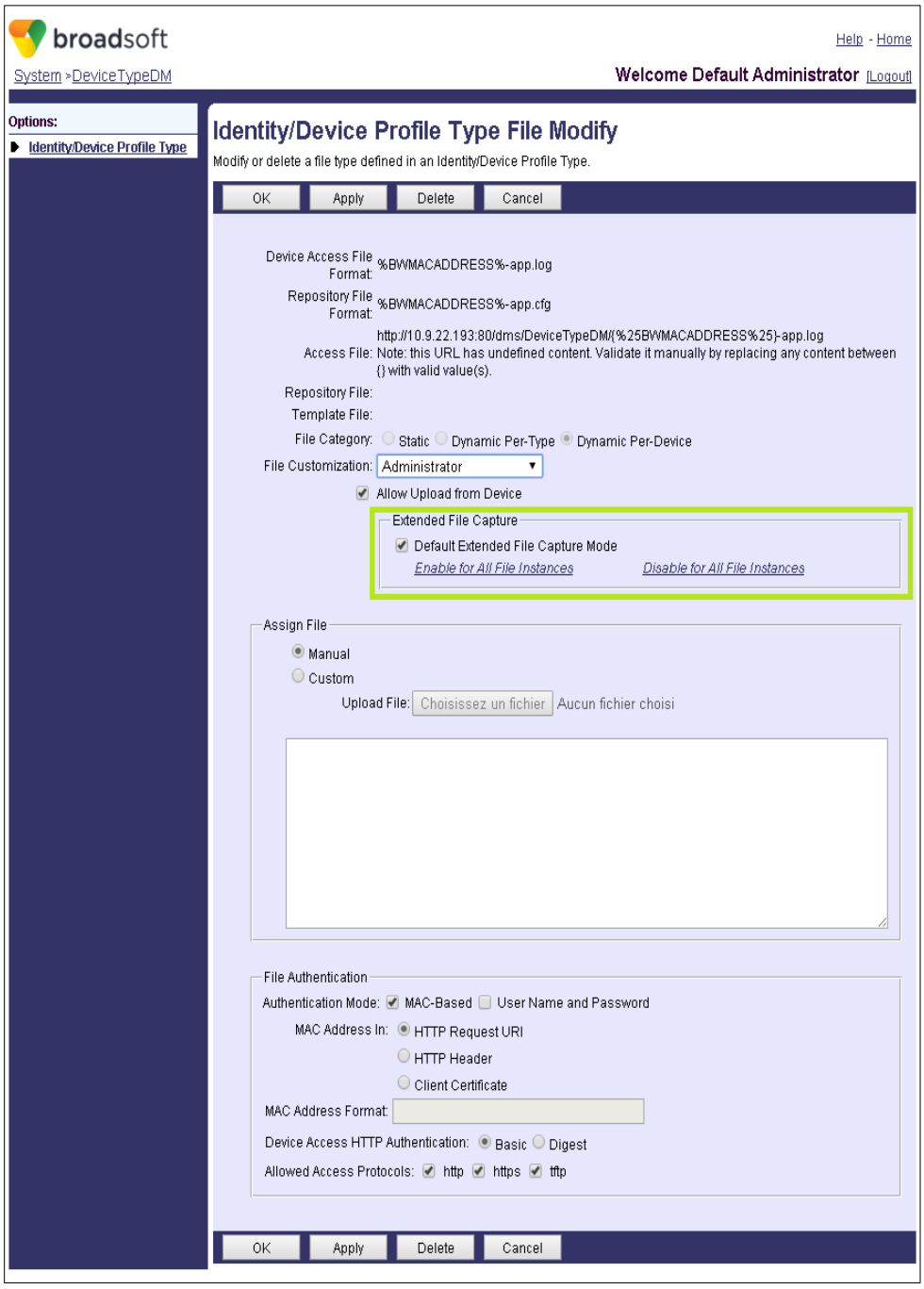


Figure 47 Device Type File Options for Extended File Capture Mode

Note that at this level, the check box only controls the value to be assigned to newly created instances of the file. Therefore, modifying the value of the check box never interferes with the value of the existing file instances.

To modify the value of all file instances, two links are provided: *Enable for All File Instances* and *Disable for All File Instances*.

6) Optionally, the option could be set on a single file instance of the device profile file (at either the system, service provider/enterprise, or group level).

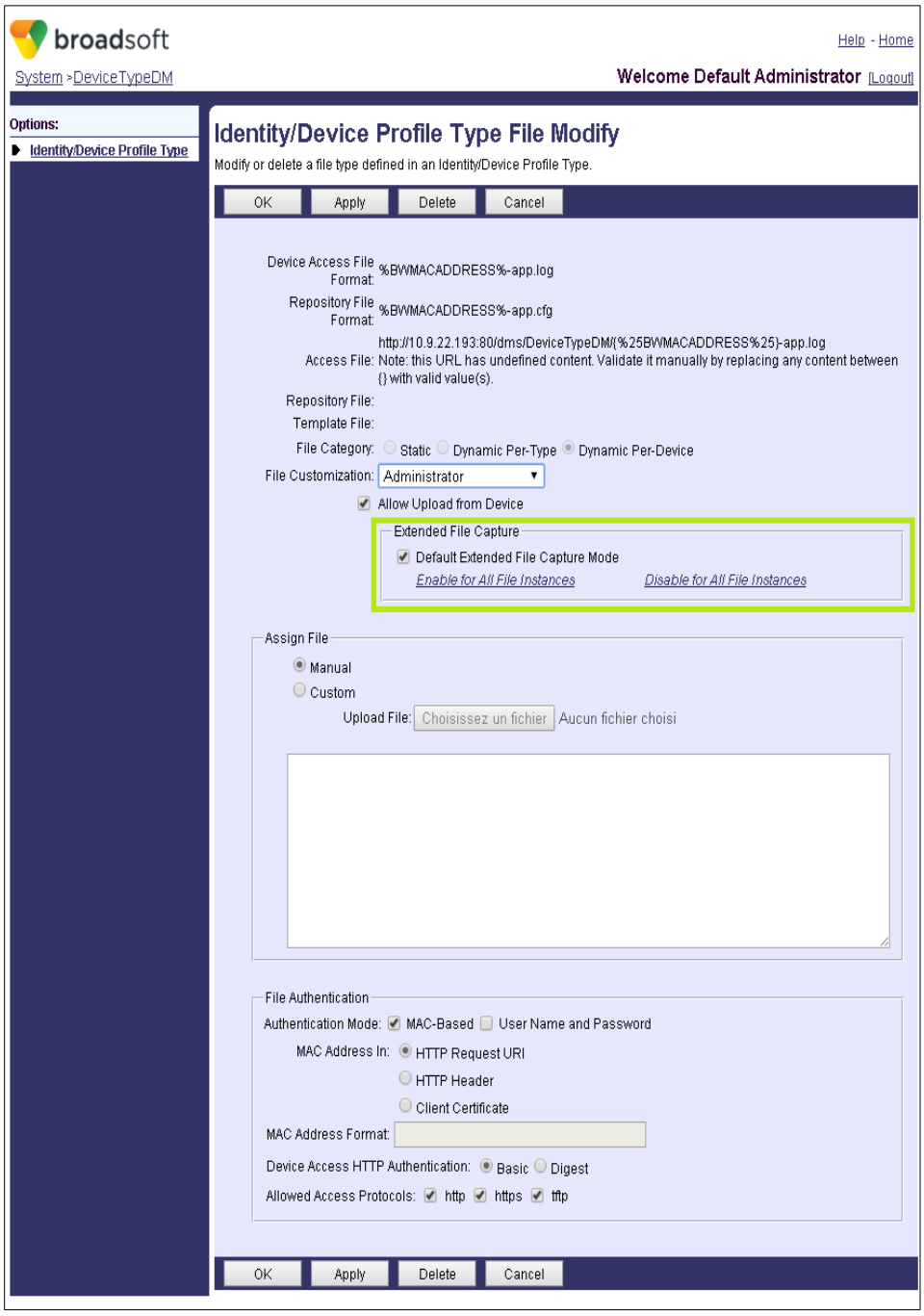


Figure 48 Device Profile File Modify for Extended File Capture

5.10 Set Device Language

The way the Provisioning Server defines languages is not the same as the way a device type defines its supported languages. In fact, there are many ways to define a language. Therefore, the feature introduces a new web page (and related Open Client Interface and CLI commands) at the device-type level to allow an administrator to map Provisioning Server languages to device-type-specific languages.

From the CLI:

```
AS_CLI/System/DeviceType/SIP/Languages>set <deviceType>
<broadworksLanguage> <deviceLanguage>
```

… where

- **<deviceType>** is the name of the device type for which the language is set.
- **<broadworksLanguage>** contains the Cisco BroadWorks name for the chosen language.
- **<deviceLanguage>** represents the equivalent language as understood by the device.

For example, if English is called “english” in the Provisioning Server, the administrator can map this to “en” for a device type. If a Provisioning Server language does not have a supported equivalent, the administrator can choose another language that the device understands. The administrator can also leave this field empty. In this case, the %BWLANGUAGE-x% variable is mapped to the provisioned device-type default language string.

From the web:

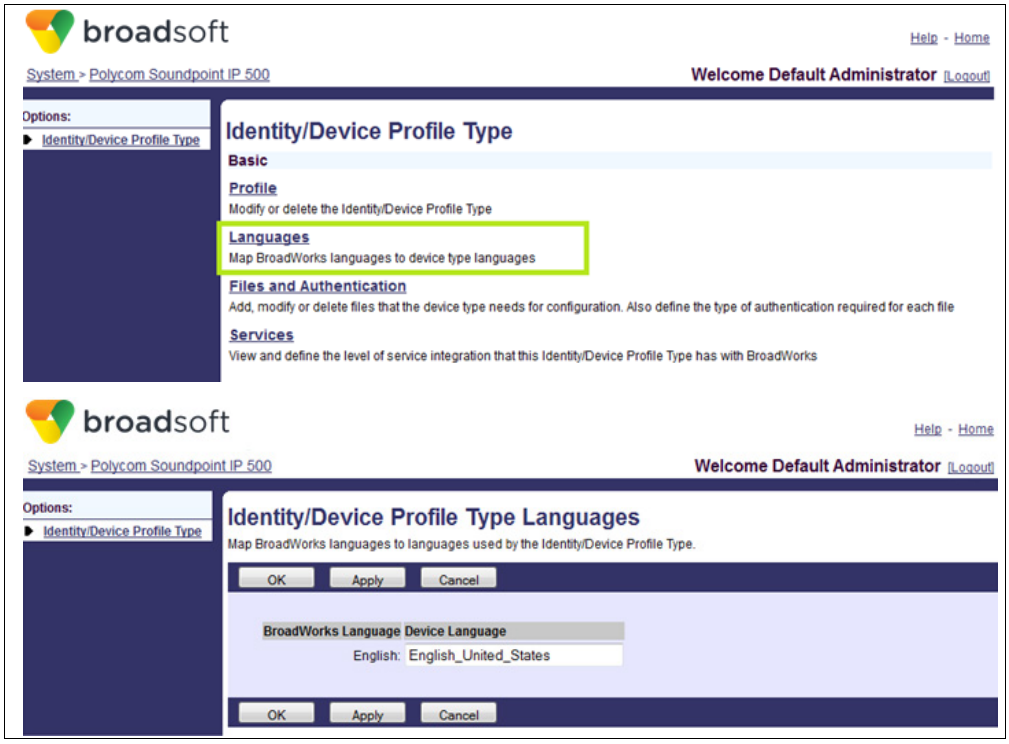


Figure 49 Associate Cisco BroadWorks to Device Language

5.11 Set Time Zone

To support time zones, there must be a mapping between the Provisioning Server time zones and device-type time zones because devices use different ways to define time zones. As is currently done for the presentation of time zones in different languages, the feature uses a file to provide a mapping from Provisioning Server time zone IDs to device-type-specific time zone strings. There is a one-time zone-mapping file per device type. These files are stored in */usr/local/broadworks/bw_base/conf/dms/* and the file name format is:

TimeZoneAliasLabels_<URL-encoded device type name>.properties

Note that these characters cannot be encoded: a-z, A-Z, 0-9, "*", "-", and "_". However, all remaining characters can be encoded.

If the device type name contains ".", it must be URL encoded. The encoded value is "%2E".

Also note that if the device type name contains spaces " ", it must be URL encoded as well. The encoded value is "+".

NOTE: The URL encoding is done with regards to the HTML Specifications, which explains why the spaces are encoded to "+" and not "%20". For example, for the device type name "Polycom.ACD%.IND", the file name would be *TimeZoneAliasLabels_Polycom%2EACD%25%2EIND.properties.* The following is an example of file content for the file *TimeZoneAliasLabels_Polycom+Soundpoin+IP+500.properties*.

```
CANADA_PACIFIC_TIME=-8
US_PACIFIC_TIME=-8
CANADA_MOUNTAIN_TIME=-7
US_MOUNTAIN_TIME=-7
CANADA_CENTRAL_TIME=-6
US_CENTRAL_TIME=-6
CANADA_EASTERN_TIME=-5
US_EASTERN_TIME=-5
CANADA_ALTANTIC_TIME=-4
CANADA_NEWFOUNDLAND=-3.5
```

The feature offers no specific interface to create, edit, or manage these time zone mapping files. The administrator must manually edit them when device types are created or deleted. The files in */usr/local/broadworks/bw_base/conf/dms/* are replicated to other Provisioning Servers in the cluster using rsync.

NOTE: The Application Server must be restarted for a time zone file to be picked up by the server.

5.12 Primary User

Currently, only the Polycom Phone Services uses the primary user settings to select the user by which the device's directory file is built. The administrator can change the primary user on a device as shown in *Figure 50 Specifying Primary Line for User of Device Profile*. However, viewing and setting the primary user on devices is a generic functionality that simply has no effect when the device type does not support the Polycom Phone Services.

Reordering the lines on a device does not change the primary user setting.

Setting and unsetting the Polycom Phone Services support on a device type does not change the primary user setting on the device-type devices; however, the Polycom Phone Services files cease to be generated for these devices.

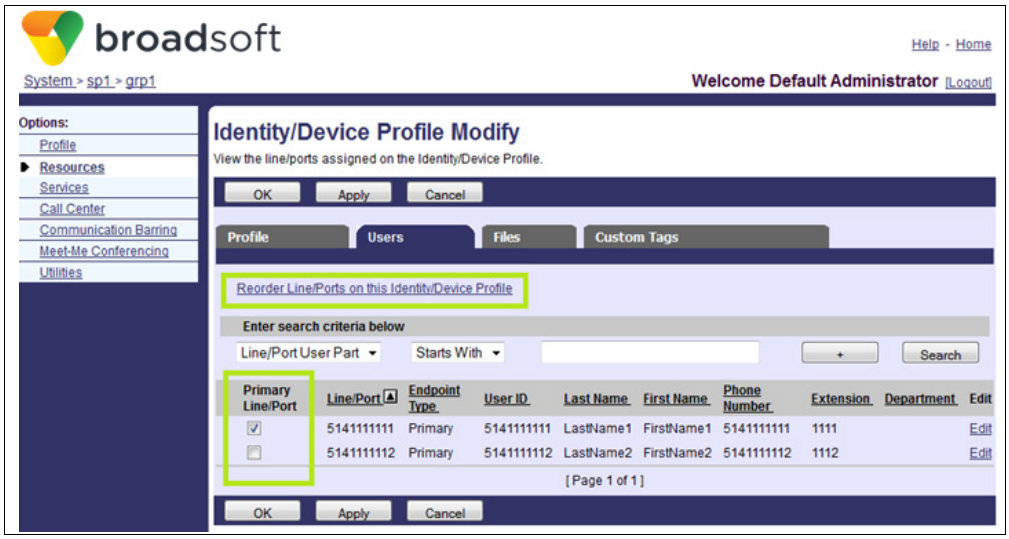


Figure 50 Specifying Primary Line for User of Device Profile

Figure 51 Reordering Lines at System, Service Provider, or Group Level

5.13 Use Tags and Tag Sets

5.13.1 Overview

Service integration on Cisco BroadWorks is based on the concept of “Tags”. Tags are variables that can be embedded in the configuration template files. When Cisco BroadWorks generates a configuration file from a configuration template, the tags are replaced with actual values. Tags are delimited with a beginning and ending % sign.

There are two types of tags:

- Dynamic Built-in Tags – These tags are predefined by Cisco. The value of each built-in tag is dynamically evaluated based on the context of the device profile. Depending on the services assigned to users of the device and the values of various service attributes, a built-in tag for one device evaluates differently from a built-in tag for another device. Built-in tags provide rich service integration with Cisco BroadWorks. There are over 100 built-in tags defined on the system. All built-in tags are prefixed with “BW”.

- Static Tags – These tags are defined by the operator when building the device profile type. The value of each static tag is assigned by the operator.

To simplify the management of static tags, the operator can define "Tag Sets" at the system level. Tag sets allow the operator to define a list of static tags that can be re-used across device profile types. Note that in AS mode, the Reseller level allows to create tag sets as well.

There is one "System Default Tag Set" that is initially empty. The operator can populate the System Default Tag Set with tags that are common to all device types, such as domain names, DNS servers, session border controller addresses, and so on.

The operator can define any number of additional tag sets. For every device profile type, the administrator can indicate if the "System Default Tag Set" applies, and can select one additional tag set to be applied to the device profile type.

Individual static tags can be created or customized for a device profile type at the group level, and for a device profile at the service provider, group, and device profile levels. By default these capabilities are disabled and must be enabled separately for each device profile type.

The users assigned to a device are classified into two separate types of users:

- Standard users – These are users assigned to a group.
- Trunk group users – These are users (including pilot users) who belong to a trunk group.

Each type of user has its own set of tags as follows:

- Device Management tags (BWDN, BWNAME, BWLINEPORTHOST, and so on) are used for resolving user data associated with Cisco BroadWorks standard users.
- Trunk group user's tags that resolve data for users assigned to the device, which are part of a trunk group.

NOTE 1: Only AS mode allows trunk group users. Therefore, the tags for trunk group users, although available in both modes, are only resolved in AS mode.

NOTE 2: When the static tag is contained within an xml file, certain characters used in the tag's value are xml-encoded when the tag is resolved. These characters are &, <, >, /, and ".

5.13.1.1 Line Number Correlation

Device Management currently supports tags having a suffix "-x", where "x" is an integer between 1 through 1024 that specifies the line port number in the device.

This line port number does not apply to trunk group users as such users do not consume a port; therefore, there is no correlation between the x value used for a standard user and a trunk group user.

The x value used for trunk group users is dynamically maintained by Device Management. The first trunk group user in the list of trunk group users assigned to the device is automatically assigned the x value of "1", the second user is assigned the x value of "2", and so on. This allows, for example, pulling the relevant DN, line/port, and Calling Line ID values based on the users' position in this list.

For example, the tags for a device with five regular users and with ten trunk users have:

- Tags for regular users have the format BWTAGNAME-1 … BWTAGNAME-5.
- Tags for trunk group users have the format BWTRUNKTAGNAME-1 … BWTRUNKTAGNAME-10.
- The values of BWTAGNAME-x and BWTRUNKTAGNAME-x do not denote the same user/line port even though their "-x" value is the same.
- The ordering of trunk group users is always dynamic whereas the ordering of regular users can be either dynamic or static.

5.13.1.2 Encrypted Custom Tags

NOTE: Only XS mode supports encrypted custom tags. Therefore, all mentioned about encrypted custom tags in this document, all CLI commands, OCI-P commands, and web pages concerning encrypted custom tags do not apply to AS mode.

Also take note that encrypted tags are only available to a system administrator. Meaning that encrypted tags entries do not appear in the list of tags when shown to other administrators and similarly, the column *Is Encrypted* is not shown to other administrators. Encrypted tag option in the CLI and in related OCI-P commands can only be used by a system administrator.

Custom tag values are stored in clear text within Cisco BroadWorks. This may bring some concerns, especially when custom tags contain sensitive data to populate into Device Management (DM) configuration files.

Therefore, DM offers the option to encrypt their values. An encrypted custom tag value does not appear in clear text in logs, web portal, and CLI, and is stored encrypted in the database.

The value is encrypted with a 128-bit AES/CBC/PKCS5Padding key that is generated at installation time. The Java KeyStore functionality is used for securely storing the encryption keys.

DM decrypts the value of an encrypted custom tag when the tag is present in a Device Management template file and populates the value into the resulting configuration file.

5.13.2 Use of Tags in Templates

A template file is a configuration file that contains one or more variables identified by a keyword starting with % and ending with % (for example, %BWMACADDRESS%). These variables are called "tags". A tag name is case-sensitive. The tags are replaced with actual contextual values (for example, replace %BWCLID-1% with 5143331234) before the file is sent to the device. If a device file already includes a % character that is not used for tags and there is no other % character on the rest of the line, it is ignored as the start of a tag, it can also be escaped with a second %.

For example:

```
<template>
<value>MyValue%WithEmbeddedPercent</value>
</template>
```

…can be escaped as follows:

```
<template>
<value>MyValue%%WithEmbeddedPercent</value>
```

```
</template>
```

NOTE: For definitions and tag locations in the system, see the *Cisco BroadWorks Device Management Tag Reference Guide* [11].

5.13.3 Use of Tags in File Names

As seen previously, tags can be used in a template file destined to a device. However, tags can also be used in the file name of the template itself. *Table 3* lists the tags to be used in template file names.

Example:

/directoryExample/%BWDEVICEID%_myFileExample.cfg

…can be resolved to:

/directoryExample/MyProfileNameExample_myFileExample.cfg

Table 3 List of Tags used in Template File Names

Type of File	Device Access File Name	File Name on File Repository
Static	Can be a path followed by a name or just a name. The path and/or name can contain the following variables: %BWLOGIN-ID-x% %BWTIMEZONE-x% %BWLANGUAGE-x% %BWDEPARTMENT-x% %BWGROUP-x% %BWGROUPID-x% %BWENTERPRISE-x% %BWENTERPRISEID-x% %BWSERVICEPROVIDER-x% %BWSERVICEPROVIDERID-x% %BWDEVICEUSERNAME% %BWMACADDRESS% %BWMACADDRESSUPPER% %DEVICE_FILE% %SYSTEM_FILE% %BWSERVERADDRESS% %BWSOFTWARELOAD% %BWFILESERVERLOCATION% %BWFILESERVERDIR% %BWDEVICEID% %BWFQDEVICEID%	Can be a path followed by a name or just a name. The path and/or name cannot contain variables (they are not resolved).

Type of File	Device Access File Name	File Name on File Repository
Dynamic per device type (customizable per group)	Can be a path followed by a name or just a name. The path and/or name can contain the following variables: %BWLOGIN-ID-x% %BWTIMEZONE-x% %BWLANGUAGE-x% %BWDEPARTMENT-x% %BWGROUP-x% %BWGROUPID-x% %BWENTERPRISE-x% %BWENTERPRISEID-x% %BWSERVICEPROVIDER-x% %BWSERVICEPROVIDERID-x% %BWDEVICEUSERNAME% %BWMACADDRESS% %BWMACADDRESSUPPER% %DEVICE_FILE% %SYSTEM_FILE% **%BWSERVERADRESS%** **%BWSOFTWARWELOAD%** **%BWFILESERVERLOCATION%** **%BWFILESERVERDIR%** %BWDEVICEID% %BWFQDEVICEID% **NOTE**: Some of these tags (the ones not in bold) can only be resolved when the file has been overwritten through *Utilities – Device Configuration*. Otherwise, they appear unresolved in their template format (for example, %BWGROUPID%).	Can be a path followed by a name or just a name. The path and/or name can contain the following variables: %BWLANGUAGE-x% %BWGROUP-x% %BWGROUPID-x% %BWENTERPRISE-x% %BWENTERPRISEID-x% %BWSERVICEPROVIDER-x% %BWSERVICEPROVIDERID-x% **NOTE**: These tags can only be resolved when the file has been overwritten through *Utilities – Device Configuration*. Otherwise, they appear unresolved in their template format (for example, %BWGROUPID%).

Type of File	Device Access File Name	File Name on File Repository
Dynamic per device profile	Can be a path followed by a name or just a name. The path and/or name can contain the following variables: %BWLOGIN-ID-x% %BWTIMEZONE-x% %BWLANGUAGE-x% %BWDEPARTMENT-x% %BWGROUP-x% %BWGROUPID-x% %BWENTERPRISE-x% %BWENTERPRISEID-x% %BWSERVICEPROVIDER-x% %BWSERVICEPROVIDERID-x% %BWDEVICEUSERNAME% %BWMACADDRESS% %BWMACADDRESSUPPER% %DEVICE_FILE% %SYSTEM_FILE% %BWSERVERADDRESS% %BWSOFTWARELOAD% %BWFILESERVERLOCATION% %BWFILESERVERDIR% %BWDEVICEID% %BWFQDEVICEID%	Can be a path followed by a name or just a name. The path and/or name can contain the following variables: %BWLOGIN-ID-x% %BWTIMEZONE-x% %BWLANGUAGE-x% %BWDEPARTMENT-x% %BWGROUP-x% %BWGROUPID-x% %BWENTERPRISE-x% %BWENTERPRISEID-x% %BWSERVICEPROVIDER-x% %BWSERVICEPROVIDERID-x% %BWDEVICEUSERNAME% %BWMACADDRESS% %BWMACADDRESSUPPER% %DEVICE_FILE% %SYSTEM_FILE% %BWSERVERADDRESS% %BWSOFTWARELOAD% %BWFILESERVERLOCATION% %BWFILESERVERDIR% %BWDEVICEID% %BWFQDEVICEID%

5.13.4 Use of Tags for Dynamic Per-Type Files

Some system tags, whether they are in the file name or in the file itself, cannot be resolved unless they are customized. The tag always shows in its original template format (such as %BWGROUP%) for the device files that are not overwritten, but is resolved in the customized files.

The tags that can be resolved at the device type level are tags that have no relationship with an enterprise, service provider, group, profile, or user (such as %BWTIMESTAMP% or %BWSERVERADDRESS%).

5.13.5 Ensure Unique File Repository Names using Remote File Format

For a given unique URI, it is useful to serve different configuration files depending on which device makes the request. It is possible that all devices try to grab a file of the same name that must be device-specific. This can be accomplished using authentication on the file and tags in the remote file format. When a device makes a request, the authentication allows Cisco BroadWorks to determine the exact profile that is being accessed. Using this profile, tags in the remote file name format are resolved at request time, resulting in different files being served for different profiles as shown in *Figure 52*.

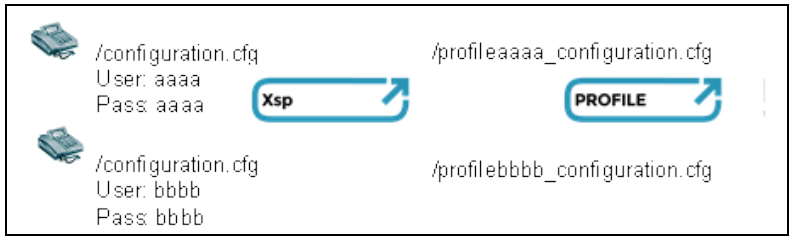


Figure 52 Requesting Same URI can Result in Different Files

For this scenario to take place, only one file is created for the device type. The file formats are defined as follows:

- Device access file format: *configuration.cfg*
- Repository file format: *%BWFQDEVICEID%_configuration.cfg*

5.13.6 Create Static Tags

The feature introduces a new type of variable “static tags” for device files. For each static tag, there is a corresponding value.

A static tag is a <key> = <value> pair, where:

- <key> is a string that starts with % and ends with % (with no other %). This is the name of the static tag. This name must be unique within a set and the name cannot start with %BW to avoid conflicts with Cisco BroadWorks tags. The name is case-sensitive.
- <value> is a string that replaces the static tag name when the tag name is used in a template. The value can be empty, in which case the static tag is replaced by an empty string.

Once a static tag is defined, it can be used in a template file.

Static tags can be created at the system level as part of a tag set which can then be assigned to a device type.

If a device type is configured to allow tags to be created for device profiles, tags can be added or customized for individual device profiles at the service provider, group, and device level.

If a device type is configured to allow tags to be created at the group level, individual static tags can be added or modified for a device type at the group level.

5.13.6.1 Create Tags at System Level (AS Mode Only)

You can either add tags to a system tag set (*systemTagSet*), which is always available, or create one or more new tag sets for your tags.

To create a new tag set from the CLI:

```
AS_CLI/System/DeviceTagSet>add <tagSetName>
```

… where:

- **<tagSetName>** is the name of the newly created tag set. If you do not want to create a new tag set, you can skip this step and use the *systemTagSet*, which is always available.

To create a static tag from the CLI:

```
AS_CLI/System/DeviceTagSet/Tags>add <tagSetName> <tagName> <tagValue>
```

… where:

- **<tagSetName>** is either *systemTagSet* or the name of the tag set that contains the new tag.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and not start by "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this group.

From the web:

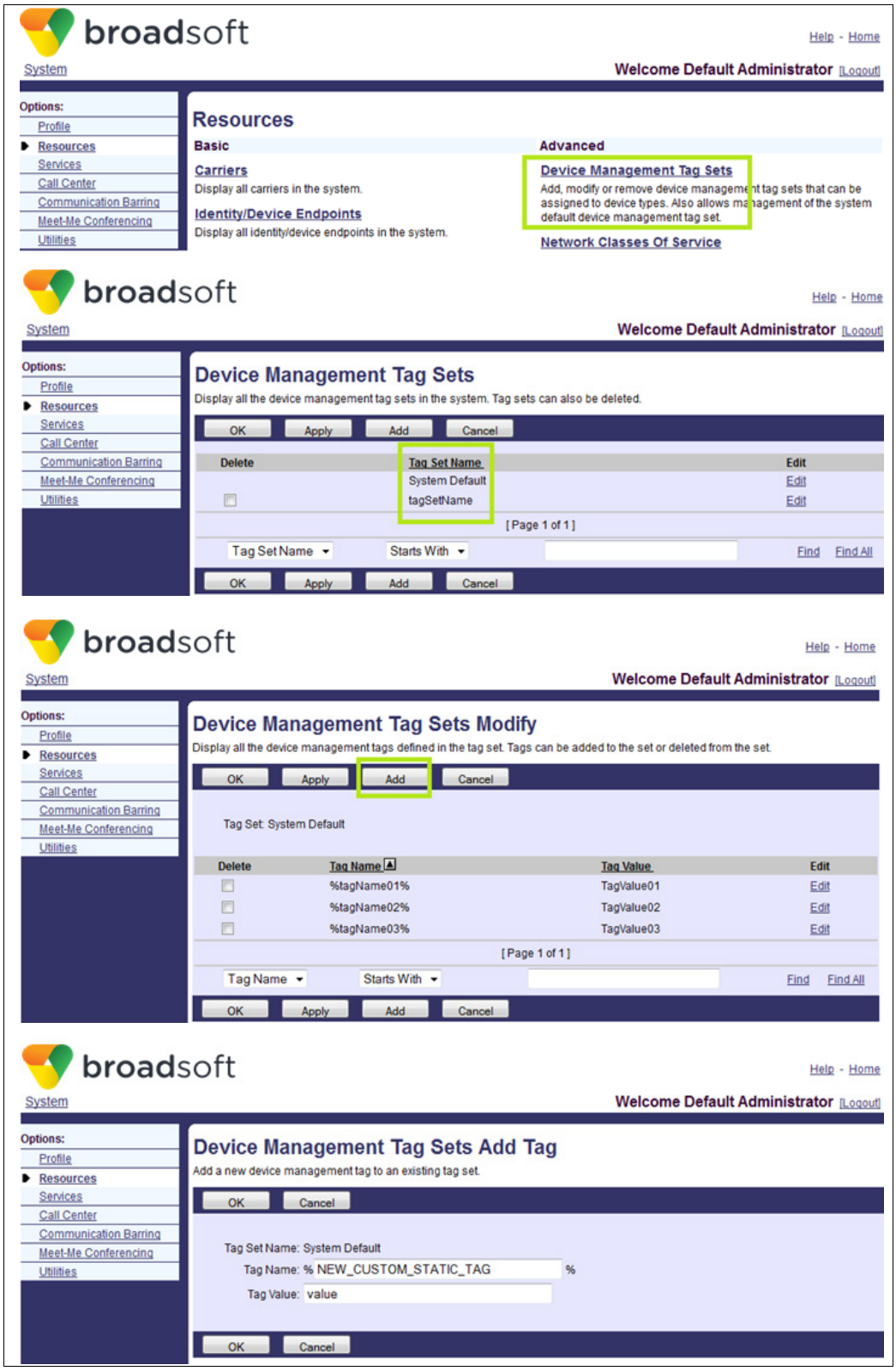


Figure 53 Managing Tag Sets

5.13.6.2 Create Tags for Device Profiles (AS Mode Only)

If a device type is configured to allow custom tags (that is, if the *allowDeviceProfileCustomTagSet* parameter is set to "true") then you can also create individual static tags for device profiles at the service provider, group, and device profile levels.

To create a tag at the service provider level from the CLI:

```
AS_CLI/SubscriberMgmt/ServiceProvider/Device/CustomTags>add <sp>
<deviceName> <tagName> <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and not start by "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this service provider.

To create a tag at the group level from the CLI:

```
AS_CLI/SubscriberMgmt/Group/Device/CustomTags>add <sp> <group>
<deviceName> <tagName> <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and not start by "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this group.

To create a tag at the profile level from the CLI:

```
AS_CLI/SubscriberMgmt/Device/CustomTags>add <deviceName> <tagName>
<tagValue>
```

… where:

- **<deviceName>** is the name of the profile for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and not start by "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this device profile.

From the web:

To add a tag at the system, service provider, or group level; access the *Identity/Device Profiles* menu item from the Resources menu page for the service provider, group, or system respectively.

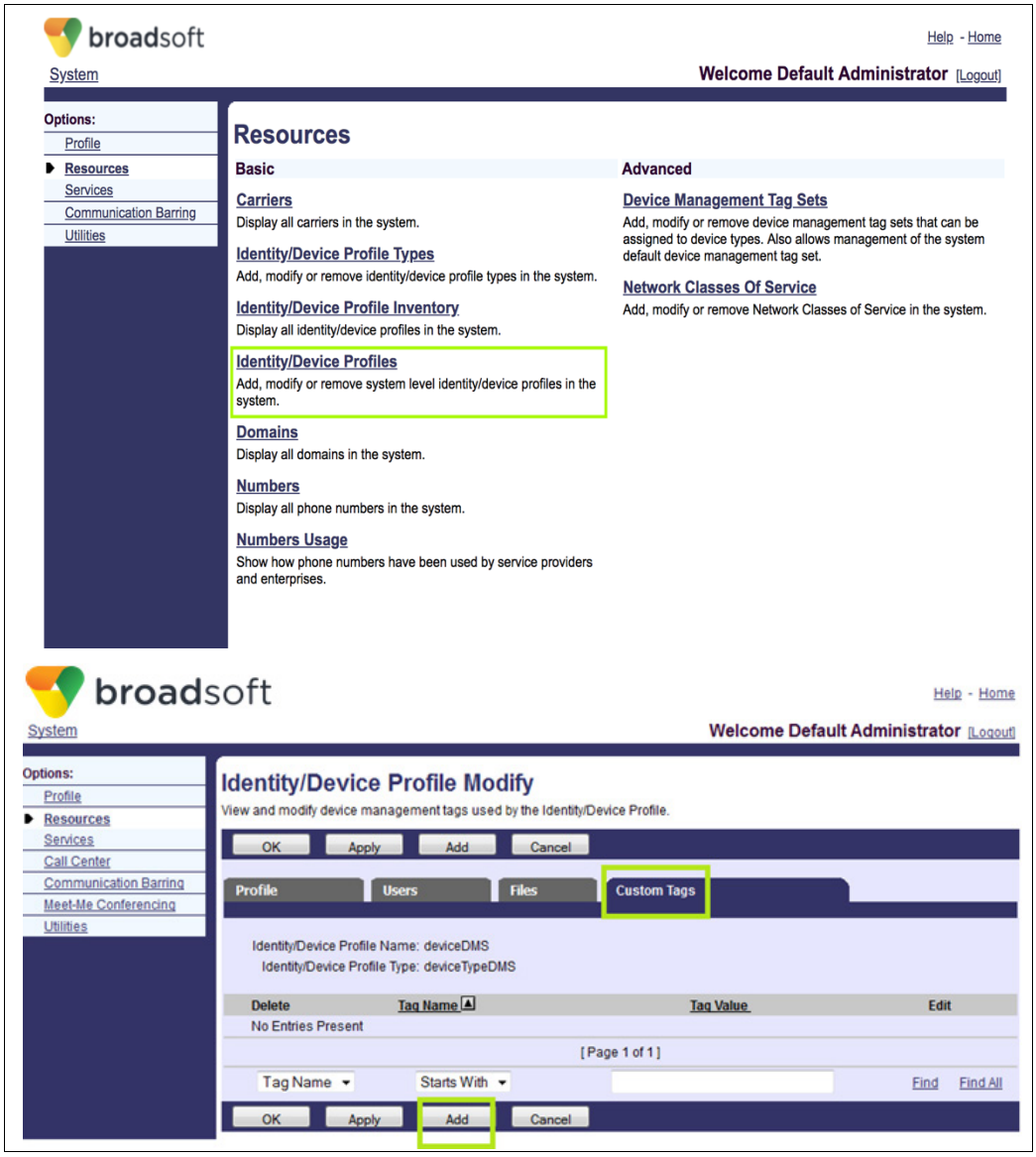


Figure 54 Creating Tags for a Device Profile

5.13.6.3 Create Tags for Device Types at the Group Level (AS Mode Only)

If a device type is configured to allow custom tags at the group level (that is if the *allowGroupCustomTagSet* parameter is set to "true") then you can also create individual static tags for device types at the group level.

