Cisco Meeting Server

Cisco Meeting Server Release 2.2
API Reference Guide

November 21, 2018
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change History</td>
<td>10</td>
</tr>
<tr>
<td>1 General Information</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>1.1 How to Use this Document</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.2 Cisco Meeting Server Release 2.2 API Additions</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>2 General Structure of Objects</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>2.1 Object Hierarchy</td>
</tr>
<tr>
<td>3 Accessing the API</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>3.1 Configuration Settings</td>
</tr>
<tr>
<td></td>
<td>3.2 Authentication</td>
</tr>
<tr>
<td></td>
<td>3.3 Tools to Use</td>
</tr>
<tr>
<td>4 API Methods</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>4.1 GET Methods</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.2 POST Methods</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.3 PUT Methods</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.4 DELETE Methods</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>4.5</td>
<td>Failure Reasons</td>
</tr>
<tr>
<td>5</td>
<td>Example Requests and Responses for Specific Methods</td>
</tr>
<tr>
<td>5.1</td>
<td>Retrieval of Current Active Calls</td>
</tr>
<tr>
<td>5.2</td>
<td>Creating a New Call Leg (SIP dial out to 10.1.144.129)</td>
</tr>
<tr>
<td>6</td>
<td>coSpace Related Methods</td>
</tr>
<tr>
<td>6.1</td>
<td>Retrieving coSpaces</td>
</tr>
<tr>
<td>6.2</td>
<td>Creating and Modifying a coSpace</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Secondary coSpace URLs</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Auto-generation of coSpace callId</td>
</tr>
<tr>
<td>6.3</td>
<td>Retrieving Detailed Information about a Single coSpace</td>
</tr>
<tr>
<td>6.3.1</td>
<td>Retrieving entry details for a specific coSpace</td>
</tr>
<tr>
<td>6.4</td>
<td>coSpace Member Methods</td>
</tr>
<tr>
<td>6.4.1</td>
<td>Retrieving the members of a coSpace</td>
</tr>
<tr>
<td>6.4.2</td>
<td>Adding and modifying a coSpace member</td>
</tr>
<tr>
<td>6.4.3</td>
<td>Retrieving Information on a coSpace member</td>
</tr>
<tr>
<td>6.5</td>
<td>coSpace Chat/Message Board Methods</td>
</tr>
<tr>
<td>6.5.1</td>
<td>Posting to the message board of a coSpace</td>
</tr>
<tr>
<td>6.5.2</td>
<td>Deleting messages from a coSpace message board</td>
</tr>
<tr>
<td>6.6</td>
<td>Multiple coSpace Access Methods</td>
</tr>
<tr>
<td>6.6.1</td>
<td>General information</td>
</tr>
<tr>
<td>6.6.2</td>
<td>Retrieving coSpace access methods</td>
</tr>
<tr>
<td>6.6.3</td>
<td>Creating and modifying coSpace access methods</td>
</tr>
<tr>
<td>6.6.4</td>
<td>Retrieving information on an individual coSpace access method</td>
</tr>
<tr>
<td>6.7</td>
<td>Calling Out from a coSpace</td>
</tr>
<tr>
<td>6.8</td>
<td>Bulk creating, updating and deleting coSpaces</td>
</tr>
<tr>
<td>6.8.1</td>
<td>Creating /coSpaceBulkParameterSets</td>
</tr>
<tr>
<td>6.8.2</td>
<td>Retrieving the parameter sets for creating coSpaces in bulk</td>
</tr>
<tr>
<td>6.8.3</td>
<td>Retrieving information on an individual /coSpaceBulkParameterSet</td>
</tr>
<tr>
<td>6.8.4</td>
<td>Queueing the bulk sync operations</td>
</tr>
<tr>
<td>6.8.5</td>
<td>Retrieving the bulk sync operations</td>
</tr>
<tr>
<td>6.8.6</td>
<td>Retrieving a specific bulk sync operation</td>
</tr>
<tr>
<td>6.8.7</td>
<td>Examples</td>
</tr>
<tr>
<td>6.9</td>
<td>coSpace Diagnostics Methods</td>
</tr>
<tr>
<td>7</td>
<td>Dial Plan Methods</td>
</tr>
<tr>
<td>7.1</td>
<td>Outgoing Dial Plan API Methods</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>7.1.1</td>
<td>Access to the outgoing dial plan</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Retrieving outbound dial plan rules</td>
</tr>
<tr>
<td>7.1.3</td>
<td>Creating and modifying outbound dial plan rules</td>
</tr>
<tr>
<td>7.1.4</td>
<td>Retrieving information on an individual outbound dial plan rule</td>
</tr>
<tr>
<td>7.2</td>
<td>Incoming Call Matching Dial Plan API Methods</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Access to incoming domain matching rules</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Retrieving incoming dial plan rules</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Creating and modifying incoming dial plan rules</td>
</tr>
<tr>
<td>7.2.4</td>
<td>Retrieving information on an individual incoming dial plan rule</td>
</tr>
<tr>
<td>7.3</td>
<td>Incoming Call Forwarding Dial Plan API Methods</td>
</tr>
<tr>
<td>7.3.1</td>
<td>Access to incoming call forwarding rules</td>
</tr>
<tr>
<td>7.3.2</td>
<td>Retrieving incoming call forwarding dial plan rules</td>
</tr>
<tr>
<td>7.3.3</td>
<td>Creating and modifying incoming call forwarding dial plan rules</td>
</tr>
<tr>
<td>7.3.4</td>
<td>Retrieving information on an individual incoming call forwarding dial plan rule</td>
</tr>
<tr>
<td>8</td>
<td>Call Related Methods</td>
</tr>
<tr>
<td>8.1</td>
<td>Call Methods</td>
</tr>
<tr>
<td>8.1.1</td>
<td>Retrieving Information on Active Calls</td>
</tr>
<tr>
<td>8.1.2</td>
<td>Creating a New Call and Modifying an Active call</td>
</tr>
<tr>
<td>8.1.3</td>
<td>Retrieving Information on an Individual Active Call</td>
</tr>
<tr>
<td>8.1.4</td>
<td>Generating diagnostics for an individual call</td>
</tr>
<tr>
<td>8.1.5</td>
<td>Retrieve participants in a conference</td>
</tr>
<tr>
<td>8.1.6</td>
<td>Creating a new participant for a specified call</td>
</tr>
<tr>
<td>8.1.7</td>
<td>Set properties for all participants in a conference</td>
</tr>
<tr>
<td>8.2</td>
<td>Call Profile Methods</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Retrieving call profiles</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Setting up and modifying call profiles</td>
</tr>
<tr>
<td>8.2.3</td>
<td>Retrieving detailed information about an individual call profile</td>
</tr>
<tr>
<td>8.3</td>
<td>Call Leg Methods</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Retrieving Information on Active Call Legs</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Adding and Modifying Call Legs</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Retrieving Information on an Individual Call Leg</td>
</tr>
<tr>
<td>8.4</td>
<td>Call Leg Profile Methods</td>
</tr>
<tr>
<td>8.4.1</td>
<td>General information</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Retrieving call leg profiles</td>
</tr>
<tr>
<td>8.4.3</td>
<td>Creating and modifying a call leg profile</td>
</tr>
</tbody>
</table>
8.4.4 Retrieving information on an individual call leg profile ........................................... 102
8.4.5 Example call leg profile & access method use .......................................................... 103
8.4.6 /callLegProfiles/<call leg profile id>/usage object method .................................... 103
8.4.7 /callLegs/<call leg id>/callLegProfileTrace object method .................................... 103
8.4.8 /callLegs/<call leg id>/generateKeyframe ................................................................. 106

8.5 DialTransform Methods ................................................................................................. 106
8.5.1 Retrieving dial transforms .......................................................................................... 106
8.5.2 Setting up and modifying dial transforms ................................................................... 107
8.5.3 Retrieving detailed information about an individual dial transform ......................... 108
8.5.4 Examples ................................................................................................................... 108

8.6 Setting Individual Features for a Call Leg .................................................................... 109

8.7 Call Branding Profile Methods ....................................................................................... 109
8.7.1 Retrieving Call Branding Profiles ............................................................................. 109
8.7.2 Setting up and modifying call branding profiles ......................................................... 110
8.7.3 Retrieving detailed information about an individual call branding profile ............ 110

8.8 DTMF Profile Methods .................................................................................................. 110
8.8.1 Retrieving DTMF Profiles .......................................................................................... 110
8.8.2 Setting up and modifying DTMF Profiles .................................................................. 112
8.8.3 Retrieving detailed information about an individual dtmfProfile ................................ 114

8.9 IVR Methods .................................................................................................................. 114
8.9.1 Retrieving IVRs .......................................................................................................... 114
8.9.2 Setting up and modifying IVRs ................................................................................. 115
8.9.3 Retrieving detailed information about an individual IVR ........................................ 115

8.10 IVR Branding Profile Methods ..................................................................................... 115
8.10.1 Retrieving IVR branding profiles ............................................................................ 115
8.10.2 Setting up and modifying an IVR branding profile .................................................. 116
8.10.3 Retrieving detailed information about an individual IVR branding profile ............ 116

8.11 Participant Related Methods .......................................................................................... 116
8.11.1 Retrieving participants ............................................................................................. 117
8.11.2 Making a participant important if they are already in a conference ....................... 118
8.11.3 Retrieving detailed information on an individual participant ................................. 118
8.11.4 Retrieving a participant’s call legs ............................................................................ 119
8.11.5 Limiting a call’s participants ................................................................................... 120

9 User Related Methods ....................................................................................................... 121
9.1 Retrieving Users ........................................................................................................... 121
9.2 Retrieving Detailed Information on an Individual User ................................................ 121
9.3  User Profile Methods ................................................................. 122
  9.3.1  Retrieving user profiles ......................................................... 122
  9.3.2  Setting up and modifying user profiles .................................... 124
  9.3.3  Retrieving detailed information about an individual user profile .. 125

10  System Related Methods ............................................................. 126
  10.1  Retrieving System Status ......................................................... 126
  10.2  Retrieving System Alarm Status .............................................. 127
  10.3  Retrieving System Database Status ......................................... 130
  10.4  CDR Receiver URI Methods .................................................... 131
        10.4.1  Retrieving the CDR Receivers URI ................................. 131
        10.4.2  Setting the CDR Receivers URI ..................................... 132
  10.5  Global Profile Methods ......................................................... 132
        10.5.1  Retrieving the Global Profile ....................................... 132
        10.5.2  Setting the Global Profile .......................................... 132
  10.6  Licensing Methods ............................................................... 133
        10.6.1  Retriving information on licensing .................................. 133
  10.7  TURN Server Methods ........................................................... 136
        10.7.1  Retrieving Information on TURN Servers ......................... 136
        10.7.2  Setting up and modifying TURN servers ............................ 137
        10.7.3  Retrieving detailed information about an individual TURN server 137
        10.7.4  Retrieving individual TURN Server status ....................... 138
  10.8  Web Bridge Methods ............................................................ 139
        10.8.1  Retrieving Information on Web Bridges ......................... 139
        10.8.2  Setting Up and Modifying a Web Bridge ......................... 139
        10.8.3  Retrieving detailed information about an individual Web Bridge 140
        10.8.4  Updating the Web Bridge customization .......................... 141
        10.8.5  Retrieving diagnostics on a Web Bridge (from 2.2) .......... 141
  10.9  Call Bridge Methods ............................................................ 141
        10.9.1  Retrieving Information on Call Bridges ......................... 141
        10.9.2  Setting Up and Modifying a Call Bridge ....................... 141
        10.9.3  Retrieving detailed information about an individual Call Bridge 142
  10.10 Call Bridge Group Methods .................................................. 142
        10.10.1  Retrieving Information on Call Bridge Groups ............... 142
        10.10.2  Setting Up and Modifying a Call Bridge Group ............. 142
        10.10.3  Retrieving detailed information about an individual Call Bridge Group ... 143
  10.11  XMPP Methods ................................................................... 143
10.11.1 Retrieving the XMPP server details ..................................................... 143
10.11.2 Setting Up and Modifying an XMPP server ........................................... 143
10.12 Call Bridge Cluster Methods ................................................................. 144
  10.12.1 Retrieving the Call Bridge Cluster details ........................................... 144
  10.12.2 Setting Up and Modifying the Call Bridge Cluster ......................... 144
10.13 Recorder methods .................................................................................. 145
  10.13.1 Retrieving recorder operations ........................................................... 145
  10.13.2 Setting up and modifying recording operations ............................... 146
  10.13.3 Retrieving detailed information about an individual recording node ... 146
  10.13.4 Retrieving diagnostics on a Recorder (from 2.2) .............................. 146
10.14 Streamer methods .................................................................................. 147
  10.14.1 Retrieving streamer operations ........................................................... 147
  10.14.2 Setting up and modifying streaming operations .............................. 147
  10.14.3 Retrieving detailed information about an individual streaming node ... 148
  10.14.4 Retrieving diagnostics on a Streamer (from 2.2) .............................. 148
10.15 System Load Method ............................................................................... 148
10.16 Compatibility Profile Methods ............................................................... 149
  10.16.1 Retrieving compatibility profile operations ......................................... 149
  10.16.2 Setting up and modifying compatibility profile operations ............. 149
10.17 System Diagnostics Methods ................................................................ 150
  10.17.1 Retrieving system diagnostics ............................................................ 150
  10.17.2 Retrieving an individual system diagnostic ........................................ 151
  10.17.3 Retrieving the content of an individual system diagnostic ............... 151
11 LDAP Methods ........................................................................................... 152
  11.1 LDAP Server Methods .......................................................................... 153
    11.1.1 Retrieving Information on LDAP Servers ........................................ 153
    11.1.2 Adding and modifying an LDAP Server .......................................... 154
    11.1.3 Retrieving detailed information about an individual LDAP Server .... 154
  11.2 LDAP Mapping Methods ....................................................................... 155
    11.2.1 Adding and modifying an LDAP Mapping ....................................... 155
    11.2.2 Secondary LDAP Mapping parameter ............................................ 156
    11.2.3 Retrieving information on LDAP Mappings .................................... 156
    11.2.4 Retrieving detailed Information about an individual LDAP Mapping ... 157
  11.3 LDAP Source Methods .......................................................................... 157
    11.3.1 Retrieving Information on LDAP Sources ....................................... 157
    11.3.2 Adding and modifying an LDAP Source ........................................ 158
11.3.3 Retrieving detailed information on a LDAP Source ........................................ 158
11.4 LDAP Sync Methods ......................................................................................... 158
  11.4.1 Retrieving scheduled LDAP sync methods .............................................. 159
  11.4.2 Initiating a new LDAP sync ..................................................................... 159
  11.4.3 Cancelling a scheduled LDAP sync .......................................................... 160
  11.4.4 Retrieving information on a single LDAP sync method ....................... 160
11.5 External Directory Search Locations ............................................................. 160
  11.5.1 Retrieving Information on external directory search locations .......... 161
  11.5.2 Adding and modifying external directory search locations ................. 161
  11.5.3 Retrieving detailed information on external directory search locations ... 162
  11.5.4 Example of adding external directory search locations ..................... 162

12 Multi-tenancy ........................................................................................................ 166
  12.1 Tenants .......................................................................................................... 167
    12.1.1 Retrieving Tenants ................................................................................. 167
    12.1.2 Creating and Modifying a Tenant ......................................................... 168
    12.1.3 Retrieving Detailed Information about an Individual Tenant .......... 169
  12.2 Tenant group operations ............................................................................. 169
    12.2.1 Retrieving Tenant Groups .................................................................... 169
    12.2.2 Creating and Modifying a Tenant Group .............................................. 169
    12.2.3 Retrieving Detailed Information about an Individual Tenant Group .. 169

13 Query Methods .................................................................................................... 170
  13.1 accessQuery Method ................................................................................... 170

14 Using Profiles ...................................................................................................... 171

Appendix A Additional Call Leg Information ....................................................... 173
  A.1 Call Leg Information ..................................................................................... 173
    A.1.1 Call leg configuration ............................................................................ 173
    A.1.2 Call leg status ....................................................................................... 174

Appendix B Additional Multiparty Licensing Information .................................... 176
  B.1 /system/licensing Information .................................................................. 176
  B.2 /system/multipartyLicensing Information .................................................. 177

Appendix C Installing API tools ............................................................................. 178
  C.1 Installing Chrome Postman ......................................................................... 178
  C.2 Installing Firefox Poster ............................................................................. 180
C.3 Installing Chrome Advanced Rest Client .........................................................183
Appendix D Using Firefox Poster with the API .........................................................187
Cisco Legal Information .........................................................................................190
Cisco Trademark .................................................................................................191

Figures:

Figure 1: Cisco Meeting Server documentation for release 2.2 ...............................13
Figure 2: Outline LDAP process ............................................................................153
Figure 3: Outline multi-tenancy process ................................................................167
Figure 4: Value inheritance with profiles ..............................................................172
Figure 5: Value inheritance with profiles when using tenants ..................................172
## Change History

<table>
<thead>
<tr>
<th>Date</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nov 21, 2018</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>Oct 16, 2018</td>
<td>Minor improvements.</td>
</tr>
<tr>
<td>Sept 24, 2018</td>
<td>Introduced the new <a href="#">interactive API reference tool</a>.</td>
</tr>
<tr>
<td>July 11, 2018</td>
<td>Corrections to <a href="#">Compatibility Profile Methods</a>.</td>
</tr>
<tr>
<td>June 13, 2018</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>June 1, 2018</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>May 10, 2018</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>April 24, 2018</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>February 14, 2018</td>
<td>Minor correction.</td>
</tr>
<tr>
<td>January 12, 2018</td>
<td>Minor correction.</td>
</tr>
<tr>
<td>December 11, 2017</td>
<td>Updated with missing information on /calls/&lt;call id&gt;/participants/ and /calls/&lt;call id&gt;/participants/*</td>
</tr>
<tr>
<td>October 31, 2017</td>
<td>Minor corrections.</td>
</tr>
<tr>
<td>June 17, 2017</td>
<td>Added note on weightedCallsActive, see Licensing Methods section in System Related Methods.</td>
</tr>
<tr>
<td>May 03, 2017</td>
<td>New version for Cisco Meeting Server 2.2, see Section 1.2 for an overview of the objects and parameters added.</td>
</tr>
</tbody>
</table>
1 General Information

The Cisco Meeting Server was formerly called the Acano Server. The Cisco Meeting Server software can be hosted on specific servers based on Cisco Unified Computing Server (UCS) technology as well as on the Acano X-Series hardware, or on a specification-based VM server. Cisco Meeting Server is referred to as the Meeting Server throughout this document.