To create a new tag for a device type from the CLI:

```
AS_CLI/SubscriberMgmt/Group/DeviceConfiguration/CustomTags>add <sp>
<group> <deviceType> <tagName> <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and not start by "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced device type templates for this device type and group.

From the web:

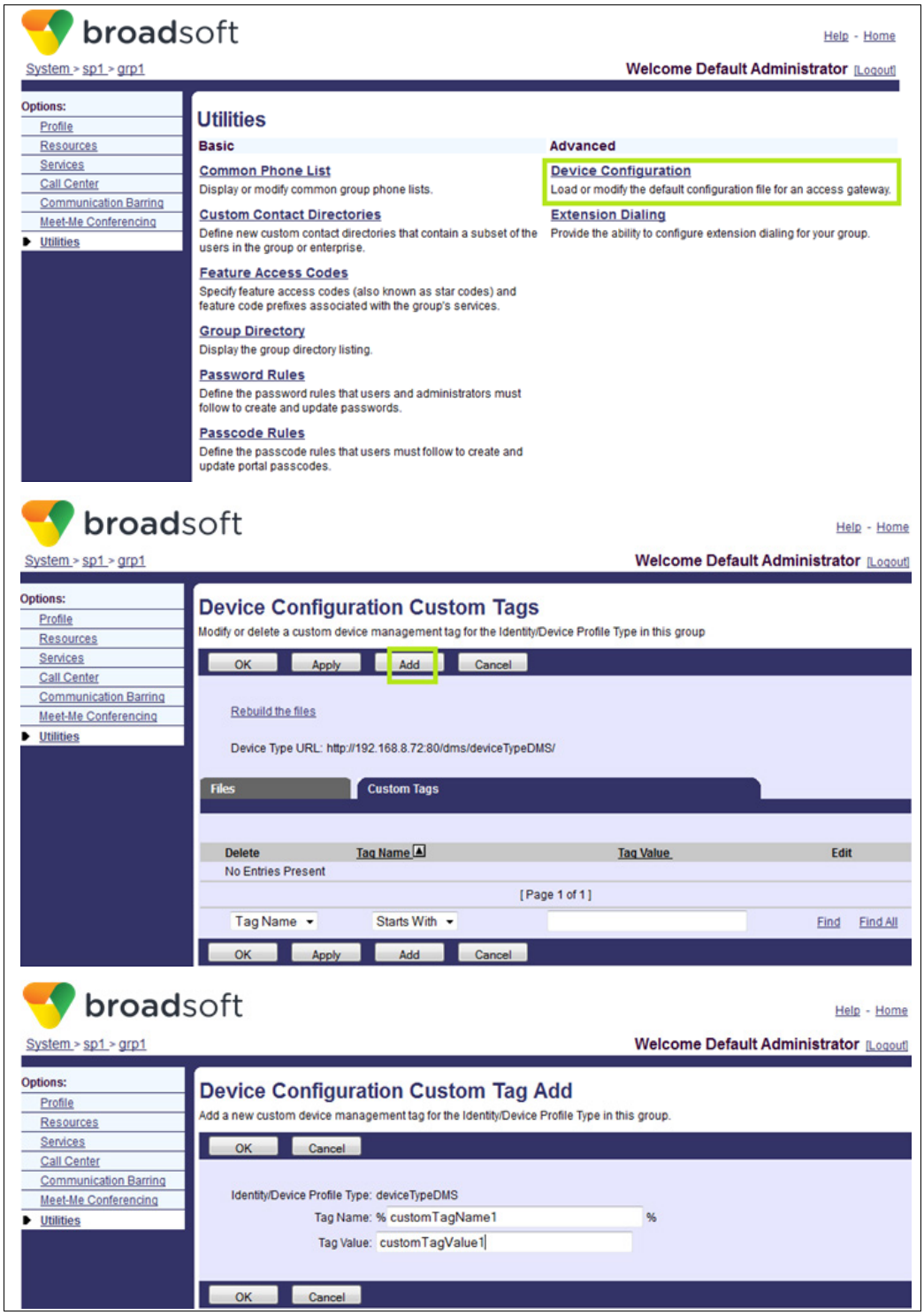


Figure 55 Creating Tags for a Device Type at the Group Level

5.13.6.4 Create Tags for Device Types at the Service Provider/Enterprise Level (AS Mode Only)

If a device type is configured to allow custom tags at the group level (that is if the *allowSPCustomTagSet* parameter is set to “true”) then you can also create individual static tags for device types at the service provider/enterprise level.

To create a new tag for a device type from the CLI:

```
AS_CLI/SubscriberMgmt/ServiceProvier/DeviceConfiguration/CustomTags>add
<sp> <deviceType> <tagName> <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the “%” character and not start by “BW”, for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced device type templates for this device type and group.

From the web:

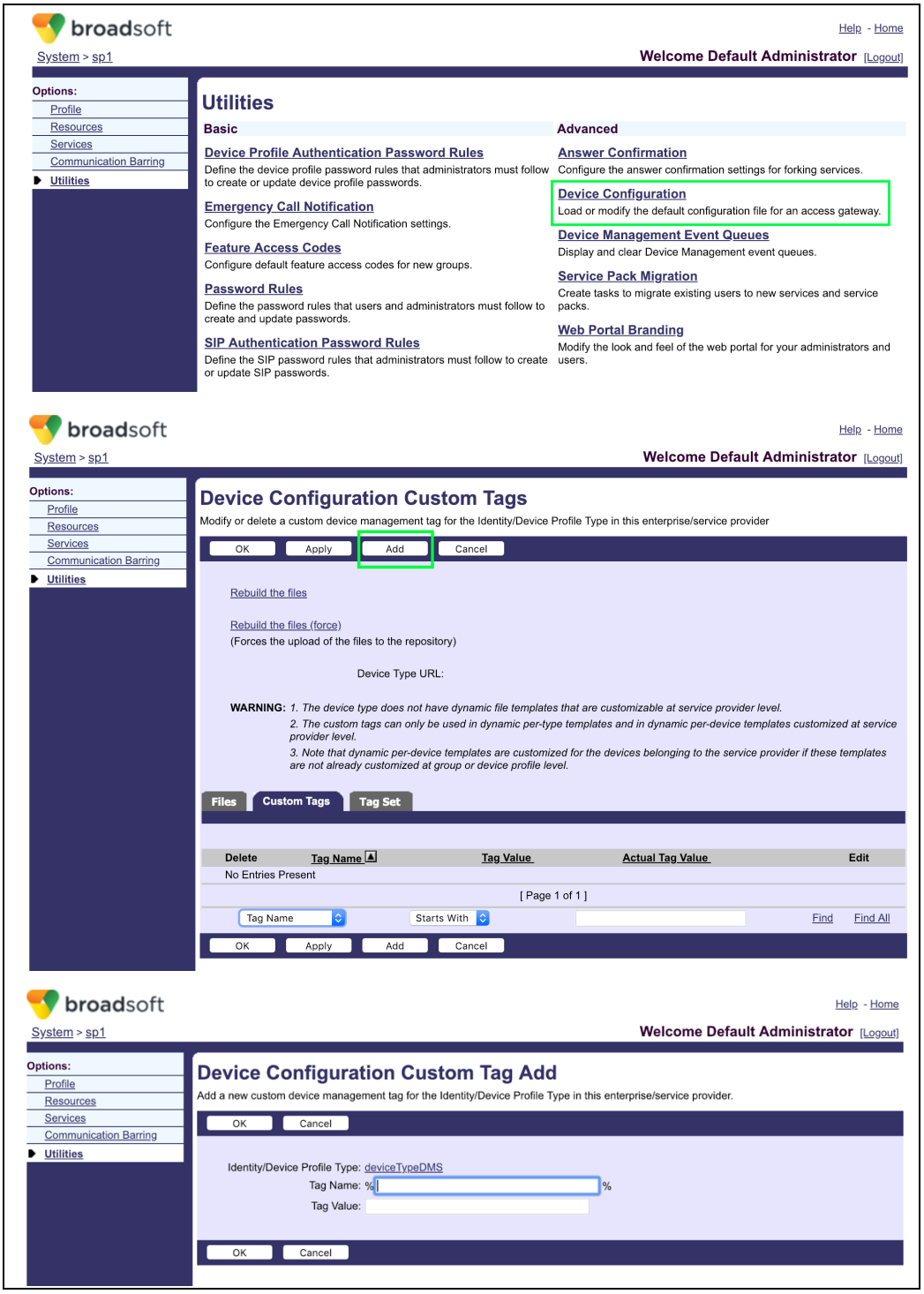


Figure 56 Creating Tags for a Device Type at the Service Provider/Enterprise Level

5.13.6.5 Create Tags at the System Level (XS Mode Only)

You can either add tags to a system tag set (*systemTagSet*), which is always available, or create one or more new tag sets for your tags.

To create a new tag set from the CLI:

```
AS_CLI/System/DeviceTagSet>add <tagSetName>
```

… where:

- **<tagSetName>** is the name of the newly created tag set. If you do not want to create a new tag set, you can skip this step and use the *systemTagSet*, which is always available.

To create an unencrypted static tag from the CLI:

```
AS_CLI/System/DeviceTagSet/Tags>addUnEncrypted <tagSetName> <tagName>
tagValue <tagValue>
```

… where:

- **<tagSetName>** is either *systemTagSet* or the name of the tag set that contains the new tag.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates.

From the web:

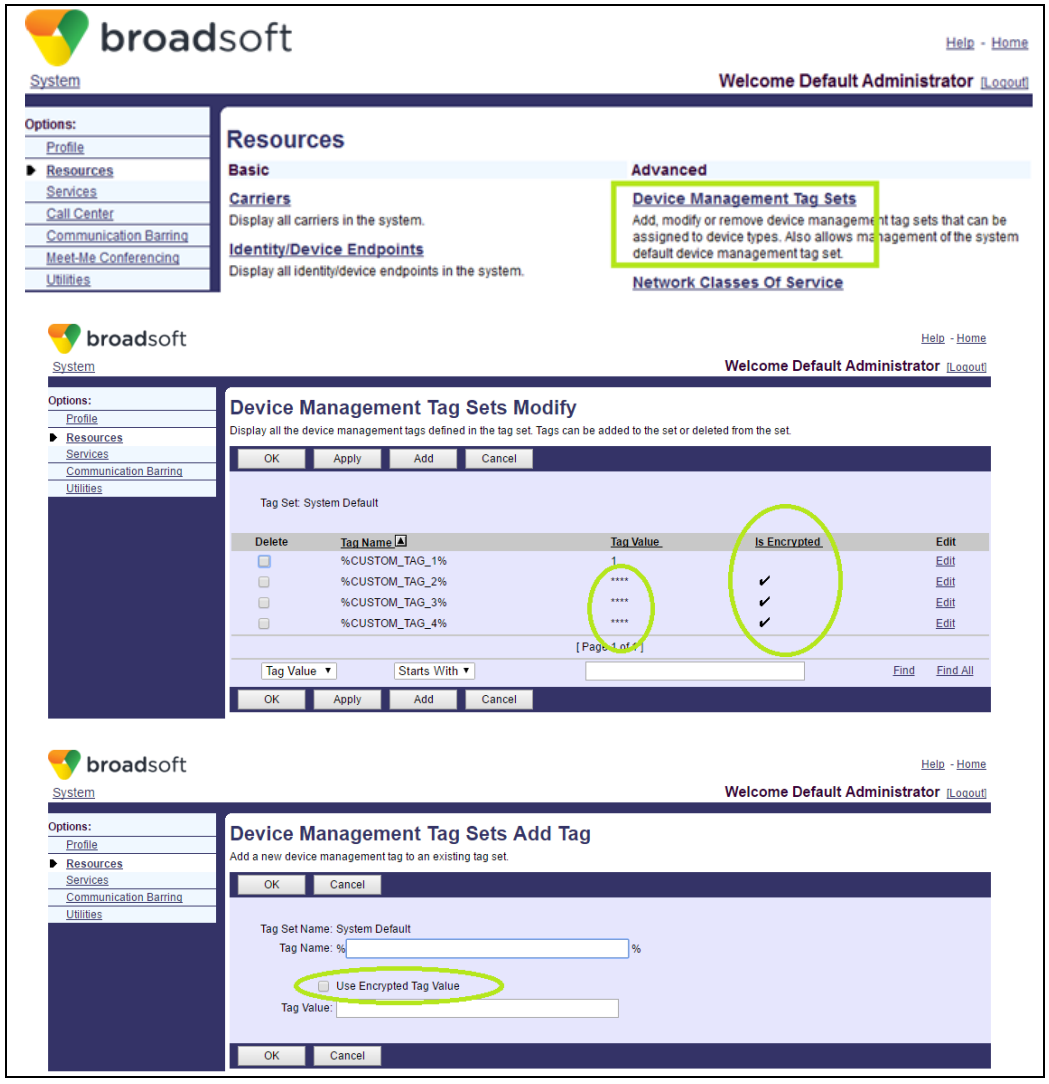


Figure 57 Managing Unencrypted Tag Sets

To create an encrypted static tag from the CLI:

```
AS_CLI/System/DeviceTagSet/Tags>addEncrypted <tagSetName> <tagName>
tagValueToEncrypt
```

… where:

- **<tagSetName>** is either *systemTagSet* or the name of the tag set that contains the new tag.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and re-type initial value.

```
CLI/System/DeviceTagSet/Tags> addEncrypted systemTagSet %CUSTOM_TAG_2%
tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

From the web:

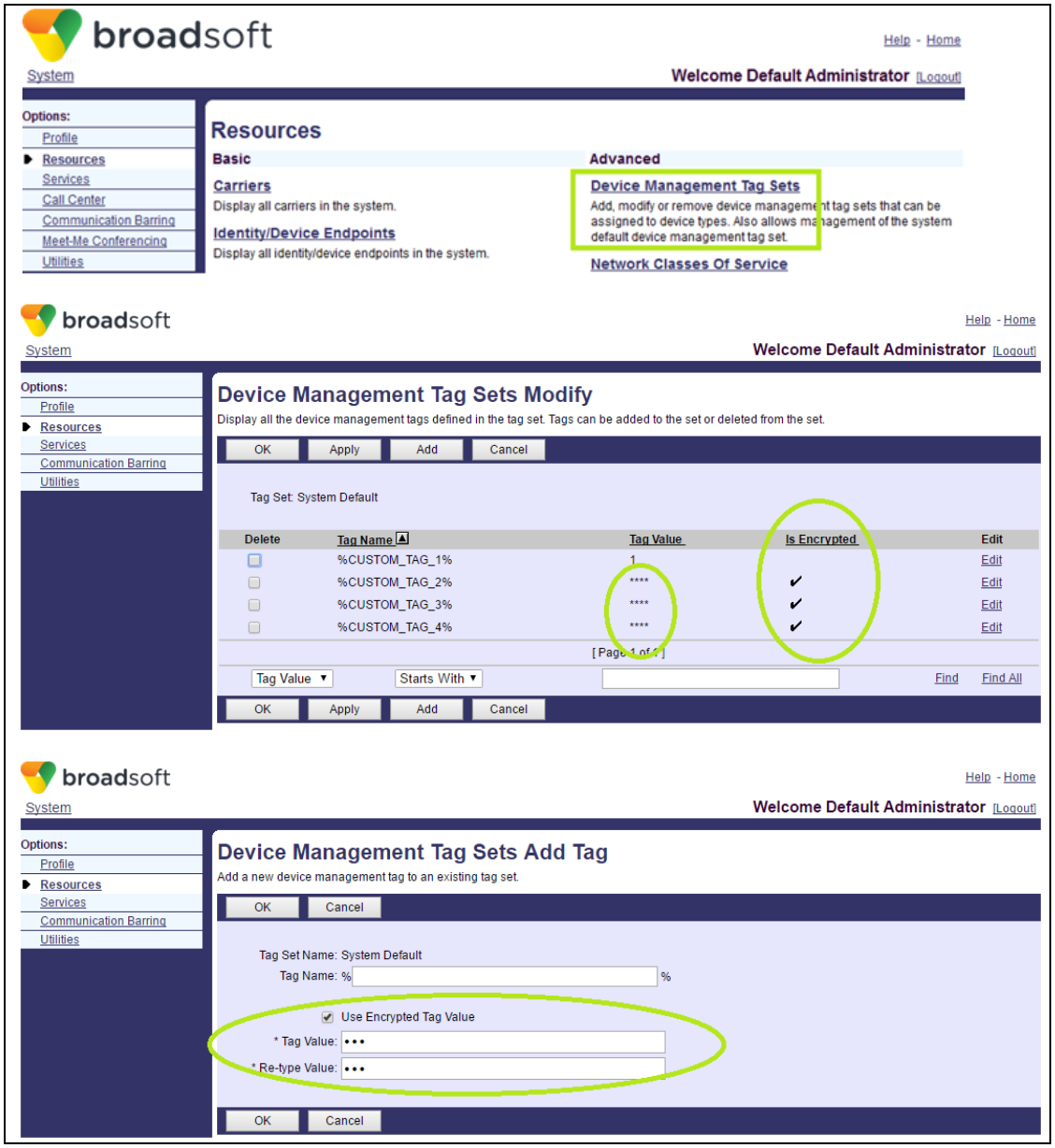


Figure 58 Managing Encrypted Tag Sets

5.13.6.6 Create Tags for Device Profiles (XS Mode Only)

If a device type is configured to allow custom tags (that is, if the *allowDeviceProfileCustomTagSet* parameter is set to "true") then you can also create individual static tags for device profiles at the service provider, group, and device profile levels.

To create an unencrypted tag at the service provider level from the CLI:

```
AS_CLI/SubscriberMgmt/ServiceProvider/Device/CustomTags>addUnEncrypted
<sp> <deviceName> <tagName> tagValue <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this service provider.

To create an unencrypted tag at the group level from the CLI:

```
AS_CLI/SubscriberMgmt/Group/Device/CustomTags>addUnEncrypted <sp> <group>
<deviceName> <tagName> tagValue <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this group.

To create an unencrypted tag at the system profile level from the CLI:

```
AS_CLI/SubscriberMgmt/Device/CustomTags>addUnEncrypted <deviceName>
<tagName> tagValue <tagValue>
```

… where:

- **<deviceName>** is the name of the device profile for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced in templates for this device profile.

From the web:

To add a tag at the system, service provider, or group level, access the *Identity/Device Profiles* menu item from the *Resources* menu page for the service provider, group, or system respectively.

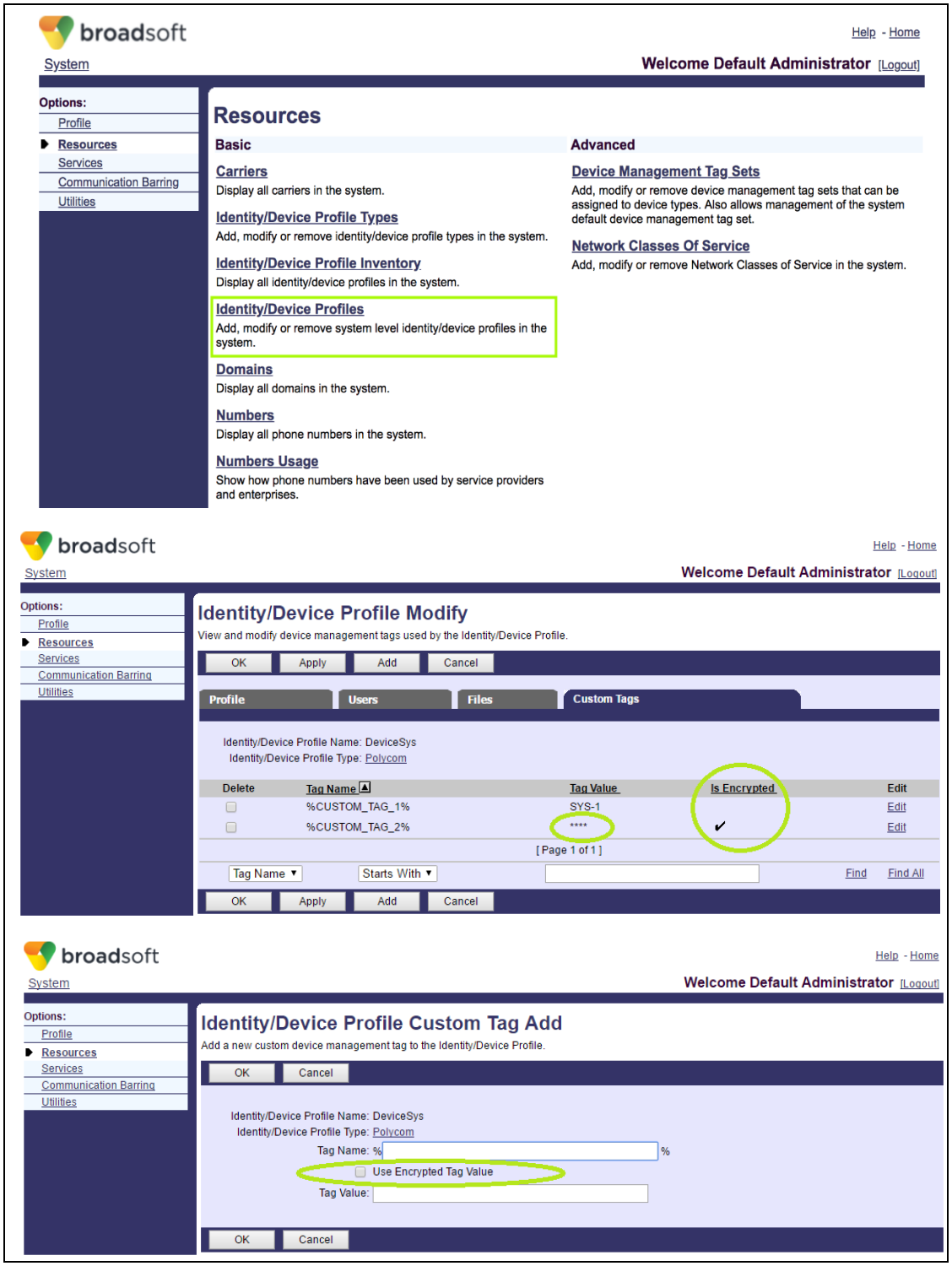


Figure 59 Creating Unencrypted Tags for a Device Profile

To create an encrypted tag at the service provider level from the CLI:

```
AS_CLI/SubscriberMgmt/ServiceProvider/Device/CustomTags>addEncrypted <sp>
<deviceName> <tagName> tagValueToEncrypt
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and re-type initial value.

```
$ CLI/SubscriberManagement/ServiceProvider/Device/CustomTags>
addEncrypted sp1 DeviceSp1 %CUSTOM_TAG_3% tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

To create an encrypted tag at the group level from the CLI:

```
AS_CLI/SubscriberMgmt/Group/Device/CustomTags>addEncrypted <sp> <group>
<deviceType> <tagName> tagValueToEncrypt
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceName>** contains the specific device for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and then re-type initial value.

```
$ CLI/SubscriberManagement/Group/Device/CustomTags> addEncrypted sp1 grp1
DeviceGp1 %CUSTOM_TAG_3% tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

To create an encrypted tag at the system profile level from the CLI:

```
AS_CLI/SubscriberMgmt/Device/CustomTags>addUnEncrypted <deviceName>
<tagName> tagValueToEncrypt
```

… where:

- **<deviceName>** is the name of the device profile for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and then re-type initial value.

```
$ CLI/SubscriberMgmt/Device/CustomTags> addEncrypted DeviceSys
%CUSTOM_TAG_3% tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

From the web:

To add a tag at the system, service provider, or group level, access the *Identity/Device Profiles* menu item from the *Resources* menu page for the service provider, group, or system respectively.

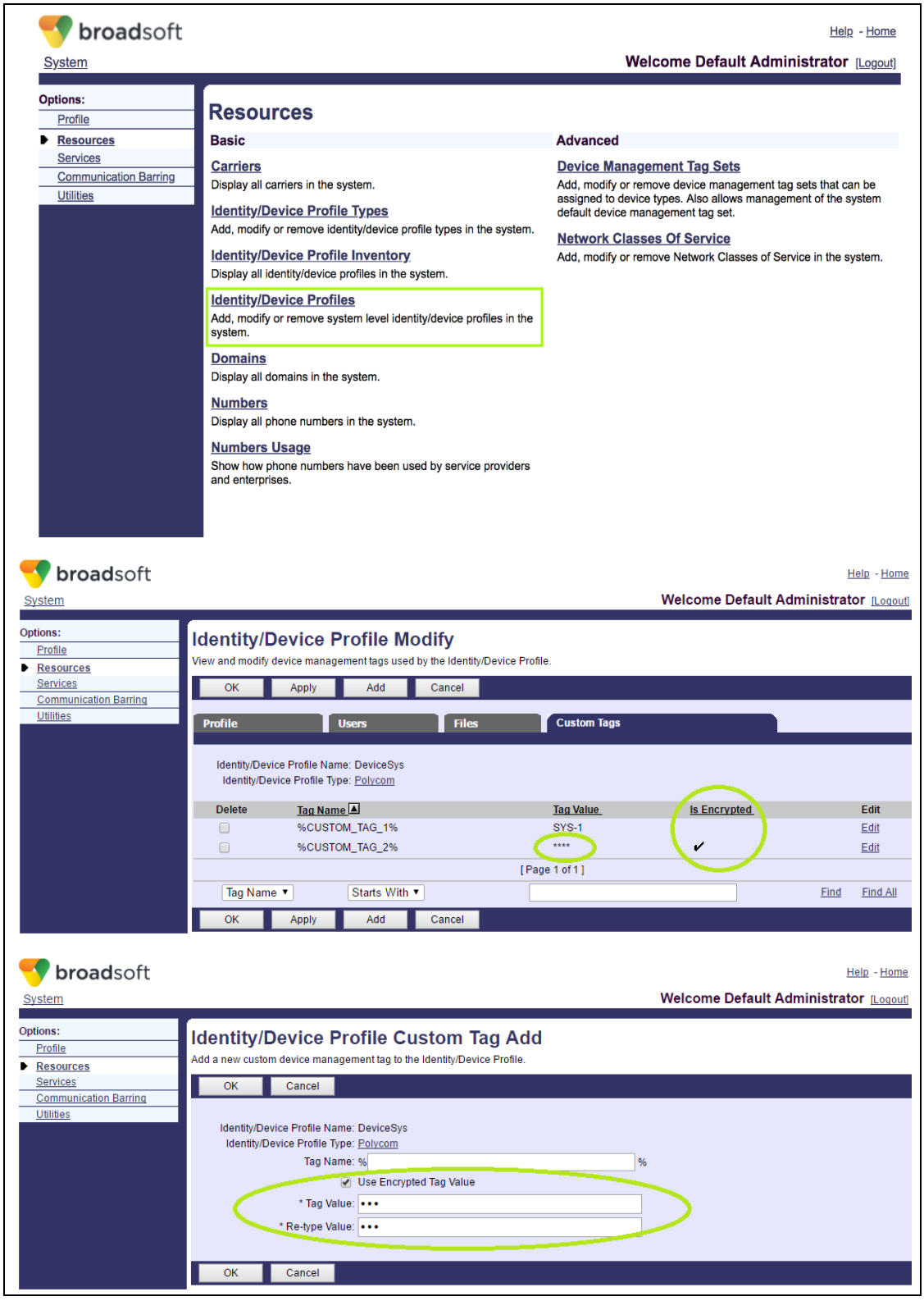


Figure 60 Creating Encrypted Tags for a Device Profile

5.13.6.7 Create Tags for Device Types at the Group Level (XS Mode Only)

If a device type is configured to allow custom tags at the group level (that is, if the *allowGroupCustomTagSet* parameter is set to "true") then you can also create individual static tags for device types at the group level.

To create a new unencrypted tag for a device type from the CLI:

```
PS_CLI/SubscriberMgmt/Group/DeviceConfiguration/CustomTags>addUnEncrypted
<sp> <group> <deviceType> <tagName> tagValue <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced device type templates for this device type and group.

From the web:

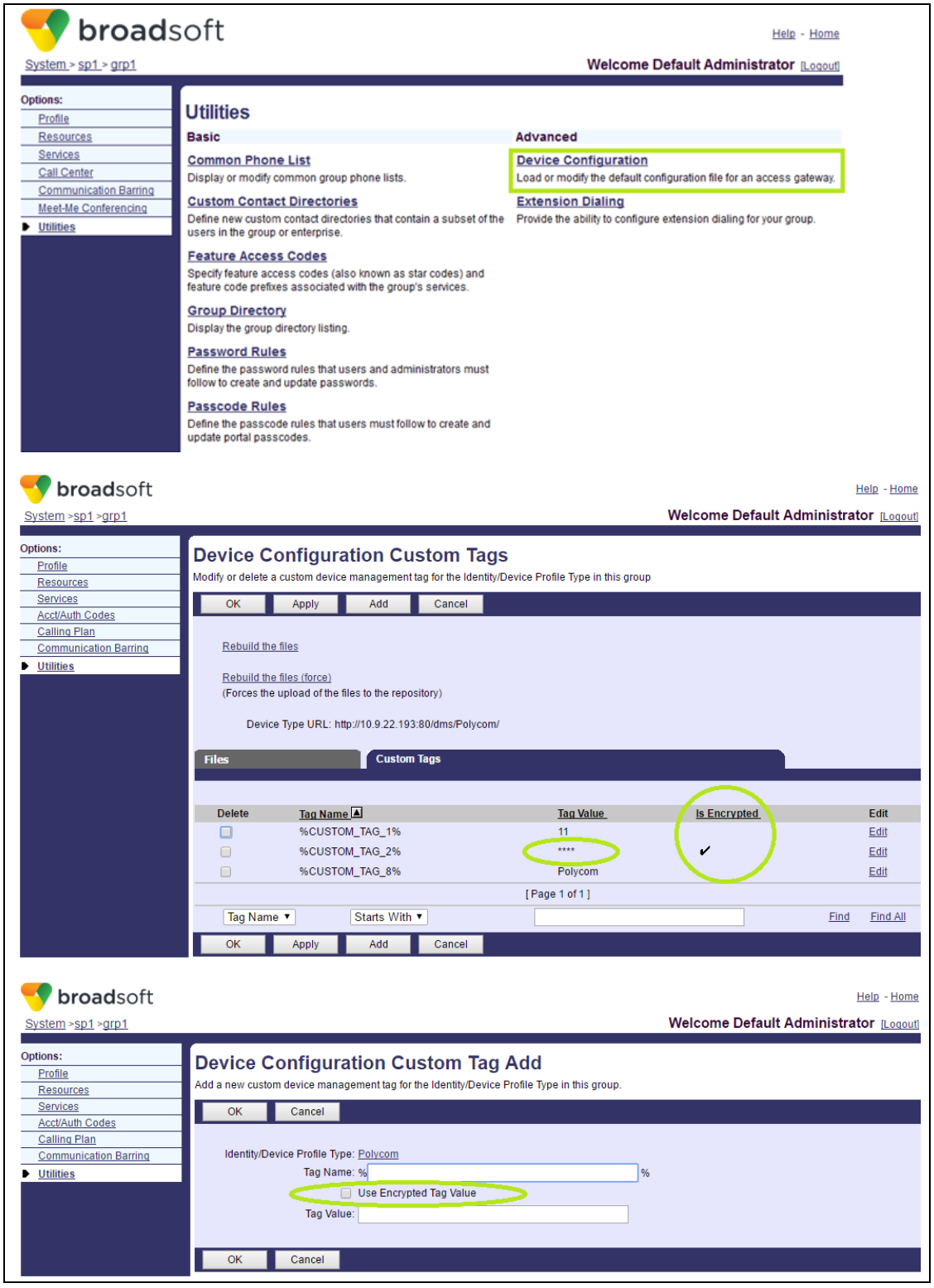


Figure 61 Creating Unencrypted Tags for a Device Type at the Group Level

To create a new encrypted tag for a device type from the CLI:

```
PS_CLI/SubscriberMgmt/Group/Device/CustomTags>addEncrypted <sp> <group>
<deviceType> <tagName> tagValueToEncrypt
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and then re-type initial value.

```
PS_CLI/SubscriberMgmt/Group/Device/CustomTags>addEncrypted sp1 grp1
Polycom %CUSTOM_TAG_5% tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

From the web:

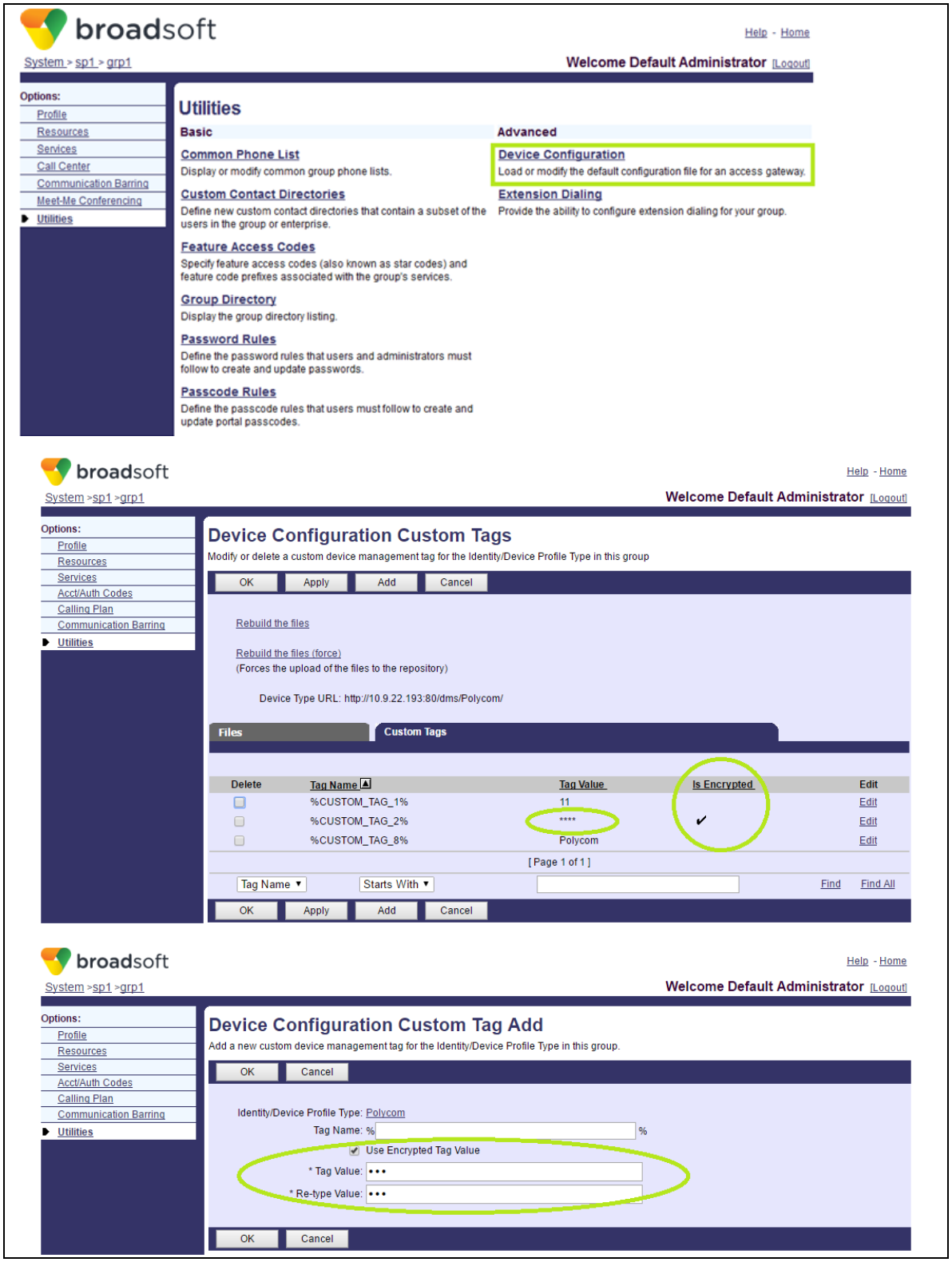


Figure 62 Creating Encrypted Tags for a Device Type at the Group Level

5.13.6.8 Create Tags for Device Types at the Service Provider/Enterprise Level (XS Mode Only)

If a device type is configured to allow custom tags at the service provider/enterprise level (that is, if the *allowSPCustomTagSet* parameter is set to "true") then you can also create individual static tags for device types at the service provider/enterprise level.

To create a new unencrypted tag for a device type from the CLI:

```
PS_CLI/SubscriberMgmt/ServiceProvider/DeviceConfiguration/CustomTags>addU
nEncrypted <sp> <deviceType> <tagName> tagValue <tagValue>
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.
- **<tagValue>** is the string by which the tag is replaced device type templates for this device type and group.