This document covers release 2.2 of the Application Program Interface for the Cisco Meeting Server.

Note: The Cisco Meeting Server software is referred to as the Meeting Server throughout the remainder of this guide.

1.1 How to Use this Document

We suggest that you start by reading sections 2 to 5 inclusive and in order. They provide the concepts behind the API, the way to use the API methods and some examples of usage.

Then use the remaining sections as reference material for the method(s) that you want to use.

This guide is part of the documentation set (shown in Figure 1) for the Meeting Server.

1.1.1 Interactive API Reference tool

We recently introduced a new interactive API reference tool enabling you to see a high level view of the API objects and drill down to lower levels for the detail. There are also learning labs to help you get started, these will be added to over time. We encourage you to try out this tool; sometime in the future we will discontinue publishing the pdf version of the API Reference Guide.

https://developer.cisco.com/cisco-meeting-server/

Steps to use the tool:

1. Click View the docs
2. Select a category from the list in the left pane. For example: Call Related Methods.
3. Click on any method to see URI: GET/POST/PUT. Refer to the table of parameters and response elements with descriptions. For example: GET https://ciscocms.docs.apiary.io/api/v1/calls?

Note: If you are using a POST/PUT methods, the related 'Attributes' with descriptions appear on the right-hand pane when you select the method.

Learning labs

https://learninglabs.cisco.com/modules/cisco-meeting-server
The learning labs are intended as a starting point, covering a broad cross-section of what is possible with the Cisco Meeting Server API. Every learning lab is a step-by-step tutorial which takes you through the steps to complete the task from start to finish.

Example: The 'Setting up host and guest access with Cisco Meeting Server API' provides instructions to configure ways in which users can join meetings in a space with different options.
1.2 Cisco Meeting Server Release 2.2 API Additions

New API functionality for the Meeting Server 2.2 includes support for:
Office 365 Dual Home Experience with OBTP Scheduling
setting the maximum quality levels for main video and content
load balancing of outbound calls to SIP endpoints
diagnostics for recordings, streamings and Web Bridges
enabling and disabling the on-screen participant counter

there are also some other miscellaneous additions.
You are advised not to use beta features in a production environment. Only use them in a test environment until they are fully released.

1.2.1 Office 365 Dual Home Experience with OBTP Scheduling
New request parameter added to /inboundDialPlanRules and
/inboundDialPlanRules/<inbound dial plan rule ID>:resolveToLyncSimpleJoin

1.2.2 Setting the maximum quality levels for main video and content
New request parameters to /callLegProfile: qualityMain, qualityPresentation

1.2.3 Support for load balancing of outbound calls to SIP endpoints
New request parameter to /callBridgeGroups: loadBalanceOutgoingCalls
New request parameters to /calls/<call id>/participants: callBridgeGroup, callBridge for POST operations only
New request parameter to /outboundDialPlanRules and
/outboundDialPlanRules/<outbound dial plan rule id>: callBridgeGroup
Added value to scope parameter for /outboundDialPlanRules: callBridgeGroup
New request parameter to /calls: activeWhenEmpty

1.2.4 Assigning an Importance level to participants to control the screen layout
New request parameter to /calls/<call id>/participants/:importance (POST only)
New request parameter to /participants/<participant id>:importance (PUT only)
NewAPI object: /calls/<call id>/participants/* with request parameter: importance (PUT only)

1.2.5 Retrieving diagnostics on a Recorder/Streamer/Web Bridge
New node added to /recorders/<recorder id>, /streamers/<streamer id>, /webBridges/<web bridge id>: status
1.2.6  Support to disable and re-enable the on-screen participant counter

New request parameter added to /callLegProfiles: participantCounter

1.2.7  Miscellaneous additions

- New request parameter added to /coSpaces and /coSpaces/<coSpace ID>: meetingScheduler
- New read-only field added to /calls/<call ID>: ownerName
- New request parameter added to /system/status: cdrCorrelatorIndex. This support external tools to the Meeting Server determining whether they have received all CDR records that have been sent.
- New operation to /calls/<call id>/participants/* with attributes layout or (rx|tx) (Audio|Video)Mute
- New request parameter added to /compatibilityProfiles and /compatibilityProfiles/<compatibilityProfile ID>: sipMediaPayloadTypeMode
- New object /callLegs/<call leg id>/generateKeyframe. POST to /callLegs/<call leg id>/generateKeyframe to trigger the generation of a new keyframe in outgoing video streams for the call leg in question. This is a debug facility, and Cisco Support may ask you to use the feature when diagnosing an issue.
2 Growth of Objects

The Meeting Server’s Application Programming Interface (API) is designed as a hierarchy of objects, like the trunk and roots of a tree. For example, each configured coSpace exists as a node in this tree, and all of the users who are members of that coSpace exist as nodes “beneath” the coSpace object’s node. The API objects are accessed using a suitable REST client, see Section 3.3.

**Note:** Although the Cisco Meeting App and other Cisco Meeting Server guides refer to "spaces" rather than "coSpaces", the API still uses /coSpace objects.

The Meeting Server has the potential to host a large number of active calls and coSpaces. To reduce the overhead of retrieving the entire collection of objects in a single response, responses typically return the first “N” matching entries and a count of the total number of objects of that type. To find an individual object’s active status, or to modify or delete it, use filters on the initial retrieval in order to identify the object in question. Refer to Section 4.1 for more information on filters and the GET command.

2.1 Object Hierarchy

The hierarchy of objects addressable via the API is:

```
/accessQuery
/callBrandingProfiles
/callBrandingProfiles/<call branding profile id>
/callBridges
/callBridges/<call bridge id>
/callBridgeGroups (2.1 onwards)
/callBridgeGroups/<call bridge group id> (2.1 onwards)
/calls
/calls/<call id>
/calls/<call id>/callLegs
/calls/<call id>/diagnostics
/calls/<call id>/participants
/calls/<call id>/participants/* (2.2 onwards)
/callProfiles
/callProfiles/<call profile id>
/callLegs
/callLegs/<callLeg id>
/callLegs/<callLeg id>/callLegProfileTrace
/callLegProfiles
```
/callLegProfiles/<call leg profile id>
/callLegProfiles/<call leg profile id>/usage

/compatibilityProfiles (2.1 onwards)
/compatibilityProfiles/<compatibility profile id> (2.1 onwards)

/coSpaceBulkParameterSets (2.0 onwards)
/coSpaceBulkParameterSets/<coSpace bulk parameter set id> (2.0 onwards)
/coSpaceBulkSyncs (2.0 onwards)
/coSpaceBulkSyncs/<coSpace bulk sync id> (2.0 onwards)

/coSpaces
/coSpaces/<coSpace id>
/coSpaces/<coSpace id>/accessMethods
/coSpaces/<coSpace id>/accessMethods/<access method id>
/coSpaces/<coSpace id>/coSpaceUsers
/coSpaces/<coSpace id>/coSpaceUsers/<coSpaceUser id>
/coSpaces/<coSpace id>/diagnostics
/coSpaces/<coSpace id>/messages

dialTransforms
/dialTransforms/<dial transform id>

directorySearchLocations
/directorySearchLocations/<directory search location id>

dtmfProfiles
/dtmfProfiles/<dtmf profile id>

-forwardingDialPlanRules
/-forwardingDialPlanRules/<forwarding dial plan rule id>

/inboundDialPlanRules
/inboundDialPlanRules/<inbound dial plan rule id>

/ivr
/ivr/<ivr id>
/ivrBrandingProfiles
/ivrBrandingProfiles/<ivr branding profile id>

/ldapMappings
/ldapMappings/<ldap mapping id>
/ldapServers
/ldapServers/<ldap server id>
/ldapSources
/ldapSources/<ldap source id>
/ldapSyncs
/ldapSyncs/<ldap sync id>

/outboundDialPlanRules
/outboundDialPlanRules/<outbound dial plan rule id>
In each case, the top level plural term sits above potentially many individual object nodes; these individual object nodes are identified by an <ID> which is a GUID, typically. For example, if there
are 100 coSpaces configured on an Meeting Server, conceptually there would be 100 nodes directly beneath “/coSpaces” in the hierarchy.
3 Accessing the API

The API uses HTTPS as a transport mechanism.

**Note:** The time taken for API requests can vary depending on factors, including but not limited to, the request type, number of outstanding requests, database size, server loading, latency between API client and the Call Bridge receiving API requests, and latency between the Call Bridge receiving the API request and the database master. We recommend that when developing applications, you test API performance on a representative system.

### 3.1 Configuration Settings

To use the API, you need to connect via HTTPS via the same TCP ports as you would use to access the Web Admin Interface – typically port 443; that is, they use the same interface.

You also need to configure a username and password: you must provide these credentials in order to use the API. Set them using the MMP command `user add <username> (admin|crypto|audit|appadmin|api)`. This command prompts for the user’s password; see the MMP Command Reference for details.

### 3.2 Authentication

The API user supplies a shared secret username and password to the Meeting Server configured with the same username and password. The username and password are set in the MMP command line.

While the authentication credentials are sent in essentially plain text within the HTTP traffic, by using HTTPS the traffic itself cannot be read by an external party.

### 3.3 Tools to Use

Suitable tools to access and update the API include:

- Firefox Poster
- Chrome Postman
- Chrome Advanced Rest Client

see Appendix C for the installation instructions.

Appendix D provides an example of using the Firefox Poster tool.
4 API Methods

There are four methods:

- **GET** is used for retrieval of existing information
- **POST** is used to create new objects in the hierarchy
- **PUT** is used to modify an existing object
- **DELETE** is used to destroy an object in the tree

These methods are described in more detail below.

**URL format**

For the purposes of addressing or creating individual objects, the URL format mirrors the conceptual hierarchy of objects, with some additional preceding tags in order to identify that the request is for the API. By way of example, to retrieve information on API object “/calls/dbfca0dd-dbe1-43bb-b101-beb9a7ef35f4” it would be necessary to issue:

```
GET /api/v1/calls/dbfca0dd-dbe1-43bb-b101-beb9a7ef35f4 HTTP/1.1
```

Specifically, at the top level, including “/api” means that the on-board HTTP server process can distinguish this HTTP method from a normal browser method, and including “v1” means the API handler knows that the request is being made by an object that understands version 1 of the API.

If an API method is successful, it yields a “200 OK” response from the Meeting Server. If an error occurs, the Meeting Server responds with a 4xx or 5xx HTTP status code.

A 503 (“Service Unavailable”) status code is returned for API calls unable to be serviced due to a temporary “busy” condition on the Meeting Server – this can be used as an indication to the requestor that it would be useful to re-attempt the same method later.

Equally, a request supplying a `<coSpace id>` which does not correspond to a valid coSpace object yields “404 Not Found”.

For 4xx and 5xx error cases, extended error information may be returned as “text/xml” body data, for example:

```
<failureDetails>
<coSpaceDoesNotExist/>
</failureDetails>
```

More generally, such a response consists of a “failureDetails” section and a list of errors; in the above case a method was attempted using a coSpace ID that did not correspond to an active coSpace. The possible failure reasons are described in Section 4.5.

4.1 GET Methods

As mentioned above, GET methods allow retrieval of information about existing API-accessible objects, and are used at two levels: Collections level and Individual object level.
4.1.1 Collections level

If the GET method is performed at the Collections level (the pluralized noun: “calls”, “coSpaces” etc.) then multiple matching child nodes will be retrieved. By design this is not guaranteed to be the entire list, but the total number of objects of that type present in the Meeting Server can be learnt via this mechanism.

In order to retrieve just specific items, most GET methods at the Collections level allow the use of a filter expression. The idea here is that the interface of a management tool would initially present the API user with the summed count of coSpaces (for example), basic details on the first “N” coSpaces (e.g. their names) and a filter box which the human user can use to search for the specific coSpace(s) of interest.

With no other additional parameters, a GET method at the Collections level will return items from the start of the Meeting Server’s notional complete list. By comparing the number of items returned with the "total" value, it is straightforward to determine whether all elements have been returned (if the number of elements returned is equal to the "total" value).

4.1.2 Using limit and offset at the Collections level

It is possible to restrict the number of elements returned to a limit chosen by the requestor, by including a "limit=<limitValue>" in the API request. This guarantees that no more than the specified "limitValue" number of elements will be returned - the Meeting Server will have its own limit in these cases too, and therefore the number of elements returned will be the lower of any supplied "limitValue" and the Meeting Server’s own limit.

In order to retrieve elements other than the first "N" on the Meeting Server’s notional list, it is also possible to supply an "offset=<offsetValue>" in the API request. This causes the Meeting Server to return elements which start at the specified position in its list, skipping the first "offsetValue" number of elements. If "offsetValue" is greater than the number of objects of that type, then no elements will be returned.

Note: The offset value should not be viewed as a general mechanism for retrieving a large complete list – sequential retrievals of one "page" of data followed by a second "page" will not necessarily be operating on the same complete list if any objects have been deleted or added in the interval between these methods.

The expectation is that, for each request and response, the requestor will keep track of the offset and limit values used, and combine this knowledge with the number of elements returned in the response and the "total" indicated by the Meeting Server. If the "offsetValue" supplied by the requestor plus the number of elements returned is less than the "total" value indicated in the response, the requestor then knows that there are more values present. The following table shows some examples:
4.1.3 Individual object level

If the GET method is performed at the Individual object level, full information about just that one object will be returned. For example, after the unique ID of a coSpace has been learnt via a (potentially filtered) GET of the “/coSpaces” node, a subsequent GET of the “/coSpaces/<coSpace id>” node would return expanded information about just that one coSpace, for example how many members it has, and when it was last activated.

4.1.4 HTTP specifics

GET methods contain the complete node location and any parameters specific to the retrieval being performed in the URI supplied by the API user. For example, to retrieve basic information on the first “N” coSpaces, the URI would be:

/api/v1/coSpaces

whereas to list just those whose name includes “sales”, the GET would be performed on:

/api/v1/coSpaces?filter=sales

If a GET method has been successful and yields a “200 OK” response, the Meeting Server returns the retrieved information as “text/xml” body data.
4.1.5 How GET methods are detailed in this document

For each GET method at the Collections level the following information is provided:

- The node it operates on
- A table of form parameters, such as filter, offset and limit mentioned above, some of which may be optional. Mandatory parameters are marked with an asterisk (*)
- A table showing the returned information

Both tables show the format of the parameter (e.g. ID or string) or the possible values (e.g. true|false)

For each GET method at the Individual level the following information is provided:

- The node it operates on
- A table showing the returned information

The form parameters are those for the Collections level, unless otherwise indicated.

4.2 POST Methods

POST methods create new objects; for example, to create a new configured coSpace or dial plan rule. Using a POST method to create a new call leg associated with a coSpace is the way to make a new outbound SIP connection.

4.2.1 HTTP specifics

Most POST methods require some parameters to be supplied: for example, creation of a coSpace requires the new coSpace’s name to be specified, and a new call leg can only be created if the remote party’s address is known. Such parameters must be supplied by the initiator of the POST method via the standard HTTP “x-www-form-urlencoded” format, as used by “<form>” elements in an HTML document.

If a POST method has been successful in adding a new object to the hierarchy, that object’s id, and its position within the hierarchy are returned in the “Location” field of the response.

4.2.2 How POST methods are detailed in this document

For each POST method the following information is provided:

- The node it operates on
- A table of form parameters, some of which may be optional. Mandatory parameters are marked with an asterisk (*)
The format of each parameter (e.g. ID or string) or the possible values (e.g. true|false). If appropriate the default value of a parameter (the value used if you do not specify a parameter) is shown in bold e.g. true|false.

4.3 PUT Methods

PUT methods modify existing objects; for example, changing the name of a coSpace, muting a specific call leg or changing the layout.

In general, when using PUT in an object:

- omit a parameter to leave its value unchanged
- use a parameter with a new value to change to this value. Supply an empty value to unset a value. For example, to remove a tenant association from a coSpace, modify that coSpace with a parameter set including “tenant=”.

4.3.1 HTTP specifics

Parameters for a request must be supplied in “x-www-form-urlencoded” format.

4.3.2 How PUT methods are detailed in this document

Each PUT method is in the same section as the POST method for the same object e.g. creating and modifying a coSpace are dealt with together. Form parameters for modifying an object (PUT) are only noted if they differ from the POST method; for example, for callLegs.

4.4 DELETE Methods

A DELETE method removes an individual object from the hierarchy; for example, disconnecting a call leg or disassociating a user from a coSpace so that the user is no longer a member.

Therefore the DELETE method is typically performed at the Individual level e.g. DELETE on /api/v1/coSpace/<id>/accessMethods/<id>

The object’s ID is known either from a previous retrieval (GET) method at the Collections level or from the “Location” field in the response to a previous creation (PUT) method. (coSpace can be deleted at the Collections level.)

If the object is removed successfully, the Meeting Server sends a “200 OK” response.

Because of the relative simplicity of this method, it is not detailed elsewhere in this document – with the exception of deleting chat messages.