From the web:

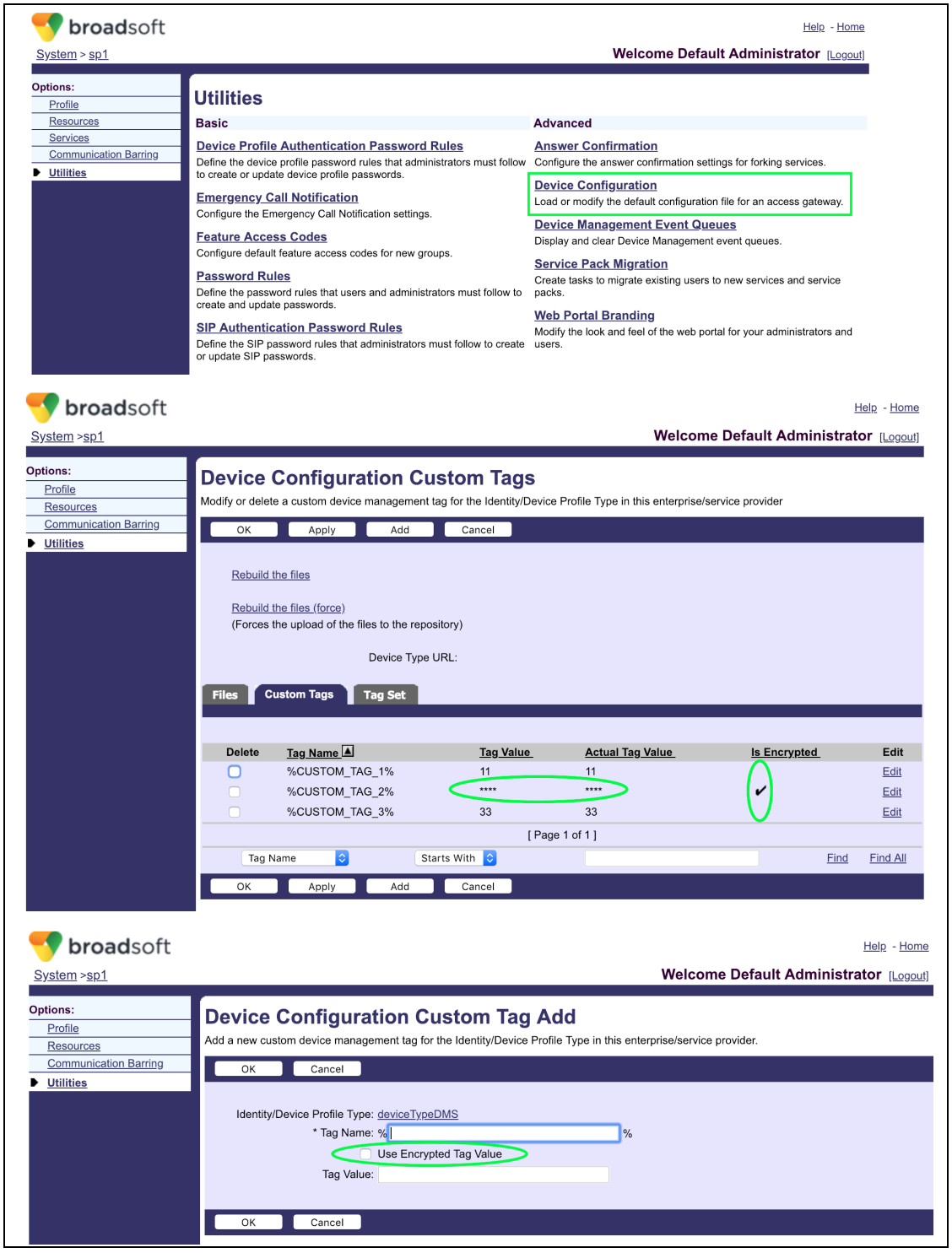


Figure 63 Creating Unencrypted Tags for a Device Type at the Group Level

To create a new encrypted tag for a device type from the CLI:

```
PS_CLI/SubscriberMgmt/Group/Device/CustomTags>addEncrypted <sp> <group>
<deviceType> <tagName> tagValueToEncrypt
```

… where:

- **<sp>** is the service provider for which the tag is added.
- **<group>** represents the group for which the tag is added.
- **<deviceType>** contains the specific device type for which the tag is added.
- **<tagName>** is the case-sensitive tag name that must be enclosed by the "%" character and must not start with "BW", for example, %MYTAG%.

The CLI prompts the administrator to enter the value to encrypt in a password-like fashion. Enter the initial value and then re-type initial value.

```
PS_CLI/SubscriberMgmt/Group/Device/CustomTags>addEncrypted sp1 grp1
Polycom %CUSTOM_TAG_5% tagValueToEncrypt
Initial Value:
Re-type Initial Value:
...Done
```

From the web:

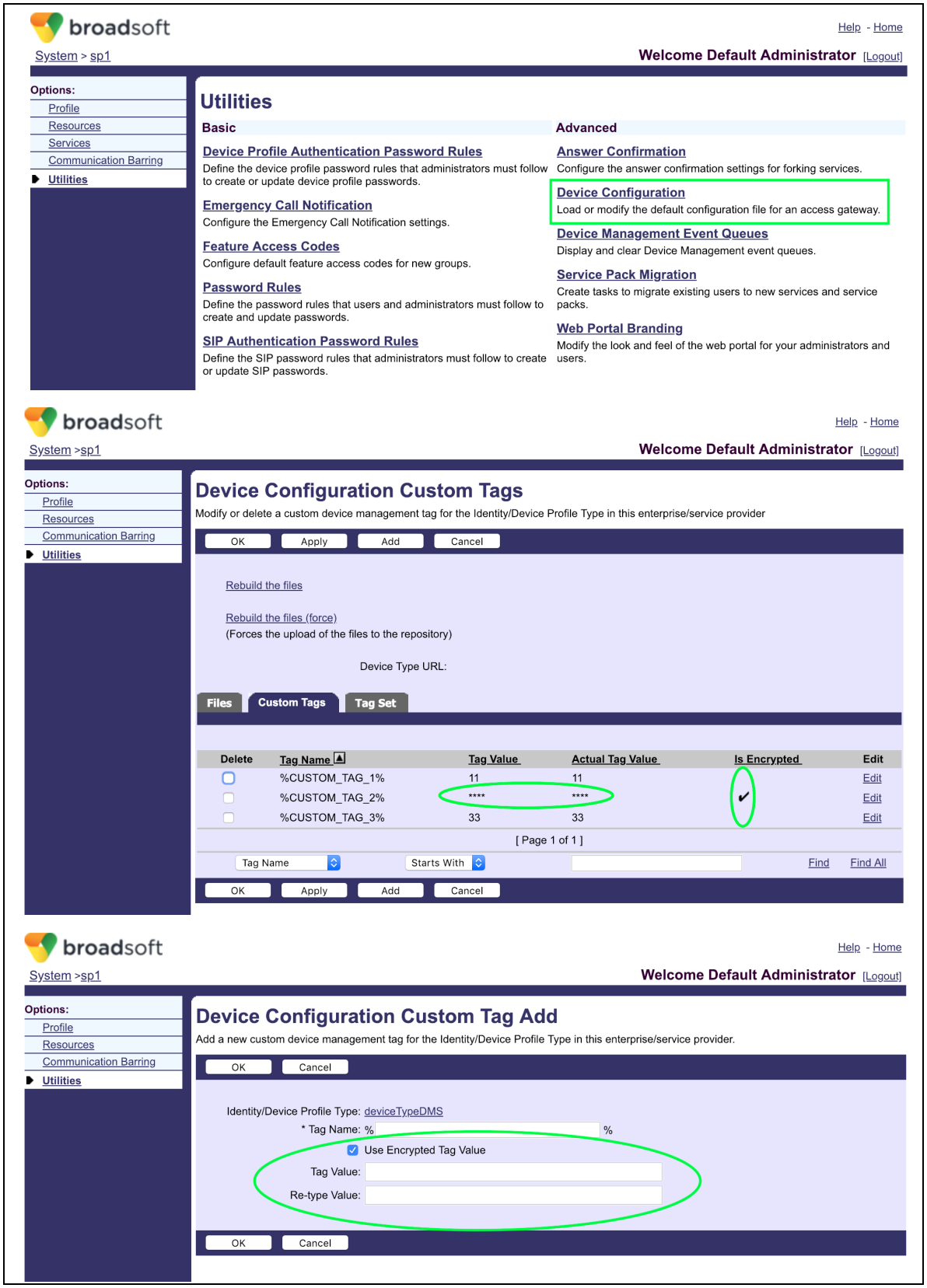


Figure 64 Creating Encrypted Tags for a Device Type at the Group Level

5.13.6.9 Create Tags for Device Types at the Group Level

The web portal displays a message when the admin is adding new custom tags but no customized file template has been defined yet.

The problem here is that when using custom tags with device type level templates, the custom tags are not honored (which is well understood since it would overwrite the common built file).

To correct the situation, the customer must use custom files at the group level as well. That requires more provisioning. The following figure shows the warning message when the administrator is adding new custom tags but no customized file template has yet been defined.

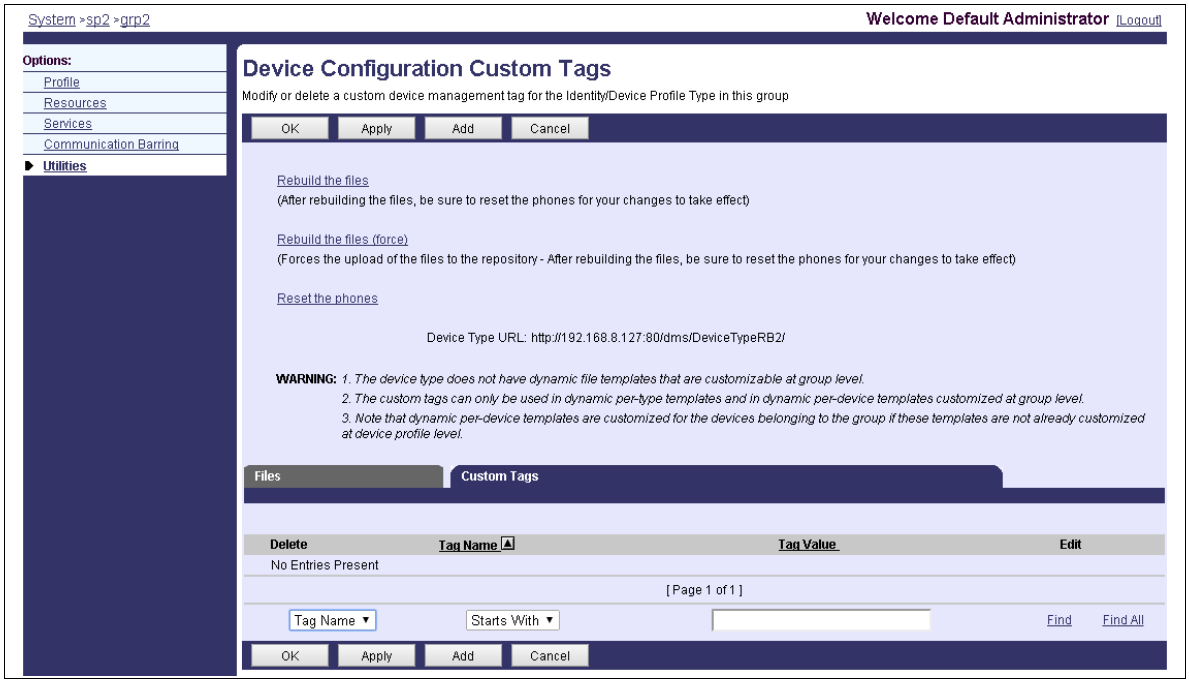


Figure 65 Warning Message

5.13.7 Customize Static Tags

To customize an existing tag, create a tag with the same name at a more specific level. The general rule is that the most specific tag is resolved for a given template. For example, if a tag %TEST% has the value "VALUE1" in the default tag set and there also exists a tag named %TEST% at the group level with the value "VALUE2", then the most specific value "TEST2" is used when resolving tags for a template in this group. The tag hierarchy is shown in *Figure 63*.

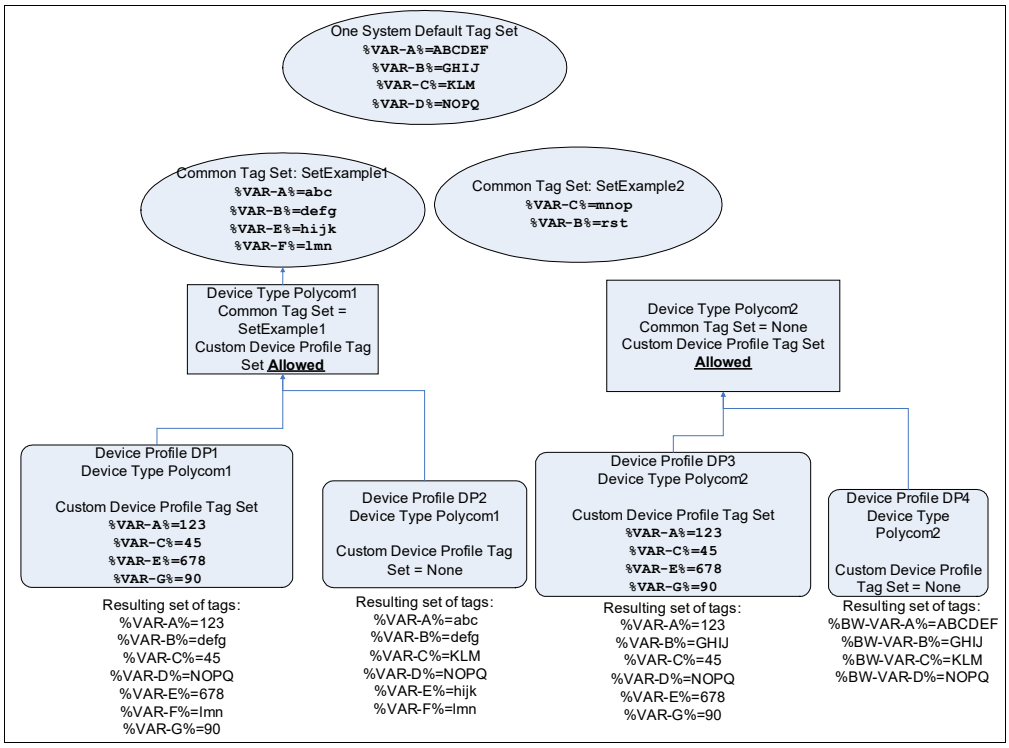


Figure 66 Static Tags Hierarchy

5.13.7.1 Tag Customization on Dynamic Per-Type Files

Device Management allows customizing tags at the group level (usually overriding existing tags from a system-defined tag set). The web page used for this customization is located at the group level in the web portal under *Utilities → Device Configuration*, this page lists the device types that use tag sets defined at system level.

For dynamic per-type files, these customizations are ignored by the system unless a custom template is assigned to a file. As their names indicate, dynamic per-type files have a one-to-one relationship with the device profile type. There is only one file generated for each device profile level template file. Note that if a group customization tag applied to these types of files, it would cause logical errors for tag resolution.

For example, in a scenario where two different groups override the same tag (see the following figure) both overrides cannot be applied since the file is resolved once for the whole profile type (when using the default template).

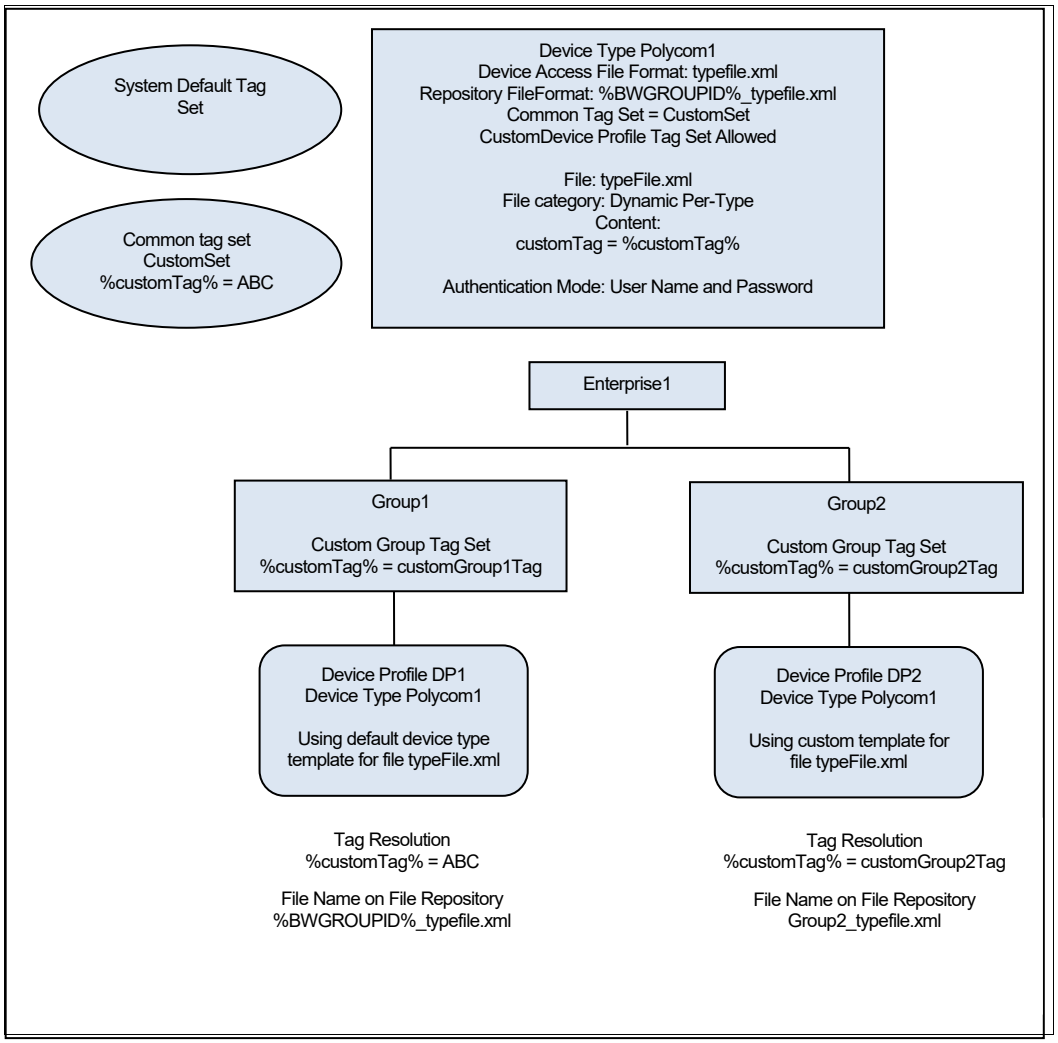


Figure 67 Tag Customization on Dynamic Per-Type Files

As shown in *Figure 64*, the solution is to customize the template used by the device profile, in which case the system resolves custom tags within the device per-type file.

Note that when using a dynamic per-type file, authentication is necessary to allow the system to find the right file when requested. Without authentication, when a file is customized, there is no way of knowing if the customized or generic file should be provided.

5.13.7.2 Make Static Tags Non-Overridable

Device Management allows to control whether custom tags in a tag set at the system level can be overridden at any other level.

The following figure demonstrates how the overridable flag affects the tag values used in the configuration files.

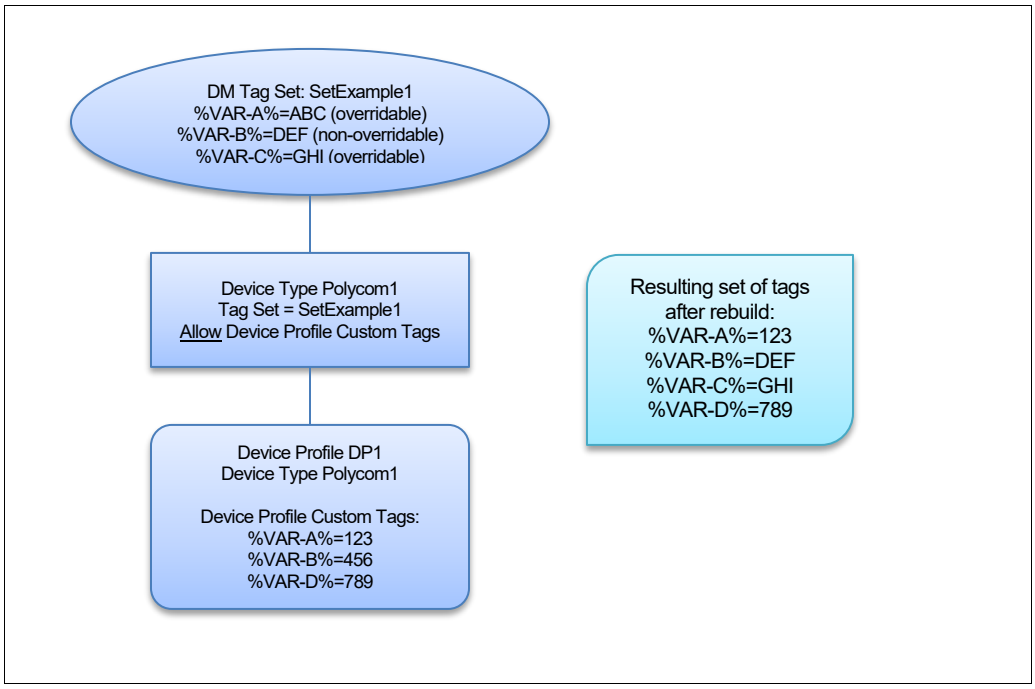


Figure 68 Basic Tag Hierarchy Using Overridable Flag

In the example, %VAR-A% is overridable and %VAR-B% is non-overridable in the system-level tag set. They are also defined again for the device profile. When the tags are resolved, %VAR-A% is the value defined in the device profile since it is overridable in the system-level tag set, and %VAR-B% is the value defined in the system-level tag set since it is non-overridable.

5.13.8 Use of Static Tags to Gradually Introduce New Firmware File

This section provides an example of the static tag customizing mechanism. In this example, the administrator must gradually introduce new firmware files to a few Polycom devices to test the file before it is rolled out to all the devices. One way to do this would be to use a static tag at the system level that points to the firmware file. For example, the administrator would create a static tag called %POLYCOM_500_FIRMWARE_FILE% with a value of “Polycom500_2.3.ld”. The Polycom template for all devices would look similar to the following.

```
<?xml version="1.0" standalone="yes"?>
<!-- Master SIP Configuration File-->
<APPLICATION APP_FILE_PATH="%POLYCOM_500_FIRMWARE_FILE%"
             CONFIG_FILES="phone1.cfg, sip.cfg"
             MISC_FILES=""
             LOG_FILE_DIRECTORY="log/"/>
```

To test a new firmware on a few devices, the administrator would first upload the new firmware file as a static file name *Polycom500_2.4.ld*. Then the administrator would go to a specific device profile and create a new static tag with the same name, but with a value that points to the new firmware file. After rebuilding the file for this specific Polycom device, the master file would point to the new firmware file. Once the test is over, the administrator would customize the same static tag at the group level to gradually introduce the new firmware to a group or simply change the value at the device-type level to apply the new file to all devices.

5.13.9 Tag Set Assignment at Device Customization Levels

At the system level it is possible to assign a Tag Set to your Device Type. To improve the provisioning requirements, it is also possible to assign an additional tag set at each of the customization levels (*enterprise/service provider → group → device profile*).

5.13.9.1 Tag Set Assignment at Device Profile Level

If a device type is configured to allow custom tags at the device profile level (that is if the *allowDeviceProfileCustomTagSet* parameter is set to "true") then you can also assign existing tag sets for the device types at the device profile level.

To assign a tag set to a group level access device from the CLI:

```
CLI/SubscriberMgmt/Group/Device/TagSet>set <sp> <group> <deviceName>
<tagSetName>
```

… where:

- **<sp>** is the service provider for which the tag set is assigned.
- **<group>** represents the group for which the tag set is assigned.
- **<deviceName>** represents the device name for which the tag set is assigned.
- **<tagSetName>** represent the tag set name to assign.

To assign a tag set to a service provider/enterprise level access device from the CLI:

```
CLI/SubscriberMgmt/ServiceProvider/Device/TagSet>set <sp> <deviceName>
<tagSetName>
```

… where:

- **<sp>** is the service provider for which the tag set is assigned.
- **<deviceName>** represents the device name for which the tag set is assigned.
- **<tagSetName>** represent the tag set name to assign.

To assign a tag set to a systemp level access device from the CLI:

```
CLI/SubscriberMgmt/Device/TagSet>set <deviceName> <tagSetName>
```

… where:

- **<deviceName>** represents the device name for which the tag set is assigned.
- **<tagSetName>** represent the tag set name to assign.

From the web:

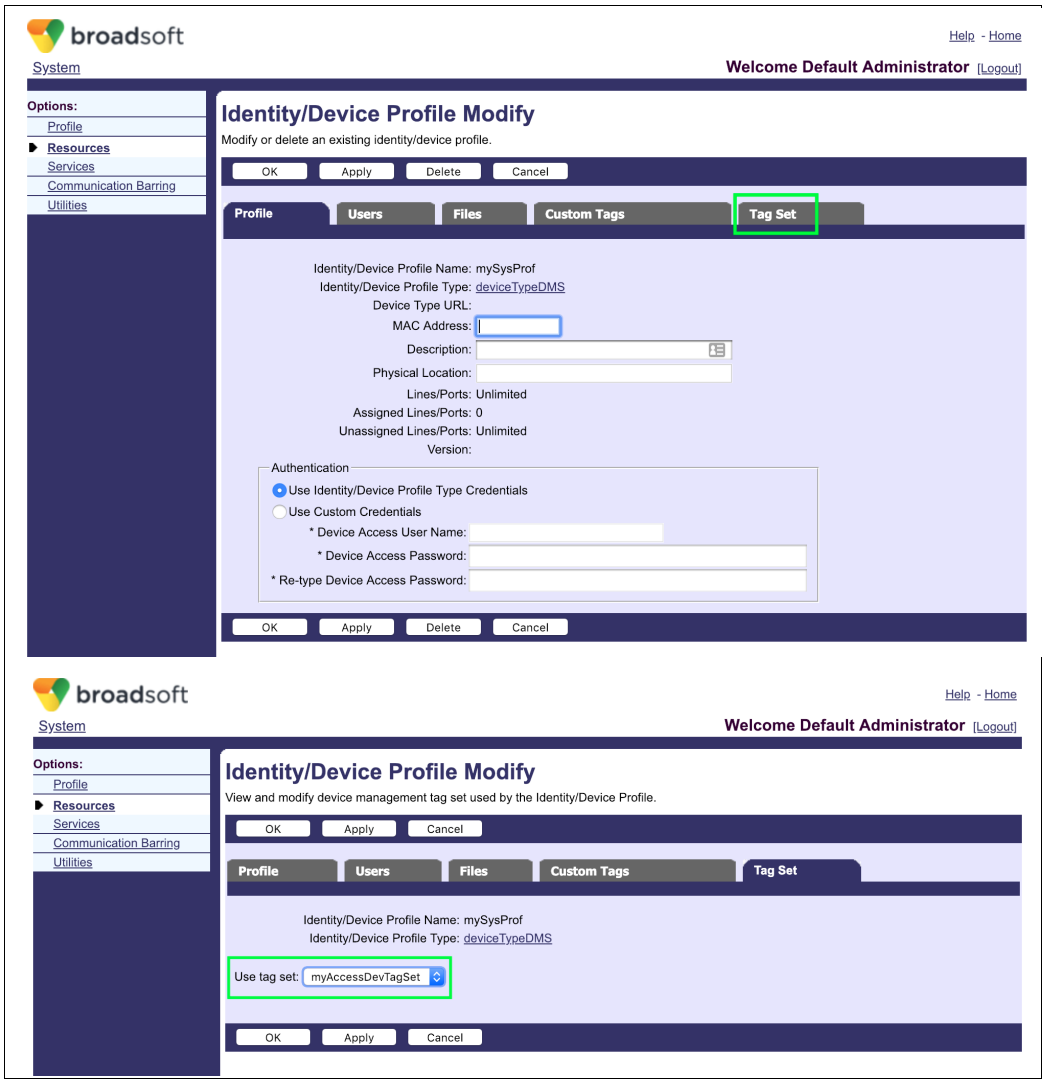


Figure 69 Assigning a Tag Set for a System Level Access Device

NOTE: The screen capture above represent a system level access device, the web pages are identical for service provider/enterprise and group level access devices.

5.13.9.2 Tag Set Assignment at Group Level

If a device type is configured to allow custom tags at the group level (that is if the *allowGroupCustomTagSet* parameter is set to "true") then you can also assign existing tag sets for the device types at the group level.

To assign a tag set to a group level access device from the CLI:

```
CLI/SubscriberMgmt/Group/DeviceConfiguration/TagSet>set <sp> <group>
<deviceType> <tagSetName>
```

… where:

- **<sp>** is the service provider for which the tag set is assigned.
- **<group>** represents the group for which the tag set is assigned.
- **<deviceType>** contains the specific device type for which the tag set is assigned.
- **<tagSetName>** represent the tag set name to assign.

From the web:

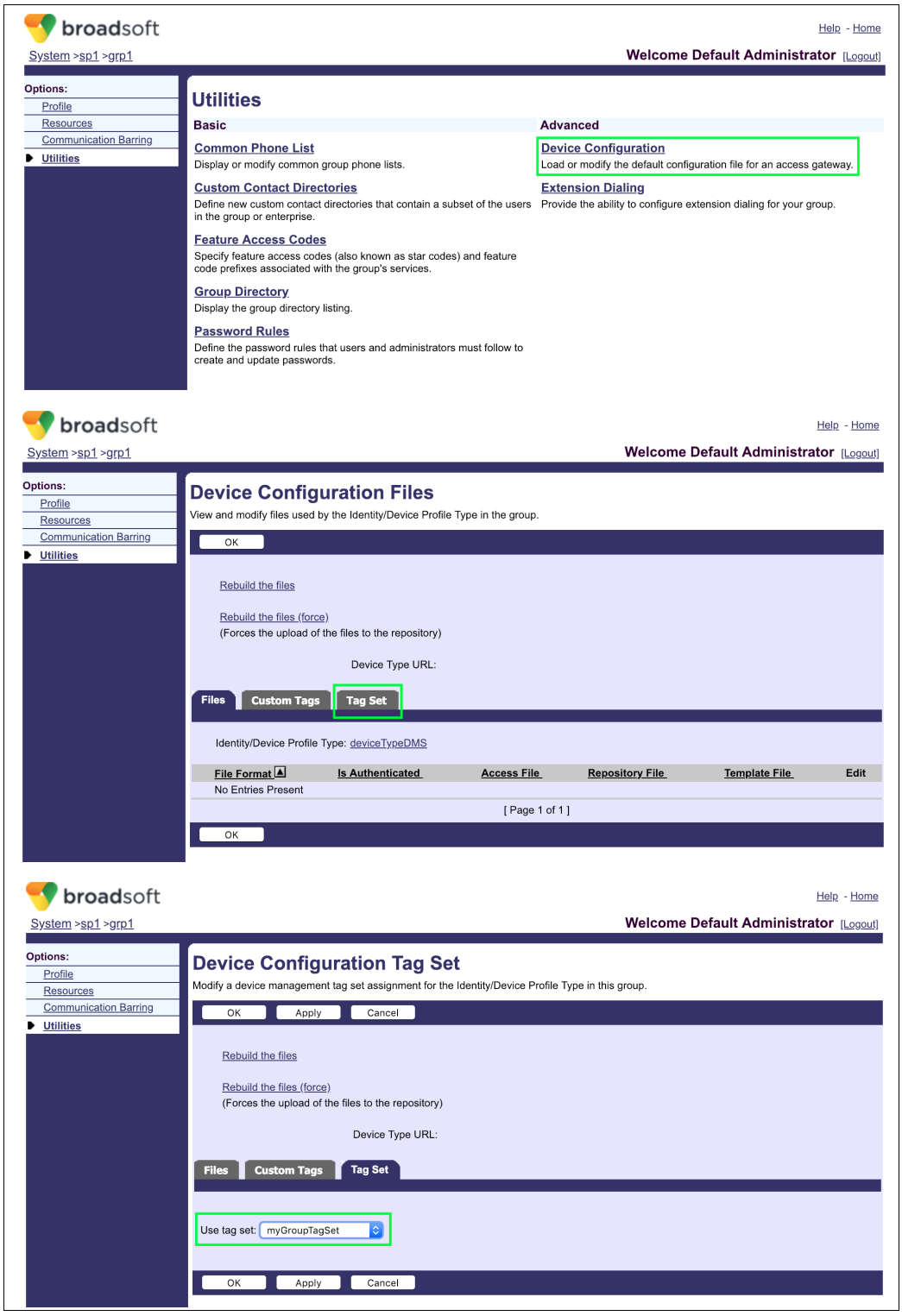


Figure 70 Assigning a Tag Set at the Group Level for a Device Type

5.13.9.3 Tag Set Assignment at Service Provider/Enterprise Level

If a device type is configured to allow custom tags at the service provider/enterprise level (that is if the *allowSPCustomTagSet* parameter is set to “true”) then you can also assign existing tag sets for the device types at the service provider/enterprise level.

To assign a tag set to a group level access device from the CLI:

```
CLI/SubscriberMgmt/ServiceProvider/DeviceConfiguration/TagSet>set <sp>
<deviceType> <tagSetName>
```

… where:

- **<sp>** is the service provider for which the tag set is assigned.
- **<deviceType>** contains the specific device type for which the tag set is assigned.
- **<tagSetName>** represent the tag set name to assign.

From the web:

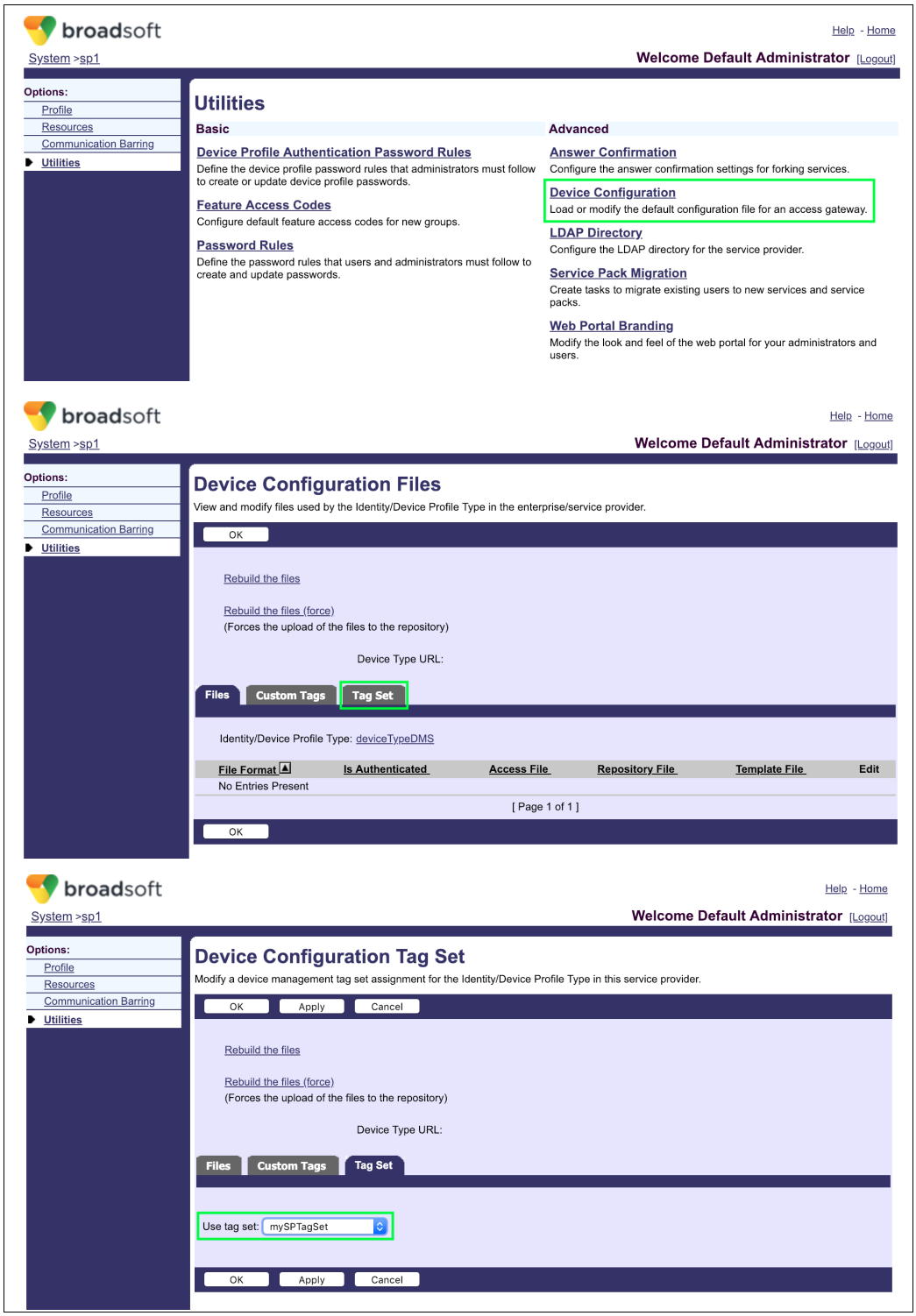


Figure 71 Assigning Tag Set for Device Type

5.14 Phone Services

5.14.1 Overview

Device Management provides the framework for a new class of user service called Phone Services. Phone Services are end-user services designed to expose specific capabilities of a specific make and model of a phone to the end user. The intent is to provide a tightly integrated end-user experience for a specific phone on Cisco BroadWorks.

When defining a device profile type, it is possible to specify whether the device profile type supports one or more Phone Services. When a Phone Service is enabled on the device profile type, it manages the creation of any template configuration files required for the Phone Service. This eliminates some of the complexity around defining the template configuration files for some advanced service integration.

5.14.2 Integrate Polycom Phone Directory with Cisco BroadWorks

The new Device Management introduces a user service called Polycom Phone Services. It allows the user to select people from their Cisco BroadWorks directory and add them as contacts to their phone directory. The directory is then downloaded by the phone from Cisco BroadWorks. When a new entry is added into the phone, the directory is also uploaded from the phone to Cisco BroadWorks. This functionality is called the Polycom Directory Phone Services.