4.5 Failure Reasons

The following "failureDetails" codes can be returned by the API for any of the above methods, in the form in response to a user error:
<failureDetails>
<tenantDoesNotExist />
</failureDetails>

<table>
<thead>
<tr>
<th>Reason code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessMethodDoesNotExist</td>
<td>You tried to modify or remove an accessMethod using an ID that did not correspond to a valid access method.</td>
</tr>
<tr>
<td>callBrandingProfileDoesNotExist</td>
<td>You tried to modify or remove a call branding profile using an ID that did not correspond to a valid call branding profile.</td>
</tr>
<tr>
<td>callBridgeDoesNotExist</td>
<td>You tried to modify or remove a configured clustered Call Bridge using an ID that did not correspond to a valid clustered Call Bridge.</td>
</tr>
<tr>
<td>callBridgeGroupDoesNotExist</td>
<td>You tried to modify, remove or use a Call Bridge group using an ID that did not correspond to a valid Call Bridge group (from version 2.1).</td>
</tr>
<tr>
<td>callBridgeGroupUnavailable</td>
<td>You tried to create a participant on a Call Bridge Group that is unavailable or could not accept the call (from version 2.2).</td>
</tr>
<tr>
<td>callBridgeUnavailable</td>
<td>You tried to create a participant on a Call Bridge that is unavailable or could not accept the call (from version 2.2).</td>
</tr>
<tr>
<td>callDoesNotExist</td>
<td>You tried to perform a method on a call object using an ID that did not correspond to a currently active call.</td>
</tr>
<tr>
<td>callRecordingCannotBeModified</td>
<td>You tried to start/stop recording a call that cannot be modified.</td>
</tr>
<tr>
<td>callStreamingCannotBeModified</td>
<td>You tried to start/stop streaming a call that cannot be modified (from version 2.1).</td>
</tr>
<tr>
<td>callLegCannotBeDeleted</td>
<td>You tried to delete a call leg that can’t be deleted.</td>
</tr>
<tr>
<td>callLegDoesNotExist</td>
<td>You tried to perform a method on a call leg object using an ID that did not correspond to a currently active call leg.</td>
</tr>
<tr>
<td>callLegProfileDoesNotExist</td>
<td>You tried to modify or remove a callLegProfile using an ID that did not correspond to a valid call leg profile.</td>
</tr>
<tr>
<td>callProfileDoesNotExist</td>
<td>You tried to modify or remove a callProfile using an ID that is not valid.</td>
</tr>
<tr>
<td>cdrReceiverDoesNotExist</td>
<td>You tried to modify or remove a CDR receiver using an ID that did not correspond to a valid CDR receiver.</td>
</tr>
<tr>
<td>coSpaceDoesNotExist</td>
<td>You tried to modify or remove a coSpace using an ID that did not correspond to a valid coSpace on the system.</td>
</tr>
<tr>
<td>coSpaceUserDoesNotExist</td>
<td>You tried to modify or remove a coSpace user using an ID that did not correspond to a valid coSpace user.</td>
</tr>
<tr>
<td>databaseNotReady</td>
<td>You tried a method (e.g. initiation of an LDAP sync method) before the database was ready.</td>
</tr>
<tr>
<td>directorySearchLocationDoesNotExist</td>
<td>You tried to reference, modify or remove a directory search location using an ID that did not correspond to a valid directory search location.</td>
</tr>
<tr>
<td>Reason code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>dtmfProfileDoesNotExist</td>
<td>You tried to reference, modify or remove a DTMF profile using an ID that did not correspond to a valid DTMF profile.</td>
</tr>
<tr>
<td>duplicateCallBridgeName</td>
<td>You tried to create or modify a clustered Call Bridge to use a name that would clash with an existing configured clustered Call Bridge.</td>
</tr>
<tr>
<td>duplicateCoSpaceId</td>
<td>You tried to create or modify a coSpace call ID to use a call ID that clashed with one used by another coSpace.</td>
</tr>
<tr>
<td>duplicateCoSpaceUri</td>
<td>You tried to create or modify a coSpace to use a URI that clashed with one that corresponds to another coSpace. (Two coSpaces can’t share the same URI, because the Meeting Server must be able to uniquely resolve an incoming call to a coSpace URI).</td>
</tr>
<tr>
<td>duplicateCoSpaceIdPasscode</td>
<td>You tried to modify a coSpace, or create or modify a coSpace access method, using a call ID/passcode combination that clashed with another call ID/passcode that is already used by that coSpace or one of its access methods.</td>
</tr>
<tr>
<td>duplicateCoSpaceUriPasscode</td>
<td>You tried to modify a coSpace, or create or modify a coSpace access method, using a URI/passcode combination that clashed with URI/passcode combination that is already used by that coSpace or one of its access methods.</td>
</tr>
<tr>
<td>duplicateCoSpaceSecret</td>
<td>You tried to modify a coSpace, or create or modify a coSpace access method, using a secret that clashed with one that is already used by that coSpace or one of its access methods.</td>
</tr>
<tr>
<td>forwardingDialPlanRuleDoesNotExist</td>
<td>You tried to modify or remove an forwarding dial plan rule using an ID that did not correspond to a valid forwarding dial plan rule.</td>
</tr>
<tr>
<td>inboundDialPlanRuleDoesNotExist</td>
<td>You tried to modify or remove an inbound dial plan rule using an ID that did not correspond to a valid inbound dial plan rule.</td>
</tr>
<tr>
<td>inboundDialPlanRuleUriConflict</td>
<td>You tried to make modifications to an inbound dial plan rule which would have caused a URI conflict. For example, this can happen if you try to add a rule which matches multiple tenants and more than one tenant has a coSpace with the same URI.</td>
</tr>
<tr>
<td>invalidOperation</td>
<td>You tried an operation which isn’t supported; for example, you attempted to POST to /api/v1/system/profiles or issue a DELETE for a configured user generated from an LDAP sync.</td>
</tr>
<tr>
<td>invalidVersion</td>
<td>You attempted an operation with an invalid API version.</td>
</tr>
<tr>
<td>ivrBrandingProfileDoesNotExist</td>
<td>You tried to modify or remove an IVR branding profile object using an ID that did not correspond to a valid IVR branding profile on the system.</td>
</tr>
<tr>
<td>ivrDoesNotExist</td>
<td>You tried to modify or remove an IVR object using an ID that did not correspond to a valid IVR on the system.</td>
</tr>
<tr>
<td>Reason code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ivrUriConflict</td>
<td>You tried to make modifications to an IVR object which would have caused a URI conflict.</td>
</tr>
<tr>
<td>ldapMappingDoesNotExist</td>
<td>You tried to modify or remove an LDAP mapping using an ID that did not correspond to a valid LDAP mapping.</td>
</tr>
<tr>
<td>ldapServerDoesNotExist</td>
<td>You tried to modify or remove an LDAP server using an ID that did not correspond to a valid LDAP server.</td>
</tr>
<tr>
<td>ldapSourceDoesNotExist</td>
<td>You tried to modify or remove an LDAP source using an ID that did not correspond to a valid LDAP source.</td>
</tr>
<tr>
<td>ldapSyncCannotBeCancelled</td>
<td>You tried to cancel an LDAP synchronization that has either started or completed – only LDAP synchronization methods that have not started yet can be canceled.</td>
</tr>
<tr>
<td>ldapSyncDoesNotExist</td>
<td>You tried to query or cancel an LDAP synchronization with an ID that did not correspond to a valid LDAP synchronization.</td>
</tr>
<tr>
<td>loadBalancingDisabled</td>
<td>You tried to create a participant in a Call Bridge Group with load balancing for outgoing calls disabled (from version 2.2).</td>
</tr>
<tr>
<td>messageDoesNotExist</td>
<td>You tried to remove a coSpace message using an ID that did not correspond to a valid coSpace message.</td>
</tr>
<tr>
<td>outboundDialPlanRuleDoesNotExist</td>
<td>You tried to modify or remove an outbound dial plan rule using an ID that did not correspond to a valid outbound dial plan rule</td>
</tr>
<tr>
<td>parameterError</td>
<td>One or more parameters in a request were found to be invalid. Supporting parameter and error values give more detail about the failure.</td>
</tr>
<tr>
<td>participantCannotBeDeleted</td>
<td>You tried to add a new participant beyond the maximum number allowed for the call.</td>
</tr>
<tr>
<td>participantCannotBeModified</td>
<td>You tried to modify or remove a participant using an ID that did not correspond to a valid recorder.</td>
</tr>
<tr>
<td>participantLimitReached</td>
<td>You tried to start recording a call beyond the maximum number allowed.</td>
</tr>
<tr>
<td>recorderDoesNotExist</td>
<td>You tried to modify or remove a recorder using an ID that did not correspond to a valid recorder.</td>
</tr>
<tr>
<td>recordingLimitReached</td>
<td>You tried to start streaming a call beyond the maximum number allowed.</td>
</tr>
<tr>
<td>streamerDoesNotExist</td>
<td>You tried to modify or remove a streamer using an ID that did not correspond to a valid streamer (from version 2.1).</td>
</tr>
<tr>
<td>streamingLimitReached</td>
<td>You tried to start streaming a call beyond the maximum number allowed (from version 2.1).</td>
</tr>
<tr>
<td>tenantDoesNotExist</td>
<td>You tried to modify or remove a tenant using an ID that did not correspond to a valid tenant</td>
</tr>
<tr>
<td>Reason code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tenantGroupCoSpaceIdConflict</td>
<td>Your request to remove or use a tenant group would have resulted in a coSpace ID conflict.</td>
</tr>
<tr>
<td>tenantGroupDoesNotExist</td>
<td>You tried to modify, remove or use a tenant group that does not exist.</td>
</tr>
<tr>
<td>tenantParticipantLimitReached</td>
<td>You tried to add a new participant beyond the maximum number allowed for the owning tenant.</td>
</tr>
<tr>
<td>tooManyCdrReceivers</td>
<td>You tried to add a new CDR receiver when the maximum number were already present. Currently, up to 2 CDR receivers are supported.</td>
</tr>
<tr>
<td>tooManyLdapSyncs</td>
<td>A method to create a new LDAP synchronization method failed. Try again later.</td>
</tr>
<tr>
<td>recognisedObject</td>
<td>There are elements in the URI you are accessing that are not recognized; for example, you tried to perform a GET on /api/v1/system/profile rather than (the correct) /api/v1/system/profiles</td>
</tr>
<tr>
<td>userDoesNotExist</td>
<td>You tried to modify or remove a user using an ID that did not correspond to a valid user.</td>
</tr>
<tr>
<td>userProfileDoesNotExist</td>
<td>You tried to modify a user profile using an ID that did not correspond to a valid user profile.</td>
</tr>
</tbody>
</table>
5 Example Requests and Responses for Specific Methods

5.1 Retrieval of Current Active Calls

As described in Section 4.1, retrieval methods using GET involve no body content posted by the retriever. If the request is valid, the Meeting Server returns XML response data.

Request:
GET /api/v1/calls HTTP/1.1\nHost: test.example.com\nUser-Agent: API console\nConnection: keep-alive\nAuthorization: Basic Ym9iOmJ1aWxkZXI=\n\n
Response:
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 187
Connection: close
\n<?xml version="1.0"?>
<calls total="1">
<call id="527089d6-6581-4331-8417-971c05c9e274">
<name>Sales coSpace</name>
<coSpace>2dcf2b7a-3410-4066-b638-46273698d469</coSpace>
</call>
</calls>

5.2 Creating a New Call Leg (SIP dial out to 10.1.144.129)

As described above, any parameters needed for the creation method (in this case, the address of the remote party), need to be supplied by the issuer as form data. If the request is successful, details about the new object are returned by the Meeting Server in the “Location” header field.

Request:
POST /api/v1/calls/527089d6-6581-4331-8417-971c05c9e274/callLegs HTTP/1.1\nHost: test.example.com\nUser-Agent: API console\nConnection: keep-alive\nAuthorization: Basic Ym9iOmJ1aWxkZXI=\nContent-Type: application/x-www-form-urlencoded\nContent-Length: 24\n\n
remoteParty=10.1.144.129

Response:
HTTP/1.1 200 OK
Location: /api/v1/callLegs/5a3b907a-7641-42fb-ae8c-b3424a3e923f
Connection: close
6 coSpace Related Methods

Note: Although the Cisco Meeting App and other Cisco Meeting Server guides refer to "spaces" rather than "coSpaces", the API still uses /coSpace objects. The Web Admin interface has been changed to refer to "spaces".

This chapter details the API methods related to management of coSpaces. The chapter covers:

- retrieving coSpaces
- creating and modifying a coSpace
- retrieving detailed information about a single coSpace
- retrieving the members of a coSpace
- adding and modifying a coSpace member
- posting to the message board of a coSpace
- deleting messages from a coSpace message board
- retrieving coSpace access methods
- creating and modifying coSpace access methods
- calling out from a coSpace
- bulk creating, updating and deleting coSpaces
- coSpace diagnostics

6.1 Retrieving coSpaces

GET method on the "/coSpaces" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve coSpaces other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply &quot;filter=&lt;string&gt;&quot; in the URI to return just those coSpaces that match the filter</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter=&lt;tenant id&gt; to return just those coSpaces associated with that tenant</td>
</tr>
<tr>
<td>callLegProfileFilter</td>
<td>ID</td>
<td>Supply callLegProfileFilter=&lt;call leg profile id&gt; to return just those coSpaces using that call leg profile</td>
</tr>
</tbody>
</table>
The response includes the total count of the number of coSpaces present which match the filter if provided, irrespective of the number returned within the response. (With no filter, this value is the total number of configured coSpaces.)

The human-readable name that will be shown on clients’ UI for this coSpace

The URI that a SIP system would use to dial in to this coSpace

The secondary URI for this coSpace - this provide the same functionality as the “uri” parameter, but allows more than one URI to be configured for a coSpace

The numeric ID that a user would enter at the IVR (or via a web client) to connect to this coSpace

If provided, associates the specified call leg profile with this tenant

Whether this coSpace has been added automatically or manually

- true - this coSpace has been added automatically as part of an LDAP sync operation. therefore it is not possible, to remove it except by modifying the parameters of the sync operation
- false - this coSpace has been added either via an API method or by using Cisco Meeting App; it can be modified or removed via the API

6.2 Creating and Modifying a coSpace

Creating: POST method to the “/coSpaces” node. If the coSpace was created successfully, a “200 OK” response is received, and the “Location” header contains the ID for the new coSpace

Modifying: PUT method on a “/coSpaces/<coSpace ID>” node
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>String (URI user part)</td>
<td>The URI that a SIP system would use to dial in to this coSpace. (The URI &quot; user part&quot; is the part before any '@' character in a full URI.)</td>
</tr>
<tr>
<td>secondaryUri</td>
<td>String (URI user part)</td>
<td>The secondary URI for this coSpace – this provide the same functionality as the “uri” parameter, but allows more than one URI to be configured for a coSpace. (The URI &quot; user part&quot; is the part before any '@' character in a full URI.)</td>
</tr>
<tr>
<td>callId</td>
<td>String</td>
<td>The numeric ID that a user would enter at the IVR (or via a web client) to connect to this coSpace</td>
</tr>
<tr>
<td>cdrTag</td>
<td>String</td>
<td>Up to 100 characters of free form text to identify this coSpace in a CDR; when a &quot;callStart&quot; CDR is generated for a call associated with this coSpace, this tag will be written (as &quot;cdrTag&quot;) to the callStart CDR. See the Cisco Meeting Server CDR Reference for details. The cdrTag can be modified in a PUT method.</td>
</tr>
<tr>
<td>passcode</td>
<td>String</td>
<td>The security code for this coSpace</td>
</tr>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If provided, associates the specified tenant with this coSpace</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If provided, associates the specified call leg profile with this coSpace</td>
</tr>
<tr>
<td>callProfile</td>
<td>ID</td>
<td>If provided, associates the specified call profile with this coSpace</td>
</tr>
<tr>
<td>callBrandingProfile</td>
<td>ID</td>
<td>If provided, associates the specified call branding profile with this coSpace</td>
</tr>
<tr>
<td>requireCallId</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>secret</td>
<td>String</td>
<td>If provided, sets the security string for this coSpace. If absent, a security string is chosen automatically if the coSpace has a callId value. This is the security value associated with the coSpace that needs to be supplied with the callId for guest access to the coSpace.</td>
</tr>
</tbody>
</table>
| regenerateSecret | true|false | • If provided as true - a new security value is generated for this coSpace and the former value is no longer valid (for instance, any hyperlinks including it will cease to work)  
  • If provided as false - do not generate a new secret value for this coSpace; this has no effect  
  This parameter is only valid for the modify (PUT) case. |
| nonMemberAccess  | true|false | Controls whether non-members of the coSpace are able to have access to the coSpace. If not provided, behaviour defaults to true. (From version 2.0). |
| ownerJid         | String     | Indicates the coSpace is owned by the user with the specified JID. (From version 2.0).                                                                 |
| streamUrl        | URL        | Indicates where the coSpace is streamed to, if streaming is initiated. (From version 2.1).                                                                 |
| ownerAdGuid      | ID         | If provided, the coSpace will be owned by the user with the given AD GUID. (From version 2.1).                                                                 |
| meetingScheduler | String     | Name of person (not necessarily a user) who scheduled the creation of this coSpace, which if set is propagated to any call objects as the “ownerName” field. (From version 2.2). |

**Note:** You can also use this PUT to modify the values of a coSpace created in a Cisco Meeting App. For example, the coSpace will have been created with the cdrTag of the user who created it but you can change that value with an API call. (This is unlike the cdrTag of an automatically generated coSpace, which cannot be updated with an API call.)

**Default layout options**

The naming of the defaultLayout options varies between the API and the Web Admin Interface Configuration > coSpaces page. The “mapping” is shown in the table below.

<table>
<thead>
<tr>
<th>API</th>
<th>Web Admin Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>allEqual</td>
<td>all equal</td>
</tr>
<tr>
<td>speakerOnly</td>
<td>full screen</td>
</tr>
<tr>
<td>telepresence</td>
<td>overlay (the loudest speaker is in a large pane and a number of the previous speakers are in small panes which overlay the bottom of the loudest speaker’s pane.)</td>
</tr>
</tbody>
</table>
6.2.1 Secondary coSpace URIs

Per coSpace, there is an optional secondaryUri parameter as shown above. This allows flexibility; for example, numeric dialing in addition to a name.