When authorized to a group, Polycom Phone Services introduces the group administration for Polycom Phone Services. When assigned to a user, Polycom Phone Services introduces the user personalization for Polycom Phone Services. To authorize the Polycom Phone Services for a device type, see *Figure 66*.

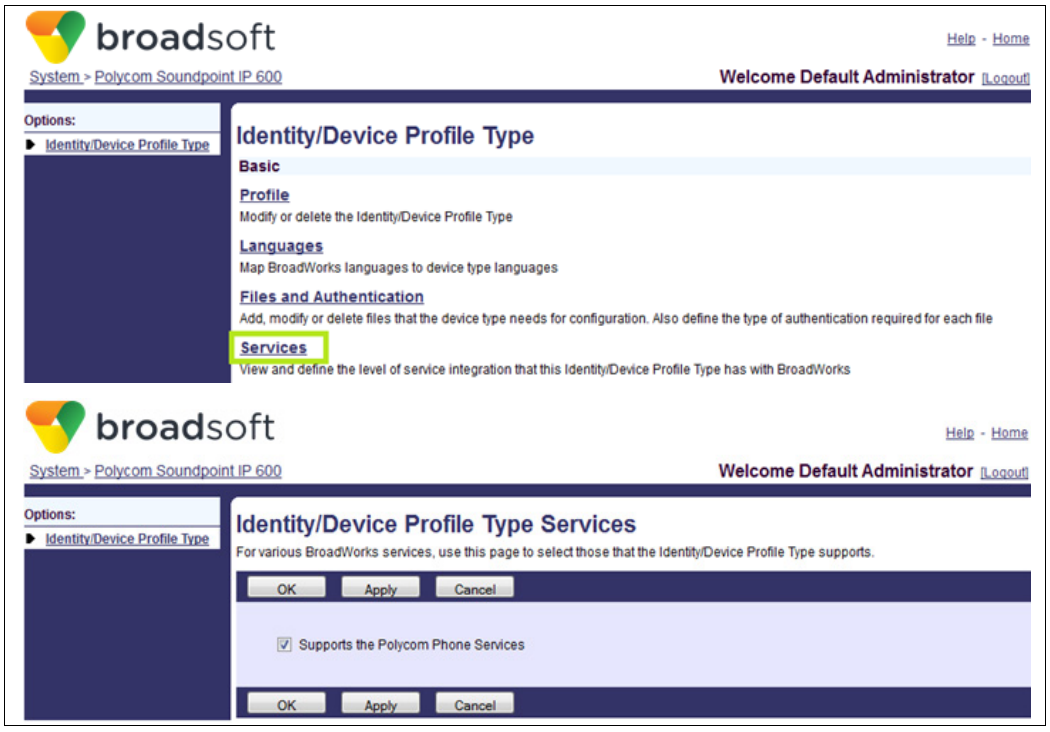


Figure 72 Enabling Polycom Phone Services for Specific Device Type

5.14.3 Group Authorization

This allows the group administrator to specify the contacts that all users in the group should see on their Polycom phones. This is called the Polycom Group Phone Directory List. It is shared among all users who have been assigned the Polycom Phone Services User service and have the Polycom Phone Directory enabled.

The administrator can configure the Polycom Group Phone Directory by doing any of the following:

- Electing to have the group's Common Phone List included

  -and/or-

- Choosing a Custom Contact Directory to include in the phone directory

Figure 67 shows how the service is assigned at the group level and *Figure 68* shows how to configure it.

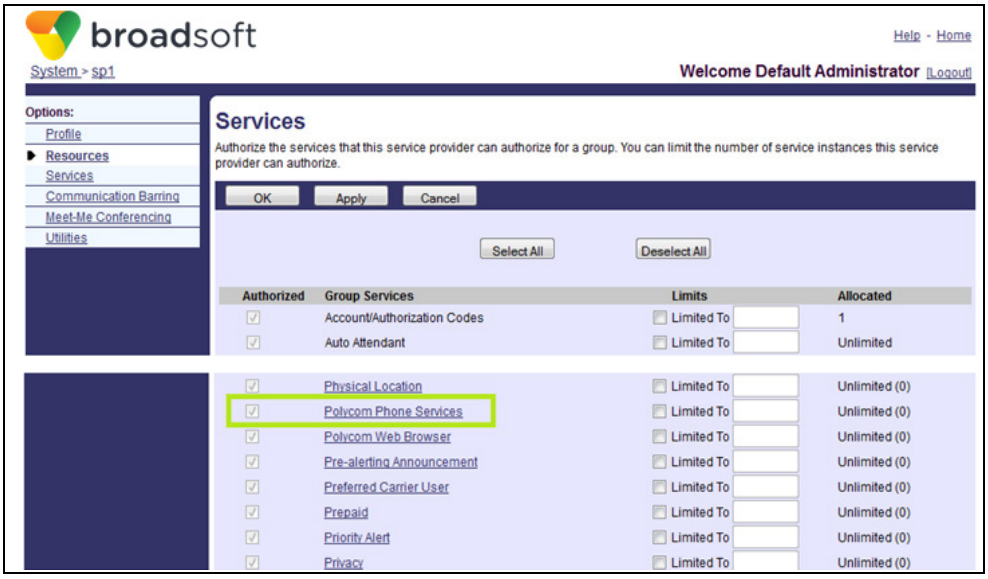


Figure 73 Authorizing Polycom Phone Services at Group Level

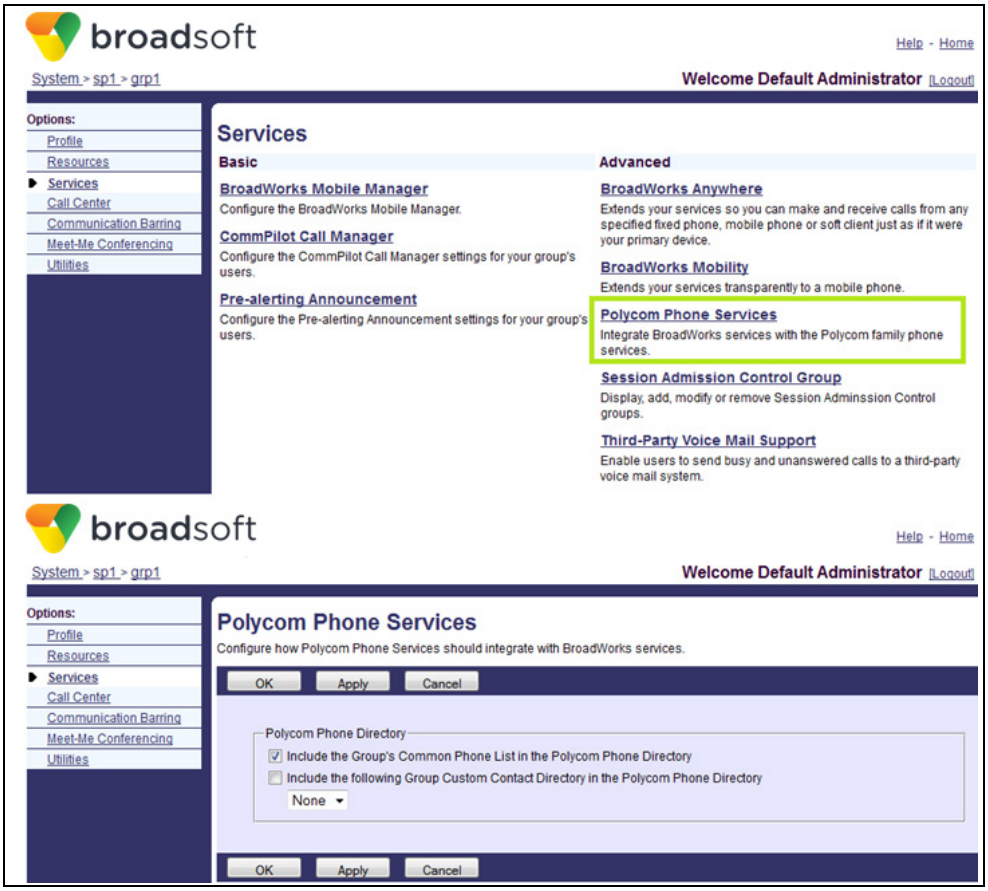


Figure 74 Configuring Polycom Phone Services at Group Level

5.14.4 User Assignment

If a user has been assigned the user service called Polycom Phone Services, then the user is also given the ability to enable and disable the Polycom Phone Directory on the user's phone. The phone directory includes all the contacts in the Polycom Group Phone Directory, set up by the administrator. The user can also personalize the list by doing any of the following:

- Electing to have the user's Personal Phone List included

  -and/or-

- Choosing a Custom Contact Directory to include in the phone directory

Figure 69 shows how to assign the service to a user and *Figure 70* shows how to configure the service for a specific user.

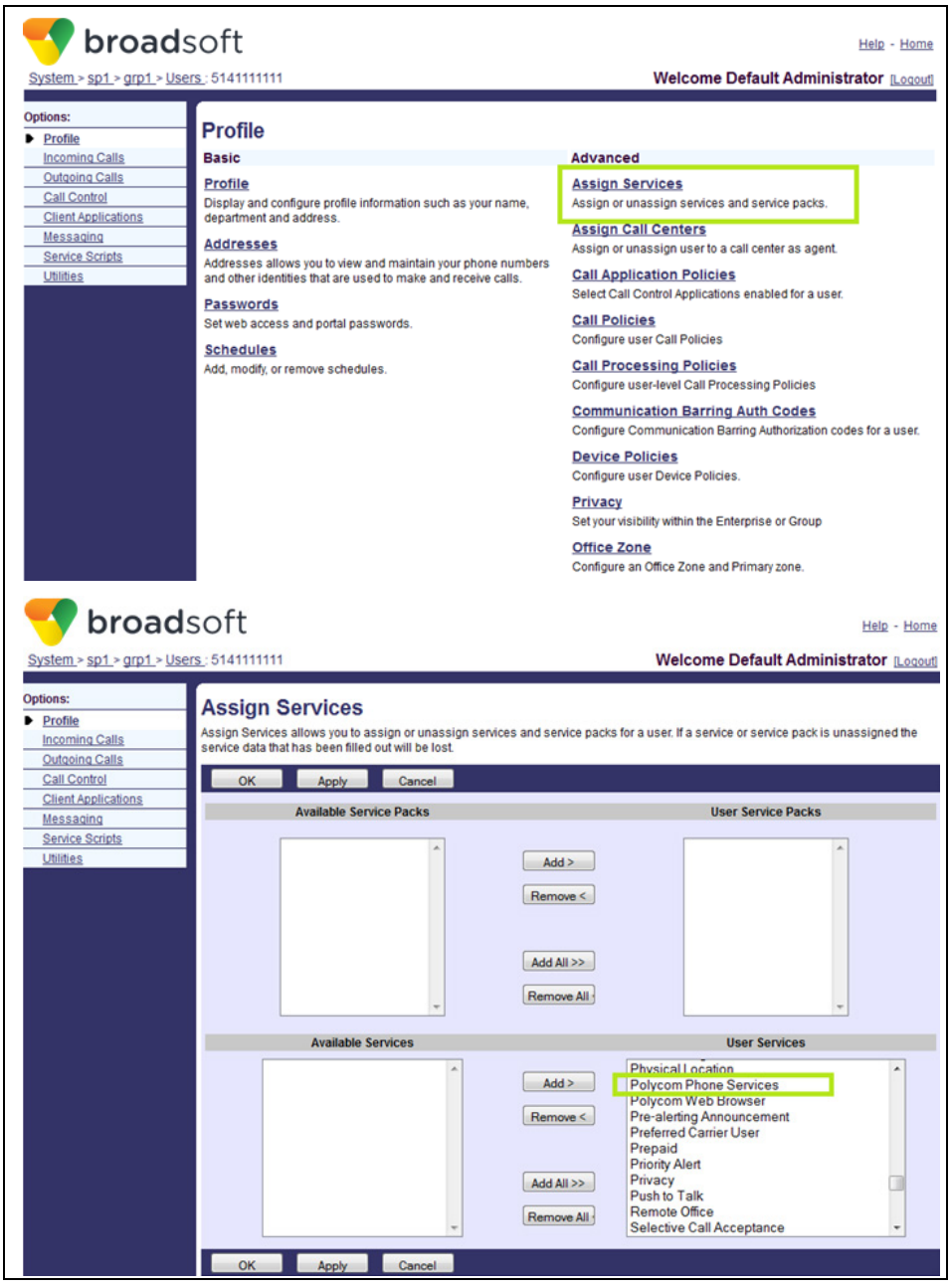


Figure 75 Authorizing Polycom Phone Services to Specific User

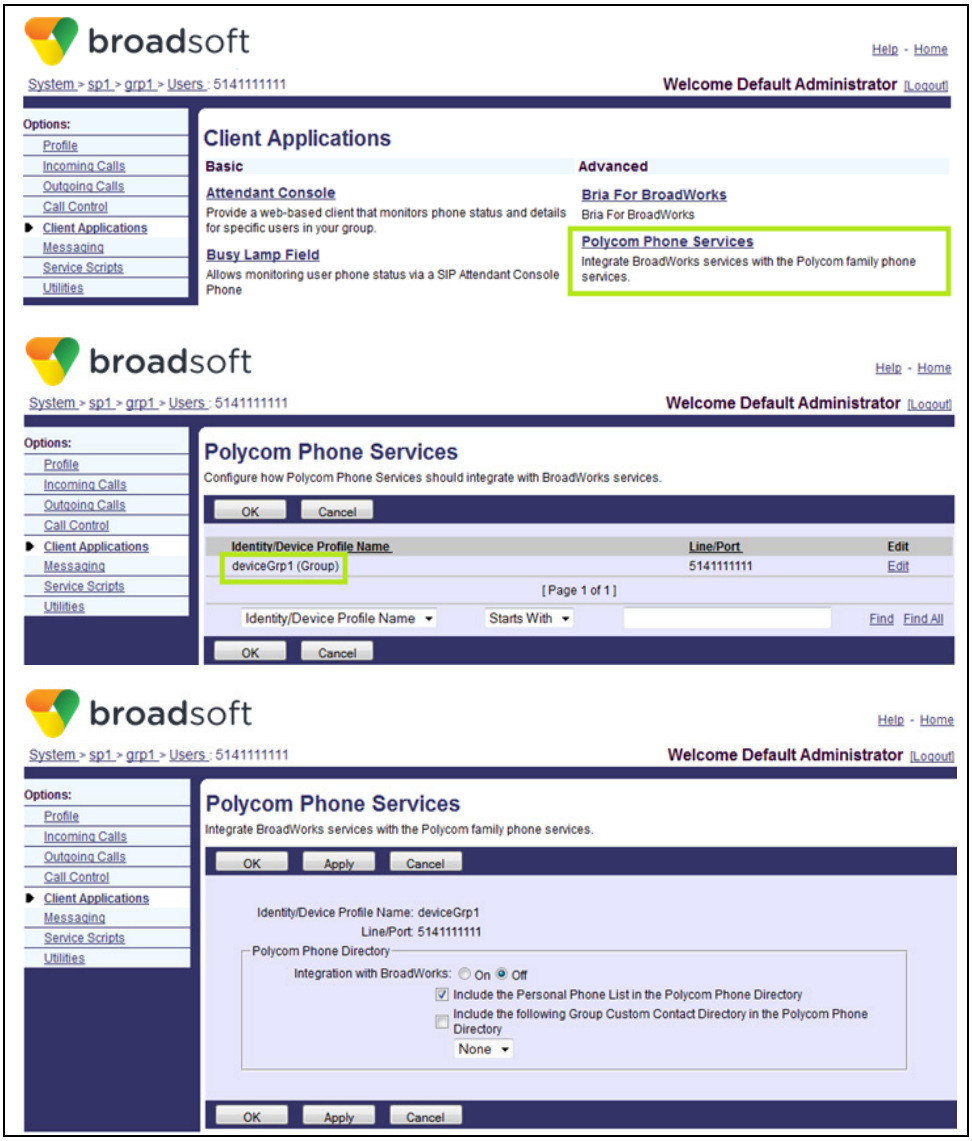


Figure 76 Overriding Polycom Phone Service Configuration at User Level

5.14.5 Directory File

When the administrator sets up the Polycom Phone Services at the device-type level, the system automatically creates a new file using the following attributes:

- File key is *BW_DIRECTORY.*
- File name is *%BWMACADDRESS%-directory.xml*.
- File type is dynamic per profile.
- File repository file name is *%BWFQDEVICEID%-directory.xml*.
- The new device type file inherits the authentication settings (authentication mode, MAC address in, and Device Access HTTP Authentication) of the device type. If the device type does not have any authentication enabled, MAC authentication is enabled.

NOTE: The file is only built for the device profiles that fulfill the following:

1- Polycom Phone Services is enabled for the associated Device Type (see section *5.14.2 Integrate Polycom Phone Directory with Cisco BroadWorks*).

2- The device profile has a user set to "Primary Line/Port" (see section *5.12 Primary User*).

3- The user specified at step 2 has enabled "Integration with BroadWorks" under Polycom Phone Services (see section *5.14.4 User Assignment*).

Also note that given the device access file format for the directory file (*%BWMACADDRESS%-directory.xml*), it is mandatory to specify a MAC address for the device profiles using Polycom Phone Services.

The system automatically loads the template that can be shared by all device profiles. If required, it is also possible for the administrator to load a device profile-specific template if some device profiles are handled slightly differently or the administrator changes the default. The template automatically loaded by the system is as follows.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<directory>
.<item_list>
%BW-MULTI-BLOCK-START%
..<item>
...<ln>%BWLASTNAME-LIST-RIGHT%</ln>
...<fn>%BWFIRSTNAME-LIST-LEFT%</fn>
...<ct>%BWNUMBER-LIST%</ct>
...<sd/>
   <dc/>
   <rt/>
   <ad>0</ad>
   <ar>0</ar>
..</item>
%BW-MULTI-BLOCK-END%
.</item_list>
</directory>
```

As can be seen in the previous example, the feature introduces new dynamic tags as described in *Table 4*. To better understand the tag's behavior for free-format names, assume that the name "John William S. Smith" is defined as a contact.

Table 4 New Dynamic Tags for Polycom Phone Services

Tag Name	Description
%BW-MULTI-BLOCK-START% %BW-MULTI-BLOCK-END%	These tags can be used in a template to mark the start and end of a repeatable block of lines that can be added any number of times when the Provisioning Server builds the file using this template. After the file is built, these tags do not appear in the resulting file. It is not possible to embed a repeatable block within another repeatable block. In addition, a template can only contain zero or one repeatable block.
%BWLASTNAME-CLID-LIST-RIGHT%	This tag represents the last name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks Calling Line ID (CLID) last name defined for the user. Otherwise, the tag is replaced with the last word of the contact name, such as "Smith" in our example.

Tag Name	Description
%BWLASTNAME-CLID-LIST-LEFT%	This tag represents the last name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks CLID last name defined for the user. Otherwise, the tag is replaced with the first word of the contact name, such as “John” in our example.
%BWFIRSTNAME-CLID-LIST-LEFT%	This tag represents the first name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks CLID first name defined for the user. Otherwise, the tag is replaced with all the words of the contact name, except the last word, such as “John William S.” in our example.
%BWFIRSTNAME-CLID-LIST-RIGHT%	This tag represents the first name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks CLID first name defined for the user. Otherwise, the tag is replaced with all the words of the contact name, except the first word, such as “William S. Smith” in our example.
%BWLASTNAME-LIST-RIGHT%	This tag represents the last name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks last name defined for the user. Otherwise, the tag is replaced with the last word of the contact name, such as “Smith” in our example.
%BWLASTNAME-LIST-LEFT%	This tag represents the last name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks last name defined for the user. Otherwise, the tag is replaced with the first word of the contact name, such as “John” in our example.
%BWFIRSTNAME-LIST-LEFT%	This tag represents the first name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks first name defined for the user. Otherwise, the tag is replaced with all the words of the contact name, except the last word, such as “John William S.” in our example.
%BWFIRSTNAME-LIST-RIGHT%	This tag represents the first name of a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with the Cisco BroadWorks first name defined for the user. Otherwise, the tag is replaced with all the words of the contact name, except the first word, such as “William S. Smith” in our example.
%BWNUMBER-LIST%	This tag represents the number associated with a contact in a directory. When a contact in the phone directory links to a user in the Cisco BroadWorks enterprise/group directory, the tag is replaced with: User’s phone number, if present -or- User’s extension, if present -or- User’s SIP URI If the contact does not map to a Cisco BroadWorks user, the tag is replaced by the number defined for the contact in the contact list.

When the Provisioning Server must build the directory file for a given device profile, it fetches the contact entries from the following lists:

- The group's Common Phone List, if the Polycom Phone Directory is configured as such at the group level.
- A group's Custom Contact Directory, if the Polycom Phone Directory has one selected at the group level.
- The user's Personal Phone List, if the Polycom Phone Directory user settings is configured as such.
- A user-selected Custom Contact Directory, if the Polycom Phone Directory has one selected at the user level.

The Provisioning Server then merges all the entries found together. Duplicates are removed. Two entries are considered duplicated if all their fields are the same (first name, last name, number, and so on). The resulting set is also ordered (ascending) when the file is built. Ordering is done on the last name, first name, and number (in this order). Ordering uses the user's locale to provide a natural ordering of entries.

5.14.6 Directory Updates from Devices

The feature is able to support directory uploads by the device. This occurs when a user updates the directory directly on the phone. Seconds after the last directory change, the phone tries to push the updated directory file to Cisco BroadWorks. When the Xtended Services Platform receives the new directory, it authenticates the request similar to any other type of request.

After parsing the received file, the Application Server determines whether an entry must be added to the user's personal phone list. An entry in the phone's directory is considered to already be present on Cisco BroadWorks if the entry exactly matches an entry on one of the phone lists or on one of the selected custom contact directories (set of Cisco BroadWorks users). In the case of phone lists (common or personal), entries are identical if "<fn content>SPACE<ln content>" matches the Cisco BroadWorks phone list name attribute and the <ct content> matches the Cisco BroadWorks phone list *Phone Number* attribute.

If an entry does not exist and if the user is allowed to have the user's personal phone list included in the phone directory, then the Application Server adds this entry to the user's personal phone list. Otherwise, the entry is not added.

NOTE: The Application Server allows a maximum of 1,000 entries (not configurable) to be added to the user's database via the phone directory interface.

5.15 Device File Application Server Location Lookup

This is in the deployment model where there is an Xtended Services Platform (Xsp) that fronts multiple Application Server clusters. The Xtended Services Platform must locate the Application Server, where a device that is requesting a file, is authenticated and authorized. The Device Management system provides three methods by which this operation can be performed:

- Full Application Server lookup
- Meta information for Application Server cluster locations into URLs
- Network Server device file location lookup

5.15.1 Full Application Server Lookup

Using this method, the Xtended Services Platform obtains the list of all Application Server clusters from the Network Server. Using this list, the Xtended Services Platform sends the authorization OCI request to each Application Server until it finds those that can serve the requested file. This method does not require any special configuration. This method is suitable for deployment where there are a low number of Application Server clusters (typically one) or when a lengthy device response time is acceptable (light load on servers).

5.15.2 Include Meta Information for Application Server Cluster Locations into URLs

This second method is a performance enhancement to the full Application Server lookup. It relies on the possibility for the devices to embed meta information on the Application Server cluster in the URL of requested files. Section *5.17.5.1 URI Scheme* describes why a device is redirected to a URL with embedded meta information on its Application Server cluster. Information on the Xtended Services Platform front-end access must be kept in the Application Server. These parameters are usually set when a new device type is created. To modify them for an existing device type from the CLI:

```
AS_CLI/System/DeviceType/SIP>set deviceAccessFQDN <xsp_fqdn>
AS_CLI/System/DeviceType/SIP>set deviceAccessPortNumber <xsp_port>
AS_CLI/System/DeviceType/SIP>set deviceAccessContextName <webapp_context>
```

… where:

- **<deviceAccessFQDN>** represents the FQDN of the Xtended Services Platform used by the device to fetch its files.
- **<deviceAccessPortNumber>** is the port number on the Xtended Services Platform used by the device to fetch its files.
- **<deviceAccessContextName>** is the context name given to the BroadworksDms web application when it was deployed on the Xtended Services Platform (see section *5.17.5.3 BroadworksDms Web Application*).

NOTE: This information must be kept manually synchronized if the Xtended Services Platform deployment changes.

The first time a configuration file is fetched from a generic URL (probably mass configured into the devices prior to deployment), the full Application Server lookup is performed. When the device downloads the file and reads it, it contains the new URL for subsequent accesses. This new URL embeds the Application Server cluster location. On subsequent queries, the device uses the new URL. For this to occur, the template itself contains tags resolved dynamically to rewrite the access URL and match the Application Server cluster on which it was generated.

For example, a Polycom configuration file would need to set the *device.prov.serverName* to the following value:

```
"http://%BWDEVICEACCESSFQDN%:%BWDEVICEACCESSPORT%/%BWDMSCONTEXT%/bw/host/
%BWASCLUSTERFQDN%/%BWFILESERVERDIR% %BWDEVICEACCESSURI%"
```

For more information, see the *SoundPoint IP Soundstation IP Administrator's Guide* (*soundpoint_ip_soundstation_ip_administrators_guide_v2_2.pdf*), page 211, available from http://www.polycom.com/global/siteselector/site_selector.html.

If the Polycom device accessed on the first try is http://xsp_fqdn/dms/Polycom500/myConfiguration.cfg, then this URL is converted to http://xsp_fqdn/dms/bw/host/as_cluster_fqdn/Polycom500/myConfiguration.cfg when the template is generated.

The Polycom device would update its internal flash setting to use the new URL for subsequent accesses, thus avoiding the costly full Application Server lookup.

This method is suitable for deployment where there are multiple Application Server clusters and the devices are able to embed meta information in the URL.

5.15.3 Network Server Device File Location Lookup

This method uses the Network Server as the device file location server. All the Application Server clusters synchronized the following information to the Network Server:

- The device access Uniform Resource Identifier (URI) (one per device type).
- The device access file format (many per device types).
- The device identifier type – MAC, user name or none (one per device access format).
- When the MAC address is not in the URL, but from the HTTP Header then the HTTP header in which it can be found and the pattern to find it (one per device access format), or the pattern to find it when the MAC address is from the client certificate CN.
- When the device user name is used to identify a device, the type of challenge used, basic or digest (one per device access format).
- The device MAC addresses.
- The device user names.

When a request for a file arrives at the Xtended Services Platform, the Xtended Services Platform sends a Network Server portal API command to the Network Server. This command contains the requested URL. Based on the received URL and the data synchronized from the Application Server(s), the Network Server returns the list of potential Application Servers on which the file can reside. To look up a request for a file, the Network Server must associate a URL with one (or many) Application Server clusters. Since individual devices can be identified using their MAC address or a device user name, the Network Server also requires associating a specific device with one Application Server. The combination of both portions of information (URL and MAC/user name device) is used in the algorithm to locate the device file.

The combination of the device access URI and the device access file format files forms the device URL lookup table, in which each row indicates the device identifier type (MAC, user name, none), for the MAC address where it can be found and for user name, the type of authentication used (basic or digest).

The device file location algorithm is as follows.

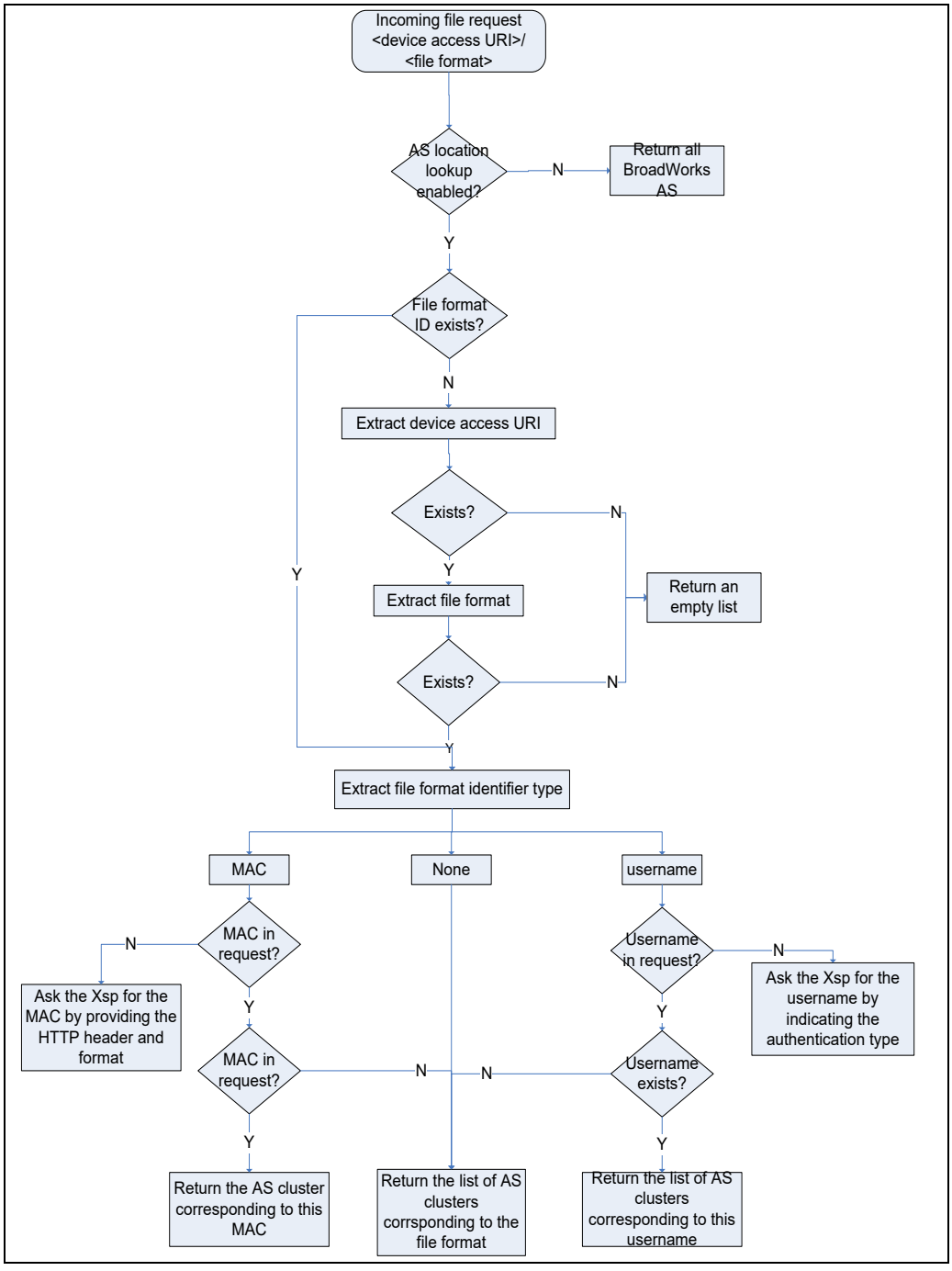


Figure 77 Device File Location Algorithm

When the device access URI contains a wildcard symbol (%BW_WILDCARD%), the algorithm looks up all the file formats corresponding to the device access URI received by replacing the wildcard symbol with a regular expression of type ".*". The first entry that matches the received URL is then used.

NOTE: It is not recommended to use the wildcard symbol (%BW_WILDCARD%) by itself. The recommended approach is to use the wildcard symbol with a prefix, for example, "myPrefix%BW_WILDCARD%". The wildcard symbol without a prefix could potentially match many files, which could lead to unexpected behavior.

As part of the identification of a file format, any dynamic tags are replaced with a general regular expression of type ".*". Only the %BWMACADDRESS% tag is replaced with "(.*)", which indicates to extract it.

This method is suitable for deployment where there are multiple Application Server clusters and the devices are not capable of embedding meta information in their URL. It can also be combined with the previous method to replace the initial full location Application Server lookup and provide the most efficient device file location algorithm.

This method requires the configuration that follows.

5.15.3.1 Application Server

On the Application Server, synchronizing the device file information is configurable through the CLI under *AS_CLI/Interface/NetServSync* as follows.

```
AS_CLI/Interface/NetServSync> set syncFlag true syncDeviceManagementInfo
true
```

5.15.3.2 Network Server

On the Network Server, the device file location lookup is enabled using the CLI under *NS_CLI/System/DeviceManagement* as follows.

```
NS_CLI/System/DeviceManagement> set asLocationLookupEnabled true
```

5.16 Deployment Models

The following figures show the typical deployments for Device Management. *Figure 72* shows the FTP deployment. Deploying with an FTP server is fundamentally less secure for the following reasons:

- The password is sent clear text over the network (no SSL or Secure File Transfer Protocol [SFTP] support).
- There is no authentication challenge available.

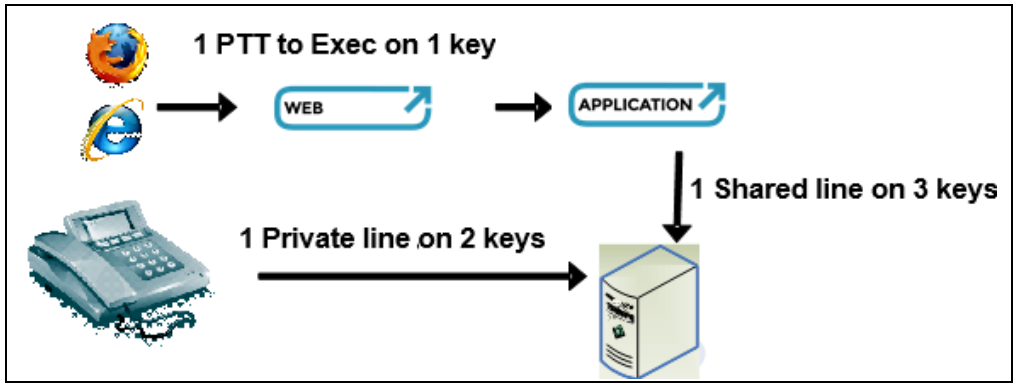


Figure 78 Legacy FTP Deployment

Figure 73 shows a standard deployment using Xtended Services Platform, Network Server, and Profile Server farms. The Xtended Services Platform servers may need to talk to a Network Server before they can contact the appropriate Application Server. If the information as to which Application Server cluster to use is missing from the phone HTTP/TFTP request, the Xtended Services Platform asks a list of all the Application Server and polls them to find the appropriate one.

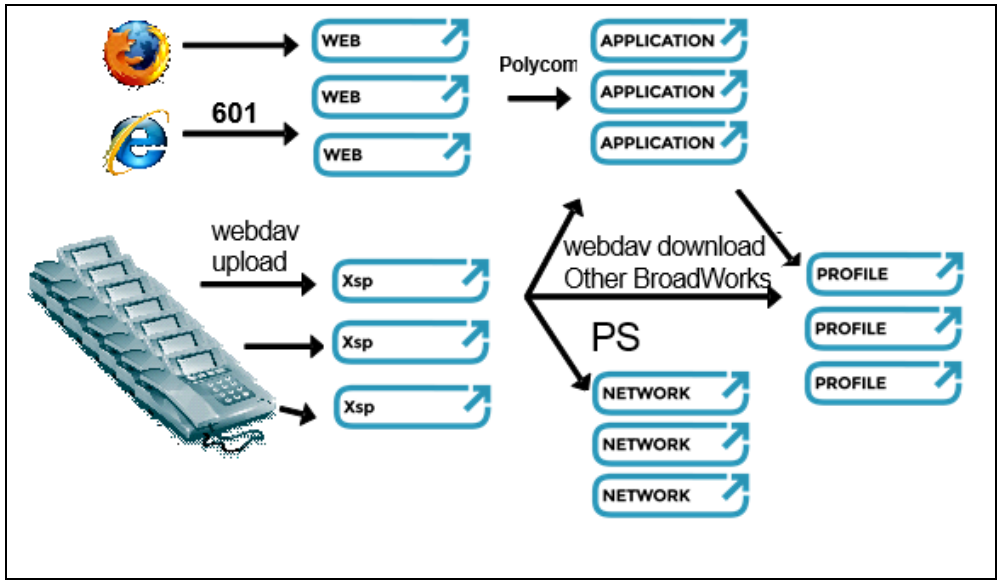


Figure 79 Standard Deployment with Xtended Services Platform, Network Server, Application Server, and Profile Server Farms

Figure 74 shows a special deployment in which only a single Application Server cluster is available. When all provisioning is accessed on a single Application Server cluster, the Xtended Services Platform does not require the presence of a Network Server.

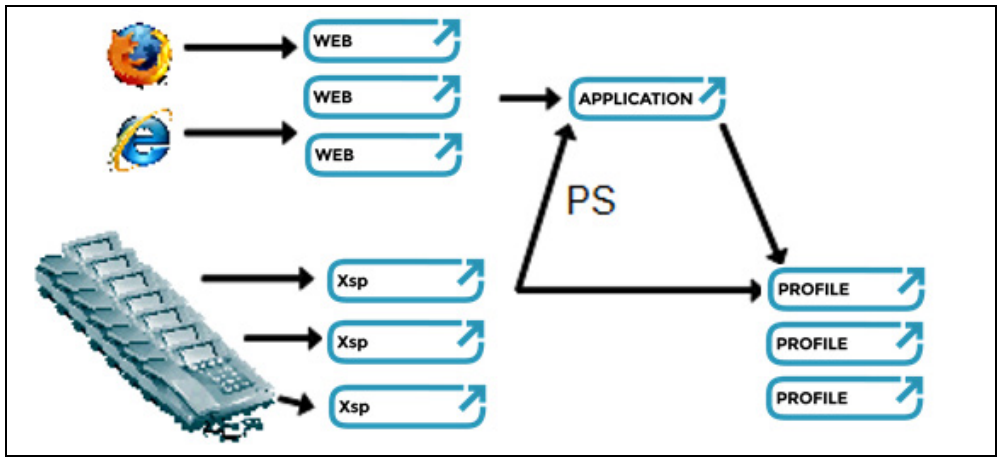


Figure 80 Single Application Server Deployment (Application Server Mode)

5.16.1 Dedicated Repository

A dedicated repository bypasses the Network Server and Application Server lookups, which Device Management uses to determine where a phone's data resides.

For a dedicated repository to work properly, all phone types must have a common directory on the Profile Server (PS) where the configuration files are kept.

In the following subsections, it is assumed that the directory called "sccp" is used on the Profile Server to store all SCCP data files.

5.16.1.1 Xtended Services Platform

Enable the dedicated repository functionality. In the following procedure, replace the host address by the actual address of the Profile Server and set the root directory to "sccp".

1) On the Xtended Services Platform, enable the dedicated repository functionality as follows.