- When creating or modifying a coSpace (see the previous section) you can supply a secondaryUri parameter in addition to the form parameters in the table above e.g. uri
- The secondary URI will be checked for validity and uniqueness in the same way as the uri, and if valid, establishes a new URI by which the coSpace can be reached
- When retrieving information on an individual coSpace (see below) the secondaryUri value will be returned, if it is defined for this coSpace
- The secondaryUri can be created automatically during an LDAP sync if the new LDAP mapping parameter is used. See coSpaceSecondaryUriMapping

6.2.2 Auto-generation of coSpace callId

A new auto-generated Call Id is assigned if "requireCallId=true" is set via a create (POST) or modify (PUT) method on the coSpace, and no callId is currently specified for the coSpace.

6.3 Retrieving Detailed Information about a Single coSpace

GET method performed on a "/coSpaces/<coSpace ID>" node. If the coSpace ID supplied is valid, a "200 OK" response is received, containing a single "<coSpace id=<ID>>" object with data as described above for the creating and modifying case.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The human-readable name that will be shown on clients’ UI for this coSpace</td>
</tr>
<tr>
<td>uri</td>
<td>String (URI user part)</td>
<td>The URI that a SIP system would use to dial in to this coSpace. (The URI &quot;user part&quot; is the part before any '@' character in a full URI.)</td>
</tr>
<tr>
<td>secondaryUri</td>
<td>String (URI user part)</td>
<td>The secondary URI for this coSpace – this provide the same functionality as the &quot;uri&quot; parameter, but allows more than one URI to be configured for a coSpace. (The URI &quot;user part&quot; is the part before any '@' character in a full URI.)</td>
</tr>
<tr>
<td>callId</td>
<td>String</td>
<td>The numeric ID that a user would enter at the IVR (or via a web client) to connect to this coSpace</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cdrTag</td>
<td>String</td>
<td>Up to 100 characters of free form text to identify this coSpace in a CDR; when a &quot;callStart&quot; CDR is generated for a call associated with this coSpace, this tag will be written (as &quot;cdrTag&quot;) to the callStart CDR. See the Cisco Meeting Server CDR Reference for details. The cdrTag can be modified in a PUT method.</td>
</tr>
<tr>
<td>passcode</td>
<td>String</td>
<td>The security code for this coSpace</td>
</tr>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If provided, associates the specified tenant with this coSpace</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If provided, associates the specified call leg profile with this coSpace</td>
</tr>
<tr>
<td>callProfile</td>
<td>ID</td>
<td>If provided, associates the specified call profile with this coSpace</td>
</tr>
<tr>
<td>callBrandingProfile</td>
<td>ID</td>
<td>If provided, associates the specified call branding profile with this coSpace</td>
</tr>
<tr>
<td>autoGenerated</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>nonMemberAccess</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>numAccessMethods</td>
<td>Number</td>
<td>If additional access methods have been defined for this coSpace, this returns a count of the number of additional access methods for the coSpace.</td>
</tr>
<tr>
<td>secret</td>
<td>String</td>
<td>If provided, sets the security string for this coSpace. If absent, a security string is chosen automatically if the coSpace has a callId value. This is the security value associated with the coSpace that needs to be supplied with the callId for guest access to the coSpace.</td>
</tr>
<tr>
<td>ownerId</td>
<td>ID</td>
<td>Indicates the coSpace is owned by the user with the specified GUID. (From version 2.0).</td>
</tr>
<tr>
<td>ownerJid</td>
<td>String</td>
<td>Indicates the coSpace is owned by the user with the specified JID. (From version 2.0).</td>
</tr>
<tr>
<td>streamUrl</td>
<td>URL</td>
<td>Indicates where the coSpace is streamed to, if streaming is initiated. (From version 2.1).</td>
</tr>
<tr>
<td>meetingScheduler</td>
<td>String</td>
<td>Name of person (not necessarily a user) who scheduled the creation of this coSpace, which if set is propagated to any call objects as the &quot;ownerName&quot; field. (From version 2.2).</td>
</tr>
</tbody>
</table>

### 6.3.1 Retrieving entry details for a specific coSpace

From 2.1, a "meetingEntryDetail" node is added to allow retrieval of entry details for a specific coSpace meeting. Perform a GET on /coSpaces/<coSpace id>/meeting EntryDetail.

Response values are uri and callId.

### 6.4 coSpace Member Methods

#### 6.4.1 Retrieving the members of a coSpace

GET method on a “/coSpaces/<coSpace ID>/coSpaceUsers” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those coSpace users that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve coSpaces other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>callLegProfileFilter</td>
<td>ID</td>
<td>Supply callLegProfileFilter=&lt;ID&gt; to return just members using that call leg profile</td>
</tr>
</tbody>
</table>
The response includes the total count of coSpace users configured for the queried coSpace which match the filter, irrespective of the number returned within the response. (With no filter, this value is the total number of users associated with the coSpace.)

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpaceUser id</td>
<td>ID</td>
<td>&lt;coSpaceUser&gt; elements have their own ID and also contain an ID for the user.</td>
</tr>
<tr>
<td>userJid</td>
<td>String</td>
<td>The XMPP ID of the user.</td>
</tr>
<tr>
<td>userId</td>
<td>ID</td>
<td>Identifies the user with no relationship to any coSpace association, and may or may not be the same as the ID of the &quot;coSpaceUser&quot; object.</td>
</tr>
<tr>
<td>autoGenerated</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- true - this coSpaceUser has been added automatically as part of an LDAP sync operation. Therefore, it is not possible to remove it except by modifying the parameters of the sync operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- false - this coSpaceUser has been added either via an API method or by using Cisco Meeting App; it can be modified or removed via the API</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If provided, associates the specified call leg profile with this coSpace user</td>
</tr>
<tr>
<td>canDestroy</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canAddRemoveMember</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeName</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeUri</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeCallId</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangePasscode</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canPostMessage</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canRemoveSelf</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canDeleteAllMessages</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
6.4.2 Adding and modifying a coSpace member

- Adding: POST method to a “/coSpaces/<coSpace ID>/coSpaceUsers” node
- Modifying: PUT method performed on a “/coSpaces/<coSpace ID>/coSpaceUsers/<coSpaceUser ID>” node. The parameters that you can modify are listed below, with the exception of “userJid”

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>userJid *</td>
<td>String</td>
<td>JID of the user to be added as a member</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If provided, associates the specified call leg profile with this coSpace user</td>
</tr>
<tr>
<td>canDestroy</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canAddRemoveMember</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeName</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeUri</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangeCallId</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canChangePasscode</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canPostMessage</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canRemoveSelf</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>canDeleteAllMessages</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

If the member was added successfully, a “200 OK” response is received, and the “Location” header in the response contains the new user ID.

**coSpace Permissions**

Members with canAddRemoveMember set to true can add other users as members of the coSpace from a Cisco Meeting App (depending on Meeting App type and version). New members have identical permissions to the member who added them, except in one case: when the original member also has canRemoveSelf set to false.

Members who cannot remove themselves from the coSpace (as controlled by canRemoveSelf) should not be able to create a second member in order to delete their own membership. Therefore any member created from Cisco Meeting App by another member in this situation will have canAddRemoveMember set to false and canRemoveSelf set to true (see the table below). All other permissions are copied from the original member.
Using the API provides more flexibility: it is possible to create coSpaces with members who cannot remove themselves, but who can be removed by another member. Members can always be removed via the API.

Auto-generated members (created by an LDAP sync) have auto-generated permissions because it makes no sense to allow them to make changes that will be overwritten by the next LDAP sync. Therefore, for these users the following parameters are always set to false: canDestroy, canChangeName, canChangeUri, canChangeCallId and canRemoveSelf. The other “can” parameters are set to True. Note that changing any of these settings for an auto-generated member via the API will only have a temporary effect and will be overwritten at the next LDAP sync: you can discover whether a member is auto-generated – see the next section.

Finally, if a user creates a coSpace from the Cisco Meeting App, then canDeleteAllMessages is set to false for all members, and all other permissions are set to true by default for all members.

For a summary of default settings for the permissions, see the table below:

<table>
<thead>
<tr>
<th>Permission</th>
<th>coSpace created by:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cisco Meeting App</td>
</tr>
<tr>
<td>canDestroy</td>
<td>true</td>
</tr>
<tr>
<td>canAddRemoveMember</td>
<td>true</td>
</tr>
<tr>
<td>canChangeName</td>
<td>true</td>
</tr>
<tr>
<td>canChangeUri</td>
<td>true</td>
</tr>
<tr>
<td>canChangeCallId</td>
<td>true</td>
</tr>
<tr>
<td>canChangePasscode</td>
<td>true</td>
</tr>
<tr>
<td>canPostMessage</td>
<td>true</td>
</tr>
<tr>
<td>canRemoveSelf</td>
<td>true</td>
</tr>
<tr>
<td>canDeleteAllMessages</td>
<td>false</td>
</tr>
</tbody>
</table>
6.4.3 Retrieving Information on a coSpace member

GET method performed on a “/coSpaces/<coSpace ID>/coSpaceUsers/<coSpaceUser ID>” node. If the retrieval is valid, a “200 OK” response is received, with containing a single <coSpaceUser id=<ID>> object with data as described above for the creating and modifying case. In addition

- the autoGenerated value shows whether the coSpace member was added to the coSpace automatically as part of an LDAP sync operation
- the canDeleteAllMessages shows whether this member is allowed to delete chat in the coSpace

6.5 coSpace Chat/Message Board Methods

6.5.1 Posting to the message board of a coSpace

POST method performed on the “/coSpaces/<coSpace id>/messages” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>message *</td>
<td>String</td>
<td>The message string to be posted to the message board</td>
</tr>
<tr>
<td>from</td>
<td>String</td>
<td>A “from” name to be shown to message board viewers as the originator of the message</td>
</tr>
</tbody>
</table>

If successful, the message is posted to the message board and an ID for the message is returned in the “Location” field of the response header.

6.5.2 Deleting messages from a coSpace message board

DELETE method on a ” /coSpaces/<coSpace ID>/messages” node with the form parameters below which select messages by age. With no additional parameters, all messages for that coSpace will be deleted.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>minAge</td>
<td>Number</td>
<td>If supplied (in the URL) deletes only messages whose age is at least the specified value (in seconds)</td>
</tr>
<tr>
<td>maxAge</td>
<td>Number</td>
<td>If supplied specifies (in the URL) an upper limit (in seconds) on the age of messages to be deleted</td>
</tr>
</tbody>
</table>

6.6 Multiple coSpace Access Methods

6.6.1 General information

There are two related tables of objects:
- Access method per coSpace, "/coSpaces/<cospace ID>/accessMethods[/<accessMethod ID>]"
- Call leg profile, "/callLegProfiles/<callLegProfile ID>".

**Access method per coSpace**

Access methods define combinations of URI, passcode, callId and secret that can be used to access a coSpace.

Optionally, Access methods can have an associated callLegProfile; any call leg joining via such an Access method has that call leg profile applied to it. If the Access method has no call leg profile but the coSpace does, then so does coSpace’s call.

**Note:** When you send an email invitation from a Cisco Meeting App to one or more people to join a coSpace or active call, only one set of URI, passcode, callId, secret information is included. If the scope field for an access method is set to public then this information is used. If no access methods have a public scope then the call information from the coSpace’s own configuration is included.

**Call leg profile**

A call leg profile can be associated with a coSpace object, making it the default call leg profile for all call legs in that coSpace (for instance, those that connect via its configured URI and **secondaryUri**). (The effect of the coSpace call leg profile can still be overridden by more specific overrides imposed via call leg profiles configured for additional coSpace access methods. See the **Call Leg Profile** section.

### 6.6.2 Retrieving coSpace access methods

GET method on the " /coSpaces/<coSpace id>/accessMethods/" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those coSpace access methods that match the filter</td>
</tr>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve coSpaces other than the first &quot;page&quot; in the notional list (see <strong>Section 4.1.2</strong>).</td>
</tr>
<tr>
<td>callLegProfileFilter</td>
<td>ID</td>
<td>Supply callLegProfileFilter=&lt;ID&gt; to return just accessMethods for coSpaces using that call leg profile</td>
</tr>
</tbody>
</table>
### Response elements

<table>
<thead>
<tr>
<th>Description/Notes</th>
<th>Type/Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response is a collection of &quot;&lt;accessMethod id=&lt;access method id&gt;&gt;&quot; objects contained within an &quot;&lt;accessMethods&gt;&quot; object. &lt;accessMethod&gt; elements follow the general form on the left.</td>
<td>accessMethod id (ID)</td>
</tr>
<tr>
<td>The URI to be used for dialing in via this access method.</td>
<td>uri (String)</td>
</tr>
<tr>
<td>The &quot;call ID&quot; to be used for connecting via this access method (using the IVR or Web Bridge login).</td>
<td>callId (ID)</td>
</tr>
<tr>
<td>A passcode required for this access method.</td>
<td>passcode (String)</td>
</tr>
<tr>
<td>The ID of a call leg profile to apply to calls in via this access method.</td>
<td>callLegProfile (ID)</td>
</tr>
</tbody>
</table>

### 6.6.3 Creating and modifying coSpace access methods

- **Creating**: POST method to the " /coSpaces/<coSpace id>/accessMethods" node
- **Modifying**: PUT method on a " /coSpaces/<coSpace id>/accessMethods/<access method id>" node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>String (URI user part)</td>
<td>The URI to be used for dialing in via this access method. (The URI &quot;user part&quot; is the part before any '@' character in a full URI.)</td>
</tr>
<tr>
<td>callId</td>
<td>ID</td>
<td>The &quot;call ID&quot; to be used for connecting via this access method (using the IVR or Web Bridge login)</td>
</tr>
<tr>
<td>passcode</td>
<td>String</td>
<td>A passcode required for this access method.</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>The ID of a call leg profile to apply to calls in via this access method</td>
</tr>
<tr>
<td>secret</td>
<td>String</td>
<td>If provided, sets the security string for this coSpace access method. If absent, a security string is chosen automatically if the coSpace access method has a callId value. This is the security value associated with the coSpace access method that needs to be supplied with the callId for guest access to the coSpace via this access method.</td>
</tr>
</tbody>
</table>
| regenerateSecret | true|false | **If provided as true** - a new security value is generated for this coSpace access method and the former value is no longer valid (for instance, any hyperlinks including it will cease to work)  
**If provided as false** - do not generate a new secret value for this coSpace access method; this has no effect  
This parameter is only valid for the modify (PUT) case |
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
</table>
| scope      | public|private | The visibility of this coSpace access method to users of Cisco Meeting App who are members of the coSpace  
- If provided as public - details of this coSpace access method can be made available to members of the coSpace  
- If provided as private - details of this coSpace access method will not be made available to members of the coSpace  
Note: if you set the scope to public then the Cisco Meeting App can no longer edit the coSpace details. In addition, the uri shown under the name is that from the access method. |

If the coSpace access method is created successfully, a “200 OK” response will be received, and the “Location” header in the response will contain the new coSpace access method ID.

**6.6.4 Retrieving information on an individual coSpace access method**

GET method on a " /coSpaces/<coSpace id>/accessMethods/<access method id>" node.  
If the access method ID supplied is valid, a " 200 OK" response and a single <accessMethod id=<access method id> object will be returned with data in the previous section.

**6.7 Calling Out from a coSpace**

Adding a remote party to a coSpace requires that this coSpace has an active call from which connections can be made. Essentially this makes an initial call out from a coSpace a combination of two other API methods:  
1. Creation of a new call.  
2. Adding a new outgoing call leg to a call.  

These methods are described in the Section 8.