```
XSP_CLI/Applications/BroadworksDms/DedicatedRepository> get
enabled = false
requestType = tftp
host =
port = 80
rootDirectory = /
secure = false
username =

XSP_CLI/Applications/BroadworksDms/DedicatedRepository> set enabled true
XSP_CLI/Applications/BroadworksDms/DedicatedRepository> set requestType
all
XSP_CLI/Applications/BroadworksDms/DedicatedRepository> set host
10.16.105.16
XSP_CLI/Applications/BroadworksDms/DedicatedRepository> set rootDirectory
/dedicatedDir

XSP_CLI/Applications/BroadworksDms/DedicatedRepository> get
enabled = true
requestType = all
host = 10.16.105.16
port = 80
rootDirectory = /sccp
secure = false
username =
```

2) Clear the *deviceAccessUri* of any wildcards. In the following example, it is *CiscoSccp%BW_WILDCARD%*.

```
XSP_CLI/Applications/DeviceManagementTFTP/GeneralSettings> get
serverPort = 69
clientPortLow = 30000
clientPortHigh = 30100
deviceAccessContextName = dms
deviceAccessUri = CiscoSccp%BW_WILDCARD%
maxActiveTransfer = 999
XSP_CLI/Applications/DeviceManagementTFTP/GeneralSettings> clear
deviceAccessUri
...Done
XSP_CLI/Applications/DeviceManagementTFTP/GeneralSettings> get
serverPort = 69
clientPortLow = 30000
clientPortHigh = 30100
deviceAccessContextName = dms
```

```
deviceAccessUri =
maxActiveTransfer = 999
```

3) Restart the Xtended Services Platform.

5.16.1.2 Application Server

In a dedicated repository deployment, the Application Server is set up to point files to the Profile Server based on the various configured folders for the Device Type profile. All of these folders must point to the same directory.

1) Using the Application Server CLI, list the Profile Server directories where the files are stored.

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> get
                Device Type      File Repository Name  Directory
================================================
Cisco Sccp 7940(G)         ProfileServer
CiscoSccp7940G
Cisco Sccp 7960(G)         ProfileServer                  CiscoSccp7960G
Cisco Sccp 7941G(E)        ProfileServer                  CiscoSccp7941G
Cisco Sccp 7961G(E)        ProfileServer                  CiscoSccp7961G
Cisco Sccp 7970G(E)        ProfileServer                  CiscoSccp7970G
Cisco Sccp 7971G           ProfileServer                  CiscoSccp7971G
Cisco Sccp 7942G           ProfileServer                  CiscoSccp7942G
Cisco Sccp 7945G           ProfileServer                  CiscoSccp7945G
Cisco Sccp 7962G           ProfileServer                  CiscoSccp7962G
```

2) Write down the directory names listed for each phone type. These names are used to modify the Profile Server.

3) Change the directory names to a common name for all the device types.

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> set "Cisco Sccp 7940(G)"
directory sccp
...Done
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> get
Device Type      File Repository Name    Directory
================================================
Cisco Sccp 7940(G)        ProfileServer                sccp
Cisco Sccp 7960(G)        ProfileServer                sccp
Cisco Sccp 7941G(E)       ProfileServer                sccp
Cisco Sccp 7961G(E)       ProfileServer                sccp
Cisco Sccp 7970G(E)       ProfileServer                sccp
Cisco Sccp 7971G          ProfileServer                sccp
Cisco Sccp 7942G          ProfileServer                sccp
Cisco Sccp 7945G          ProfileServer                sccp
Cisco Sccp 7962G          ProfileServer                sccp
```

4) Restart the Application Server.

5.16.1.3 Profile Server

Create the *sccp* directory on the Profile Server and in the *sccp* directory, create symbolic links to each SCCP phone type directory. Make the symbolic links point back to the *sccp* directory.

1) Log in to the Profile Server shell and change the directory path to the file repository.

```
bwadmin@e1vm$ cd /var/broadworks/fileRepos
```

2) Create a directory called “sccp”.

```
bwadmin@e1vm$ mkdir sccp
```

3) Go to the newly created directory.

```
bwadmin@e1vm$ cd sccp
```

4) Using the list of SCCP phone types that you wrote down in the previous procedure, create a symbolic link for each SCCP phone type.

```
bwadmin@e1vm$ ln -s /var/broadworks/fileRepos/sccp CiscoSccp7940
```

5) Perform a directory listing of the *sccp* directory. The output should look similar to the following.

```
bwadmin@e1vm$ ls -l Cisco*
lrwxrwxrwx 1 bwadmin 35 Aug 2 15:12 CiscoSccp7940 ->
/var/broadworks/fileRepos/sccp
lrwxrwxrwx 1 bwadmin bwadmin 35 Aug 3 09:18 CiscoSccp7960 ->
/var/broadworks/fileRepos/sccp
...
```

5.17 System Configuration

5.17.1 System Planning Overview

As shown in *Figure 75*, eight major steps are required to enable the configuration profile management capabilities of Cisco BroadWorks Device Management. You may want to skip the installation of the servers or the migration steps if you are starting from an existing base. These steps are guidelines for the rest of this section.

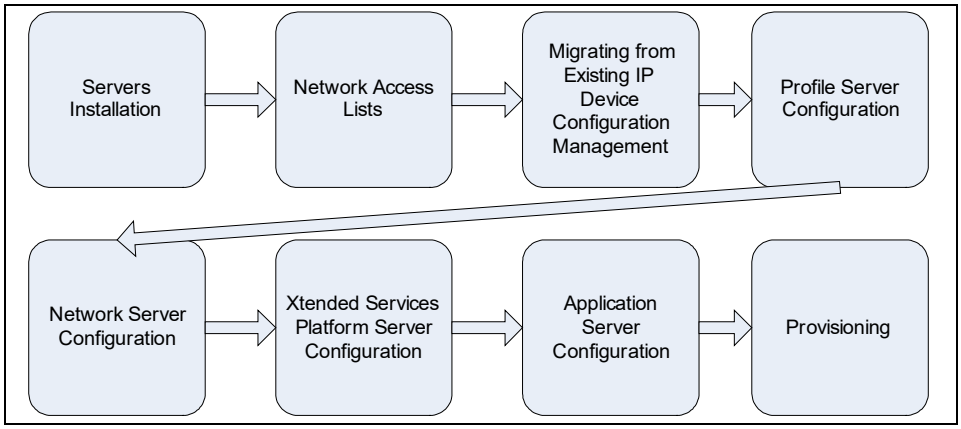


Figure 81 High-level Steps to Set Up Device Management Installation

5.17.2 Install Cisco BroadWorks on Servers

For more information about the installation procedure, see the *Cisco BroadWorks Software Management Guide* [1].

5.17.3 Position Servers on Network

Figure 76 shows how a full installation would look.

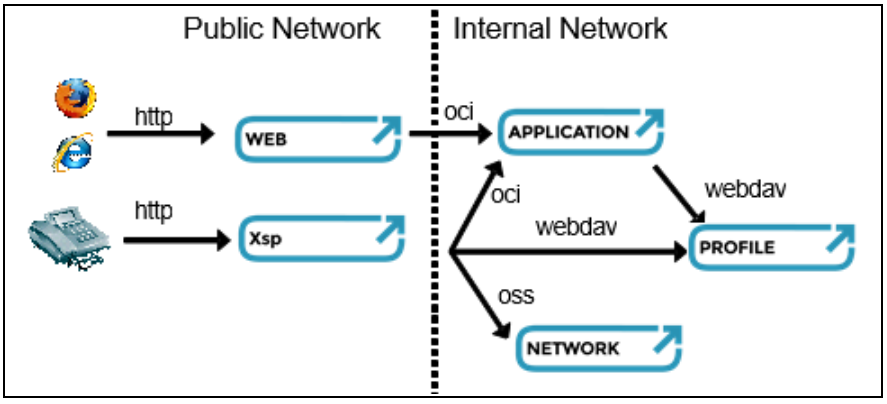


Figure 82 Typical Installation View

You can avoid installing the following servers if you do not require all the functionality as follows:

- If the provisioning/configuration of Device Management is entirely done via the Application Server CLI, you can avoid installing/using the Web Server.
- If you are using a single Application Server cluster, you can avoid installing/using the Network Server.

- If you are using an FTP repository, you can avoid installing/using the Xtended Services Platform, Network Server, and Profile Server.

5.17.4 Network Access Lists

All the servers must communicate with each other. This is done via the access lists through the CLI for each server as follows:

1) On the Application Server, allow the Xtended Services Platform to make OCI requests, as follows.

```
AS_CLI/System/NetworkAccessLists/OCI/Provisioning>add <ip of xsp>
```

2) On the Profile Server, allow the Xtended Services Platform to make WebDAV requests, as follows.

```
PS_CLI/Applications/BroadworksFileRepos_16.0/NetworkAccessLists/WebDav>ad
d <ip of xsp>
```

3) On the Profile Server, allow the Application Server to make WebDAV requests, as follows.

```
PS_CLI/Applications/BroadworksFileRepos_16.0/NetworkAccessLists/WebDav>ad
d <ip of as>
```

4) If you are using a Network Server, add the Xtended Services Platform IP address to the Network Server portal API access control list, as follows.

```
NS_CLI/System/NetworkAccessLists/PortalAPI>add <ip of xsp>
```

5.17.5 Xtended Services Platform Configuration

The Xtended Services Platform is required for HTTP/HTTPS and TFTP device file access. Devices are configured to point to an HTTP or TFTP address resolving to any Xtended Services Platform in a farm.

As the front end for file requests, the Xtended Services Platform server handles download/upload requests from devices. To successfully serve these requests, the Xtended Services Platform communicates with the back-end servers. This section covers the BroadworksDms web application, the DeviceManagementTFTP application, and the BWCommunication configuration. The following figure shows the Xtended Service Platform (Xsp) in a typical Device Management deployment:

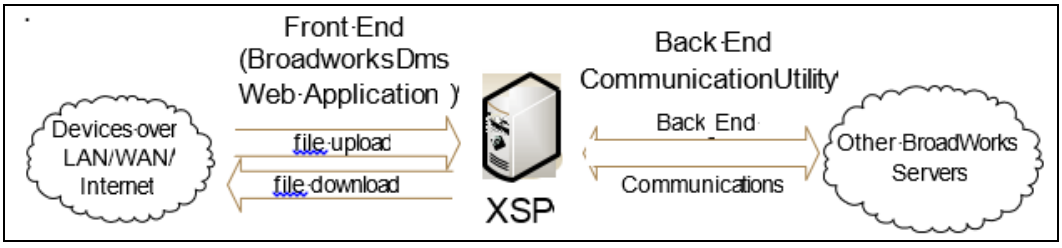


Figure 83 Xtended Services Platform Viewed as Bridge Relaying Requests to Other Servers in Back End

5.17.5.1 URI Scheme

This section explains the URI scheme for multiple Application Server deployment. If you only have a single Application Server cluster, you can skip this whole section. If you are looking for the CLI commands to actually configure the URIs for device access on the Xtended Services Platform, see section *5.17.5.2 HTTP Configuration*. Conceptually speaking, there are two important concepts related to URIs:

- A URI can embed meta information about the back-end topology. When multiple Application Servers are deployed, the Xtended Services Platform must know which Application Server to ask for the requested files. To do this, special tokens in the URI inform the Xtended Services Platform as to which Application Server cluster to use. If these tokens are missing from the URI, the Xtended Services Platform performs a full lookup on the back end and tries to locate the proper Application Server for the requested files.

NOTE: Once a device successfully fetches its files for the first time, the configuration files embed the proper meta information into the access URI. This avoids the full Application Server lookup tax on subsequent requests from a same device. For information on how to configure this for the Application Server, see section *5.14 Phone Services*.

- Each unique URI represents a specific version of a file. If two devices share a common file name but their files are different, their URIs must be different. URIs are case-insensitive.

A typical URI with meta information has the following “look”.

```
<protocol>://<xsp_fqdn>/<dms_context_name>/bw/host/<as_cluster_fqdn>/<dev
ice_type_path>/<device_resource>
```

… where:

- **<protocol>** is the protocol to be used (HTTP, HTTPS, or TFTP).
- **<xsp_fqdn>** is an alias resolving to the Xtended Services Platform farm FQDN. Many different aliases can be used to point to the same farm.
- **<dms_context_name>** is the configurable context name for the BroadworksDms web application. This token must be unique for the BroadworksDms web application because the Xtended Services Platform can host other web applications as well. In addition, this token must be manually synchronized between the BroadworksDms web application and the Application Server provisioning of device files. For information on how to configure this for the Xtended Services Platform, see section *5.17.5.2 HTTP Configuration*. For information on how to configure this for the Application Server when a device type is created, see section *5.3.2.2 Device Management Options*.
- **bw/host** is the non-configurable keyword that Cisco BroadWorks uses to specify that the next token in the URI contains meta information about the back-end topology. The next token informs the Xtended Services Platform as to the exact Application Server cluster FQDN alias that contains the provisioning information for the requested files.

- **<as_cluster_fqdn>** is the configurable Application Server cluster FQDN on which the requesting device information has been provisioned. The device embeds this meta information in the requested URI if configured to do so manually. Alternatively, this meta information is embedded in the configuration files by the Application Server. The device should switch from a generic URI to a new URI with meta information on its Application Server cluster after it first fetches its configuration file.
- **<device_type_path>** is the unique path per device type that allows a device to gain access to the appropriate set of files. This path is configurable on a device type-basis. It can be empty or it can contain one or more directory names. A good name would be the device type name, for example "polycom_soundpoint_ip_500".
- **<device_resource>** is the actual file name that the device wants to "get".

5.17.5.2 HTTP Configuration

From the CLI, make sure that the HTTP parameters are properly configured.

```
XSP_CLI/Interface/Http/HttpServer> get

    Interface    Port                Name    Secure         Cluster FQDN
=============================================================================
192.168.12.90      80    xsp1.example.com    false    web.broadsoft.com
```

Note that for Device Management with authorization based on MAC address in client certificate to work, *clientAuthReq* must be enabled to request a certificate from clients. If it is preferred that the certificate is not required for all clients on the server, *clientAuthReq* can be enabled for the *BroadworksDMS* application only.

```
XSP_CLI/Interface/Http/ClientAuthentication/WebApps> get

       Interface  Port  Application Name  Client Auth Req
=============================================================
192.168.12.90      443     BroadworksDms             true
```

Also, the *XSP_CLI/Interface/Http/ClientAuthentication/WebApps/Resources>* level can be used to enable *clientAuthReq* on specific URLs if needed.

If required, define aliases.

```
XSP_CLI/Interface/Http/HttpAlias> get

    Interface              Alias          Cluster FQDN
=========================================================
192.168.12.90    my1.example.com        my.example.com
```

It is also important to note that certain devices do not support a redirect response (HTTP status code 302) sent back from the Xtended Services Platform. The redirect response can occur when the device is provisioned to contact the Xtended Services Platform via the Xtended Services Platform farm FQDN, but this FQDN is not provisioned as an alias within the Xtended Services Platform.

In this case, it is recommended that an alias for the Xtended Services Platform farm FQDN be defined using the CLI, thus allowing it to accept the request from the device without sending back a 302 status code.

Example:

```
XSP_CLI/Interface/Http/HttpAlias> get

      Interface            Alias  Cluster FQDN
================================================
            ALL              XSP
            ALL  xsp.example.com
```

In the previous example, the Xtended Services Platform farm FQDN is *xsp.example.com*, which is the same value that is defined on the Application Server as the *deviceAccessFQDN* parameter set for the device-type level.

```
AS_CLI/System/DeviceType/SIP>set <device-type> deviceAccessFQDN
xsp.example.com
```

This value can then be referred to in device configuration files using the %BWDEVICEACCESSFQDN% tag.

For more information about the HTTP configuration and enabling an HTTPS secure interface, see the *Cisco BroadWorks Xtended Services Platform Configuration Guide* [2].

NOTE: Most devices do not support self-signed certificates. For more information, see your respective devices' integration guide.

5.17.5.3 BroadworksDms Web Application

The BroadworksDms web application must be manually activated and deployed. From the CLI, activate and deploy the BroadworksDms web application under the context name of "dms". You can use any contextual name you wish to (shorter is better) as long as it does not conflict with other web applications. Use the "get broadworks full" command to see all deployed web applications.

```
XSP_CLI/Maintenance/ManagedObjects> activate application BroadworksDms
18.0_1.179 /dms
XSP_CLI/Maintenance/ManagedObjects> deploy application /dms
```

NOTE: The version you enter (in the previous example, 18.0_1.179), depends on the software version you have.

You can test that the BroadworksDms web application has been properly deployed by pointing your browser at *http://<xsp_fqdn>/dms/test*. Because you only configured the front-end communication and not the back-end communication (yet), you should see:

```
500 Internal Server Error
```

For information, see the server logs.

For common configuration issues, see section *9.4 Common Problems*.

5.17.5.4 DeviceManagementTFTP Application

The DeviceManagementTFTP application must be manually activated and deployed to use the TFTP. The application must also be started. Use the "get broadworks full" command to see all deployed applications.

```
XSP_CLI/Maintenance/ManagedObjects> activate application
DeviceManagementTFTP 18.0_1.179
```

```
XSP_CLI/Maintenance/ManagedObjects> deploy application
DeviceManagementTFTP
XSP_CLI/Maintenance/ManagedObjects>start application DeviceManagementTFTP
```

NOTE: The version you enter (in the previous example, 18.0_1.179), depends on the software version that you have.

The DeviceManagementTFTP must be configured to point to the BroadworksDms web application deployed context path.

```
XSP_CLI/Applications/DeviceManagementTFTP_16.0_1.517/GeneralSettings>
set deviceAccessContextName dms
```

The following information is also configurable through the CLI or through the Element Management System (EMS) using the configuration system:

- Device access URI
- Maximum simultaneous requests
- TFTP server port
- TFTP client ports range (used as the source UDP port to respond to TFTP requests)

The device access URI is optional and should be present if one is specified for the device type using the TFTP. A device URI can help speed up the process of locating the Application Server on which a specific file resides. The device access URI contains wildcard symbols (%BW_WILDCARD%). The wildcard symbol is used to replace one or many characters in the device access URI, and allows the Device Management TFTP application to point to many device type URIs. For example, one can define a device access URI this way: Type1%BW_WILDCARD%. This device access URI can point to Type1Model1, Type1Model2, and so on.

When the device requests a file, for example, <MAC Address>.cfg, the request sent to the Device Management servlet would be Type1%BW_WILDCARD%/<MAC Address>.cfg. The code in the Network Server and Application Server recognizes the wildcard symbol and performs its search based on the potential replacement for it. Since the MAC address is unique, a single entry is returned. For static files that are common to all models (with common content), many entries correspond to the device access URI. In this case, the first one the system found is returned. For static files that are specific to a model, only a single entry is found and returned.

Note that the wildcard symbol is especially useful for SCCP devices as there is no ability to specify a device access URI in these devices. The wildcard symbol can be used to allow the definition of many SCCP device types. So, in the case of SCCP devices, the device access URI is set on the Xtended Services Platform.

This is not the case for non-SCCP devices. These devices generally allow the specification of a device access URI and therefore, the device access URI is not specified in the Xtended Services Platform.

It poses a problem if a deployment is to support SCCP and non-SCCP devices at the same time. In one case, the device access URI is specified in the Xtended Services Platform and in the other, it is not set. The solution is to use two Xtended Services Platforms, one for SCCP devices and one for non-SCCP devices.

NOTE: It is not recommended to use the wild card symbol (%BW_WILDCARD%) by itself. The recommended approach is to use the wild card symbol with a prefix, for example, "myPrefix%BW_WILDCARD%". The wild card symbol without a prefix could potentially match many files, which could lead to unexpected behavior.

For example, consider the following device types and files in the system.

```
DeviceType 1
Access Device URI: /device_type_1
File format: static.cfg
Authentication mode: User Name and Password
File location: AS1

DeviceType 2
Access Device URI: /device_type_2
File format: static.cfg
Authentication mode: User Name and Password
File Location: AS1

DeviceType 3
Access Device URI: /sccp_device_type_3
File format: static.cfg
Authentication mode: none
File location: AS2

DeviceType 4
Access Device URI: /sccp_device_type_4
File format: static.cfg
Authentication mode: none
File location: AS2
```

Examine what happens when trying to retrieve the file for the device type 3 or 4. If a wildcard without a prefix is used for the device access URI, the system could return a file either for device type 1, 2, 3, or 4 since it returns the first match found.

Trying to locate the *static.cfg* file from the Network Server device file location could also lead to being directed to the wrong Hosting NE (when asLookupEnabled is enabled), or retrieving the wrong file on the wrong Hosting NE (when asLookupEnabled is not enabled) since using the wildcard without a prefix means trying to randomly find a matching file on the full list of Hosting NEs returned by the Network Server.

Also note that the files for device type 1 and 2 have authentication enabled. Therefore, the retrieval of the file could fail given the fact of expecting the file for device type 3 or 4, which do not have any authentication enabled.

You can test that the DeviceManagementTFTP application has been properly deployed by trying to do a TFTP get on *test*. Because you only configured the front-end communication and not the back-end communication (yet), you should see the following.

```
500 Internal Server Error
```

For information, see the server logs.

5.17.5.5 Back-end Communication

From an Xtended Services Platform perspective, there are two main deployment scenarios for back-end servers. This section describes the configuration for both:

Single Application Server cluster – In this scenario, all files are provisioned on the same Application Server cluster. Any Xtended Services Platform always communicates with the same Application Server cluster no matter what the requested URI is.

```
XSP_CLI/System/CommunicationUtility/DefaultSettings> set mode AS
<as_cluster_fqdn>
```

Multiple Application Server clusters – In this scenario, the provisioning of files is spread across multiple Application Server clusters. Any Xtended Services Platform must extract a specific Application Server cluster FQDN from the meta information in the URI or by doing a full lookup on all Application Server clusters returned by the Network Server.

```
XSP_CLI/System/CommunicationUtility/DefaultSettings> set mode NS
<ns_cluster_fqdn>
```

After you restart Cisco BroadWorks on the Xtended Services Platform, you can test the back-end communication by pointing the browser to http://<xsp_fqdn>/dms/test. You should see the following.

```
HTTP Status 404
```

General Communication Utility configurations impact Device Management back-end communications. For example, the standard Communication Utility settings provide overload control through the following CLI settings.

```
XSP_CLI/System/CommunicationUtility/DefaultSettings> set
userTransactionLimit <nb of user requests>

XSP_CLI/System/CommunicationUtility/DefaultSettings> set
transactionLimitPeriodSecs <seconds>
```

NOTE: The user transaction limit is calculated per user and not per phone, that is, all requests from all devices using the same authenticated user are bound by a unique user transaction limit.

For common configuration issues, see section *9.4 Common Problems*.

5.17.6 Profile Server Configuration

As a central file repository, the Profile Server cluster handles upload/download requests for files from the Application Server and Xtended Services Platform servers. The following figure illustrates the Profile Server in a typical deployment.

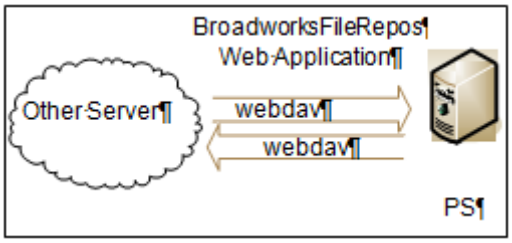


Figure 84 Profile Server as Centralized File Repository Serving Cisco BroadWorks Servers

From the CLI, make sure that the HTTP parameters are properly configured.

```
PS_CLI/Interface/Http/HttpServer> get

Interface   port  Name      Secure
```

```
=====================
ALL          80     192.168.12.94    false
```

Make sure that you have the proper aliases defined.

```
PS_CLI/Interface/Http/HttpAlias> get

Interface     Alias      Cluster FQDN
=======================================
ALL             PS        ps.example.com
```

For more information about the HTTP configuration, see the *Cisco BroadWorks Profile Server Configuration Guide* [3].

5.17.6.1 File Repository Web Application

The *BroadworksFileRepos* web application is automatically deployed when the Profile Server is installed. It handles the WebDAV protocol. You can ensure that the web application is properly deployed via the CLI as follows.

```
PS_CLI/Maintenance/ManagedObjects>get broadworks full
```

You should see:

```
                Name      Version Context Path Deployed
=========================================================
...
BroadworksFileRepos  18.0_1.179            /      true
...
```

NOTE: The version you see (in the previous example, 18.0_1.179), depends on the software version that you have.

The file repository behavior can be customized with the following options.

```
PS_CLI/Applications/BroadworksFileRepos/General> get
rootDirectory = /var/broadworks/fileRepos/
userAuthentication = none
```

… where:

- **rootDirectory** is the path where all the files are stored.
- **userAuthentication** can be set to “none”, “basic”, or “digest”. It controls the HTTP authentication that is performed when a client tries to make a WebDAV request on the server. For example, when the Xtended Services Platform must fetch a file from the Profile Server, it is “digest-challenged” by the server if this item is configured to “digest”.

IMPORTANT NOTE: The *fileRepos* user authentication is enforced only when the *userAuthentication* field is set to “basic” or “digest”.

```
PS_CLI/Applications/WebContainer/Tomcat/OverloadProtection/Server> get
period = 1
limit = 100
```

```
PS_CLI/Applications/WebContainer/Tomcat/OverloadProtection/Webapps> get
                 Name   Period  Limit
=====================================
  BroadworksFileRepos       50     2500
```

Overload Protection – Server

- **limit** is the maximum number of transactions for all clients for a given period of time. The server answers with HTTP *503 Service Unavailable* when this limit is exceeded.
- **period** controls the unit of time used to calculate the global transaction limit. The number of transactions is allowed within this time limit.

Overload Protection - Webapps

- **limit** is the maximum number of transaction per client for the configured webapp, for a given period of time. The server answers with HTTP *503 Service Unavailable* when the limit is reached.
- **period** controls the unit of time used to calculate the number of transaction for the configure webapp. The number of transactions is allowed within this time limit.

The Profile Server supports multiple users and permissions that can be configured via the CLI as follows.

```
PS_CLI/Applications/BroadworksFileRepos/Users>add <username>
<permissions>
```

The user on the Profile Server must match what is to be provisioned on the Application Server, that is, the selected credentials on the Application Server must match a user on the Profile Server who has sufficient rights for the requested operation.

You can test that the Profile Server is responding by pointing your browser at http://<ps_fqdn>/ test. You should see the following.

```
HTTP Status 403 (if the IP from which you made the request is not on the
access list) or HTTP Status 404
```

For common configuration issues, see section *9.4 Common Problems*.

5.17.7 Network Server Configuration

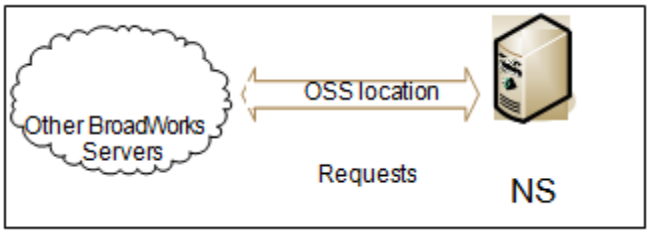


Figure 85 Network Server Configuration

There is no configuration specific to Device Management on the Network Server. If the deployment includes a Network Server, there must be standard configuration; however, there is nothing additional or particular to be done. If the Network Server is already deployed, you can skip this section.

The Network Server must know all the Application Servers (there may be many entries, one for each machine in each cluster in a farm).

```
NS_CLI/System/Device/HostingNE/Address>add <host name> <cluster id> <ip
address> <type>
```

Make sure that the Network Server knows the IP address of all the Xtended Services Platforms (Xsp). The address must be added to the Network Server portal API access control list (ACL).

```
NS_CLI/System/NetworkAccessLists/PortalAPI> get
```

For information on adding the Xtended Services Platform IP address to the Network Server portal API ACL, see section *5.17.4 Network Access Lists*. For more information, see the *Cisco BroadWorks Network Server Command Line Interface Administration Guide* [4].

5.17.8 Application Server Cluster Configuration

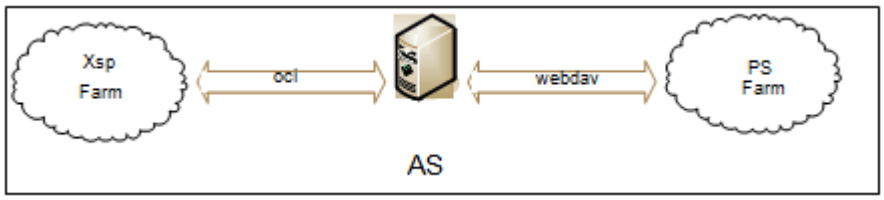


Figure 86 Application Server in Typical Deployment for Device Management

The Application Server holds the bulk of the Device Management configuration. This section employs a use case-based approach. If you must search for a specific web or CLI setting, see the web administration or CLI administration guide, which lists them all in detail. The Application Server answers queries from the Xtended Services Platform as to which Profile Server contains the requested files. It also deals with the provisioning of all the files.

5.17.8.1 CLI Configuration Overview

The following figure shows the Application Server CLI menus introduced by the Device Management feature.

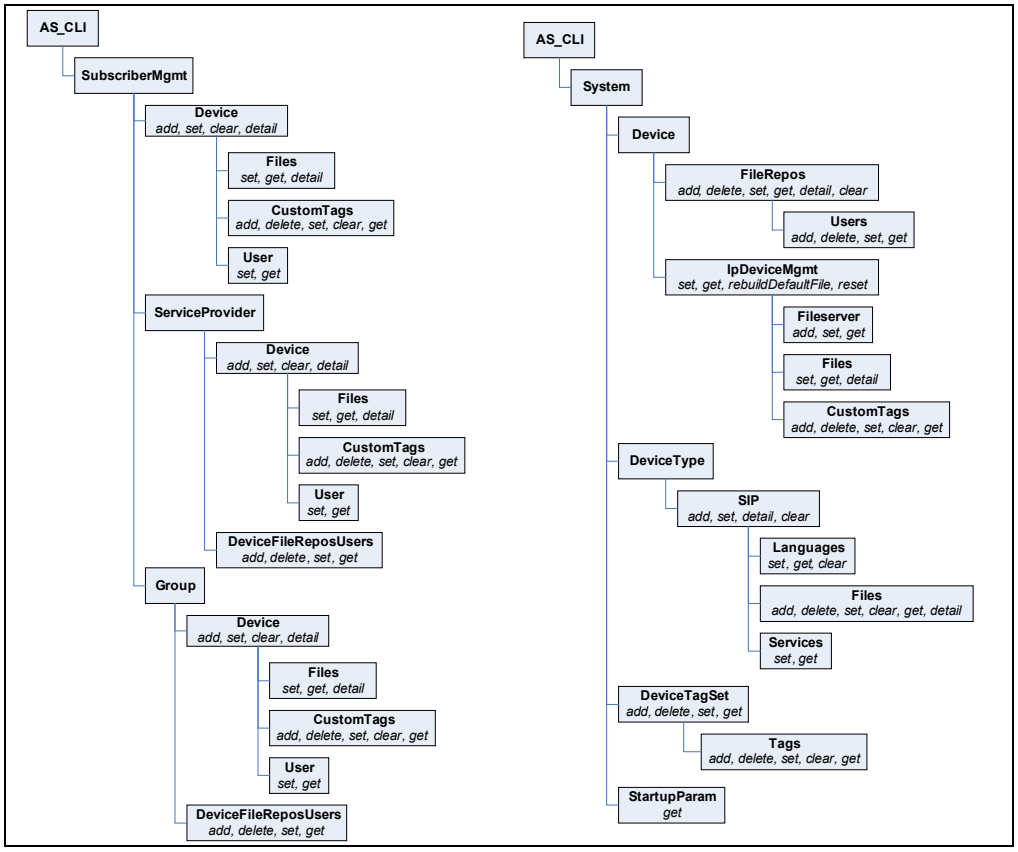


Figure 87 Overview of Device Management CLI Structure on Provisioning Server

5.17.8.2 Device Management Settings

```
AS_CLI/System/Device/IpDeviceMgmt> set […]
```

… where:

- **<enableIPDeviceMgmt>** indicates whether or not Device Management is enabled.
- **<ftpConnectTimeoutSeconds>** contains the connection timeout for the file repository connection (either FTP or the Profile Server).
- **<ftpFileTransferTimeoutSeconds>** contains the communication timeout for transfer to the file repository (either FTP or the Profiler Server).
- **<pauseBetweenFileRebuildMilliseconds>** contains a value in milliseconds to throttle the sending of files from the Application Server to the file repository. The Application Server waits (this value) between each time a file is sent.
- **<maxBusyTimeMinutes>** is the maximum time Device Management should take to process all the file rebuild events currently queued.
- **<deviceAccessAppServerClusterName>** contains the Hosting NE name provisioned on the Network Server corresponding to the current Application Server cluster. This name is used to resolve the %BWASCLUSTERFQDN% tag.

- **<allowDeviceCredentialsRetrieval>** determines whether the Device Management credentials are returned to authenticated client applications.

5.17.8.3 Add File Repository

From the Application Server CLI:

```
AS_CLI/System/Device/FileRepos>add <repoName> <protocol> <FQDN>
AS_CLI/System/Device/FileRepos>set <repoName> rootDirectory "rootDirName"
```

… where:

- **<repoName>** is the name given to the file repository. "ProfileServer" is recommended because it is automatically picked up when new device types are added. If you give another name, you may have to manually associate a device type with a Profile Server.
- **<protocol>** must be WebDAV for a Profile Server deployment or for any custom file repository that supports WebDAV. Use FTP for a file repository server external from Cisco BroadWorks.
- **<FQDN>** is the fully qualified domain name of the Profile Server farm. The DNS should resolve the FQDN in a fixed order fashion, that is, the Profile Server nodes should always be in the same order.
- **<rootDirectory>** represents the base directory under which files are stored.

The root directory can be misleading on an FTP server. Depending on the server configuration, the root directory is appended to the currently logged-in user's directory, or it is used directly from the "/" directory on the server. To find out which case applies to your FTP server, log in as a Device Management user. Perform an "ls" command and then a "cd /". If you are still in the logged-in user's directory, then the root directory of the file repository is relative to that. If you ended on the root of the server, then the root directory is relative to root of the server. This confusion is absent from the WebDAV configuration because the root directory on the server is always relative to the path configured in the Apache *httpd.conf* file.

The Application Server can be configured to force a secure WebDAV connection to the file repository. From the Application Server CLI:

```
AS_CLI/System/Device/FileRepos>set <repoName> secure true port <port>
```

… where:

- **<repoName>** is the name given to the file repository that should be accessed using a secure connection.
- **secure true** forces a secure connection to be established.
- **<port>** the port used to establish the secure connection.

Note that the values set for the *port* and *secure* parameters must match what is set on the Profile Server under the following CLI level.

```
PS_CLI/Interface/Http/HttpServer>
```

For more information on this CLI level, see section *8.1.1 Profile Server Configuration*.

Once the Application Server has been configured as described, both the Application Server and the Xtended Services Platform server use a secure connection to access files on the file repository.

5.17.8.4 Change Association between File Repository and Device Type

The file repository on the receiving end of generated files is configured on a device-type basis. From the CLI:

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> add <deviceType>
<fileRepoName> directory <remoteDirectory>
```

… where:

- **<deviceType>** is the device type for which files are associated to the repository.
- **<fileRepoName>** names the existing file repository to hold the files.
- **<remoteDirectory>** is the remote directory name for files of this device type.

An interesting behavior of the file repository association is that manually associating a device type with a file repository can be avoided. When a new device type is created, if the new Device Management is enabled and a file repository named "ProfileServer" exists, then the new device type is automatically associated with this repository. If no file repository of that name exists, then the first one that supports the WebDAV protocol is picked up. Nevertheless, you may still need to manually modify the association in cases where the device type was created prior to the enabling of the new Device Management or simply to move files from one repository to another.

5.17.9 Possible Security Measures

Many communication channels for Device Management can be secured through the mechanisms such as network access lists, authentication, and SSL. The following figure shows the different communication channels and the way they can be protected for a typical deployment.