**6.8 Bulk creating, updating and deleting coSpaces**

**6.8.1 Creating /coSpaceBulkParameterSets**

- Creating: POST method to the " /coSpaceBulkParameterSets" node. Creates a new parameter set, see table below. Returns location of new parameter set /cospaceBulkParameterSets/<bulk parameter set guid>  
- Modifying: PUT method to the " /coSpaceBulkParameterSets" node. Updates the parameters within this parameter set, but needs to be synchronized for it to take effect.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>startIndex</td>
<td>Number</td>
<td>Index that coSpace mappings start from (inclusive)</td>
</tr>
<tr>
<td>endIndex</td>
<td>Number</td>
<td>Index that coSpace mappings end at (inclusive)</td>
</tr>
</tbody>
</table>
| coSpaceUriMapping  | String  | If specified, this is the mapping that describes what URLs will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set, coSpace will not have a dialable URI. Syntax: uri-mapping = [uri-component] ["$index$"] [uri-component] Where: uri-component = *( uri-character / escaped-character ) unescaped-character = any character EXCLUDING ‘@’ and ‘\’ escaped-character = \\
|                    |         | producing ‘\’ and ‘$’ respectively. These need to be unique so if an index is not used there will be clashes, unless the field is just left completely blank. |
| coSpaceNameMapping | String  | If specified, this is the mapping that describes what names will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. Syntax: name-mapping = [name-component] ["$index$"] [name-component] Where: name-component = *( unescaped-character / escaped-character ) unescaped-character = any character EXCLUDING ‘$’ and ‘\’ escaped-character = \\
|                    |         | producing ‘\’ and ‘$’ respectively. These are not required to be unique.                                                                          |
| coSpaceCallIdMapping | String | If specified, this is the mapping that describes what call IDs will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not have a callId. Syntax: id-mapping = [id-component] ["$index$"] [id-component] Where: id-component = *( unescaped-character / escaped-character ) unescaped-character = any character EXCLUDING ‘$’ and ‘\’ escaped-character = \\
<p>|                    |         | producing ‘\’ and ‘$’ respectively. These need to be unique so if index is not used there will be clashes, unless the field is just left completely blank. Secrets will be auto-generated if CallIdMapping is set. |
| tenant             | ID      | If specified this is the tenant to be associated with the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not be associated with a tenant. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callProfile</td>
<td>ID</td>
<td>If specified this is the call profile to be associated with the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not be associated with a call profile.</td>
</tr>
<tr>
<td>callBrandingProfile</td>
<td>ID</td>
<td>If specified this is the call branding profile to be associated with the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not be associated with a call branding profile.</td>
</tr>
<tr>
<td>nonMemberAccess</td>
<td>true</td>
<td>Whether non-members will be able to access the bulk created coSpaces. If this parameter is not supplied in a create (POST) operation, it defaults to &quot;true&quot; and non members can access the coSpace.</td>
</tr>
</tbody>
</table>

### 6.8.2 Retrieving the parameter sets for creating coSpaces in bulk

GET method on "/coSpaceBulkParameterSets" node.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>startIndex</td>
<td>Number</td>
<td>Index that coSpace mappings start from (inclusive)</td>
</tr>
<tr>
<td>endIndex</td>
<td>Number</td>
<td>Index that coSpace mappings end at (inclusive)</td>
</tr>
</tbody>
</table>

### 6.8.3 Retrieving information on an individual /coSpaceBulkParameterSet

GET method on "/coSpaceBulkParameterSets/<coSpace bulk parameter set id>" node

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td></td>
<td>Response is structured as a top-level &lt;coSpaceBulkParameterSets total=&quot;N&quot;&gt; tag with potentially multiple &lt;coSpaceBulkParameterSet&gt; elements within it. Each &lt;coSpaceBulkParameterSet&gt; element may include the following elements.</td>
</tr>
<tr>
<td>startIndex</td>
<td>Number</td>
<td>Index that coSpace mappings start from (inclusive)</td>
</tr>
<tr>
<td>endIndex</td>
<td>Number</td>
<td>Index that coSpace mappings end at (inclusive)</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| coSpaceUriMapping          | String  | If specified, this is the mapping that describes what URIs will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set, coSpace will not have a dialable URI.  
  Syntax:  
  uri-mapping = [uri-component] ["$index$"] [uri-component]  
  Where:  
  uri-component = *( uri-character / escaped-character )  
  uri-character = *( unescaped-character EXCLUDING ‘@’)  
  unescaped-character = any character EXCLUDING ‘$’ and ‘\’  
  escaped-character = “\” / “$” ; producing ‘\’ and ‘$’ respectively.  
  These need to be unique so if an index is not used there will be clashes, unless the field is just left completely blank.  |
| coSpaceNameMapping         | String  | If specified, this is the mapping that describes what names will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet.  
  Syntax:  
  name-mapping = [name-component] ["$index$"] [name-component]  
  Where:  
  name-component = *( unescaped-character / escaped-character )  
  unescaped-character = any character EXCLUDING ‘$’ and ‘\’  
  escaped-character = “\” / “$” ; producing ‘\’ and ‘$’ respectively.  
  These are not required to be unique.  |
| coSpaceCallIdMapping       | String  | If specified, this is the mapping that describes what call IDs will be used for the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not have a callId  
  Syntax:  
  id-mapping = [id-component] ["$index$"] [id-component]  
  Where:  
  id-component = *( unescaped-character / escaped-character )  
  unescaped-character = any character EXCLUDING ‘$’ and ‘\’  
  escaped-character = “\” / “$” ; producing ‘\’ and ‘$’ respectively  
  These need to be unique so if index is not used there will be clashes, unless the field is just left completely blank.  
  Secrets will be auto-generated if CallIdMapping is set.  |
| tenant                     | ID      | If specified this is the tenant to be associated with the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not be associated with a tenant.  |
| callProfile                | ID      | If specified this is the call profile to be associated with the coSpaces created with a /coSpaceBulkSync using this coSpaceBulkParameterSet. If not set then the coSpace will not be associated with a call profile.  |
### 6.8.4 Queueing the bulk sync operations

- **Creating:** POST method to the " /coSpaceBulkSyncs" node. Queues the bulk sync operations for execution as soon as possible. Returns location /cospaceBulkSync/<bulk sync guid>

**Note:** Bulk Sync will iterate between startIndex and endIndex (inclusive at both end) and expand and insert the mapping parts.

- **Modifying:** PUT method to the " /coSpaceBulkSyncs" node not supported.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type/Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpaceBulkParameterSet</td>
<td>ID</td>
<td>Parameter set GUID that is going to be synchronised</td>
</tr>
<tr>
<td>removeAll</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

### 6.8.5 Retrieving the bulk sync operations

GET method on " /coSpaceBulkSyncs" node.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Response is structured as a top-level &lt;coSpaceBulkSyncs total=&quot;N&quot;&gt; tag with potentially multiple &lt;coSpaceBulkSync&gt; elements within it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;coSpaceBulkSync&gt; elements follow the general form on the left.</td>
</tr>
</tbody>
</table>
### Response elements

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpaceBulkParameterSet</td>
<td>ID</td>
<td>Parameter set that was used for this bulk sync</td>
</tr>
</tbody>
</table>
| status                        | pending| running| complete| failedCoSpaceUriConflict| failedCallIdConflict| failedIndexRangeInvalid| failedIndexRangeTooGreat| failedNoSuchParameterSet| failed | removeAll | true| false | If supplied, determines whether the sync will remove all entries that were created using the parameter set. Used only to remove all spaces that were created previously. If set to true then no spaces will be created. If set to false, or omitted, then all spaces previously created using this parameter set will be removed and new spaces based on the new mappings will be created. If this parameter is not supplied in a create (POST) operation, it defaults to "false"

#### Status of the sync operation:

- **pending** - the sync operation is in a queue waiting to execute
- **running** - the sync operation is currently running
- **complete** - the sync operation has successfully completed
- **failedCoSpaceUriConflict** - the sync failed because it would involve creating a URI that conflicts with one that already exists
- **failedCallIdConflict** - the sync failed because it would involve creating a call ID that conflicts with one that already exists
- **failedIndexRangeInvalid** - the sync failed because the "startIndex" was greater than the "endIndex"
- **failedIndexRangeTooGreat** - the sync failed because the difference between "endIndex" and "startIndex" was too large
- **failedNoSuchParameterSet** - the "coSpaceBulkParameterSet" refered to in the sync command did not exist
- **failed** - the sync operation failed

---

### 6.8.6 Retrieving a specific bulk sync operation

GET method on "/coSpaceBulkSyncs/<coSpace bulk sync id>" node.
<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpaceBulkSyncs</td>
<td>total=&quot;N&quot;</td>
<td>Response is structured as a top-level <code>&lt;coSpaceBulkSyncs total=&quot;N&quot;&gt;</code> tag with potentially multiple <code>&lt;coSpaceBulkSync&gt;</code> elements within it. <code>&lt;coSpaceBulkSync&gt;</code> elements follow the general form on the left.</td>
</tr>
<tr>
<td>coSpaceBulkParameterSet</td>
<td>ID</td>
<td>Parameter set that was used for this bulk sync</td>
</tr>
<tr>
<td>status</td>
<td>pending</td>
<td>running</td>
</tr>
<tr>
<td>removeAll</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
6.8.7 Examples

Creating coSpaces in bulk

1. Create a cospaceBulkParameterSet with parameters:
   - startIndex=1000
   - endIndex=1999
   - coSpaceUriMapping=space.$index$
   - coSpaceNameMapping=Space $index$
   - coSpaceCallIdMapping=811$index$

2. Create a cospaceBulkSync with parameters:
   - cospaceBulkParameterSet=<GUID from above>

This will create 1000 spaces starting with

“Space 1000” space.1000@domain.com, callID=8111000

and ending in

“Space 1999” space.1999@domain.com, callID=8111999

To update the range:

1. PUT new range to cospaceBulkParameterSets/<GUID from above>

2. Create a cospaceBulkSync with parameters:
   - cospaceBulkParameterSet=<GUID from above>

This deletes all the previous spaces and creates a new set. This whole operation will succeed or fail. In failure the transaction will be rolled back and the spaces that previously existed will still be there.

To delete a range:

1. Create a cospaceBulkSync with parameters:
   - cospaceBulkParameterSet=<GUID from above>&removeAll=true

This removes all spaces that were created using this parameter set. They will get removed even if they have been renamed, or edited in any other way.

6.9 coSpace Diagnostics Methods

A POST to "/coSpaces/<coSpace id>/diagnostics" triggers the generation of call diagnostics for the specified coSpace.
7 Dial Plan Methods

This chapter details the API methods related to configuring dial plans for outbound calls, inbound calls and call forwarding. The chapter covers:

- retrieving outbound dial plan rules
- creating and modifying outbound dial plan rules
- retrieving information on an individual outbound dial plan rule
- retrieving dial plan rules for incoming calls
- creating and modifying dial plan rules for incoming calls
- retrieving information on the dial plan rule for an individual incoming call
- retrieving dial plan rules for forwarding incoming calls
- creating and modifying dial plan rules for forwarding incoming calls
- retrieving information on the dial plan rule to forward an individual incoming call

7.1 Outgoing Dial Plan API Methods

7.1.1 Access to the outgoing dial plan

Typically, the configuration of which trunks / proxies to use for outbound calls is based on the domain of the (SIP) destination being called, which is specified in the outgoing dial plan. The outgoing dial plan sits in the API object tree under the " /outboundDialPlanRules" node, use the POST method to create the outgoing dial plan or set it up via the Web Admin Interface (see note below).

If you are deploying Call Bridge clustering, use the API parameter scope to choose whether to apply each outbound dial plan rule to every Call Bridge in the cluster, or just to a particular Call Bridge so the Call Bridge can be trunked to its local Call Control solution (if appropriate).

**Note:** The API parameter callRouting specifies the mechanism for traversal of outgoing SIP/Lync calls, use this parameter to set up firewall traversal for SIP and Lync devices. This is still a beta feature.

**Note:** On the Web Admin Interface, the table of outbound rules is configured through the Configuration > Dial plan page. All rules added via the Web Admin Interface are global and applied to every Call Bridge in the cluster. You cannot use the Web Admin interface to specify the call routing for outbound SIP/Lync calls using a specific Call Bridge or Call Bridge group.
7.1.2 Retrieving outbound dial plan rules

GET method on the "/outboundDialPlanRules/" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those outbound dial plan rules that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve coSpaces other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>If supplied, this filter only returns those outbound dial plan rules associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>outboundDialPlanRule id</td>
<td>ID</td>
<td>Response is a collection of &quot;&lt;outboundDialPlanRule id=&lt;ID&gt;&gt;&quot; objects contained within an &quot;&lt;outboundDialPlanRules&gt;&quot; object &lt;outboundDialPlanRule&gt; elements follow the general form on the left.</td>
</tr>
<tr>
<td>domain</td>
<td>String</td>
<td>The domain to match in order to apply the dial plan rule; either a complete value (e.g. &quot;example.com&quot;) or a &quot;wildcarded&quot; one (e.g. &quot;*.com&quot;)</td>
</tr>
<tr>
<td>priority</td>
<td>Number</td>
<td>A numeric value which determines the order in which dial plan rules (including rules with wild-carded domains) will be applied. Rules with higher priority values are applied first</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If a tenant is specified, this rule will only be used to make outbound call legs from calls associated with that tenant; otherwise, this rule may be used from any call.</td>
</tr>
</tbody>
</table>

7.1.3 Creating and modifying outbound dial plan rules

- Creating: POST method to the "/outboundDialPlanRules" node. If the outgoing dial plan rule is created successfully, a “200 OK” response will be received, and the “Location” header in the response will contain the new outgoing dial plan rule ID.
- Modifying: PUT method on an "/outboundDialPlanRules/<outbound dial plan rule ID>" node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>String</td>
<td>The domain to match in order to apply the dial plan rule; either a complete value (e.g. &quot;example.com&quot;) or a &quot;wildcarded&quot; one (e.g. &quot;*.com&quot;)</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>priority</td>
<td>Number</td>
<td>A numeric value which determines the order in which dial plan rules (including rules with wildcarded domains) will be applied. Rules with higher priority values are applied first. If a rule is matched, but the call cannot be made, then other lower priority rules may be tried depending on the failureAction parameter for the rule.</td>
</tr>
<tr>
<td>localContactDomain</td>
<td>String</td>
<td>Used when forming an explicit contact domain to be used: if you leave this field blank then the localContactDomain is derived from the local IP address. If you are using Lync, we suggest that you set localContactDomain. If you are not using Lync, we recommend that localContactDomain is not set to avoid unexpected issues with the SIP call flow.</td>
</tr>
<tr>
<td>localFromDomain</td>
<td>String</td>
<td>Used when forming the calling party for outgoing calls using this dial plan rule.</td>
</tr>
<tr>
<td>sipProxy</td>
<td>String</td>
<td>The address (IP address or hostname) of the proxy device through which to make the call. If not set, it is a direct call.</td>
</tr>
<tr>
<td>trunkType</td>
<td>sip</td>
<td>lync</td>
</tr>
<tr>
<td>failureAction</td>
<td>stop</td>
<td>continue</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sipControlEncryption</td>
<td>auto, encrypted, unencrypted</td>
<td>Whether to enforce use of encrypted control traffic on calls made via this rule:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• encrypted: allow only encrypted SIP control traffic (TLS connections)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• unencrypted: use only unencrypted traffic (TCP or UDP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• auto: attempt to use encrypted control connections first, but allow fall back to unencrypted control traffic in the event of failure.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: Ensure all &quot;Lync&quot; outbound dialing rules are explicitly set to <strong>Encrypted</strong> mode to prevent the Call Bridge attempting to use unencrypted TCP for these connections in the event of the TLS connection attempt failing.</td>
</tr>
<tr>
<td>scope</td>
<td>global, callBridge, callBridgeGroup</td>
<td>The entities for which this outbound dial plan rule is valid:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• global – all Call Bridges are able to use this outbound dial plan rule to reach a matching domain.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• callBridge – this outbound dial plan rule is only valid for a single nominated Call Bridge – whose ID is given in callBridge parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• callBridgeGroup – this outbound dial plan rule is only valid for a single nominated Call Bridge Group – whose ID is given in the callBridgeGroup parameter. (From version 2.2).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this parameter is not supplied in a create (POST) operation it defaults to “global”.</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>If the rule has a scope of callBridge (see above), this is the id of the Call Bridge for which the rule is valid.</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>If the rule has a scope of callBridgeGroup (see above), this is the id of the Call Bridge Group for which the rule is valid (from version 2.2).</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If a tenant is specified, this rule will only be used to make outbound call legs from calls associated with that tenant; otherwise, this rule may be used from any call.</td>
</tr>
<tr>
<td>callRouting</td>
<td>default, traversal</td>
<td>This is the media routing that should be used for SIP calls originating from this rule:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• default - calls using this rule will use normal, direct, media routing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• traversal - media for calls using this rule will flow via a TURN server</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If this parameter is not supplied in a create (POST) operation, it defaults to “default”.</td>
</tr>
</tbody>
</table>
7.1.4 Retrieving information on an individual outbound dial plan rule

GET method on a " /outboundDialPlanRules/<outbound dial plan rule ID>" node. If the outbound dial plan rule ID supplied is valid, a " 200 OK" response and a single " <outboundDialPlanRule id=<ID>>" object will be returned with data as per the previous section.

7.2 Incoming Call Matching Dial Plan API Methods

7.2.1 Access to incoming domain matching rules

When an incoming SIP call is routed to the Meeting Server, the Call Bridge looks through the configured inbound dial plan rules first and tries to match the " domain" part of the destination URI " <user>@<domain>" against the rules. Use the POST method on API object /inboundDialPlanRules to create a new inbound dial plan rule to match against incoming SIP calls, or set it up via the Web Admin Interface (see note below).

**Note:** On the Web Admin Interface, the table of inbound rules is configured through the Configuration > Incoming calls page.

7.2.2 Retrieving incoming dial plan rules

GET method on the " /inboundDialPlanRules" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those incoming dial plan rules that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve incoming dial plan rules other than the first &quot; page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those inbound dial plan rules associated with the specified tenant</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>inboundDialPlanRule id</td>
<td>ID</td>
<td>Response is a collection of &quot; &lt;inboundDialPlanRule id=&lt;ID&gt;&gt;&quot; objects contained within an &quot; &lt;inboundDialPlanRules&gt;&quot; object &lt;inboundDialPlanRule&gt; elements follow the general form on the left.</td>
</tr>
</tbody>
</table>
## 7.2.3 Creating and modifying incoming dial plan rules

- **Creating:** POST method to the `/inboundDialPlanRules` node. If the incoming dial plan rule is created successfully, a “200 OK” response will be received, and the “Location” header in the response will contain the new incoming dial plan rule ID.

- **Modifying:** PUT method on an `/inboundDialPlanRules/<inbound dial plan rule ID>` node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain *</td>
<td>String</td>
<td>The domain to match in order to apply the dial plan rule. Must be a complete value (e.g. &quot;example.com&quot;)</td>
</tr>
</tbody>
</table>
### Parameters | Type/Value | Description/Notes
--- | --- | ---
priority | numeric | Inbound dial plan rules' configured domain values are always exactly matched against incoming calls. For the purposes of generating full URIs to advertise for incoming calls (especially cases where multiple rules are applicable) you can also set a numeric priority value - higher values will be preferred.

resolveToUsers | true/false | If set to true, calls to this domain will be matched against user JIDs (if a match is then found, that incoming call leg causes a "point to point" call to that user's Meeting App.

resolveTocoSpaces | true/false | If set to true, calls to this domain will be matched against coSpace URIs (if a match is then found, the incoming call leg becomes a participant in the coSpace).

resolveToIvrs | true/false | If set to true, calls to this domain will be matched against configured IVR URIs (if a match is then found, the incoming call leg connects to that IVR).

resolveToLyncConferences | true/false | If set to true, calls to this domain will be resolved to a Lync conference URL; if the resolution is successful, the incoming call leg becomes a participant in the Lync conference. If this parameter is not supplied in a create (POST) operation, it defaults to "false".

resolveToLyncSimplejoin | true/false | If set to true, calls to this domain will be resolved by an HTTPS lookup to the given URL. If the resolution is successful, the incoming call leg becomes a participant in the Lync conference. If this parameter is not supplied in a create (POST) operation, it defaults to "false". (From version 2.2).

tenant | ID | If specified, calls to this inbound domain will only be matched against user JIDs and coSpace URIs for the specified tenant.

#### 7.2.4 Retrieving information on an individual incoming dial plan rule

GET method on a "/inboundDialPlanRules/<inbound dial plan rule ID>" node. If the incoming dial plan rule ID supplied is valid, a "200 OK" response and a single "<inboundDialPlanRule id=<ID>>" object will be returned with data as per the previous section.

### 7.3 Incoming Call Forwarding Dial Plan API Methods

#### 7.3.1 Access to incoming call forwarding rules

If the "domain" part of the destination URI of an incoming SIP call fails to match any of the inbound dial plan rules, the call will be handled according to the rules in the call forwarding dial plan rules. The rules decide whether to reject the call outright or to forward the call in bridge mode.
Call forwarding rules can overlap, and include wildcards. You order rules using the Priority value; higher numbered rules are tried first. By defining rules, you decide whether to forward the call or not. It might be appropriate to “catch” certain calls and reject them.

For calls that will be forwarded, you can rewrite the destination domain, a new call is created to the specified domain.

The call forwarding dial plan sits in the API object tree under a "/forwardingDialPlanRules" node. Use the POST method to create the forwarding rules or set them up via the Web Admin Interface (see note below)

**Note:** On the Web Admin Interface, the Incoming Call Forwarding rules are configured through the Call Forwarding section of the **Configuration > Incoming calls** page.

### 7.3.2 Retrieving incoming call forwarding dial plan rules

GET method on the " /forwardingDialPlanRules/" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those incoming call forwarding rules that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve coSpaces other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>If supplied, this filter restricts the results returned to those forwarding dial plan rules that are associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>forwardingDialPlanRule id</td>
<td>ID</td>
<td>The matchPattern and priority are described in the next section</td>
</tr>
<tr>
<td>matchPattern</td>
<td>Text</td>
<td>The domain to match in order to apply the dial plan rule. Must be a complete domain name (e.g. &quot;example.com&quot;) or a “wildcarded” one (e.g. &quot;.com&quot;)</td>
</tr>
<tr>
<td>priority</td>
<td>Number</td>
<td>Numeric value used when determining the order in which to apply forwarding dial plan rules; higher values will be applied first</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>The tenant associated with the forwardingDialPlanRule</td>
</tr>
</tbody>
</table>
7.3.3 Creating and modifying incoming call forwarding dial plan rules

- Creating: POST method to the " /forwardingDialPlanRules" node. If the forwarding dial plan rule is created successfully, a “200 OK” response will be received, and the “Location” header in the response will contain the new forwarding dial plan rule ID.
- Modifying a forwarding dial plan rule is a PUT method on a " /forwardingDialPlanRules/<forwarding dial plan rule ID>" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>matchPattern</td>
<td>String</td>
<td>The domain to match in order to apply the dial plan rule. Must be a complete domain name (e.g. &quot;example.com&quot;) or a &quot;wildcarded&quot; one (e.g. exa*.com). Wildcards are permitted in any part of a domain matching pattern, but do not use &quot;matchPattern=*&quot; as a match all, otherwise you will create call loops.</td>
</tr>
</tbody>
</table>
| destinationDomain | String                | Calls that are forwarded with this rule will have their destination domain rewritten to be this value.                                                 |}

| action          | forward[|reject | If set to “forward” causes matching call legs to become point-to-point calls with a new destination. “reject” causes the incoming call leg to be rejected. |
|-----------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| callerIdMode    | regenerate| preserve | When forwarding an incoming call to a new destination address, whether to preserve the original calling party's ID or to generate a new one. If this parameter is not supplied in a create (POST) operation, it defaults to "regenerate". |
| priority        | Number   | Numeric value used when determining the order in which to apply forwarding dial plan rules; higher values will be applied first.                 |
| tenant          | ID       | If a tenant is specified, calls using this rule will be associated with the specified tenant.                                                    |
| uriParameters   | discard| forward | When forwarding an incoming call to a new destination address, this parameter determines whether to discard any additional parameters that are present in the destination URI of the incoming call, or to forward them on to the destination URI of the outbound call. If this parameter is not supplied in a create (POST) operation, it defaults to "discard". This parameter is present from version 2.0 onwards. |

7.3.4 Retrieving information on an individual incoming call forwarding dial plan rule

GET method on a " /forwardingDialPlanRules/<forwarding dial plan rule ID>" node. If the forwarding dial plan rule ID supplied is valid, a "200 OK" response and a single "<forwardingDialPlanRule id=<ID>>" object will be returned with data as per the previous section.
8 Call Related Methods

This chapter details the API methods for:

- active calls
- call profiles
- call legs
- call leg profiles
- dial transforms
- call branding profiles
- dtmf profiles
- ivr methods
- ivr branding profiles
- participants

8.1 Call Methods

8.1.1 Retrieving Information on Active Calls

GET method performed on the " /calls" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>coSpaceFilter</td>
<td>ID</td>
<td>Supply an ID to return just those calls that match the filter</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply an ID to return just those calls that belong to the specified tenant</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>callCorrelator</td>
<td>ID</td>
<td>An id that is the same across all distributed instances of this call.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The associated (human-readable) name for the call</td>
</tr>
</tbody>
</table>
### 8.1.2 Creating a New Call and Modifying an Active call

POST method performed on the “/calls” node or PUT method to “/calls/<call id>”

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpace *</td>
<td>ID</td>
<td>Specifies the coSpace for which the call is being instantiated</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>The name of the new call</td>
</tr>
<tr>
<td>locked</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>recording</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>streaming</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllMuteSelf</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllPresentationContribution</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>joinAudioMuteOverride</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>messageText</td>
<td>String</td>
<td>Text to display to every participant in the call (only displayed if configured messageDuration is non zero). (From version 2.1)</td>
</tr>
</tbody>
</table>
### 8.1.3 Retrieving Information on an Individual Active Call

GET method performed on a “/calls/<call ID>” node. If the call ID supplied is valid, a “200 OK” response is received, with XML content of the form:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpace</td>
<td>ID</td>
<td>If the call represents the instantiation of a coSpace, this value will be present and hold the id of the coSpace</td>
</tr>
<tr>
<td>callCorrelator</td>
<td>ID</td>
<td>An id that is the same across all distributed instances of this call.</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>The ID of the tenant that owns the call</td>
</tr>
<tr>
<td>durationSeconds</td>
<td>Number</td>
<td>The duration of the call, as the number of seconds since the call started</td>
</tr>
<tr>
<td>numCallLegs</td>
<td>Number</td>
<td>The number of call legs currently active within this call</td>
</tr>
<tr>
<td>maxCallLegs</td>
<td>Number</td>
<td>The highest number of call legs that have been simultaneously present within this call</td>
</tr>
<tr>
<td>numParticipantsLocal</td>
<td>Number</td>
<td>The number of participants within this call locally hosted by the Call Bridge to which the request is being made</td>
</tr>
<tr>
<td>numParticipantsRemote</td>
<td>Number</td>
<td>The number of participants in this call hosted by other Call Bridges</td>
</tr>
<tr>
<td>numDistributedInstances</td>
<td>Number</td>
<td>The number of other Call Bridges hosting participants in this call</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>presenterCallLeg</td>
<td>ID</td>
<td>The presenterCallLeg value is only present if a call leg is actively presenting within this call.</td>
</tr>
<tr>
<td>locked</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>recording</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>streaming</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllMuteSelf</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllPresentationContribution</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>joinAudioMuteOverride</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>messageText</td>
<td>String</td>
<td>Text to display to every participant in the call (only displayed if configured messageDuration is non zero). (From version 2.1)</td>
</tr>
<tr>
<td>messagePosition</td>
<td>top</td>
<td>middle</td>
</tr>
<tr>
<td>messageDuration</td>
<td>Number</td>
<td>Time in seconds to display configured messageText on screen (can also be &quot;permanent&quot; to cause the message to be displayed until it is reconfigured). (From version 2.1)</td>
</tr>
<tr>
<td>messageTimeRemaining</td>
<td>Number</td>
<td>Time remaining in seconds for configured messageText to be displayed on screen. (From version 2.1)</td>
</tr>
<tr>
<td>ownerName</td>
<td>String</td>
<td>If set, displays the owner of this call. This can be the meetingScheduler of the coSpace of this call, or the name of the owner of this call or the Jid of the owner. (From version 2.2)</td>
</tr>
<tr>
<td>activeWhenEmpty</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

### 8.1.4 Generating diagnostics for an individual call

POST method performed on “/calls/<call id>diagnostics” generates call diagnostics for the call in question.
8.1.5 Retrieve participants in a conference

GET method performed on the " /calls/<call id>/participants" node. Retrieves a list of all of the participants associated with the specified call.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve active participants other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>coSpaceFilter</td>
<td>ID</td>
<td>Supply an ID to return just those active participants that match the filter.</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply an ID to return just those active participants that belong to the specified tenant.</td>
</tr>
<tr>
<td>callBridgeFilter</td>
<td>ID</td>
<td>Supply an ID to return just those active participants located on the specified Call Bridge.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The associated (human-readable) name associated with this participant</td>
</tr>
<tr>
<td>call</td>
<td>ID</td>
<td>The call that this participant is part of</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>The specific tenant with which this participant is associated</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>The remote, clustered Call Bridge to which this participant is connected</td>
</tr>
</tbody>
</table>

8.1.6 Creating a new participant for a specified call

POST method on the " /calls/<call id>/participants" node.

**Note:** Due to load balancing across clustered Meeting Servers, an explicit selection of a Call Bridge or Call Bridge Group or from configured dial plan rules, may result in the call leg instantiation (" owned" by the participant object) occurring on a remote clustered Call Bridge.

**Note:** See also the section on participant related methods Section 8.11.
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>remoteParty *</td>
<td>String</td>
<td>For POST only, specifies the participant’s address; this could be a SIP URI, a phone number, or a user JID to invite that user to the call.</td>
</tr>
<tr>
<td>bandwidth</td>
<td>Number</td>
<td>For POST only, if supplied, sets the bandwidth for the participant, in bits per second (e.g. 2000000 for 2Mbit/s). If not supplied, the Call Bridge configured value will be used.</td>
</tr>
</tbody>
</table>
| confirmation     | true|false | For POST only, if supplied, this overrides the automatic choice of whether to require a confirmation from the remote party to join the call.  
   true – always require a confirmation from the remote party; typically this takes the form of a voice prompt requiring them to hit a key to join  
   false – never require a confirmation from the remote party; the remote party will be joined into the coSpace when they accept the incoming call |
<p>| ownerId          | ID         | If supplied must be an ID for the Meeting Server to associate with this participant. This will be returned by the Meeting Server when the participant’s call leg is later queried and therefore should be a value that has meaning to the requestor. |
| callLegProfile   | ID         | If provided, associates the specified call leg profile with this participant’s call leg. |
| needsActivation  |            | Supply any of these parameters to override the call leg profile values for this call leg. |</p>
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>participantLabels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>endCallAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>telepresenceCallsAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipPresentationChannelEnabled</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>bfcpMode</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>layout</td>
<td>unrestricted</td>
<td></td>
</tr>
<tr>
<td>disconnectOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>qualityMain</td>
<td>unrestricted</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max1080p30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max720p30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max480p30</td>
<td></td>
</tr>
</tbody>
</table>

Restricts the maximum negotiated main video call quality for this call leg based on limiting transcoding resources. Specified using a typical resolution and framerate. Note that call legs may operate at lower resolutions or framersates due to endpoint limitations or overall Call Bridge load.

unrestricted - this is the default setting if not specified, and matches the behavior of older Call Bridge versions, where no restrictions are placed on resolution or framerate.

max1080p30 - restricts the bridge to negotiating at most 1920x1080 screen size at 30 frames per second or equivalent transcoding resources, for example 1280x720 screen size at 60 frames per second.

max720p30 - restricts the bridge to negotiating at most 1280x720 screen size at 30 frames per second or equivalent transcoding resources.

max480p30 - restricts the bridge to negotiating at most 868x480 screen size at 30 frames per second or equivalent transcoding resources.

(From version 2.2)
## Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>qualityPresentation</td>
<td>unrestricted, max1080p30, max720p5</td>
<td>Restrict the maximum negotiated presentation video call quality for this call leg based on limiting transcoding resources. Specified using a typical resolution and frame rate. This only affects legs which use a separate presentation stream. Unrestricted - this is the default setting if not specified, and matches the behavior of older Call Bridge versions, where no restrictions are placed on resolution or framerate. Max1080p30 - restricts the Call Bridge to negotiating at most 1920x1080 screen size at 30 frames per second or equivalent transcoding resources. Max720p5 - restricts the Call Bridge to negotiating at most 1280x720 screen size at 5 frames per second or equivalent transcoding resources. (From version 2.2)</td>
</tr>
<tr>
<td>participantCounter</td>
<td>never, auto, always</td>
<td>Controls the behavior of the on-screen participant counter. Never - never show an on-screen participant count value. Auto - show the on-screen participant count value when appropriate; typically this will be to indicate that there are additional participants present that you cannot currently see. Always - always show the on-screen participant count value. (From version 2.2)</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>If supplied, attempt to add the participant from the specified Call Bridge (from version 2.2).</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>If supplied, attempt to add the participant from the specified Call Bridge Group (from version 2.2).</td>
</tr>
<tr>
<td>importance</td>
<td>Number</td>
<td>The importance value of the participant to be created. To remove importance leave the importance parameter as unset (leave value as blank). (From version 2.2)</td>
</tr>
</tbody>
</table>
8.1.7 Set properties for all participants in a conference

PUT to "/calls/<call id>/participants/*" node. Set properties for all participants associated with the specified call.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>rxAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>rxVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>layout</td>
<td>allEqual</td>
<td>stacked</td>
</tr>
<tr>
<td>importance</td>
<td>Number</td>
<td>Set importance of all participants. To remove importance leave the importance parameter as unset (leave value as blank). (From version 2.2)</td>
</tr>
</tbody>
</table>

8.2 Call Profile Methods

Call profiles control the maximum number of active simultaneous participants and the in-call experience for SIP (including Lync) calls. For more information see also Section 14.

8.2.1 Retrieving call profiles

GET method performed on the "/callProfiles" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>usageFilter</td>
<td>unreferenced</td>
<td>referenced</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Response is structured as a top-level <code>&lt;callProfiles total=&quot;N&quot;&gt;</code> tag with potentially multiple <code>&lt;callProfile&gt;</code> elements within it. Each <code>&lt;callProfile&gt;</code> tag may include the following elements: See the next section</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 8.2.2 Setting up and modifying call profiles

- Creating: POST method to the " /callProfiles" node
- Modifying: PUT to " /callProfiles/<call profile id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantLimit</td>
<td>Number</td>
<td>Sets the maximum number of participants for calls (coSpace instantiations or ad hoc calls) using this call profile that can be active simultaneously; new participants beyond this limit are not permitted</td>
</tr>
<tr>
<td>messageBoardEnabled</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>locked</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>recordingMode</td>
<td>disabled</td>
<td>manual</td>
</tr>
</tbody>
</table>
### Parameters | Type/Value | Description/Notes
--- | --- | ---
streamingMode | disabled| manual| automatic | Controls how this coSpace or ad hoc call can be streamed
| | disabled - call is not streamed
| | manual - users can start/stop streaming
| | automatic - call is automatically streamed and users cannot start/stop streaming
| | If this parameter is not supplied in a create (POST) operation, it defaults to "manual". (From version 2.1)

| passcodeMode | required| timeout | Determines the behavior for passcode entry when a mixture of blank and set passcodes can be used to access a coSpace via the same URI/call Id.
| | required - requires passcode to be entered, with blank passcode needing to be explicitly entered
| | timeout - after an amount of time has elapsed with no passcode being entered, interpret this as a blank passcode. Amount of timeout is determined by value of "passcodeTimeout"

| passcodeTimeout | numeric | If specified, this is the amount of time, in seconds, that the Call Bridge will wait before before interpreting passcode as a blank passcode (if "passcodeMode" is set to "timeout"). Timeout time is measured from the end of the passcode prompt.

### 8.2.3 Retrieving detailed information about an individual call profile

GET method performed on a "/callProfiles/<call profile id>" node. If the call profile id ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

### 8.3 Call Leg Methods

#### 8.3.1 Retrieving Information on Active Call Legs

GET method performed on the “/callLegs” node (to retrieve information on all active call legs within the system).

Alternatively, a GET method performed on the “/calls/<call id>/callLegs” node (to retrieve information on active call legs for a specific call).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those call legs that match the filter</td>
</tr>
<tr>
<td>participantFilter</td>
<td>ID</td>
<td>Supply participantFilter to return only those call legs associated with the specified participant.</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those call legs associated with the specified tenant.</td>
</tr>
<tr>
<td>activeLayoutFilter</td>
<td>String</td>
<td>If supplied, this filter will restrict results returned to those call legs using the specified layout.</td>
</tr>
<tr>
<td>availableVideoStreamsLowerBound</td>
<td>Number</td>
<td>If supplied, this filter will restrict results returned to those call legs with this many or more available video streams.</td>
</tr>
<tr>
<td>availableVideoStreamsUpperBound</td>
<td>Number</td>
<td>If supplied, this filter will restrict results returned to those call legs with many or fewer available video streams.</td>
</tr>
<tr>
<td>ownerIdSet</td>
<td>true/false</td>
<td>Used to return those call legs that have an owner Id set, or those that do not</td>
</tr>
</tbody>
</table>
| alarms                 | Text       | Used to return just those call legs for which the specified alarm names are currently active. Either "all", which covers all supported alarm conditions, or one or more specific alarm conditions to filter on, separated by the '|' character. The supported alarm names are:  
  * packetLoss – packet loss is currently affecting this call leg  
  * excessiveJitter – there is currently a high level of jitter on one or more of this call leg’s active media streams  
  * highRoundTripTime – the Meeting Server measures the round trip time between itself and the call leg destination; if a media stream is detected to have a high round trip time (which might impact call quality), then this alarm condition is set for the call leg |

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
</table>
| callLeg id name        | ID         | Response is structured as a top-level <callLegs total="N"> tag with potentially multiple <callLeg> elements within it.  
  <callLeg> elements follow the general form on the left. |
8. Call Related Methods

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>remoteParty</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>call ID</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>tenant ID</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>&lt;alarms&gt;</td>
<td>String</td>
<td>For call legs which have active alarm conditions, there will be an additional &quot;&lt;alarms&gt;&quot; tag under the encompassing &quot;&lt;callLeg&gt;&quot; which details the currently active alarms. Within this &quot;&lt;alarms&gt;&quot; tag there will be one or more subsidiary indications—also see the note below:</td>
</tr>
<tr>
<td>packetLoss</td>
<td>String</td>
<td>Present if packet loss is being experienced on one or more of the call leg’s active media streams</td>
</tr>
<tr>
<td>excessiveJitter</td>
<td>String</td>
<td>Present if there is a high level of jitter on one or more of the call leg’s active media streams</td>
</tr>
<tr>
<td>highRoundTripTime</td>
<td>String</td>
<td>Present if a high round trip time has been detected for one or more of the call leg’s media streams</td>
</tr>
</tbody>
</table>

**Note on alarms:**
Call leg alarms provide information that may be useful in raising alarms or troubleshooting issues after they have occurred but they should not necessarily be treated as if they are alarm conditions in themselves—unlike system level alarms.