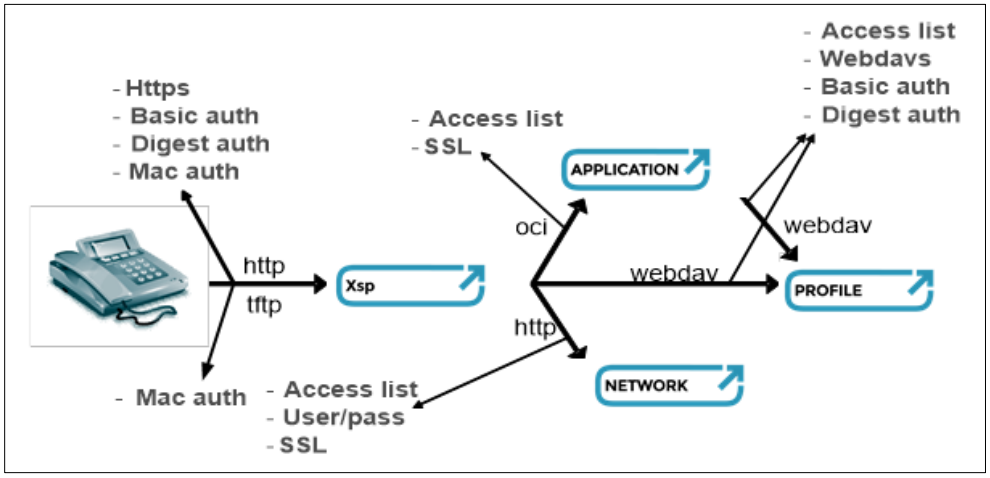


Figure 88 Possible Security Measures

6 Security Considerations

This section describes some of the recommendations to have a secure Device Management deployment.

6.1 HTTP versus HTTPS

Unencrypted traffic (HTTP) is vulnerable to multiple issues, namely; confidentiality, integrity, and authenticity. This essentially translates to being exposed to man-in-the-middle attacks, where our data can be intercepted to be accessed or tempered with. For this reason, it is strongly recommended to use HTTPS in conjunction with Device Management.

6.2 File Authentication

There are five supported file authentication mechanisms in Cisco BroadWorks Device Management, they are listed here from most secure to least secure:

1) MAC-Based – In Client Certificate
2) Username/Password – Digest
3) Username/Password – Basic
4) MAC-Based – In HTTP Header
5) MAC-Based – In HTTP Request

6.2.1 Mutual Authentication Using Signed Certificate

The most secure file authentication mode is MAC-based – In Client Certificate using mutual authentication through signed certificate. The following figure illustrates the way each request is handled through this mechanism.

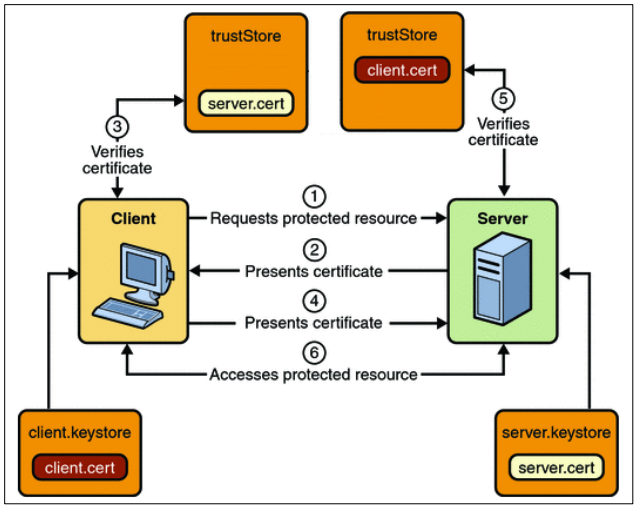


Figure 89 Mutual Authentication Using Signed Certificate

In the case where mutual certificate authentication is not available, using the authentication mode username/password is preferable to MAC authentication since forging a password is more complicated than finding a valid MAC. This is true only when strong password hardening rules have been put in place. For more information on where to set the password hardening rules in the system, see section *6.4 Device Profile Password Rules*.

For more information, see the *Cisco BroadWorks SSL Support Options Guide* [15].

6.2.2 MAC Over HTTP Request

Using file authentication with the MAC in HTTP Header or in HTTP request over HTTP can lead to multiple security vulnerabilities issues. Request can be made to the Xsp for a given file without validating the source of the request, which can lead to authenticity issues since the communication is in clear text and there is a lack of secure authentication.

A malicious entity may sniff HTTP traffic and obtain valid MAC from our system, which could then be used to retrieve configuration files that may contain sensitive information.

The use of this method (particularly MAC in header) is only recommended for cases where device authentication is reliably performed by a network element deployed in front of the Xsp. For example, a firewall terminating SSL connections from the devices and properly validating the device certificate containing the MAC address. The firewall can safely forward the authenticated requests to the Xsp including the MAC in a header. The firewall needs to prevent clients from issuing requests with the same header (or at least not allow that header to flow through un-altered). The Device Management feature takes care of doing the required authorization checks for that device (granting or denying access to the requested resources). The Xsp must be configured such that only the sanctioned network element (for example, firewall) can issue requests using the headers. This can be achieved using SSL/TLS client authentication (preferred) or iptables rules on the Xsp.

Note that as an alternative to using a proprietary header, the firewall can send the actual device certificate information in common SSL proxying headers. The Xsp can be configured to pull the client certificate information from these headers using the ClientAuthenticationProxy setting on the WebContainer. For more information, see the *Cisco BroadWorks Xtended Services Platform Configuration Guide* [2]. In this case, DeviceManagement files must be configured with *macInCert*.

6.3 Edge Node

The Xsp does not have the built-in functionality against DDoS and Brute Force attacks. Instead a firewall (FW) or web application firewall (WAF) should be used (as shown in the following figure). This provides:

- DDoS protection
- Brute Force attack protection
- Criteria-based rate limiting

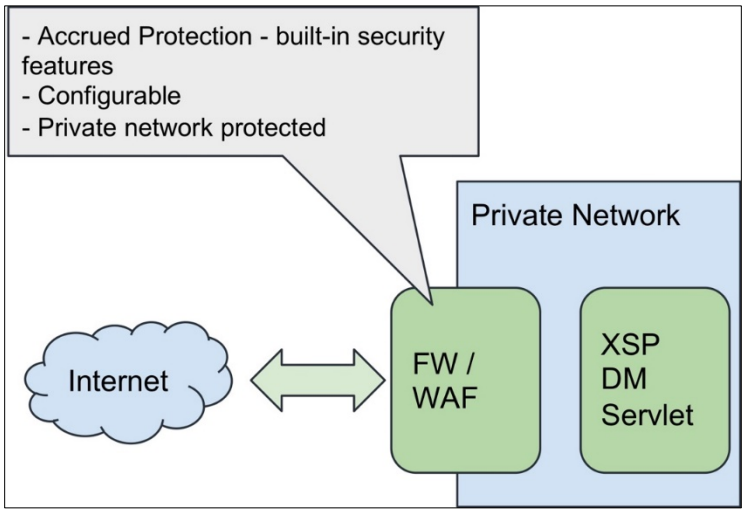


Figure 90 Recommended Edge Node Deployment

6.3.1 WAF Functionalities

Having a firewall allows some control on the traffic passing through. For instance, requests can be filtered using the User-Agent header which contains the device type and as such could allow controlling exactly what types of devices are allowed on the network.

Other notable capabilities that could be used include rate limiting based on source IP or even blacklisting of IP addresses. Note that it is not necessarily all worth deploying, each feature (functionality) must be examined for their effect on the whole system. Misuse of some functionality could lead to false positive, making end devices unusable.

FW/WAF offers various attack protection functionalities through filtering, many support SSL offloading, and all of this to reduce the attack surface on the system.

The following summarizes the functionalities that could be useful:

- Filtering using the User-Agent header, essentially whitelisting the allowed device types in the system.
- Rate limiting based on source IP (handle with care).
- Blacklisting IP addresses (handle with care).

6.4 Device Profile Password Rules

Hardening rules for device profile passwords should be used if applicable (username/password). These rules can be set under the System level under *Utilities → Device Profile Authentication Rules*.

Figure 91 Device Profile Authentication Rules

6.5 Device Profile Lockout Rules

It is possible to configure Device Profile Lockout Rules at the System level under *Utilities → Device Profile Authentication Rules*. Three options are available: Never lock out, Temporary lock out, or Permanent lock out.

The preferred approach is to use temporary lock out as it offers a lot of flexibility with regards to how long it will lock the device profile for and it also allows the system to fall back on its feet without the need of the system administrator intervention.

It is not recommended to use permanent lock out as it could be used as a denial of service attack on the system rendering end devices no longer usable.

6.6 Device Management Web App Deployment

The default access URL to BroadworksDMS is well known in the industry. By simply changing the deployment path of the application, can add a layer of complexity for a malicious entity trying to access our system.

Changing the context in which the application is deployed is fairly simple.

```
XSP_CLI/Maintenance/ManagedObjects>activate application
BroadworksDms2X.0_1.Y01 /securedms
XSP_CLI/Maintenance/ManagedObjects>deploy application /securedms
```

Note that changing this context on an existing system can be a tedious task. Changing the context path at the Xsp will be as simple, but there may be possible impacts on each device that needs to reach the Xsp. So they may need to be directed to the newly deployed path in their configuration.

6.7 File Repository Encryption

In order to avoid configuration files being readable on the file repository, BW-FileRepository can be encrypted.

```
PS_CLI/Applications/FileRepos/DiskCipher/GeneralSettings
```

This will allow all files being written to the file repository to be encrypted on disk and thus not in a readable format in the event where a malicious entity would gain access to them and try to extract sensitive information from them. The encryption keys can also be cycled if compromised.

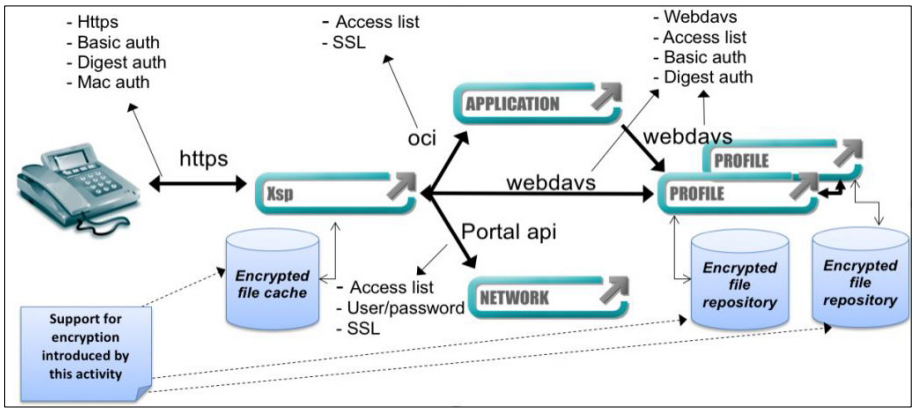


Figure 92 File Repository and File Cache Encryption

Note that the File Cache from the Xsp can also be encrypted.

```
XSP_CLI/Applications/BroadworksDms/Cache/DiskCipher/GeneralSettings
```

6.8 Fraud

Having a system deployed with weak security can lead to fraud such as the International Revenue Share Fraud (IRSF) which is a form of fraud whereby the perpetrator artificially inflates traffic by generating calls to certain portions of international number ranges with no intention of paying for the calls. To be able to make calls on behalf of a user, the malicious entity will obtain a valid config file from the system and then using the confidential user information, make SIP calls (see the following figure).

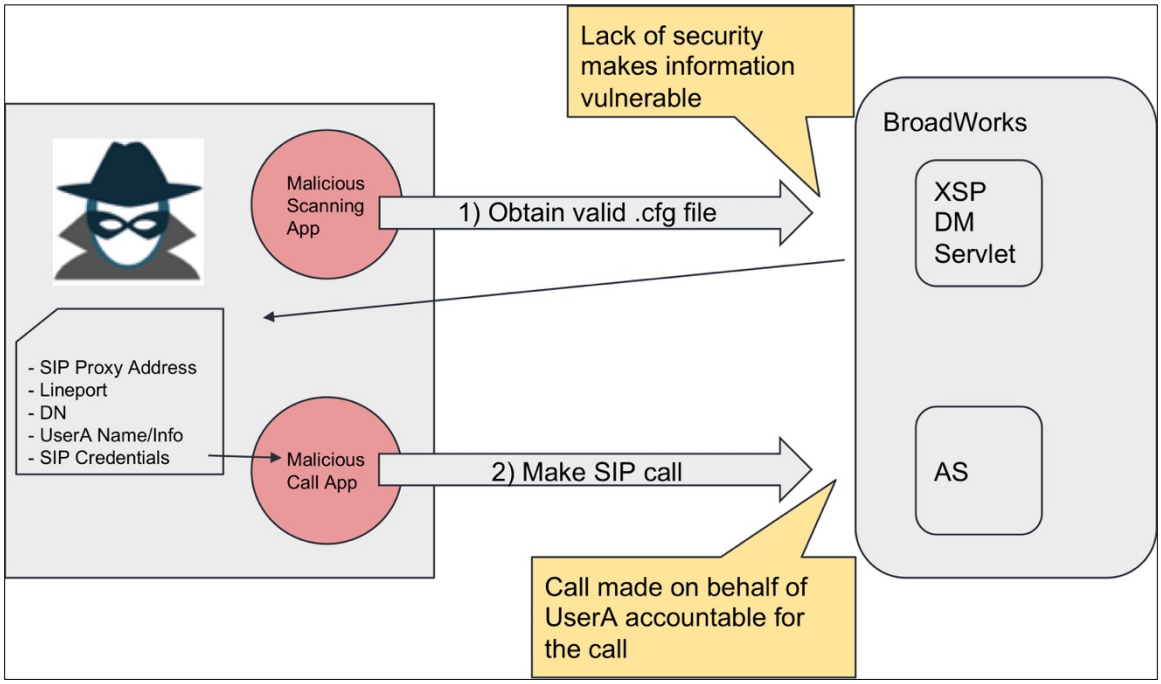


Figure 93 Fraud Example

Such frauds cost the industry billions of dollars globally. It is important to understand that a single file compromised could lead to great costs.

6.8.1 Scan for MAC

In order to obtain a valid config file, a malicious entity could scan through our system by trying to guess or find a valid MAC to access a config file. The following figure describes such a procedure.

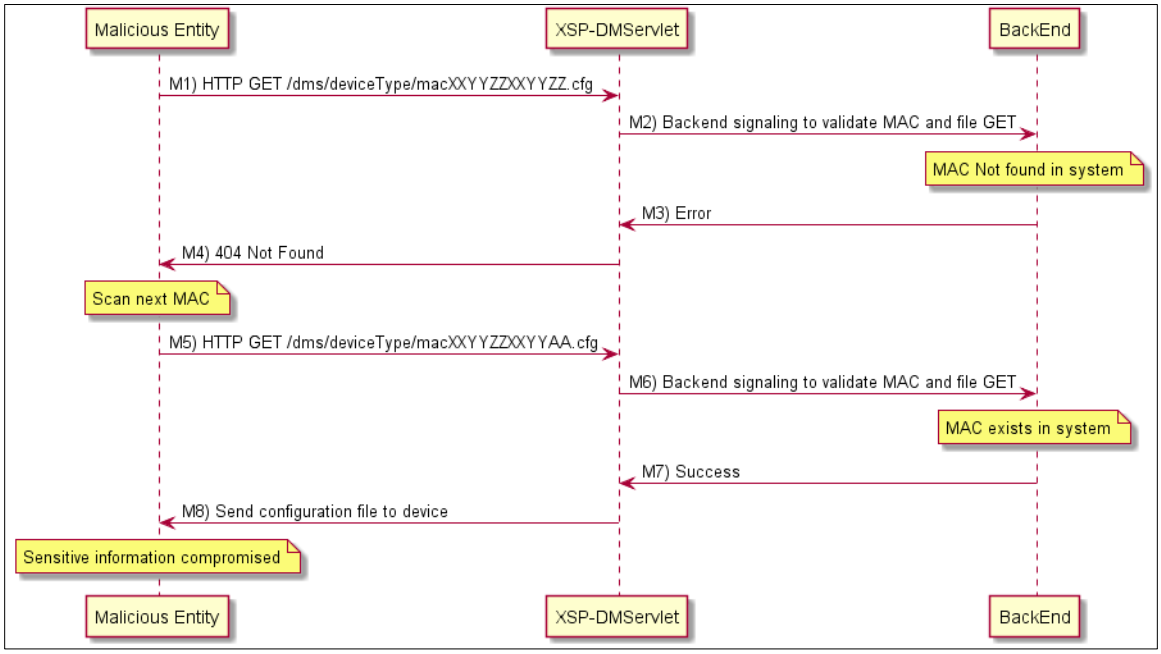


Figure 94 Scanning MAC Flow Diagram

6.8.2 Detect Scan Attacks

The HTTP metrics coupled with the thresholding notifications function provided by the Xsp can help in detecting scan attacks. The MIB table named "bwHttpWebAppResponsesTable" provides the count of responses types per webapp. More precisely, the *bwHttpWebAppResponsesNbOfResponses* counter in that table provides the count. A threshold can be configured to report specifically on 404 responses emitted by the Device Management webapp. The threshold should be configured to fire repeatedly on a desired increment of the counter. For example, to be notified on every increment of 1000 responses:

```
XSP_CLI/Monitoring/Threshold/Counter> add scancheck scancheck
bwHttpWebAppResponsesNbOfResponses 1000 1000 High true stringIndex
/.GET.404
```

The notification rate can be used to infer scan attacks or at the very least to trigger a more detailed traffic pattern analysis exercise.

6.9 Secure Device Management Deployment

Putting together all the information gathered so far in this section, the following figure represents the recommendations for a secure Cisco BroadWorks deployment with Device Management in mind.

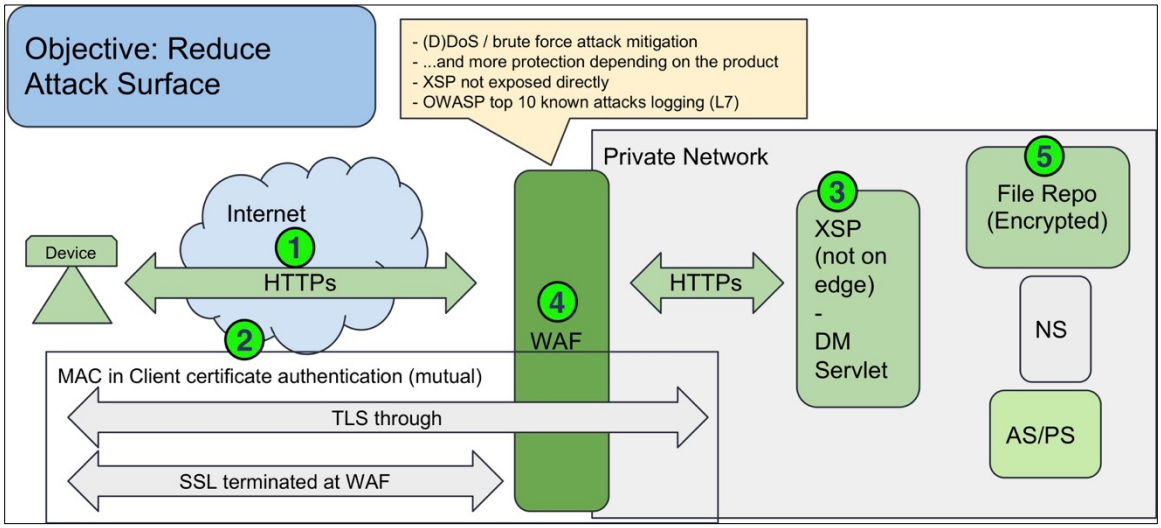


Figure 95 Secure DM Deployment

To summarize this figure, this deployment contains the following:

1) HTTPS between device and Xsp.
2) MAC in certificate authentication.
3) Xsp in private network, accessible through WAF.
4) WAF as the edge node allowing mitigation of DDoS and Brute Force attacks:
   a) TLS through when WAF not acting as SSL offloader.
   b) SSL terminated when WAF is SSL offloading.
5) Encryption of files in FileRepo.

7 Best Practices

This section describes recommended best practices.

7.1 Enable Server Thread Names in Logs

Server thread names should be displayed in the logs of all the servers. This allows a more efficient tracking of the processing of Device Management-related processing.

Xtended Services Platform (Xsp)

```
XSP_CLI/Applications/BroadworksDms/Logging> set showThreadName true
XSP_CLI/Applications/DeviceManagementTFTP/Logging> set showThreadName
true
```

Application Server (AS)

```
AS_CLI/Applications/ExecutionAndProvisioning/PS/Logging> set
showThreadName true
AS_CLI/Applications/ExecutionAndProvisioning/XS/Logging> set
showThreadName true
```

Network Server (NS)

```
NS_CLI/Applications/NSPortal/Logging> set showThreadName true
```

Profile Server (PS)

```
PS_CLI/Applications/FileRepos/Logging> set showThreadName true
```

7.2 Enable File Caching

File caching should be enabled for static and dynamic per-type files. This is enabled on a per-file basis on the Application Server.

7.3 Enable Application Server to Network Server Device Management Synchronization

The Application Server synchronizes the device file information, MAC address, and device user names with the Network Server. This allows the Network Server to be used for an Application Server location engine for Device Management. This eliminates need for an Xtended Services Platform to iterate across all Application Server clusters to find a file.

This functionality must be enabled on the Application Server using the CLI as follows.

```
AS_CLI/Interface/NetServSync> get
syncDeviceManagementInfo = true
```

The functionality must also be enabled on the Network Server using the CLI as follows.

```
NS_CLI/System/DeviceManagement> get
asLocationLookupEnabled = true
```

7.4 Increase Application Server Rebuild Throughput

The rate of rebuilds on the Application Servers is controlled by the number of threads as well as the configured pause between rebuilds. The rebuild Device Management rebuild queue should be sized to accommodate a burst of events. It should be big enough to hold a rebuild of the whole system or a number of events that can be processed in one maintenance window. The queue size can be set using the parameter *eventQueueSize* at the *AS_CLI/System/Device/IpDeviceMgmt* CLI level.

7.4.1 Increase Number of Device Management Rebuild Threads

For optimum file regeneration throughput, the *bw.ps.dms.numThreads* startup parameter should be set to half the number of CPU threads on the Application Server. This is controlled using the *AS_CLI/System/StartupParam* CLI level.

7.4.2 Tune Time between Rebuilds

The *pauseBetweenFileRebuildMilliseconds* should be set to "1". This is controlled using the *AS_CLI/System/Device/IpDeviceMgmt* CLI level.

7.5 Throttle Phone Reset Rate

The BroadWorks-controlled NOTIFY reset rate can be throttled to a number of resets per second. The reset rate serves three purposes:

- Network Protection – Ensures the reset NOTIFYs do not flood the network causing problems for other elements (for example, a session border controller).
- Maximize Device File Request Handling – Ensures that the controlled reset rate does not overwhelm the Device Management infrastructure.
- Limit Impact on Small/Medium Enterprise (SME) Customer Site Access – Controls the reboot rate for a deployment model where there are many SME customers behind a 100 Mbps pipe. Ensures that the phone reset rate does not overwhelm the available bandwidth.

Another approach for reset rate tuning is to set to the required rate to meet a specific reset requirement. For example, if the goal is to support the controlled reset of 5,000 devices in a 30-minute period then this would correspond to (5000/30*60) or approximately three resets per second.

Estimated per device reset bandwidth usage can be quickly estimated if you have a view of your phone download model, for example, a Polycom VVX 1500 4.0.2 with 20 associated files with six requiring authentication.

- Average Files Requests per Second (RPS) – four files/second
- Average File Size (AFS) – 500 KB firmware upload

The average bandwidth usage per reset for firmware upgrade = RPS * (AFS * 8)/1024 = 4 x (500 *8)/1024 = ~16 Mbps.

If a customer site requirement is to keep bandwidth usage to 60% of a 100 Mbps link, then 60/16 = ~4 reboots per second would be the limit for a full firmware download. Four reboots per second would support 7,200 device resets in 30 minutes.

The phone reset rate is controlled through the *minTimeBetweenResetMilliseconds* system parameter at the AS_CLI/System/Device/IpDeviceMgmt CLI level. As an example, set *minTimeBetweenResetMilliseconds* to 250 to achieve 4 phone reboots per seconds.

7.6 Increase Xtended Services Platform/Profile Server Thread Pool

The Xtended Services Platform uses a pool of connections to download and upload files to the Profile Server. By default, this is set to 32 connections per Profile Server. This should be increased by editing the *DmsServlet.properites* file on each Xtended Services Platform as follows:

- dms.maxHttpConnectionsPerPS=5000
- dms.maxHttpConnectionsTotal=10000

It is recommended to adjust the values of dms.maxHttpConnectionsPerPS and dms.maxHttpConnectionsTotal as the default values of the Xtended Services Platform and Profile Server thread/connection pool size are not ideal for DM. For more information, see the *System Engineering of Device Management* [14] document.

7.7 Tune Web Application Throttling

The Xtended Services Platform BroadworksDM web application (webapp) request throttling setting should not be greater than 2,000 per second.

The Profile Server *BroadworksFileRepos* web application request throttling setting should not be greater than 2,000 per second.

7.8 Profile Server File Repository

By default, the Profile Server uses the */var/broadworks/fileRepos* directory as the default file repository. For larger Device Management deployments, this should be broken out to its own physical partition. In the case where multiple disks are used within the file repository, the use of the Logical Volume Manager (LVM) is supported. The Profile Server root partition generally runs at 75 to 100 I/O operations per second (IOPS) dues to logging and replication. The file repository disks and IOPS should be dimensioned based on the throughput requirements. Using solid state drives (SSDs) can eliminate a potential disk bottleneck.

NOTE 1: It is no longer acceptable to tamper with files on the Profile Server. The Profile Server has a cache representation of the folder structures and files. Tampering with the actual files or folders on the Profile Server un-synchronizes the cache, causing unexpected errors.

NOTE 2: Soft links in the file repository are only supported in the context of a dedicated repository. Creating soft links that point outside the file repository, that is, outside of */var/broadworks/fileRepos*, is not supported.

7.9 Ensure Unique File between Application Servers

Consider the case where a device type is configured on multiple Application Servers. All the Application Servers point to the same Profile Server. The device type contains a dynamic per device file that must be unique for each device profile. The file does not use any kind of authentication.

In this case, it is important that the access file format contains tags that make the file unique per Application Server, so that the right Application Server can be identified to serve the file without conflict.

The same restriction applies to the remote file format. It must contain tags that make the file unique on the Profile Server. Otherwise, contention arises and Application Server 1's file push overwrites Application Server 2's file push on the Profile Server. The consequence is that a device on Application Server 1 can end up with the configuration file of another device produced by Application Server 2.

7.10 Good Practices and Constraints

For more detailed information on the DM deployment best practices and constraints impacting the following areas, see the *System Engineering of Device Management* [14] document:

- File rebuilds and device file requests
- File rebuilds and device reset throttling controls
- Impact of HTTPS versus HTTP
- DM scaling constraints and drivers
- DM monitoring KPIs
- DM workload/traffic patterns
- DM deployment best practices
- WebTrafficParser tool
- DM capacity dimensioning

8 Case Study

8.1 Create New Device Type on Fresh Install

In this section, the knowledge to cover an actual real-world example using a fresh install is applied. Then a new IP phone with template configuration files is provisioned.

8.1.1 Profile Server Configuration

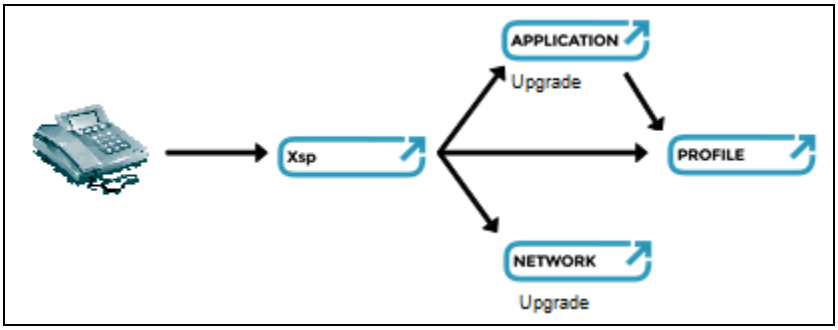


Figure 96 Fresh Install: Configuration of Profile Server

1) Using the Profile Server CLI, add all Application Server cluster members and all Xtended Services Platform to the WebDAV access list.

```
PS_CLI/Applications/BroadworksFileRepos/NetworkAccessLists/WebDav> add
192.168.12.90
PS_CLI/Applications/BroadworksFileRepos/NetworkAccessLists/WebDav> add
192.168.12.92
```

2) Ensure proper HTTP configuration.

```
PS_CLI/Interface/Http/HttpServer> get
     Interface   port              Name   Secure
=================================================
 192.168.12.94     80      192.168.12.94    false
```

NOTE: The *Name* field must be configured with the IP of the Profile Server.

3) Make sure that the BroadworksFileRepos web application is properly deployed.

```
PS_CLI/Maintenance/ManagedObjects>get broadworks full
* Hosted Applications:
                Name     Version  Context Path  Deployed
=========================================================
  BroadworksFileRepos  18.0_1.179            /      true
```

NOTE: The version you see (in the previous example, 18.0_1.179), depends on the software version that you have.

4) Optionally, create a WebDAV user name (must match credentials on the Application Server).

```
PS_CLI/Applications/BroadworksFileRepos/Users> add fileadmin
get,put,delete
```

5) Increase the Webapp overload transaction limit.

```
 PS_CLI/Applications/WebContainer/Tomcat/OverloadProtection/Webapps>
 set BroadworksFileRepos limit 500 period 1
```

6) Make sure the new WebApp overload protection does not exceed the Server overload protection.

```
PS_CLI/Applications/WebContainer/Tomcat/OverloadProtection/Webapps> get
period = 1000
limit = 1
```

7) Enforce user authentication.

```
PS_CLI/Applications/BroadworksFileRepos> set userAuthentication digest
```

-or-

```
PS_CLI/Applications/BroadworksFileRepos> set userAuthentication basic
```

8) Profile Server connectivity can be checked from telnet on the Application Server itself. A telnet test would look similar to the following.

```
bwadmin@192.168.12.92$ telnet 192.168.12.94 80
Trying 192.168.12.94…
Connected to 192.168.12.94.
Escape character 's '^]'.
GET /test.html
<html><head><title>Apache Tomcat/6.0.-4 - Error report</title> [... style
omitted ... ] </head><body><h1>HTTP Status 404 -
/test.html</h1><HR si"e""1" nosha"e="nosh"de"><p><b>type</b>
Status report</p><p><b>message</b>
<u>/test.html</u></p><p><b>description</b> <u>The requested
resource (/test.html) is not available.</u></p><HR si"e""1"
nosha"e="nosh"de"><h3>Apache Tomcat/6.0.14</h3></body></html>Connection
closed by foreign host
```

8.1.2 Network Server Configuration

As part of the basic Cisco BroadWorks configuration, the Network Server must have all the Application Servers properly defined under *NS_CLI/System/Device/HostingNE*. For more information, see the *Cisco BroadWorks System Configuration Guide* [5].

On the Network Server CLI, add all Xtended Services Platforms to the Network Server portal API access control list.

```
NS_CLI/System/NetworkAccessLists/PortalAPI> add 192.168.12.90
```

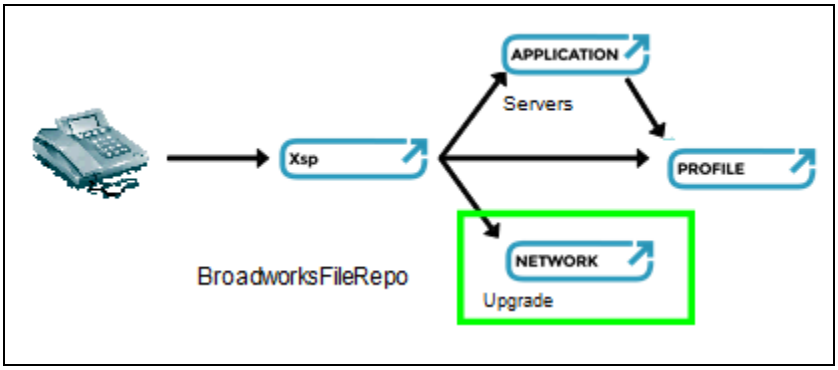


Figure 97 Fresh Install: Configuration of Network Server

8.1.3 Xtended Services Platforms Configuration

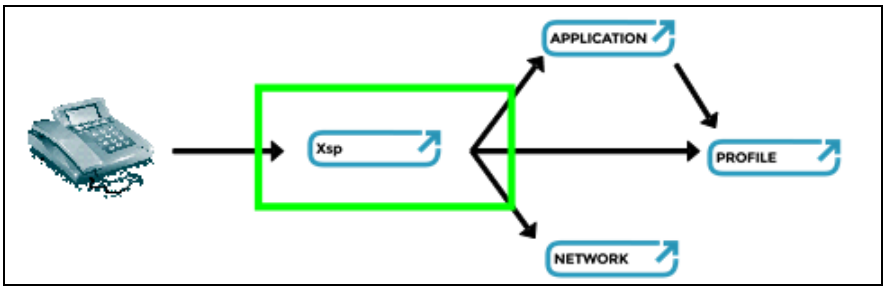


Figure 98 Fresh Install: Configuration of Xtended Services Platform

1) On the Xtended Services Platforms CLI, ensure proper HTTP settings.

```
XSP_CLI/Interface/Http/HttpServer> get
    Interface   port               Name  Secure
=============================================
192.168.12.90     80      192.168.12.90   false
```

2) Activate and deploy the BroadworksDms web application.

```
XSP_CLI/Maintenance/ManagedObjects> activate application BroadworksDms
16.0 /dms
XSP_CLI/Maintenance/ManagedObjects> deploy application /dms
XSP_CLI/Maintenance/ManagedObjects>get broadworks full
* Hosted Applications:
           Name  Version  Context Path  Deployed
=============================================
  BroadworksDms     18.0_1.179 /dms      true
```

3) Activate, deploy, and start the DeviceManagementTFTP application (if support of TFTP is required).

```
XSP_CLI/Maintenance/ManagedObjects> activate application DeviceManagementTFTP
16.0_1.517
XSP_CLI/Maintenance/ManagedObjects> deploy application DeviceManagementTFTP
XSP_CLI/Maintenance/ManagedObjects>get broadworks full
* Applications:
                Name     Version  Deployed  Administrative State  Effective State
=============================================================================
DeviceManagementTFTP  18.0_1.179 false             Unlocked          Stopped

XSP_CLI/Maintenance/ManagedObjects>start application DeviceManagementTFTP
```

4) Configure the DeviceManagementTFTP to point to the BroadworksDms web application context (if support of TFTP is required).

```
XSP_CLI/Applications/DeviceManagementTFTP/GeneralSettings> set
deviceAccessContextName dms
```

5) Configure the back-end communication.

```
XSP_CLI/System/CommunicationUtility/DefaultSettings> set mode ns
192.168.12.82
```

6) On the Application Server CLI, give access to the Xtended Services Platform.

```
AS_CLI/System/NetworkAccessLists/OCI/Provisioning>add 192.168.12.90
192.168.12.90
```

7) Check the Xtended Services Platform HTTP connectivity from an external PC. You should receive a 404 response to indicate that Apache is up.

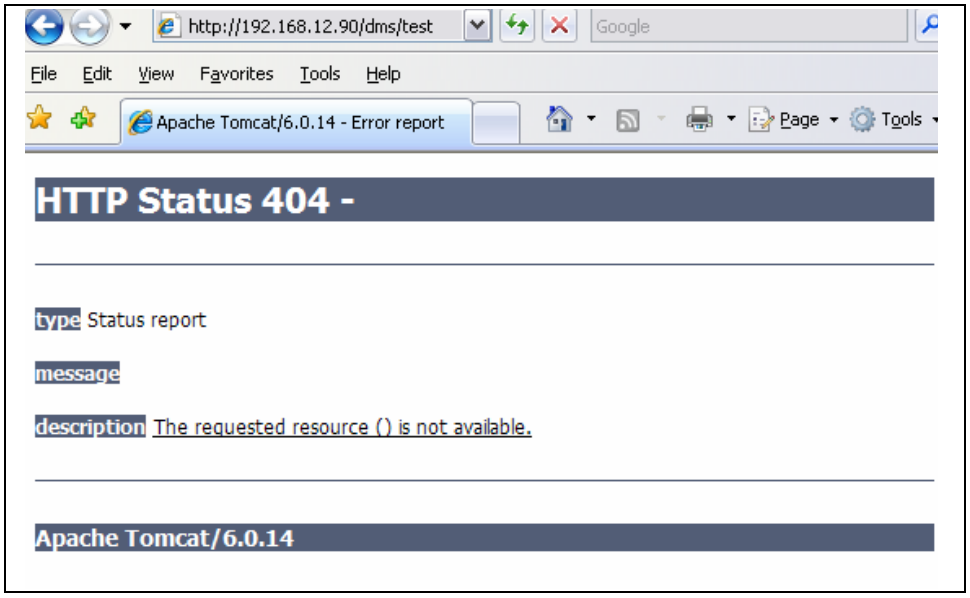


Figure 99 Connectivity Check on Xtended Services Platform

8) Check the Xtended Services Platform TFTP connectivity from an external PC. You should receive a 404 response to indicate that the DeviceManagementTFTP application is up.

```
$ tftp 192.168.12.90 -c get test
Error code 0: Error 404
```

8.1.4 Application Server Configuration

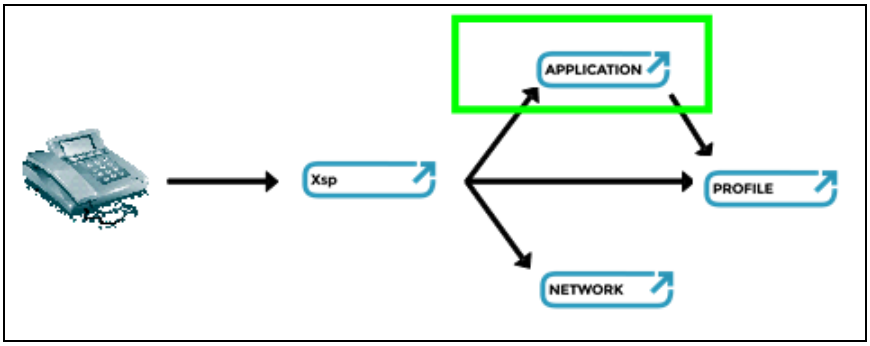


Figure 100 Fresh Install: Configuration of Application Server

1) From the Application Server CLI, make sure that the Device Management *enableIPDeviceMgmt* is set to "true".

```
AS_CLI/System/Device/IpDeviceMgmt>set enableIPDeviceMgmt true
```

2) Create a new device type for the new phone via the Web Server *System* → *Resource* → *Identity/Device Profile Type* link or the CLI as follows.