A call leg alarm may be triggered by a number of factors, not necessarily a set percentage packet loss for example. An alarm condition is attached to a call leg when the Meeting Server believes that the call leg may be being degraded. These "conditions" may include a simple threshold, but potentially other things too such as a more adaptive threshold and taking other factors into account. This does not necessarily mean that the user’s experience was poor but it provides information to troubleshoot in the event that it was. Therefore you could consider adding filters to this alarm information and deciding when to flag an event as an alarm to the operator (i.e. setting your own thresholds) and/or storing call leg alarm information alongside CDRs so that if a user reports a poor quality call you can retrieve this information after the event to determine what the cause might have been.

### 8.3.2 Adding and Modifying Call Legs
- **Adding:** POST method to a “/calls/<call ID>/callLegs” node. The <call ID> is learnt from a GET on “/calls” or from a newly created call (see Creating a new call above). If a profile has been applied to this call leg, it starts with the values set in the profile. Note: these added or modified call legs will not be load balanced across clustered Meeting Servers.
- **Modifying:** PUT method performed on a “/callLegs/<callLeg ID>” node. It makes live, dynamic, changes to an in-progress connection to a remote party.
Note: You cannot modify the remoteParty, bandwidth or confirmation.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>remoteParty *</td>
<td>String</td>
<td>For POST only, specifies the call leg's address; this could be a SIP URI, a phone number, or a user JID to invite that user to the call.</td>
</tr>
<tr>
<td>bandwidth</td>
<td>Number</td>
<td>For POST only, if supplied, sets the bandwidth for the call leg, in bits per second (e.g. 2000000 for 2Mbit/s). If not supplied, the Call Bridge configured value will be used.</td>
</tr>
</tbody>
</table>
| confirmation   | true|false | For POST only, if supplied, this overrides the automatic choice of whether to require a confirmation from the remote party to join the call  
true - always require a confirmation from the remote party; typically this takes the form of a voice prompt requiring them to hit a key to join  
false - never require a confirmation from the remote party; the remote party will be joined into the coSpace when they accept the incoming call |
<p>| ownerId        | ID         | If supplied must be an ID for the Meeting Server to associate with this call leg. This will be returned by the Meeting Server when the call leg is later queried and therefore should be a value that has meaning to the requestor. |
| chosenLayout   | allEqual|speakerOnly|telepresence|stacked|allEqualQuarters|allEqualNinths|allEqualSixteenths|allEqualTwentyFifths|onePlusFive|onePlusSeven|onePlusNine|automatic|onePlusN |
|                |            | This parameter overrides the prevailing default layout for this call leg.            |</p>
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>dtmfSequence</td>
<td>string</td>
<td>A sequence of DTMF key press commands to send to the far end either when the call leg initially connects, or during the call. In the supplied sequence, you can use the digits 0 to 9, *, and #, as well as one or more comma characters (<code>,</code>) which add a pause between digits.</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If provided, associates the specified call leg profile with this call leg. You can also supply individual values for all parameters that can be part of a call leg profile to override this call leg profiles’ value. See below</td>
</tr>
<tr>
<td>needsActivation</td>
<td></td>
<td>Supply any of these parameters to override the call leg profile values for this call leg.</td>
</tr>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>participantLabels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>endCallAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>telepresenceCallsAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipPresentationChannelEnabled</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>bfcpMode</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>layout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>disconnectOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>qualityMain</td>
<td>unrestricted</td>
<td>max1080p30</td>
</tr>
<tr>
<td>qualityPresentation</td>
<td>unrestricted</td>
<td>max1080p30</td>
</tr>
</tbody>
</table>
8.3.3 Retrieving Information on an Individual Call Leg

GET method performed on a “/callLegs/<callLeg ID>” node.

If the call leg ID supplied is valid, a “200 OK” response is received, with XML content:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>callLeg id</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>remoteParty</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>localAddress</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>call</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>sip</td>
<td>acano</td>
</tr>
<tr>
<td>subtype</td>
<td>lync</td>
<td>avaya</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>lyncSubType</td>
<td>audioVideo</td>
<td>applicationSharing</td>
</tr>
<tr>
<td>direction</td>
<td>incoming</td>
<td>outgoing</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ownerId</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>chosenLayout</td>
<td>one of:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>speakerOnly</td>
<td></td>
</tr>
<tr>
<td></td>
<td>telepresence</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stacked</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqual</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualQuarters</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualNinths</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualSixteenths</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualTwentyFifths</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusFive</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusSeven</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusNine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualSixteenths</td>
<td></td>
</tr>
<tr>
<td></td>
<td>allEqualTwentyFifths</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusFive</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusSeven</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusNine</td>
<td></td>
</tr>
<tr>
<td></td>
<td>automatic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>onePlusN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>callLegProfile fields as described above, if present show the overrides currently active specifically for this call leg (i.e. not those in force because of a higher level such as the call leg's associated tenant)</td>
<td></td>
</tr>
<tr>
<td>needsActivation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>defaultLayout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participantLabels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>endCallAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>telepresenceCallsAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipPresentationChannelEnabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bfcpMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>disconnectOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>qualityMain (from version 2.2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>qualityPresentation (from version 2.2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participantCounter (from version 2.2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>status</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td></td>
<td>state</td>
<td>initial</td>
</tr>
<tr>
<td></td>
<td>durationSeconds</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>sipCallID</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td>groupId</td>
<td>ID</td>
</tr>
<tr>
<td></td>
<td>recording</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>streaming</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>deactivated</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>encryptedMedia</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>unencryptedMedia</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>layout</td>
<td>one of:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speakerOnly</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>
| Name              | activeLayout | one of: speakerOnly |}
<p>|                   |             | telepresence | stacked | allEqual | allEqualQuarters | allEqualNinths | allEqualSixteenths| allEqualTwentyFifths | onePlusFive| onePlusSeven| onePlusNine| automatic| onePlusN |
| availableVideoStreams | Number | |</p>
<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxAudio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>codec</td>
<td>Name</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>one of:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g711u</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g711a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g722</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g728</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g729</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g722_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>g722_1c</td>
</tr>
<tr>
<td></td>
<td></td>
<td>aac</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speexNb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speexWb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speexUwb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>isacWb</td>
</tr>
<tr>
<td></td>
<td></td>
<td>isacSwb</td>
</tr>
<tr>
<td>jitter</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

Cisco Meeting Server Release 2.2 : API Reference Guide 86
<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>bitRate</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The actual measured bit rate of the incoming audio data</td>
</tr>
<tr>
<td></td>
<td>codecBitRate</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value is present for audio codec types with variants that can only be distinguished via bit rate (for example, the G.722.1 audio codec) - in these cases this field will be the expected audio bit rate rather than the observed, measured value (this parameter is present from version 2.1 onwards)</td>
</tr>
<tr>
<td></td>
<td>packetLossPercentage</td>
<td>Number</td>
</tr>
</tbody>
</table>

8 Call Related Methods
### Response elements

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>txAudio</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### txAudio

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>codec</td>
<td>one of:</td>
<td>the audio codec</td>
</tr>
<tr>
<td></td>
<td>g711u</td>
<td>used:</td>
</tr>
<tr>
<td></td>
<td>g711a</td>
<td>g711u - G.711</td>
</tr>
<tr>
<td></td>
<td>g722</td>
<td>mu law</td>
</tr>
<tr>
<td></td>
<td>g728</td>
<td>g711a - G.711 a</td>
</tr>
<tr>
<td></td>
<td>g729</td>
<td>law</td>
</tr>
<tr>
<td></td>
<td>g722_1</td>
<td>g722 - G.722</td>
</tr>
<tr>
<td></td>
<td>g722_1c</td>
<td>g728 - G.728</td>
</tr>
<tr>
<td></td>
<td>aac</td>
<td>g729 - G.729</td>
</tr>
<tr>
<td></td>
<td>speexNb</td>
<td>g722_1 - G.722.1</td>
</tr>
<tr>
<td></td>
<td>speexWb</td>
<td>g722_1c - G.722.1C</td>
</tr>
<tr>
<td></td>
<td>speexUwb</td>
<td>(G.722.1 Annex C)</td>
</tr>
<tr>
<td></td>
<td>isacWb</td>
<td>aac - AAC</td>
</tr>
<tr>
<td></td>
<td>opus</td>
<td>speexNb - Speex</td>
</tr>
<tr>
<td></td>
<td></td>
<td>narrowband</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speexWb - Speex</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wideband</td>
</tr>
<tr>
<td></td>
<td></td>
<td>speexUwb - Speex</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ultra-wideband</td>
</tr>
<tr>
<td></td>
<td></td>
<td>isacWb - iSAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(internet Speech</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Audio Codec)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>isacSwb - iSAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(internet Speech</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Audio Codec)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>superwideband</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>jitter</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>bitRate</td>
<td>Number</td>
<td>The actual measured bit rate of the incoming audio data</td>
</tr>
<tr>
<td>roundTripTime</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>packetLossPercentage</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>-------------------</td>
</tr>
<tr>
<td>rxVideo</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Parameters included in parent tag

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td>main</td>
<td>presentation</td>
</tr>
</tbody>
</table>

### Response values

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>codec</td>
<td>one of: h261</td>
<td>the video codec used</td>
</tr>
<tr>
<td></td>
<td>h263</td>
<td>h261 - H.261</td>
</tr>
<tr>
<td></td>
<td>h263+</td>
<td>h263 - H.263</td>
</tr>
<tr>
<td></td>
<td>h264</td>
<td>h263+ - H.263+</td>
</tr>
<tr>
<td></td>
<td>h264Lync</td>
<td>h264 - H.264</td>
</tr>
<tr>
<td></td>
<td>vp8</td>
<td>h264Lync - H.264 SVC for Lync</td>
</tr>
<tr>
<td></td>
<td>rtVideo</td>
<td>vp8 - VP8</td>
</tr>
</tbody>
</table>

<p>| width      | Number        |                   |
| height     | Number        |                   |
| frameRate  | Number        |                   |
| jitter     | Number        |                   |
| bitRate    | Number        |                   |
| packetLossPercentage | Number |                   |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>txVideo</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Parameters included in parent tag

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>role</td>
<td></td>
<td>The type of video stream: main or presentation</td>
</tr>
</tbody>
</table>

### Response values

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>codec</td>
<td></td>
<td>the video codec used</td>
</tr>
<tr>
<td>width</td>
<td></td>
<td></td>
</tr>
<tr>
<td>height</td>
<td></td>
<td></td>
</tr>
<tr>
<td>frameRate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jitter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bitRate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>roundTripTime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>packetLossPercentage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Name</td>
<td>activeControl</td>
<td>Description/Notes</td>
</tr>
<tr>
<td></td>
<td>(from version 2.1)</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>encrypted</td>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td></td>
<td>encrypted</td>
<td>truel</td>
</tr>
<tr>
<td></td>
<td>encrypted</td>
<td>truefalse</td>
</tr>
<tr>
<td></td>
<td>numScreens</td>
<td>Number</td>
</tr>
</tbody>
</table>

See table below.

See table below.
### Status of Active Control parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| localSubscriptions  
(from version 2.2) | | |
| capabilities | Name | If present, this indicates that the local Meeting Server has subscribed to the far end’s XCCP capabilities. |
| conferenceInfo | | If present, this indicates that the local Meeting Server has subscribed to the far end’s XCCP conference information (this includes the participant list and some conference-wide information such as whether recording is active). |
| layouts | | If present, this indicates that the local Meeting Server has subscribed to the far end’s XCCP layout information. |
| selfInfo | | If present, this indicates that the local Meeting Server has subscribed to the far end’s XCCP self information. |
| speakerInfo | | If present, this indicates that the local Meeting Server has subscribed to the far end’s XCCP speaker information. |
| remoteSubscriptions  
(from version 2.2) | | |
| capabilities | Name | If present, this indicates that the far end Meeting Server has subscribed to the local XCCP capabilities. |
| conferenceInfo | | If present, this indicates that the far end Meeting Server has subscribed to the local XCCP conference information (this includes the participant list and some conference-wide information such as whether recording is active). |
| layouts | | If present, this indicates that the far end Meeting Server has subscribed to the local XCCP layout information. |
| selfInfo | | If present, this indicates that the far end Meeting Server has subscribed to the local XCCP self information. |
| speakerInfo | | If present, this indicates that the far end Meeting Server has subscribed to the local XCCP speaker information. |

**Note:** See Appendix A for an example of the call leg configuration and status returned.

### 8.4 Call Leg Profile Methods

#### 8.4.1 General information

A call leg profile defines a set of in-call behaviors. coSpace, coSpaceUser, accessMethod, and tenant objects can optionally have a callLegProfile association – if so, call legs that correspond
to those objects inherit the in-call behavior defined by the call leg profile. For more information see also Section 14.

8.4.2 Retrieving call leg profiles

GET method on the " /callLegProfiles/" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; on the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>usageFilter</td>
<td>unreferenced</td>
<td>referenced</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>needsActivation</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

Cisco Meeting Server Release 2.2 : API Reference Guide 93
<table>
<thead>
<tr>
<th><strong>Response elements</strong></th>
<th><strong>Type/Value</strong></th>
<th><strong>Description/Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>participantLabels</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td>dualStream</td>
<td>singleStream</td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllPresentationContributionAllowed</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>endCallAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>changeJoinAudioMuteOverrideAllowed</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllMuteSelfAllowed</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td>Number</td>
<td>Number of participants up to which a &quot;join tone&quot; will be played (a value of 0 &quot;disables&quot; the feature). Cisco Meeting App users do not receive these audio indications because the Roster List provides a visual indication of who is joining and leaving.</td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td>Number</td>
<td>Number of participants up to which a &quot;leave tone&quot; will be played out (a value of 0 &quot;disables&quot; the feature)</td>
</tr>
<tr>
<td>videoMode</td>
<td>auto</td>
<td>disabled</td>
</tr>
<tr>
<td>rxAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>rxVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td>optional</td>
<td>required</td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td>Number</td>
<td>Numeric value for preferred packet size for outgoing audio streams (in milliseconds, the default value is 20ms)</td>
</tr>
</tbody>
</table>
## Call Related Methods

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>deactivationMode</td>
<td>deactivate</td>
<td>disconnect</td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td>Number</td>
<td>Number of seconds after the last &quot;activator&quot; leaves before which deactivationMode action is taken.</td>
</tr>
<tr>
<td>telepresenceCallsAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipPresentationChannelEnabled</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>bfcpMode</td>
<td>serverOnly</td>
<td>serverAndClient</td>
</tr>
<tr>
<td>callLockAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>recordingControlAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Name of profile. This parameter is present from version 2.0 onwards.</td>
</tr>
<tr>
<td>maxCallDurationTime</td>
<td>Number</td>
<td>The maximum amount of time in seconds that the call leg will exist. This parameter is present from version 2.0 onwards.</td>
</tr>
</tbody>
</table>
8.4.3 Creating and modifying a call leg profile

- **Creating:** POST method to the “/callLegProfiles” node. If the call leg profile is created successfully, a “200 OK” response will be received, and the “Location” header in the response will contain the ID of the new call leg profile.