NOTE: The following is an example. Actual device type attributes would require matching to those of the actual device.

```
AS_CLI/System/DeviceType/Sip>add "New Phone"
nonIntelligentDeviceAddressing unlimited false false false false false
disabled false false false false false false false false false
false false false false rtp-session true false reSync 2FileConfig
not_used not_used true http none true true false false false false false
deviceAccessURI NewPhone
AS_CLI/System/DeviceType/Sip> set "New Phone" defaultDeviceEncoding "utf-
8"
AS_CLI/System/DeviceType/Sip>set "New Phone" username username
AS_CLI/System/DeviceType/Sip>set "New Phone" deviceAccessPortNumber 80
AS_CLI/System/DeviceType/Sip>set deviceTypeLegacy defaultDeviceLanguage
English_United_States
```

3) From the Application Server CLI, create a file repository and associate the device type.

```
AS_CLI/System/Device/FileRepos>add ProfileServer webdav 192.168.12.94
false
AS_CLI/System/Device/FileRepos>set ProfileServer rootdirectory /
AS_CLI/System/Device/FileRepos>set ProfileServer port 80
AS_CLI/System/Device/FileRepos/Users> add ProfileServer fileadmin
put,delete,get
AS_CLI/System/Device/IpDeviceMgmt/Fileserver>add "New Phone"
ProfileServer directory /new_phone_dir
```

4) Associate the files/templates to the new device type either through the Web Server *System* → *Resource* → *Identity/Device Profile Type* → *Files* and **Authentication** link or CLI (as follows).

```
AS_CLI/System/DeviceType/SIP/Files> add "New Phone" sip.ld sip.ld static
false custom /usr/local/user/new_phone/sip.ld false
```

```
AS_CLI/System/DeviceType/SIP/Files> add "New Phone" bootrom.ld bootrom.ld
static false custom /usr/local/user/new_phone/bootrom.ld false
AS_CLI/System/DeviceType/SIP/Files> add "New Phone" %BWMACADDRESS%.cfg
%BWMACADDRESS%.cfg dynamicProfile true custom
/usr/local/user/new_phone/mac.cfg false
AS_CLI/System/DeviceType/SIP/Files> add "New Phone" app.cfg
%BWDEVICEID%_app.cfg dynamicProfile true custom
/usr/local/user/new_phone/phone.cfg false
```

5) Create a device profile for the device either from the Web Server or CLI as follows.

```
AS_CLI/SubscriberMgmt/Device> add ProfileName "New Phone" ipAddress
10.10.1.1 macAddress 000000000000
```

6) On the Application Server CLI, force a rebuild of all files associated with the "New Phone" device type.

```
AS_CLI/System/Device/IpDeviceMgmt> rebuilddefaultfile system "New Phone"
```

7) On the Profile Server shell, make sure that the files were properly uploaded.

```
bwadmin@192.168.12.94$ pwd
/var/broadworks/fileRepos/new_phone_dir

bwadmin@192.168.12.94$ ls -al
drwxrwxr-x  2 bworks bwadmin  Jul  7 15:54 ./
drwxrwxr-x  3 bworks bwadmin  Jul  7 15:50 ../
-rw-rw-r--  1 bworks bwadmin Jul  7 15:54 000000000000.cfg
-rw-rw-r--  1 bworks bwadmin Jul  7 15:50 bootrom.ld
-rw-rw-r--  1 bworks bwadmin Jul  7 15:54 ProfileName_app.cfg
-rw-rw-r--  1 bworks bwadmin Jul  7 15:50 sip.ld
-rw-rw-r--  1 bworks bwadmin Jul  7 15:54 syncinfo.xml
```

8) From the browser, access a file similar to how the device would access a file.

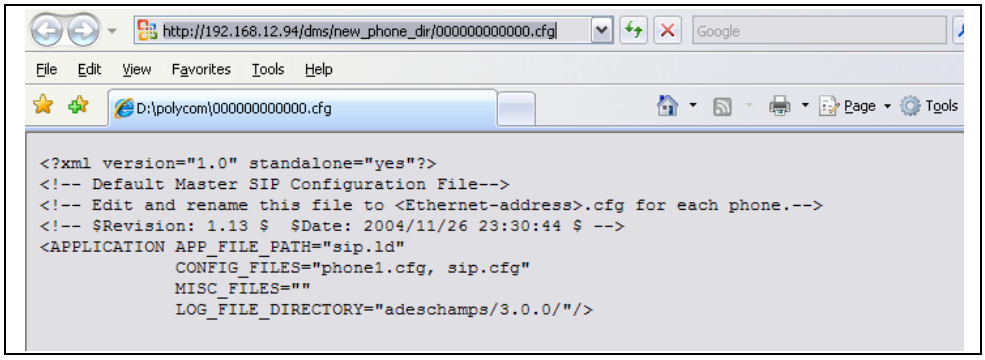


Figure 101 Real-World Example: Access "New Phone" Master File from Browser

9) From the *New Phone* device screen:

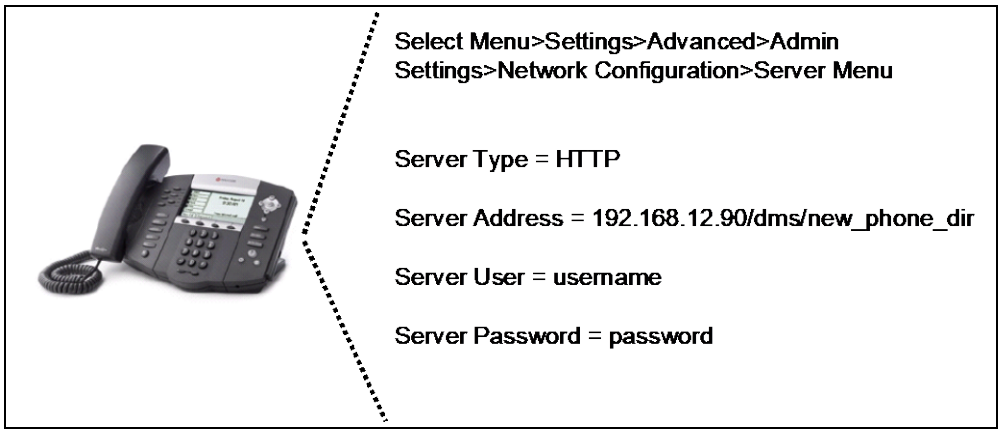


Figure 102 Fresh Install: Manually Bootstrapping Phone

8.2 Migrate to Device Management from IP Device Configuration

This section applies a "real-world" example of a migration from the legacy IP Device Configuration to the new Device Management. In this example, a Polycom 500 setup is converted to use the IpDeviceConfig default system templates, which use the new Xtended Services Platform HTTP access and Profile Server as the file repository.

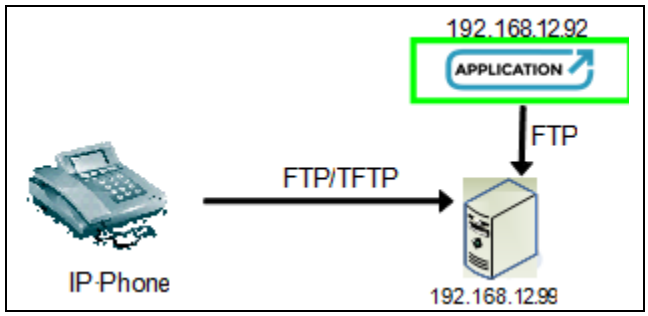


Figure 103 Migration: Configuration of Application Server (a)

On the Application Server CLI, enable Device Management.

```
AS_CLI/System/Device/IpDeviceMgmt>set enableIPDeviceMgmt true
```

NOTE: At this point, Device Management can be used in legacy mode and functions as it did previously using the legacy templates.

To leverage the new Profile Server and Xtended Services Platform servers after installing Cisco BroadWorks, configure the Profile Server and the Xtended Services Platform as you would for a fresh install. For information on Profile Server/Xtended Services Platform initial configuration, see sections *8.1.1 Profile Server Configuration* through to *8.1.4 Application Server Configuration*.

8.2.1 Application Server Configuration

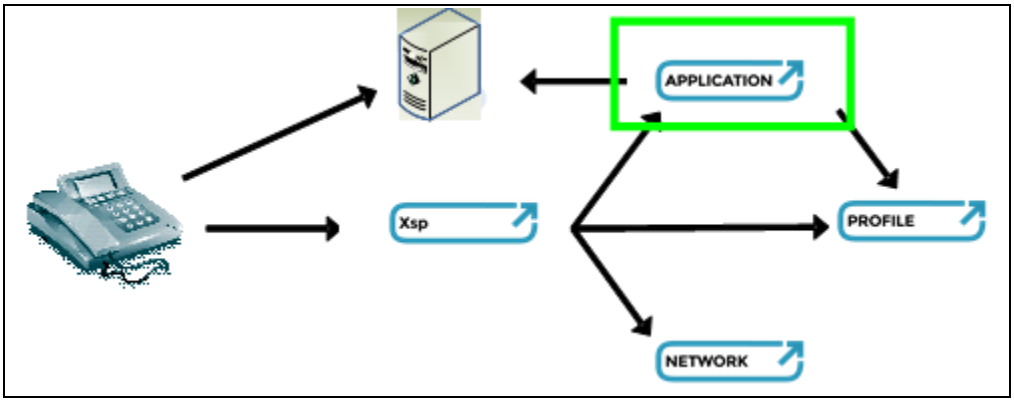


Figure 104 Migration: Configuration of Application Server (b)

8.2.1.1 Create New File Repository

On the Application Server CLI, create the new file repository for the Profile Server.

```
AS_CLI/System/Device/FileRepos>add ProfileServer webdav 192.168.12.94
false
AS_CLI/System/Device/FileRepos>set ProfileServer rootdirectory /
AS_CLI/System/Device/FileRepos>set ProfileServer port 80
AS_CLI/System/Device/FileRepos/Users> add ProfileServer fileadmin
put,delete,get
```

8.2.1.2 Modify Polycom SoundPoint IP 500 Device Type Device Management Attributes

The Polycom SoundPoint IP 500 Device Type Device Management-related attributes must be modified to include Xtended Services Platform information. Authentication mode should not be enabled at the global level since each Polycom file has a different authentication requirement. This can be done via the *System → Resource →Identity/Device Profile Type → Profile* (as shown or at the CLI *AS_CLI/System/DeviceType/SIP* level).

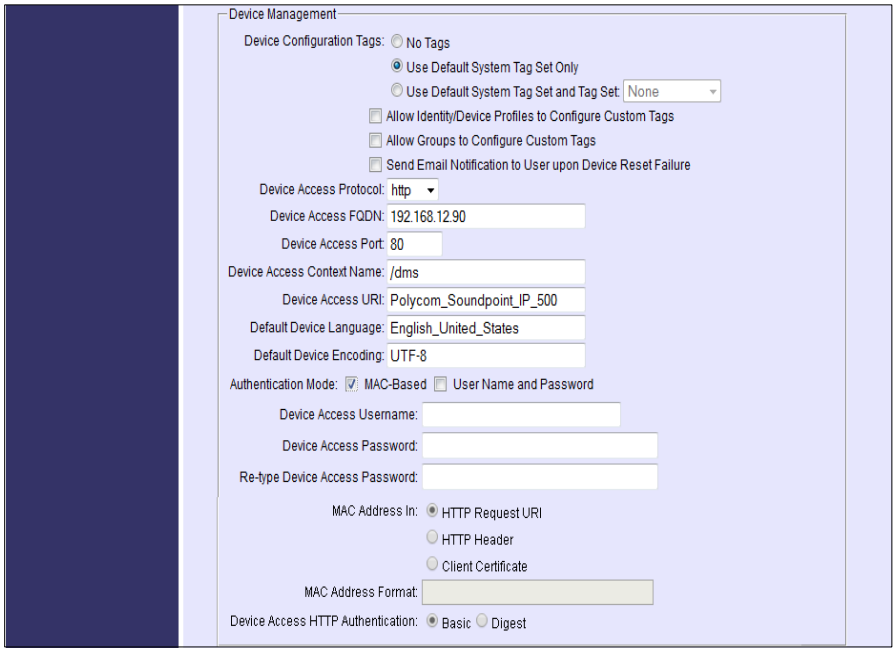


Figure 105 Modify Polycom SoundPoint IP 500 Device Type Device Management Attributes

8.2.1.3 Associate New File Templates to Polycom SoundPoint IP 500 Device Type

Unlike the legacy IP Device Configuration feature that had fixed template requirements (either two-configuration or three-configuration files), the new Device Management feature allows for the dynamic association of files to device type via the *Web Server System → Resource → Identity/Device Profile Type → Files* and **Authentication** link or CLI *AS_CLI/System/DeviceType/SIP/Files* level.

To migrate to the new Device Management feature, the existing phone templates must be loaded into the new Device Management file structure. In addition, any software load, firmware, and log files that must be pushed must be associated with the device type. The Device Management feature does not GET/PUT files that are unknown. Note that file names can now have dynamic tags present.

The existing Polycom 500 template can be re-used as templates by either pulling them to a PC for uploading via the *Web Server System → Resource → Identity/Device Profile Type → Files* and *Authentication → Add* link or directly from the Application Server directory via the CLI *AS_CLI/System/DeviceType/SIP/Files* level. A straight mapping of default legacy IP Device Configuration files to new Device Management files preserving the existing legacy naming conventions is as follows.

Device Management File Name	Legacy File Name and Location
PolyCom500System.cfg	/var/broadworks/IpDeviceConfig/System_Polycom_Soundpoint_IP_500.cfg
%BWMACADDRESS%.cfg	/var/broadworks/IpDeviceConfig/BASE_FILE_TP2_Polycom_Soundpoint_IP_500.template CONFIG_FILES="%DEVICE_FILE%,%SYSTEM_FILE%" can remain the same or be changed to CONFIG_FILES="BWDEVICE_%BWMACADDRESS%.cfg,PolyCom500System.cfg"
BWDEVICE_%BWMACADDRESS%.cfg	/var/broadworks/IpDeviceConfig/BW_DEFAULT_Polycom_Soundpoint_IP_500.template
bootrom.ld	Not currently managed by IP Device Configuration.
sip.ld	Not currently managed by IP Device Configuration.
%BWMACADDRESS%-boot.log	Not currently managed by IP Device Configuration. This log file is pushed to the repository by the phone and must be a known file.
%BWMACADDRESS%-app.log	Not currently managed by IP Device Configuration. This log file is pushed to the repository by the phone and must be a known file.

Table 7 Polycom 500 Legacy to Device Management File Mapping

The following shows the associated Polycom 500 files after they have been uploaded to the Application Server.

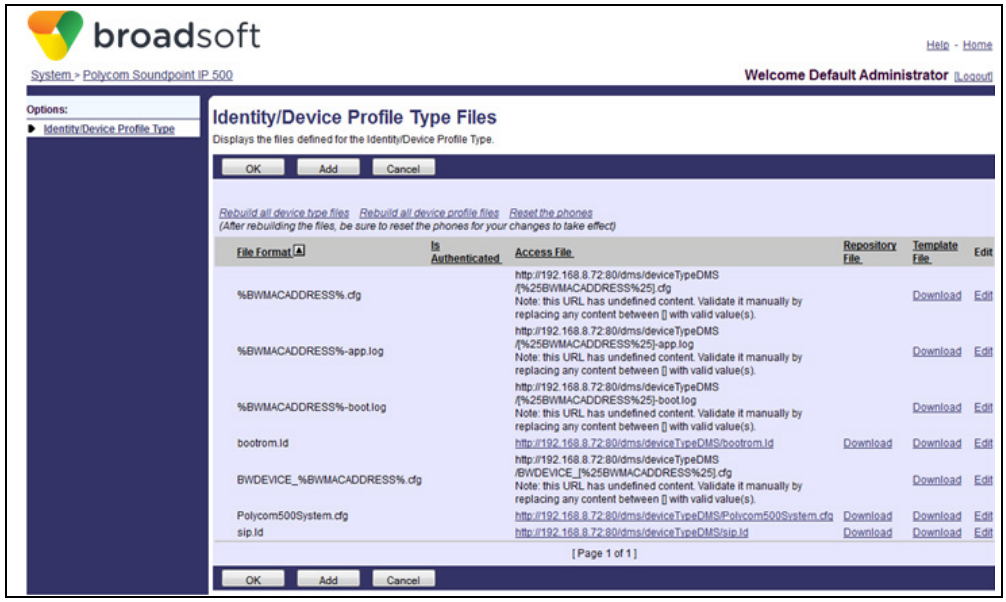


Figure 106 Associated Polycom 500 Files after Upload to Application Server

Note that some files have *Is Authenticated* enabled and some do not. In this case, basic MAC address authentication is used to emulate the legacy validation that the requested file name with MAC address must match the MAC address associated with an existing device profile. This authentication option can be set via the web for the existing file by enabling *MAC-based Authentication Mode* as shown in *Figure 101*.

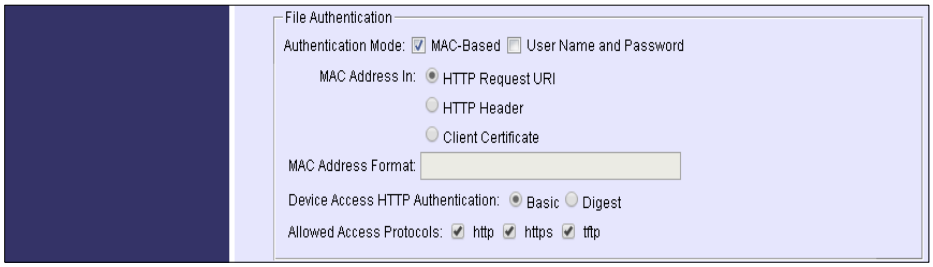


Figure 107 Authentication Mode Set

8.2.1.4 Deleting Legacy IP Device Configuration Templates

At this point, both the new Device Management templates and old IP Device Configuration templates generate on any rebuild. To stop the legacy template generation, these templates must be manually deleted or renamed. The default templates must be deleted on both Applications Servers at the same time using the peercmd command.

Example:

```
$peercmd rm
/var/broadworks/IpDeviceConfig/System_Polycom_Soundpoint_IP_500.cfg
$peercmd rm
/var/broadworks/IpDeviceConfig/BASE_FILE_TP2_Polycom_Soundpoint_IP_500.te
mplate
$peercmd rm
/var/broadworks/IpDeviceConfig/BW_DEFAULT_Polycom_Soundpoint_IP_500.templ
ate
```

NOTE: Legacy group and user customized templates, must be “unassociated” from the legacy template as well. They can be removed via the web by setting the customized file back to “default” using the normal legacy IP Device Configuration commands and then associating the custom template back using the new Device Management method.

8.2.1.5 Enable Polycom Phone Services

The Polycom Phone Services should be enabled for each device prior to performing the migration. This is accomplished by following the steps as indicated in section *5.14 Phone Services*.

When the service is enabled on a given device, a new file named *%BWFQDEVICEID%-directory.xml* is “pushed” to the repository on the next rebuild. The Application Server, before pushing this file, verifies whether there is a file named *%BWMACADDRESS%-directory.xml*. If there is, it integrates all its contacts into the personal phone list (entries are only added; no modification or deletion is performed). Once this is done, the Application Server then deletes the file *%BWMACADDRESS%-directory.xml*.

At this point, the phone no longer receives a directory file from the repository. However, it is still able to "push" one. If this occurs, a new *BWMACADDRESS%-directory.xml* is created on the repository. On the next rebuild, the same process described earlier occurs, that is, the Application Server integrates all the contacts found in *%BWMACADDRESS%-directory.xml* into the personal phone list of a user.

To get the most up-to-date information, a rebuild should be performed just before the device type repository is changed. This way, the last *%BWMACADDRESS%-directory.xml* file uploaded by the phone is integrated.

```
AS_CLI/System/Device/IpDeviceMgmt> rebuildDefaultFile system "Polycom
Soundpoint IP 500"
```

Once the repository is changed, if the phone pushes more updates, these modifications are not automatically integrated since the file *%BWMACADDRESS%-directory.xml* does not exist on the new repository. An administrator can still manually copy the *%BWMACADDRESS%-directory.xml* from the old repository to the new repository and perform a rebuild.

8.2.1.6 Automatic "Re-homing" Phones to New Repository

The Polycom 500 devices can be manually configured to point to the new repository or automatically "re-homed" by adding a new configuration file for the phone to read. The *new.cfg* configuration file that is associated with the Polycom 500 device type as shown in section *8.2.1.3 Associate New File Templates to Polycom SoundPoint IP 500 Device Type* represents the configuration file that is used for automatic repository transition and has a template as follows.

```
.<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<set>
<phone1 device.set="1"
device.prov.serverType="2"
device.prov.serverType.set="1"
device.prov.serverName="%BWDEVICEACCESSFQDN%:%BWDEVICEACCESSPORT%/%BWDMSC
ONTEXT%/bw/host/%BWASCLUSTERFQDN%/%BWDEVICEACCESSURI%"
device.prov.serverName.set="1"
device.prov.user="username"
device .prov.user.set="1"
device.prov.password="password"
device.prov.password.set="1" />
</set>
```

This *cfg* file must also be referenced in the *%BWMACADDRESS%.cfg* template file as follows.

```
CONFIG_FILES="new.cfg,BWDEVICE_%BWMACADDRESS%.cfg,PolyCom500System.cfg"
```

If the Device Management feature is in scheduled mode and an immediate system-wide file rebuild is required, the CLI rebuildDefaultFile command can be invoked as follows to push all the new files down to the existing FTP repository. This ensures that the *new.cfg* file is pushed down and all per-profile cfg files are updated to use it.

```
AS_CLI/System/Device/IpDeviceMgmt> rebuildDefaultFile system "Polycom
Soundpoint IP 500"
```

8.2.1.7 Change Polycom SoundPoint IP 500 Repository

The Polycom SoundPoint IP 500 device-type repository can now be moved to the Profile Server. The existing files on the FTP server are not deleted as part of this process and must be manually cleaned up once this migration exercise is completed.

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver>set "Polycom Soundpoint IP
500" fileReposName ProfileServer
```

This should trigger an automatic rebuild of all files on to the new Profile Server repository or this can be invoked manually as follows.

```
AS_CLI/System/Device/IpDeviceMgmt> rebuildDefaultFile system "Polycom
Soundpoint IP 500"
```

8.2.1.8 Reset Phones

The Polycom SoundPoint 500 phones can now be reset at a system level with the following CLI command.

```
AS_CLI/System/Device/IpDeviceMgmt> reset devicetype system "Polycom
Soundpoint IP 500"
```

At this point, the Polycom SoundPoint 500 phones pull the new configuration from the existing FTP server that "re-homes" them to the HTTP access Profile Server-based repository.

Once all phones have successfully migrated, the *new.cfg* file can be deleted and the reference to the file in *%BWMACADDRESS%.cfg* can be removed.

9 Troubleshooting

9.1 General

As shown in the following figure, many communication channels can be snooped for packets. Because most of the problems occur when a server is unable to talk to another for various reasons and because most of the communication occurs via HTTP, it is easy to "emulate" a phone by using a web browser to make a similar request.

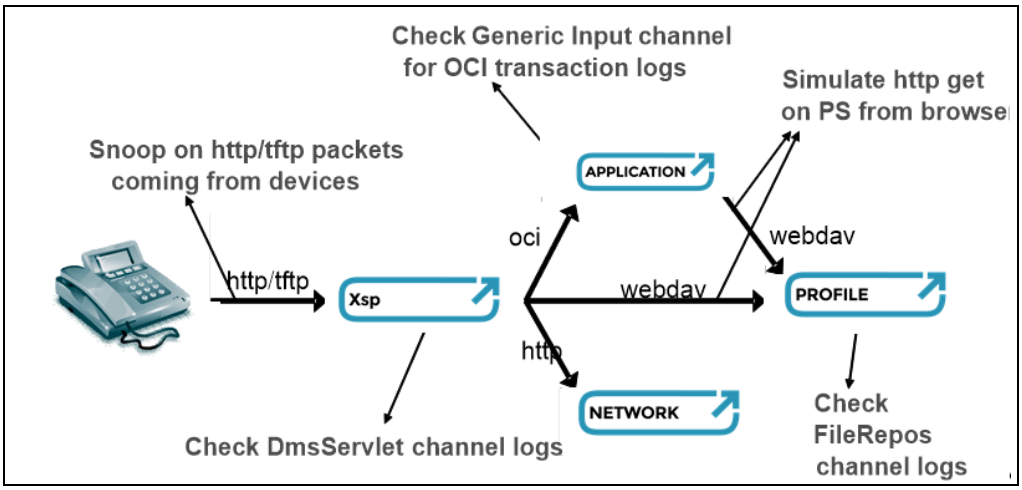


Figure 108 Generic Troubleshooting Tips

9.2 Browser Preview URLs

The web portal provides URLs that allow a preview of the files being served or a download of the templates currently in use. This interface is available at each customization level introduced in section *5.6 Customize Files*. *Figure 110* and *Figure 104* illustrate the links.

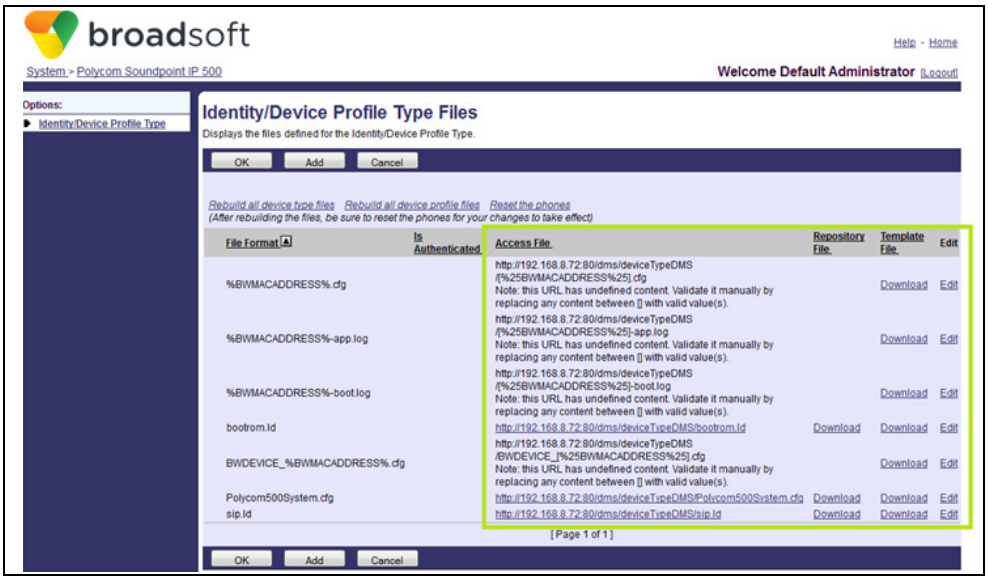


Figure 109 Browser Preview URLs on File List View

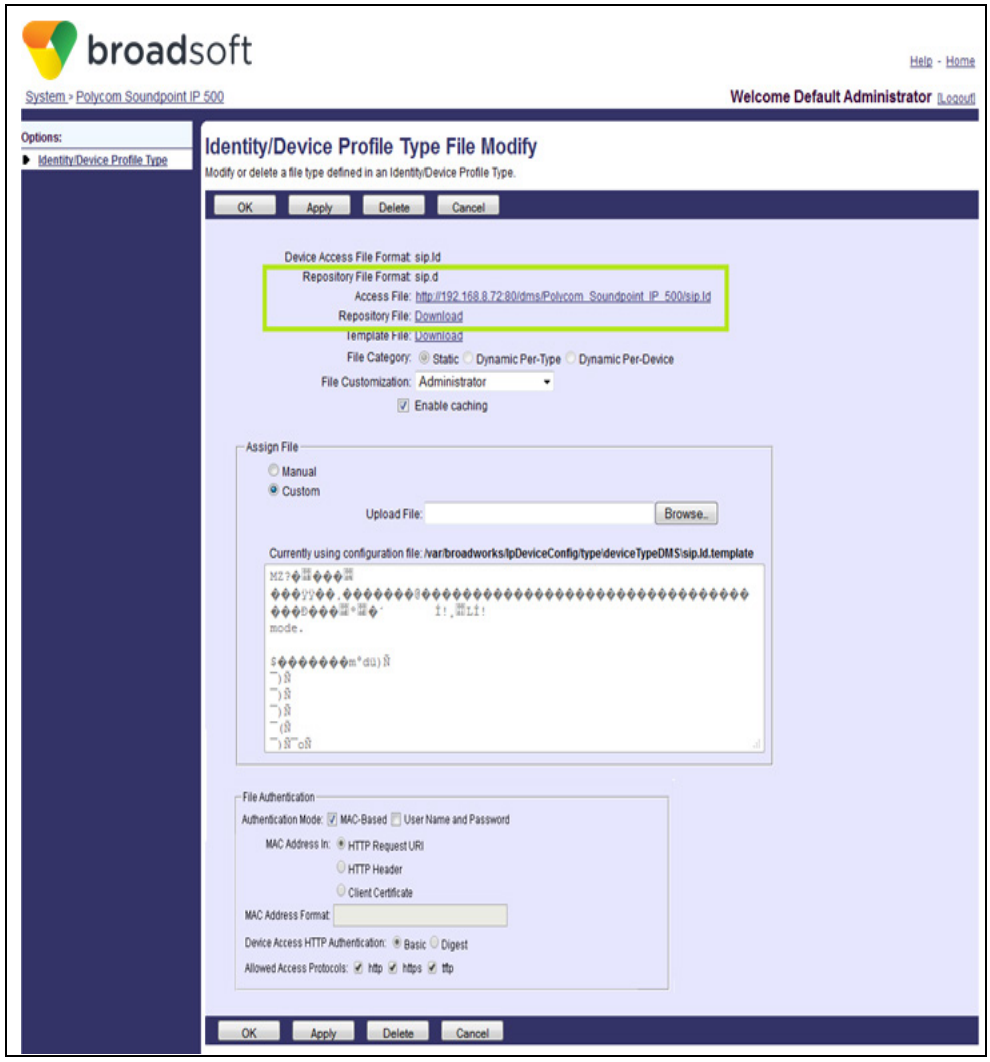


Figure 110 Browser Preview URLs on File Detail Page

Access File URL – This link opens a pop-up window in which the browser emulates a file request through the Xtended Services Platform HTTP interface just like a device would. This is useful to troubleshoot access problems such as URL typographical errors or HTTP error codes. It also helps users know what URL they should configure for a device. If the file is password protected, the browser is challenged. The Access File URL is deactivated when it contains a tag that cannot be resolved.

Repository File URL – This link downloads the file stored on the repository. It bypasses the Xtended Services Platform to retrieve the file directly from the Profile Server. This is useful if the credentials are unknown or just for a quick access to the file content when the Xtended Services Platform is down.

Template File URL – This link downloads the template stored on the Application Server. It allows an easy access to the template instead of having to retrieve it from the local disk directory structure. This is useful, for example, to download a template from a production system.

9.3 Logs

Many log files are available to help in troubleshooting the Device Management.

Profile Server Logging Capabilities

Configuration:

PS_CLI/Applications/BroadworksFileRepos_16.0/Logging

/usr/local/apache/apache_base/conf/httpd.conf

/usr/local/tomcat/tomcat_base/conf/logging.properties

Output:

/var/broadworks/logs/profileserver/

/var/tomcat/logs

/var/apache/logs

Xtended Services Platform Logging Capabilities

Configuration:

XSP_CLI/Applications/BroadworksDms_16.0/Logging

XSP_CLI/Applications/DeviceManagementTFTP_16.0_1.517/Logging

/usr/local/apache/apache_base/conf/httpd.conf

/usr/local/tomcat/tomcat_base/conf/logging.properties

Output:

/var/broadworks/logs/xsp

/var/broadworks/logs/tftp/

/var/tomcat/logs

/var/apache/logs

Application Server Logging Capabilities

Configuration:

/usr/local/broadworks/bw_base/conf/PSDebugConfig.xml

Output:

/var/broadworks/logs/appserver

9.4 Common Problems

This section describes common problems and suggests possible solutions.

Problem	Likely Cause	Suggested Action
The phone cannot download a file.	Wrong installation or network problems.	Make sure that Device Management is enabled. Application Server, Network Server, or Profile Server are unreachable or return an error (access list). Check authentication credentials (or disable authentication) on the Application Server and Profile Server. Make sure that the file exists on the Profile Server.

Problem	Likely Cause	Suggested Action
A phone downloads a file that is meant for another phone.	Access URIs are not unique per device type.	Make sure that all files have a unique URI to access them when doing the provisioning.
Tags are not replaced in the configuration file.	Tag is entered incorrectly. A line tag is not followed by a number, for example, %BWDN-1%.	Check that you are using a valid tag. An invalid tag is treated as text in the file. Check that the tag has been typed correctly.
Phone does not reset.	The phone is not registered or if registration is not used, an IP address has not been provisioned for the device. The CORBA interface is down.	Check on the *Device Modify* page to make sure that the device has registered. If registration is being used, check that authentication has been assigned and configured for the user. Check that registration has been turned on for the device; see the manufacturer's documentation. To check the CORBA interface, use the CLI *System/Util/Diag> List* command. If it returns an interface error, the CORBA interface is most likely down. You must restart the system. (This is very unlikely to happen.)
Cisco phone does not reset.	The *syncinfo.xml* file does not exist on the file server machine. This file should be in the same directory as the device configuration files.	Use the CLI *setConfigFile* command at the system or group level. Rebuild at the system or group level.
Profile Server or Xtended Services Platform returns a HTTP 404.	Wrong provisioning of the file.	Check that the web applications are properly deployed. Revisit the Apache configuration (URL rewriting, port) Make sure that the requested file exists on the Profile Server.
Permission denied when uploading files to a FTP server.	Relative versus absolute path usage.	Make sure that the root directory of the FTP server contains the full directory path.
Template generation takes forever.	The generation occurs when the server load is high.	You may want to consider the scheduled mode to avoid impacting the Application Server at peak hours.
Wrong type of authentication (The best authentication method is chosen when multiple are selected.)	More than one authentication mode is configured for a file.	Make sure that you do not have many types of authentications enabled.

Problem	Likely Cause	Suggested Action
Files are never pushed to the Profile Server.	Network issues or provisioning issues.	Make sure that the Device Management feature is enabled. Make sure that the file repository is properly configured on the Application Server. On the Profile Server, check the file permissions for the */var/broadworks/fileRepos* directory. The permissions should be as follows: drwxrwxr-x /var/broadworks/fileRepos If necessary, change the file permissions, by entering the following: chmod g+w /var/broadworks/fileRepos/

10 Known Limitations

A legacy system-level file from the web cannot be modified after the device type creation from the web (can be accomplished from the CLI).

Only the Polycom Phone Directory files, as well as any files ending with ".log" are authorized by the Application Server to be sent to the file repository. This is to prevent potential security issues with HTTP PUT when users were able to upload any files to the file repository.

11 Appendix A: Generic Device/Identity Access Profile Definitions

The following table shows the default generic identity/profile types available for Cisco BroadWorks.

Generic Device Types	Signaling Address Type [1]	Number of Ports [2]	Ringback Tone/Early Media Support [3]	Authentication [4]	Registration Capable	Static Registration Capable	E.164 Capable	Trusted	Authenticate REFER	RFC 3264 Hold	Video Capable	Use History-Info Header	Route Advance	Wireless Integration	PBX Integration	Use Business Trunking Contact	Auto Configuration Soft Client	Requires BroadWorks Call Waiting Tone	Forwarding Override	Conference Device	Mobility Manager Device	Music On Hold Device	Requires BroadWorks Digit Collection	Requires MWI Subscription	Advice of Charge
Generic SIP Phone	IP	U	L	E	X	X			X	X	X														
Generic SIP Int Proxy Domain GW	IP	U	R	E	X	X			X	X	X														
Generic SIP Non-Int Proxy Domain GW	NP	U	R	E	X	X			X	X	X														
Generic SIP Int Device Domain GW	ID	U	R	E		X			X		X														
Generic SIP Non-Int Device Domain GW	ND	U	R	E		X			X		X														
Generic SIP IP-PBX	IP	U	R	E	X	X			X	X					X										
Generic SIP IP-PBX Single Registration	IP	U	R	E	X	X			X	X					X	X									
Generic SIP TDM PBX	ID	U	R	E						X					X										
Integrated Conferencing Server (sys)	IP	U	R	D				X	X				X							X					
Generic SIP Music On Hold	IP	U	R	E	X	X			X													X			

[1] – I = Intelligent, N = Non-intelligent, D = Device Domain, P = Proxy Domain

[2] – U = Unlimited

[3] – R = RTP - Session, E = RTP – Early Session, L = Local Ringback – No Early Media

[4] – E = Enabled, D = Disabled, W= Enabled with Web Portal Credentials

The following is a description of the device types and how they can be used:

- Generic SIP Phone – This identity/device profile type can be used for any SIP phone device. It has unlimited lines for registration.
- Generic SIP Intelligent Proxy Domain GW – This identity/device profile type can be used for any SIP access gateway that uses Application Server addressing when registering, and can process enhanced services locally.

- Generic SIP Non-intelligent Proxy Domain GW – This identity/device profile type can be used for any SIP device that supports the Cisco BroadWorks INFO implementation for Call Waiting and flash hook processing that uses Application Server addressing.
- Generic SIP Intelligent Device Domain GW – This identity/device profile type can be used for any SIP access gateway that uses device addressing and can process enhanced services locally.
- Generic SIP Non-intelligent Device Domain GW – This identity/device profile type can be used for any SIP device that supports the Cisco BroadWorks INFO implementation for Call Waiting and flash hook processing that uses device addressing.
- Generic SIP IP-PBX – This identity/device profile type can be used for any intelligent, trunking, or IP-PBX device that registers each PBX user with Cisco BroadWorks. This device must use proxy domain addressing when communicating with Cisco BroadWorks. This profile type includes the PBX Integration policy to provide special processing for *Diversion* and *History-Info* headers.
- Generic SIP IP-PBX Single Registration – This identity/device profile type can be used for any intelligent trunking or IP-PBX device that registers only the main (or pilot) user with Cisco BroadWorks. This device must use proxy domain addressing when communicating with Cisco BroadWorks. This profile type includes the PBX Integration policy to provide special processing for *Diversion* and *History-Info* headers. This profile type also includes the Use Business Trunking Contact policy to populate the Request-URI with the pilot user contact.
- Generic SIP TDM PBX – This identity/device profile type can be used as any intelligent trunking or IP-PBX non-registering device that uses device domain addressing when communicating with Cisco BroadWorks. This profile type includes the PBX Integration policy to provide special processing for *Diversion* and *History-Info* headers.
- Integrated Conferencing Server (sys) – This identity/device profile type is used with the BroadWorks-integrated Instant Conferencing solution.
- Generic SIP Music On Hold – This identity/device profile type can be used for any SIP access gateway that supports an external music source, uses Application Server addressing when registering, and can process enhanced services locally.

12 Appendix B: Legacy Support and Migration Paths

The possible migration paths are shown in the following figure.

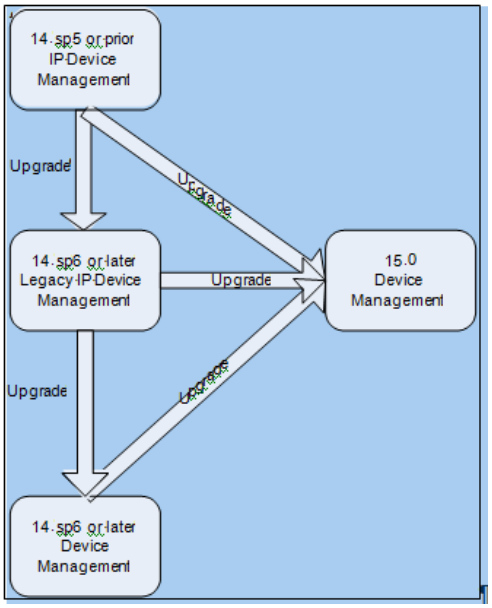


Figure 111 Possible Legacy Migration Paths

12.1 Migration from Existing IP Device Configuration Management

NOTE: To perform a migration of files from one repository to another, the servers must have been properly configured. Before attempting to do this, see section *5 Configure Device Management on Cisco BroadWorks*.

There is no special configuration required to support legacy templates. When the new Device Management functionality is enabled, from an existing FTP deployment's perspective, old templates are managed almost identically as they were previously. The main differences are that:

- The FTP client uses an absolute directory instead of a relative directory.
- Many new tags are available. For a list of tags that are available when using the new Device Management capabilities or the legacy IP Device Configuration Management, see section *5.13 Use Tags and Tag Sets*.
- An unlimited number of files can be created for a given device type.
- Template files and tags can be customized at different levels.

To empower the new Xtended Services Platform server, a file migration of existing files from the FTP to the Profile Server is mandatory. Migrating files from one repository to another means basically two things. The first is to physically move the files. The second is to point the access URI of devices in the field to the new repository. These are detailed in the next two sections.

12.2 Move Files

Assuming that the servers are already properly configured (see previous sections), a possible option is to manually copy the files from one repository to another. However, it may be easier to change the file repository address, force a template generation, and let the Application Server handle the upload to the new repository. This can be done on a device-type level or for the whole repository.

In the following examples, assume that an existing FTP server called "Server" exists and that the new Profile Server was already created. For more information, see section *5.17.8.3 Add File Repository*.

12.2.1 Move One Device Type at a Time

From the CLI:

```
AS_CLI/System/Device/IpDeviceMgmt/Fileserver> get
Device Type  File           Repository Name       Directory
=========================================
Polycom501                                 mtlcms3        /Polycom501Dir
Polycom Soundpoint IP 500        mtlcms3         /NewPhoneDir

AS_CLI/System/Device/IpDeviceMgmt/Fileserver>set <deviceType>
<fileReposName>
```

12.2.2 Move Whole File Repository

From the CLI:

```
AS_CLI/System/Device/FileRepos> get
Name    Protocol  Root Directory
=========================
Server      ftp           /
```

```
AS_CLI/System/Device/FileRepos> detail Server
name = Server
FQDN = ftp.example.com
rootDirectory = /myDirectory
protocol = ftp
secure =
ftpPassive = true
port =21
```

```
AS_CLI/System/Device/FileRepos>set Server FQDN profileserver.example.com
AS_CLI/System/Device/FileRepos>set Server protocol webdav
AS_CLI/System/Device/IpDeviceMgmt>rebuildDefaultFile all
```

The Application Server then regenerates the templates on the new file repository.

12.2.3 Point Devices to New Repository

Assuming that the files are already on the new file repository, there are three methods that can be used to modify the device access URIs to point to this new repository.

- **Option 1**: The first one is to manually go to each device and change the repository (as shown in *Figure 96* for Polycom phones). Obviously, this is not viable for large deployments.

- **Option 2**: The second option is to edit each template to replace the access URI in the file. When you force a template generation, the files with the proper access URIs are uploaded to the old repository. Devices fetch their configuration files and reconfigure their access URIs with the new file repository. When all devices have switched to the new repository, the old repository can be put offline.
- **Option 3**: The third option is a variant of option 2. Instead of performing an error-prone and time-consuming replacement of the access URIs in templates, you may want to use the proper dynamic tags in templates when you create a new template for a device. These tags point to the configured file repository. This way, when the file repository is changed (see section *12.2 Move Files*), then the templates are automatically updated with the new access URI.

Table 5 shows the common configurations required for specific manufacturers.

Table 5 Set Access URI for Supported Manufacturer Through Their Configuration Files

Manufacturer	Redirect Method	Configuration File
Polycom	Manually editing a Polycom configuration file to redirect a phone to a new repository.	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <set> <phone1 device.set="1" device.prov.serverType="2" device.prov.serverType.set="1" device.prov.serverName="10.10.1.1:80/dms/bw/host/AS/Polycom_Soundpoint_IP_500/" device.prov.serverName.set="1" device.prov.user="username" device .prov.user.set="1" device.prov.password="password" device.prov.password.set="1" /> </set>
Polycom	Using tags in a Polycom template to automatically redirect the phone to its repository.	<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <set> <phone1 device.set="1" device.prov.serverType="2" device.prov.serverType.set="1" device.prov.serverName="%BWDEVICEACCESSFQDN%:%BWDEVICEACCESSPORT%/%BWDMSCONTEXT%/bw/host/%BWASCLUSTERFQDN%/%BWDEVICEACCESSURI%/" device.prov.serverName.set="1" device.prov.user="username" device .prov.user.set="1" device.prov.password="password" device.prov.password.set="1" /> </set>

Manufacturer	Redirect Method	Configuration File
Linksys	Manually editing a Linksys configuration file to redirect a phone to a new repository.	<flat-profile> <Profile_Rule ua="na"> http://10.10.1.1:80/dms/bw/host/AS/Linksys_SPA_942/spaSystem.xml </Profile_Rule> <Profile_Rule_B ua="na"> http://10.10.1.1:80/dms/bw/host/AS/Linksys_SPA_942/$MA.xml</Profile_Rule_B> <Profile_Rule_C ua="na"/> <Profile_Rule_D ua="na"/> </flat-profile>
Linksys	Using tags in a Linksys template to automatically redirect the phone to its repository.	<flat-profile> <Profile_Rule ua="na"> http://%BWDEVICEACCESSFQDN%:%BWDEVICEACCESSPORT%/%BWDMSCONTEXT%/bw/host/%BWASCLUSTERFQDN%/%BWDEVICEACCESSURI%/spaSystem.xml </Profile_Rule> <Profile_Rule_B ua="na"> http://%BWDEVICEACCESSFQDN%:%BWDEVICEACCESSPORT%/%BWDMSCONTEXT%/bw/host/%BWASCLUSTERFQDN%/%BWDEVICEACCESSURI%/$MA.xml</Profile_Rule_B> <Profile_Rule_C ua="na"/> <Profile_Rule_D ua="na"/> </flat-profile>
Aastra	Manually editing an Aastra configuration file to redirect a phone to a new repository.	download protocol: HTTP http server: 10.10.1.1 http path: Aastra_480i http port: 80
Aastra	Using tags in an Aastra template to automatically redirect the phone to its repository.	download protocol: HTTP http server: %BWDEVICEACCESSFQDN% http path: %BWDEVICEACCESSPORT%/ %BWDMSCONTEXT%/bw/host/%BWASCLUSTERFQDN% http port: %BWDEVICEACCESSPORT%

13 Appendix C: Visual Device Management Support Provisioning

13.1 Overview and Purpose

This section describes the Leonid Systems Loki Portals and Cisco BroadWorks provisioning requirements for the Visual Device Management feature.

The Application Server/Provisioning Server feature enhances Device Management by introducing a new user-assignable service called Visual Device Management, added to the CommPilot web portal to support the Leonid Systems Loki Visual Device Management (VDM) portal (version 9285-2.26 and above).

The Cisco BroadWorks administrator (group administrator and above) is provided with a link to access the provisioning of a user's device through Leonid Systems Loki Visual Device Management (VDM) portal.

The Loki Visual Device Management portal is part of the Loki Portals framework. This suite of tools, offered by Leonid Systems, is an element provisioning platform designed for communications service providers. It offers both a native web interface and a web services API to interface to existing Business SIP Services (BSS) / Operations Support System (OSS), Customer Relationship Management (CRM) systems, like Cisco BroadWorks. It includes web 2.0 controls to simplify and enhance the user's experience, reports on call activity, provides service management, and allows device creation and configuration.

The Loki Visual Device Management portal interfaces directly to Cisco BroadWorks Device Management and provides the following:

- Can change keys on a phone using Loki Portals
- Editing of configuration files is not required
- Synchronizes with features assigned to a user
- Takes advantage of using templates for repeatability
- Can be used by Cisco BroadWorks group administrators and above

13.2 Prerequisites

This section assumes that the following provisioning steps have been done as described in the Leonid Systems *System Preparation Guide*, *Release 1.72* [10].

- Preparation of the Loki environment
- Preparation of the hosts for the Leonid Systems application suite
- Steps to prepare Red Hat or CentOS for a Loki Portals installation

13.3 SSO Integration with Loki VDM Portal

One of the enhancements provided by this feature is to allow an administrator to access the Loki Visual Device Management portal without being required to log in again, through the use of the Single Sign-On (SSO) methodology.

Loki Portals relies on two SSO methodologies to allow an end user to log in only once yet access a variety of secured systems. There are two options available for this redirection: using cookies or SAML as defined in the Leonid Systems *SSO Integration with Loki Portals, Release 1.0* [9].

The Visual Device Management feature uses the cookies method. However, the Single Sign-On method using cookies comes with limitations that impact the original requirements as follows:

- Same domain requirement – By design standard, cookies do not provide the ability to carry information across domains. Therefore, the Cisco BroadWorks servers and Loki Portals server must be on the same domain, for example, as1.broadsoft.com, xsp1.broadsoft.com, lokiportals.broadsoft.com, and so on. Otherwise, the browser does not pass the cookies to the Loki Portals server.
- As the cookies are set within the web browser, the web browser must have cookies support enabled.
- The context configuration of both Cisco BroadWorks web server and Loki web server must have cookie support enabled.
- To pass the username and password to Loki Portals, both values must be kept in the Cisco BroadWorks session. The password kept in the session memory is encrypted.
- The Loki Portals server address must be provisioned only with a FQDN. It cannot be an IP address.
- When accessing the Xtended Services Platform/CommPilot web portal, an FQDN must be used, and not an IP address. Otherwise, the browser does not pass the cookies to the Loki Portals server.

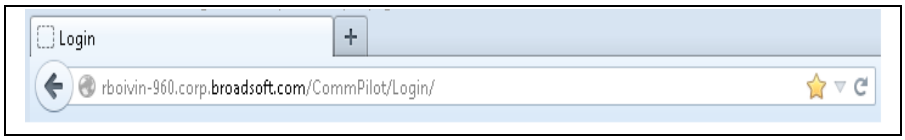


Figure 112 Using Xtended Services Platform FQDN from Web Browser

- Loki Portals server redundancy is similar to the Xtended Services Platform, so that the browser routes requests to the addresses that are resolved by the DNS. Since the redirection is internally handled by the browser, the browser goes to the next available address, which is not controlled by Cisco BroadWorks.

13.4 Cisco BroadWorks Application Server Configuration

1) Open ports 80 and 443 on the Application Server and/or Xtended Services Platform running the OCI Over SOAP application.
2) Install the OCI Over SOAP application.

```
AS_CLI/Maintenance/ManagedObjects> get versions
```

NOTE: The */webservice* path is required for the proper deployment of this application.

```
AS_CLI/Maintenance/ManagedObjects> activate application OCIOverSoap
YOUR_VERSION /webservice
AS_CLI/Maintenance/ManagedObjects> deploy application /webservice
```

3) Configure the OCI Over SOAP application as follows.

```
AS_CLI/Applications/OCIOverSoap/OCSConnectivity>
AS_CLI/Applications/OCIOverSoap/OCSConnectivity> set localhost
AS_CLI/Applications/OCIOverSoap/OCSConnectivity> set port 2208
```

```
AS_CLI/Applications/OCIOverSoap/OCSConnectivity> set
maxNumberOfOCIPConnections 2
AS_CLI/Applications/OCIOverSoap/OCSConnectivity> set
maxNumberOfCAPConnections 2
```

NOTE: The number of OCI and CAP connections depends on the number of servers that are connecting to the Application Server/Xtended Services Platform. If a web server is also deployed more connections are needed.

4) Restart Cisco BroadWorks.

5) Make sure that OCI communication is enabled correctly for the appropriate Application Server.

```
AS_CLI/System/CommunicationUtility/DefaultSettings>
```

6) Make sure the OCS (OpenClientServer) is configured and enabled for the appropriate Application Server. Review the following CLI levels for consistency.

```
AS_CLI/Applications/OpenClientServer/GeneralSettings>
AS_CLI/Applications/OpenClientServer/OCIProxy>
```

7) Make sure the Leonid host is listed in the following access lists.

```
AS_CLI/System/NetworkAccessLists/OCI/Provisioning>
```

8) Make sure the Application Server is listed in the following access lists.

```
AS_CLI/System/NetworkAccessLists/ExtAuth>
```

13.5 Loki Portals Server FQDN

Provision the FQDN of the Loki Portals server within the Xtended Services Platform as follows.

```
AS(/XSP)_CLI/Applications/CommPilot/VisualDeviceManagement> set
visualDeviceManagementServerFQDN lokiportals.xdp.broadsoft.com
```

13.6 Xtended Services Platform System Domain

1) Provision the Xtended Services Platform system domain.

```
AS(/XSP)_CLI/ Applications/CommPilot/General Settings> set systemDomain
.broadsoft.com
```

NOTE: This value is used by the Visual Device Management Single Sign-On method, which is cookie-based. Since cookies are sent across multiple servers, the cookies must be an explicitly set domain. When comparing domains, the browser does a very literal string compare of the host portion of the URL to determine the domain. As shown in the example, this value allows passing cookies across all servers in the domain broadsoft.com.

2) Restart Cisco BroadWorks.

13.7 Leonid Systems Visual Device Management and Cisco BroadWorks Interaction

The following figure shows the Visual Device Management process.

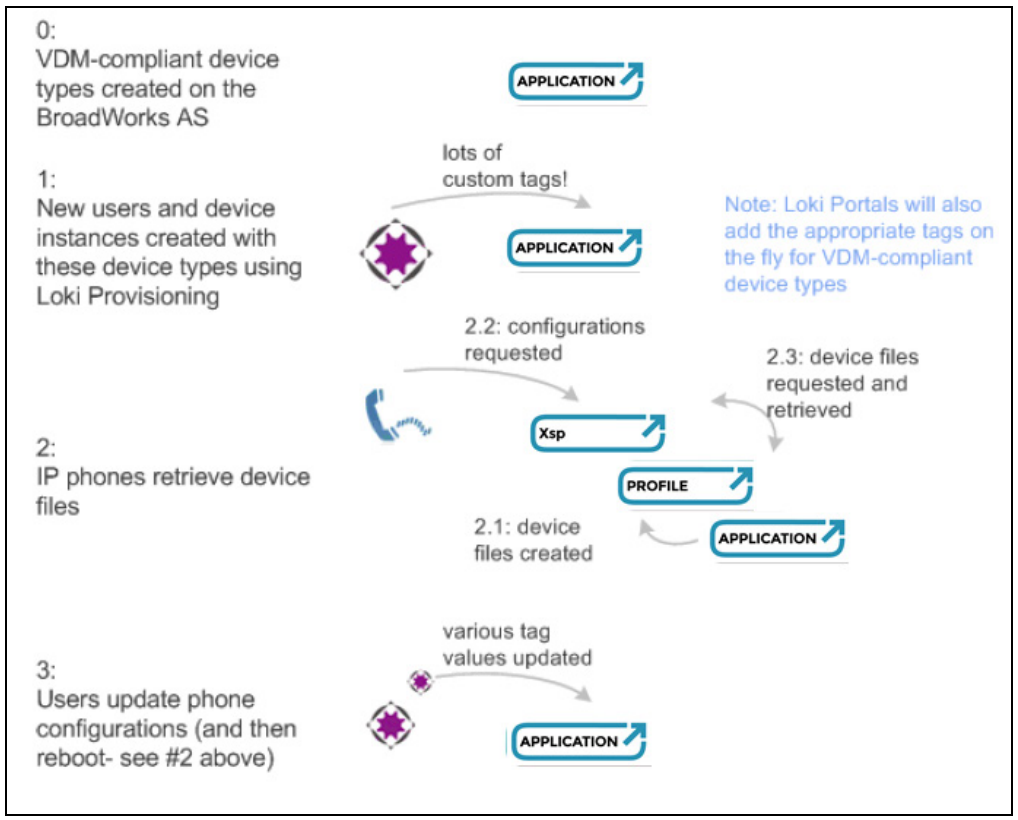


Figure 113 Visual Device Management Process (Leonid Systems)

The devices types supporting Visual Device Management are created in Cisco BroadWorks. These types consist of the following:

- Aastra 39i
- Aastra 55i
- Aastra 57i
- Polycom SoundPoint IP 3xx
- Polycom SoundPoint IP 450
- Polycom SoundPoint IP 550, 560
- Polycom SoundPoint IP 650, 670
- Cisco SPA 525(x)
- Cisco SPA 509G
- Cisco SPA 508G
- Cisco SPA 504G
- Cisco SPA 502G
- Cisco SPA 501G

The configuration file templates for these device types contain specific tags that Loki Portals Visual Device Management uses to manage the state of the device. Leonid Systems has developed DTAF packages for the various device profile types supported in Visual Device Management. These packages are available on the Leonid Systems support web site at support.leonidsystems.com.

To import the various Visual Device Management-enabled device profile types, log in to Cisco BroadWorks as a system administrator and then navigate to *Resources* → *Identity/Device Profile Types*. Next, click **Import** and select the appropriate DTAF file for the device to be added.

Note that the DTAF file import process assumes the necessary tag sets already exist. The Leonid DTAF files, similar to the Cisco CPE Kit DTAF files, attempt to use one of the following tag sets, depending on phone vendor:

- Aastra-Tags
- Cisco-Tags
- Polycom-Tags

Before importing any of the DTAF files for Visual Device Management, make sure the necessary tag set has been created.

After importing the device types, the URL for the Xtended Services Platform must be updated for each device type. Using the CommPilot interface, manually go to the profile page for each imported device type and fill in the *Device Access FQDN* field.

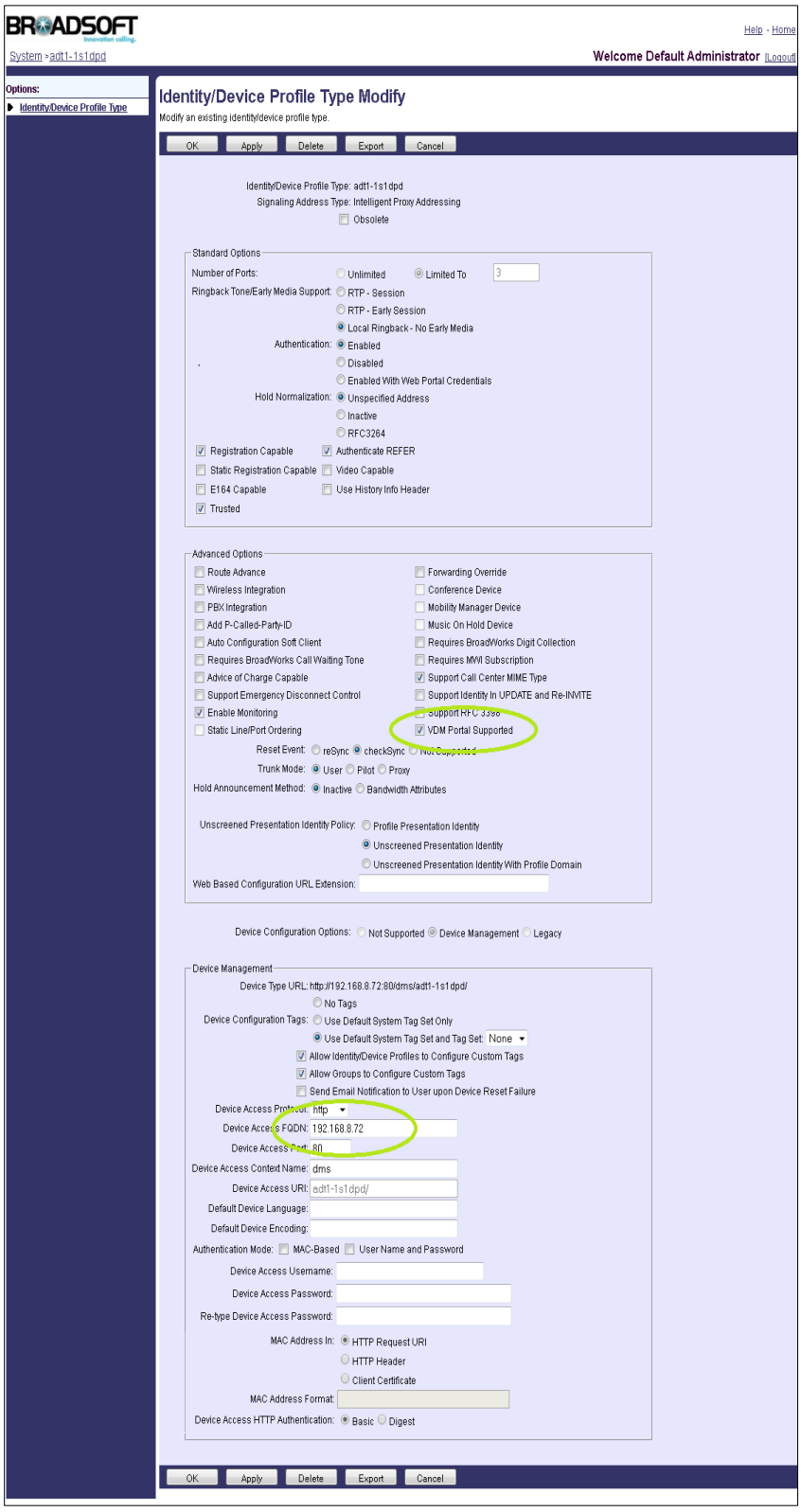


Figure 114 Device Type Profile Modify – Setting Device Access FQDN

Also, make sure that the option “Support Visual Device Management” is selected.

For more information on how to configure the Visual Device Management device types, the template files, and the list of the specific tags, see the Leonid Systems *Loki Portals Administrator’s Guide, Release 1.148* [8].

13.8 Cisco BroadWorks User Provisioning Steps

Once all the steps described in the previous sections have been completed, the administrator must perform the following steps to see the changes introduced by the feature:

1) Create a device profile of a type supporting Visual Device Management.
2) Get a new license for the Visual Device Management service.
3) Authorize Visual Device Management to a service provider.
4) Authorize Visual Device Management to a group.
5) Assign Visual Device Management to a user.
6) Assign the device previously created to the user.

14 Appendix D: Authorization Based on Device Certificate CN Provisioning

14.1 Overview and Purpose

This section describes the provisioning requirements for the Xtended Services Platform Device Management Authorization Based on Device Certificate CN feature [12].

Some devices can provide identification in the form of a certificate (often factory installed). These certificates can contain MAC addresses allowing the identification of individual devices. These certificates can be provided to servers during the Secure Sockets Layer (SSL)/Transport Layer Security (TLS) handshake used to establish secure connections. The server can authenticate the client by validating the certificate chain it provides against certificates the server knows (also known as trust anchors).

The *BroadworksDms* application hosted on the Xtended Services Platform (Xsp) has the ability to obtain a requesting device's media access control (MAC) address from a client certificate common name field (certificate CN) provided by the device when file access authorization is MAC-based.

14.2 Prerequisites on Provisioning Server

- Device Management is enabled and properly provisioned.
- A DM-managed SIP device type is created.
- A dynamic per device configuration file template that is authenticated MAC from HTTP certificate.
- Device configuration files were rebuilt from this template for all devices belonging to the device type and pushed to the file repository successfully.

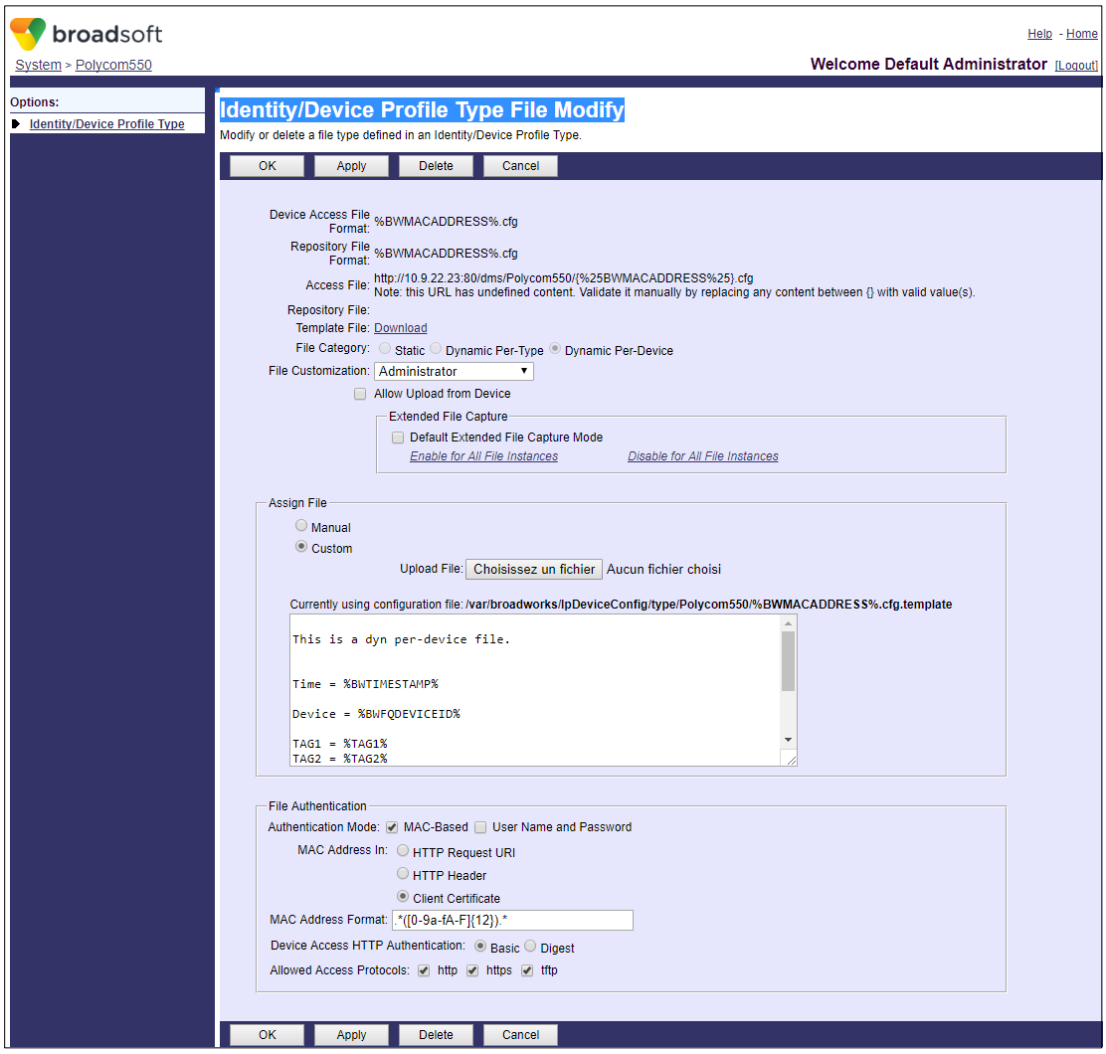


Figure 115 Identity/Device Profile Type File with MAC Address from Certificate Authentication

14.3 Xsp Provisioning Steps

1) From the Xsp CLI, enable the client authentication for the Cisco BroadWorks DMS web application.

```
XSP_CLI/Interface/Http/ClientAuthentication/WebApps> 0
    Interface   Port   Application Name   Client Auth Req
=========================================================
  10.9.22.193   443      BroadworksDms             true
```

2) Provision trusts containing the root certificate for the devices that provide identification in the form of a certificate. For a detailed explanation and steps on how to generate and export trusts, see the *Xsp Client Certificates Verification Enhancements Feature Description* [13].

```
XSP_CLI/Interface/Http/ClientAuthentication/Trusts> ?
This level is used to generate Secure Sockets Layer (SSL) trust anchors.

Commands:
    0)          get : list the trust anchors
    1)  createTrust : generate an trust anchor for a secure connection
```

```
    2)  deleteTrust : remove a trust anchor
    3)  exportTrust : export trust files
    4)    showTrust : show trust anchor certificate
    5)  updateTrust : load and update trust anchors

  h (help),  e (exit),  q (quit),  r (read),  w (write),  t (tree),
  c (config),  cd (cd),  a (alias),  hi (history),  p (pause),  re
(repeat),
  k (keyboardHelp)
XSP_CLI/Interface/Http//ClientAuthentication/Trusts>
```

For example, do the following to load a trust anchor file.

```
XSP_CLI/Interface/Http/ClientAuthentication/Trusts> updateTrust alias1
/var/broadworks/tmp/myTrustAnchorFile.cert
XSP_CLI/Interface/Http/ClientAuthentication/Trusts> 0
```

For example, do the following to generate a trust containing a certificate with a MAC address (111111111111).

```
XSP_CLI/Interface/Http/ClientAuthentication/Trusts> createTrust utRootA
commonName 111111111111
XSP_CLI/Interface/Http/ClientAuthentication/Trusts> 0

                        Alias            Owner
Issuer

=====================================================================
      autogenerated[auto-gen]  My Organization  My Organization[self-
signed]
            utRootA[auto-gen]     111111111111     111111111111[self-
signed]
```

In the previous example, the SSL Client subject DN returned from the HTTP request would look like this:

/CN=111111111111/C=US/ST=Berkshire/L=Newbury/O=unittest/OU=Org unit

In order to extract the MAC address 111111111111 from it, the MAC Address Format to be used should be: .*([0-9a-fA-F]{12}).*

Note that depending on the format of the SSL subject DN, the MAC Address Format to be used could be different. For more information, see section *5.3.2.2.2 Device Type File Configuration Options.*

Acronyms and Abbreviations

This section lists the acronyms and abbreviations found in this document. The acronyms and abbreviations are listed in alphabetical order along with their meanings.

ACL	Access Control List
Admin	Administrator
AFS	Average File Size
AoR	Address of Record
API	Application Programming Interface
AS	Application Server
BLF	Busy Lamp Field
BOOL	Boolean
BSS	Business SIP Services
BW	BroadWorks
CAP	Client Application Protocol
CLI	Command Line Interface
CLID	Calling Line ID
CPE	Customer Premises Equipment
CRM	Customer Relationship Management
DM	Device Management
DMS	Device Management System
DNIS	Dialed Number Identification Service
DNS	Domain Name System
DoS	Denial of Service
DTAF	Device Type Archive File
EMS	Element Management System
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GW	Gateway
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secured
IMS	IP Multimedia Subsystem
IP	Internet Protocol
I/O	Input/Output
IOPS	I/O Operations Per Second

IRSF	International Revenue Share Fraud
ISDN	Integrated Services Digital Network
ISUP IAM	Integrated Services User Part Initial Address Message
IVR	Interactive Voice Response
KPI	Key Performance Indicator
LCD	Liquid Crystal Display
LVM	Logical Volume Manager
MAC	Media Access Control
Mbps	Megabits per second
MGCP	Media Gateway Control Protocol
MIN	Mobile Identification Number
MSC	Mobile Switching Center
MSISDN	Mobile Station ISDN Number
MWI	Message Waiting Indicator
NS	Network Server
OCI	Open Client Interface
OCI-P	Open Client Interface-Provisioning
OCN	Original Called Number
OCS	Open Client Server
OS	Operating System
OSS	Operations Support System
PBX	Private Branch Exchange
PM	Performance Measurement
PS	Profile Server
PS	Provisioning Server
RFC	Request for Comments
RPS	Requests Per Second
SAML	Security Assertion Markup Language
SBC	Session Border Controller
SDP	Session Description Protocol
SFTP	Secure File Transfer Protocol
SIP	Session Initiation Protocol
SME	Small/Medium Enterprise
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol
SSD	Solid State Drive

SSL	Secure Sockets Layer
SSO	Single Sign-On
TDM	Time Division Multiplexing
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VDM	Visual Device Management
WAF	Web Application Firewall
WebDAV	Web-based Distributed Authoring and Versioning
Webapp	Web Application
XML	eXtensible Markup Language
Xsi	Xtended Services Interface
Xsp	Xtended Services Platform

References

[1] Cisco Systems, Inc. 2019. *Cisco BroadWorks Software Management Guide, Release 23.0*. Available from BroadSoft at xchange.broadsoft.com.

[2] Cisco Systems, Inc. 2019. *Cisco BroadWorks Xtended Services Platform Configuration Guide, Release 23.0*. Available from Cisco at xchange.broadsoft.com.

[3] Cisco Systems, Inc. 2019. *Cisco BroadWorks Profile Server Configuration Guide, Release 23.0.* Available from Cisco at xchange.broadsoft.com.

[4] Cisco Systems, Inc. 2019. *Cisco BroadWorks Network Server Command Line Interface Administration Guide, Release 23.0.* Available from Cisco at xchange.broadsoft.com.

[5] Cisco Systems, Inc. 2019. *Cisco BroadWorks System Configuration Guide, Release 23.0*. Available from Cisco at xchange.broadsoft.com.

[6] Cisco Systems, Inc. 2013. *Device Management LinePort Ordering Enhancement Feature Description, Release 19.0*. Available from Cisco at xchange.broadsoft.com.

[7] Cisco Systems, Inc. 2013. *Visual Device Management Support Feature Description, Release 20.0*. Available from Cisco at xchange.broadsoft.com.

[8] Leonid Systems 2012. *Loki Portals 2.21 Administrator's Guide, Release 1.148.* Available from Leonid Systems.

[9] Leonid Systems 2011. *SSO Integration with Loki Portals, Release 1.0.* Available from Leonid Systems.

[10] Leonid Systems 2013. *System Preparation Guide, Release 1.72.* Available from Leonid Systems.

[11] Cisco Systems, Inc. 2019. *Cisco BroadWorks Device Management Tag Reference Guide, Release 23.0*. Available from Cisco at xchange.broadsoft.com.

[12] Cisco Systems, Inc. 2015. *Xtended Services Platform Device Management Authorization Based on Device Certificate CN Feature Description*, *Release 22.0*. Available from Cisco at xchange.broadsoft.com.

[13] Cisco Systems, Inc. 2014. *Xsp Client Certificates Verification Enhancements Feature Description*, *Release 21.0*. Available from Cisco at xchange.broadsoft.com.

[14] Cisco Systems, Inc. 2013. *System Engineering of Device Management* from the *2013 Customer Summits*. Available from Cisco at xchange.broadsoft.com.

[15] Cisco Systems, Inc. 2019. *Cisco BroadWorks SSL Support Options Guide, Release 23.0*. Available from Cisco at xchange.broadsoft.com