- **Modifying a call leg profile** is a PUT method on a " /callLegProfiles/<call leg profile id>" node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>needsActivation</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>defaultLayout</td>
<td>allEqual</td>
<td>speakerOnly</td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>participantLabels</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td>dualStream</td>
<td>singleStream</td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllPresentationContributionAllowed</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>endCallAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>disconnectOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>changeJoinAudioMuteOverrideAllowed</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>allowAllMuteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td>Number</td>
<td>Number of participants up to which a &quot;join tone&quot; will be played (a value of 0 &quot;disables&quot; the feature). Cisco Meeting App users do not receive these audio indications because the Roster List provides a visual indication of who is joining and leaving.</td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td>Number</td>
<td>Number of participants up to which a &quot;leave tone&quot; will be played out (a value of 0 &quot;disables&quot; the feature)</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>videoMode</td>
<td>auto</td>
<td>disabled</td>
</tr>
<tr>
<td>rxAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txAudioMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>rxVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>txVideoMute</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td>optional</td>
<td>required</td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td>Number</td>
<td>Numeric value for preferred packet size for outgoing audio streams (in milliseconds, the default value is 20ms)</td>
</tr>
<tr>
<td>deactivationMode</td>
<td>deactivate</td>
<td>disconnect</td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td>Number</td>
<td>Number of seconds after the last &quot;activator&quot; leaves before which the deactivationMode action is taken</td>
</tr>
<tr>
<td>telepresenceCallsAllowed</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>sipPresentationChannelEnabled</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
| bfcpMode                                       | serverOnly|serverAndClient | If presentation video channel operations are enabled for SIP calls, this setting determines the Call Bridge’s BFCP behaviour  
  - serverOnly - this is the normal setting for a conferencing device, and is intended for use with BFCP client mode devices (for instance, SIP endpoints)  
  - serverAndClient - this setting allows the Call Bridge to operate in either BFCP client or BFCP server mode in calls with remote devices. This can allow improved presentation video sharing with a remote conference-hosting device such as a third party MCU |
<p>| callLockAllowed                                | true|false                     | Determines whether or not call legs using this call leg profile are allowed to lock the call. |
| recordingControlAllowed                        | true|false                     | If true, call legs using this call leg profile are allowed to start/stop recording the call. |
| streamingControlAllowed                        | true|false                     | If true, call legs using this call leg profile are allowed to start/stop streaming the call. (From version 2.1) |
| name                                           | String                      | Name of profile. This parameter is present from version 2.0 onwards.              |
| maxCallDurationTime                            | Number                      | The maximum amount of time in seconds that the call leg will exist. This parameter is present from version 2.0 onwards. |</p>
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>qualityMain</td>
<td>unrestricted</td>
<td>Restricts the maximum negotiated main video call quality for this call leg based on limiting transcoding resources. Specified using a typical resolution and framerate. Note that call legs may operate at lower resolutions or framerates due to endpoint limitations or overall Call Bridge load. (From version 2.2)</td>
</tr>
<tr>
<td></td>
<td>max1080p30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max720p30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max480p30</td>
<td></td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
</table>
| qualityPresentation| unrestricted, max1080p30, max720p5 | Restrict the maximum negotiated presentation video call quality for this call leg based on limiting transcoding resources. Specified using a typical resolution and frame rate. This only affects legs which use a separate presentation stream. (From version 2.2) 
unrestricted - this is the default setting if not specified, and matches the behavior of older Call Bridge versions, where no restrictions are placed on resolution or framerate
max1080p30 - restricts the Call Bridge to negotiating at most 1920x1080 screen size at 30 frames per second or equivalent transcoding resources
max720p5 - restricts the Call Bridge to negotiating at most 1280x720 screen size at 5 frames per second or equivalent transcoding resources. |
| participantCounter | never, auto, always         | Controls the behavior of the onscreen participant counter. (From version 2.2) 
never - never show an onscreen participant count value
auto - show the onscreen participant count value when appropriate. Typically this will be to indicate that there are additional participants present that you cannot currently see.
always - always show the onscreen participant count value |

In all cases, if you explicitly setting a parameter to an empty value in the POST or PUT, that parameter is "unset" for that profile. Those call legs then "inherit" the value for that parameter from the level above's call leg profile.

### 8.4.4 Retrieving information on an individual call leg profile

GET method on a /callLegProfiles/<call leg profile id> node. If the call leg profile ID supplied is valid, a "200 OK" response and a single "<callLegProfile id=<call leg profile ID>" object will be
returned with data as per the previous section.

8.4.5 Example call leg profile & access method use

The main use for being able to associate call leg profiles with access methods is to be able to construct separate URI / call ID / passcode combinations giving different in-call behaviors. For example, one call leg profile whose “needsActivation” value is “true” could be associated with one access method, and another call leg profile whose “needsActivation” value is “false” could be associated with a different access method.

Effectively this sets up separate “activator” and “guest” access methods for that coSpace, with callers to the “needsActivation=true” access method needing to wait until a successful call in to the other access method before their conference audio and video become active. For the multiple access methods linked to different call leg profiles in this way, you can choose to distinguish between them only by passcode; essentially, activator and guest users dial the same URI but enter a different PIN depending on whether they’re an activator or guest participant.

8.4.6 /callLegProfiles/<call leg profile id>/usage object method

There is a /callLegs/<call id>/usage object in the hierarchy. Performing a GET on this object retrieves, for the queried call leg, a list of where the specified call leg profile is used: whether it is set to be the global call leg profile or any associations it has with tenants, coSpaces, coSpace users, coSpace access methods.

8.4.7 /callLegs/<call leg id>/callLegProfileTrace object method

There is a /callLegs/<call leg id>/callLegProfileTrace object in the hierarchy. Performing a GET on this object retrieves, for the call leg that you queried, how its in-force call leg profile has been arrived at; that is, the hierarchy of overrides that have contributed to the currently "in force" call leg profile. Specifically, the response includes a section for each level in the profile hierarchy, and details which call leg profile elements have been applied at which level.

The end result for each parameter is the lowest level’s override for that parameter; for example, if the tenant-level call leg profile has set “participantLabels” to true but the coSpace call leg profile has it set to false, then call legs in that coSpace will not display participant labels.

GET method on the "/callLegProfileTrace/" node to retrieve call leg profile trace.
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>scope</td>
<td>one of:</td>
<td>Indicates at which level the set of profile parameters have been applied</td>
</tr>
<tr>
<td></td>
<td>global</td>
<td>the profile parameters given have been applied at the system-wide level; specifically, this means those parameters are present in the top-level callLegProfile, configured under system/profiles</td>
</tr>
<tr>
<td></td>
<td>tenant</td>
<td>the profile parameters given have been applied at the tenant level</td>
</tr>
<tr>
<td></td>
<td>coSpace</td>
<td>the profile parameters given have been applied at the coSpace level</td>
</tr>
<tr>
<td></td>
<td>accessMethod</td>
<td>the profile parameters given have been applied at the access method level</td>
</tr>
<tr>
<td></td>
<td>coSpaceUser</td>
<td>the profile parameters given have been applied via a callLegProfile associated with a coSpaceUser</td>
</tr>
<tr>
<td></td>
<td>callOut</td>
<td>the profile parameters given have been applied via a callLegProfile supplied when the call leg was created</td>
</tr>
<tr>
<td></td>
<td>callLeg</td>
<td>the profile parameters given have been applied specifically to this call leg (i.e. not via any configured callLegProfile object)</td>
</tr>
<tr>
<td>id</td>
<td>id</td>
<td>If present, the callLegProfile applicable to the scope of this entry</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>needsActivation</td>
<td></td>
<td>Parameters that show which call leg profile values have been overridden at this level</td>
</tr>
<tr>
<td>defaultLayout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>changeLayoutAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participantLabels</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationDisplayMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationContributionAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>presentationViewingAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>endCallAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>muteOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMuteOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>muteSelfAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMuteSelfAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>joinToneParticipantThreshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>leaveToneParticipantThreshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>videoMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txAudioMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rxVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>txVideoMute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipMediaEncryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>audioPacketSizeMs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>deactivationModeTime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>telepresenceCallsAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sipPresentationChannelEnabled</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bfcpMode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>disconnectOthersAllowed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>qualityMain (from version 2.2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>qualityPresentation (from version 2.2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participantCounter (from version 2.2)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
8.4.8 /callLegs/<call leg id>/generateKeyframe

POST to /callLegs/<call leg id>/generateKeyframe to trigger the generation of a new keyframe in outgoing video streams for the call leg in question. This is a debug facility, and Cisco Support may ask you to use the feature when diagnosing an issue.

8.5 DialTransform Methods

When dial transforms are applied to all outbound calls, then the outbound dial plan rules are applied to the transformed number.

You can use the Web Admin Interface Configuration > Outbound Calls page to control how dialed numbers are transformed. For example, the dial plan in the screen shot below ensures that outbound " +1 " (US) calls use one Call Bridge and +44 (UK) calls use another.

However, you need to use the API for dialTransforms if you use Call Bridge clustering, because the shared coSpace database is a single configuration location for all Call Bridges. In a cluster you do not need to configure the dialTransforms separately on each Call Bridge. The dialTransforms for the cluster are those defined on the Call Bridge host server (Meeting Server or virtualized deployment) that is co-located with the first coSpace database in the database cluster. Although the same dialTransforms are applied to all Call Bridge in the cluster, the outbound dial plan rules can be configured per-Call Bridge as described in Section 7.

There are three stages to the transform

- A “type” is applied, which breaks the input number/string into components $1, $2 etc.
- The components are matched using regular expressions to see if the rule is valid
- An output string is created from the components according to the defined transform

See the examples below.

Note: A phone URI is recognized as a purely numeric string (optionally prefixed by a ‘+’) when it begins with a valid international dial code (e.g. 44 for UK or 1 for US) followed by the correct number of digits for a phone number for that region.

8.5.1 Retrieving dial transforms

GET method performed on the “/dialTransforms” node.
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>Enter a filter to retrieve only those dial transforms that match the strong.</td>
</tr>
</tbody>
</table>

### Response elements

<table>
<thead>
<tr>
<th>Type</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>raw</td>
<td>strip</td>
</tr>
<tr>
<td></td>
<td>The type of pre-processing to apply to this transform</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Raw: produces one component - $1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Strip: removes dots, dashes, spaces and produces one component - $1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Phone: An international phone number - produces two components $1county code and $2number</td>
<td></td>
</tr>
</tbody>
</table>

### 8.5.2 Setting up and modifying dial transforms

- **Creating**: POST method to the "/dialTransforms" node
- **Modifying**: PUT to " /dialTransforms/<dialTransform id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>raw</td>
<td>strip</td>
</tr>
<tr>
<td></td>
<td>Raw: produces one component - $1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Strip: removes dots, dashes, spaces and produces one component - $1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Phone: An international phone number - produces two components $1county code and $2number</td>
<td></td>
</tr>
<tr>
<td>match</td>
<td>String</td>
<td>If provided, the regular expression describing whether this rule will be applied. An empty string means &quot;match all&quot;.</td>
</tr>
<tr>
<td></td>
<td>This is a logically AND’ed combination of regular expressions, each applied to a component of the pre-processed expression.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The format is:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(<code>&lt;$componentnum_1&gt;/&lt;regex_1&gt;)/</code>(<code>&lt;$componentnum_2&gt;/&lt;regex_2&gt;/</code>)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(<code>&lt;$componentnum_3&gt;/&lt;regex_3&gt;/</code>)...</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For example</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- (<code>$2/abc/</code>): Component 2 must contain 'abc'</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- (<code>$1/^0/</code>)/($1/9$/): Component 1 must start with a 0 and end with a 9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- (<code>$1/^44$/</code>)/($2/^7/): Component 1 must be '44' and component 2 must start with a 7</td>
<td></td>
</tr>
</tbody>
</table>

---

*Cisco Meeting Server Release 2.2 : API Reference Guide*
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
</table>
| transform  | String     | The replacement transform to be applied. This allows references to the pre-processed components, as well as one or more regular expression substitutions encased in curly braces with the following special strings replaced as described. $<\text{componentnum}>$ : Replace with component $<\text{componentnum}>[]$ : Replace with component $<\text{componentnum}>[/<\text{matchregex1}>/<\text{replaceregex1}>]$ $<\text{componentnum}>[/<\text{matchregex2}>/<\text{replaceregex2}>]$ $<\text{componentnum}>[/<\text{matchregex3}>/<\text{replaceregex3}>]$: Replace with component, with all instances of matchregex1 replaced by replaceregex1, and subsequently matchregex2 replaced by replaceregex2, etc. Capture groups are supported. Examples are:  
  - `abc`: Replace everything with 'abc'  
  - `$1$2@t.com`: Component 1 followed by component 2 followed by '@t.com  
  - `$1(123@t.com`: Component 1 followed by '123', followed by '@t.com'  
  - `$1(/999/123/)/@t.com`: Component 1 with all instances of '999' replaced by '123', followed by '@t.com'  
  - `$1(/\D)/(^9//)@example.com`: Component 1 with all non-digits removed and leading 9 removed, followed by '@example.com' |
| priority   | Number     | The priority this transform rule should have. Rules with higher priorities are applied first |
| action     | accept| acceptPhone| deny | The action to take if this rule matches. |

### 8.5.3 Retrieving detailed information about an individual dial transform

GET method performed on a "/dialTransform/<dialTransform id>" node. If the call branding profile id ID supplied is valid, a "200 OK" response is received, with XML content matching the section above.

### 8.5.4 Examples

<table>
<thead>
<tr>
<th>Example</th>
<th>Type</th>
<th>Match</th>
<th>Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>For US numbers, use 'vcs1' directly</td>
<td>Phone</td>
<td>$(1/01)/</td>
<td>$2@vcs1</td>
</tr>
<tr>
<td>For UK numbers, add a prefix and use 'vcs2'</td>
<td>Phone</td>
<td>$(1/44)/</td>
<td>90044$2@vcs2</td>
</tr>
</tbody>
</table>
8.6 Setting Individual Features for a Call Leg

You can set, modify and retrieve on a per active call leg basis, these settings include whether a presentation should be restricted to single screen mode (i.e. one combined main and presentation video stream), or allowed to use separate video streams if this is supported by the receiving party.

Note: Setting individual parameters for a call leg overrides the values of the call leg profile.

8.7 Call Branding Profile Methods

Call branding profiles control the in-call experience for SIP (including Lync) calls, and the ability to customize text within invitations. For more information see also Section 14.

Note: that the use of callBrandingProfiles requires a branding license.

8.7.1 Retrieving Call Branding Profiles

GET method performed on the “/callBrandingProfiles” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number, Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>usageFilter</td>
<td>referenced, unreferenced</td>
<td>Using unreferenced retrieves only those call branding profiles that are not referenced by global settings or any other object. This is a useful check before deleting a call branding profile.</td>
</tr>
</tbody>
</table>
### 8.7.2 Setting up and modifying call branding profiles

- Creating: POST method to the `/callBrandingProfiles` node
- Modifying: PUT to `/callBrandingProfiles/<call branding profile id>`

#### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>invitationTemplate</td>
<td>URL</td>
<td>The HTTP or HTTPS URL of the invitation template text which Meeting Apps will use when constructing textual invitations. Refer to the Cisco Meeting Server Customization Guide for details on how to customize the text within the invitation.</td>
</tr>
<tr>
<td>resourceLocation</td>
<td>URL</td>
<td>The HTTP or HTTPS URL that the Call Bridge call branding files will be retrieved from. This is the “directory” in which the individual audio and graphic files reside. Details of these files are in the Cisco Meeting Server Customization Guide.</td>
</tr>
</tbody>
</table>

### 8.7.3 Retrieving detailed information about an individual call branding profile

GET method performed on a “/callBrandingProfiles/<call branding profile id>” node. If the call branding profile id ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

### 8.8 DTMF Profile Methods

dtmfProfiles can be used to define a number of DTMF sequences that can be used to control audio – as explained in this section. The dtmfProfile does not define the ability to perform the actions, it defines the DTMF string that will invoke action. The definition for who has the authority to invoke that action within the coSpace is defined at the callLegProfile level. For more information see also Section 14.

If you are using the Meeting Server alongside third party solutions, or to replace an existing solution, then set the values to match the values that solution uses e.g. Lync conferencing uses *6 for both mute and unmute, so set toggleMuteSelfAudio to *6.

#### 8.8.1 Retrieving DTMF Profiles

GET method performed on the “/dtmfProfiles” node.
### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>usageFilter</td>
<td>referenced</td>
<td>unreferenced</td>
</tr>
</tbody>
</table>

### Response elements

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>muteSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to mute the audio being contributed to their call</td>
</tr>
<tr>
<td>unmuteSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute their audio</td>
</tr>
<tr>
<td>toggleMuteSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to toggle between mute and unmute audio from themselves</td>
</tr>
<tr>
<td>muteAllExceptSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to mute all other participants in the call.</td>
</tr>
<tr>
<td>unmuteAllExceptSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute all other participants in the call.</td>
</tr>
<tr>
<td>startRecording</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to start recording the active call.</td>
</tr>
<tr>
<td>stopRecording</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to stop recording the active call.</td>
</tr>
<tr>
<td>muteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to mute all new participants. Sets joinAudioMuteOverride Call object to true.</td>
</tr>
<tr>
<td>unmuteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute all new participants. Sets joinAudioMuteOverride Call object to false.</td>
</tr>
<tr>
<td>defaultMuteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to use the audio mute value from the call leg profile for new participants.</td>
</tr>
</tbody>
</table>
8.8.2 Setting up and modifying DTMF Profiles

- Creating: POST method to the /dtmfProfiles" node
- Modifying: PUT to " /dtmfProfiles/<dtmfprofile id>"
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>unmuteAllExceptSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute all other participants in the call.</td>
</tr>
<tr>
<td>startRecording</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to start recording the active call.</td>
</tr>
<tr>
<td>stopRecording</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to stop recording the active call.</td>
</tr>
<tr>
<td>startStreaming</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to start streaming the active call.</td>
</tr>
<tr>
<td>stopStreaming</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to stop streaming the active call.</td>
</tr>
<tr>
<td>allowAllMuteSelf</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to allow all participants to mute and unmute themselves. Sets allowAllMuteSelf in the Call object to true.</td>
</tr>
<tr>
<td>cancelAllowAllMuteSelf</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to cancel the permission to allow all participants to mute and unmute themselves. Sets allowAllMuteSelf in the Call object to false.</td>
</tr>
<tr>
<td>allowAllPresentationContribution</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to allow all participants to present.</td>
</tr>
<tr>
<td>cancelAllowAllPresentationContribution</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to cancel the permission for all participants to present.</td>
</tr>
<tr>
<td>muteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to mute all new participants. Sets joinAudioMuteOverride Call object to true.</td>
</tr>
<tr>
<td>unmuteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute all new participants. Sets joinAudioMuteOverride Call object to false.</td>
</tr>
<tr>
<td>defaultMuteAllNewAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to use the audio mute value from the call leg profile for new participants.</td>
</tr>
<tr>
<td>muteAllNewAndAllExceptSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to mute all new participants and all other participants in the call. Sets joinAudioMuteOverride in the call object to ‘true’ and mutes all call legs except for the issuer. This requires ‘muteOthersAllowed’ to be ‘true’ in the call leg profile of the issuer.</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>unmuteAllNewAndAllExceptSelfAudio</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to unmute all new participants and all other participants in the call. Sets joinAudioMuteOverride in the call object to ‘false’ and unmutes all call legs except for the issuer. This requires ‘muteOthersAllowed’ to be ‘true’ in the call leg profile of the issuer.</td>
</tr>
<tr>
<td>endCall</td>
<td>String</td>
<td>DTMF sequence to be used by a participant to end the call; this will disconnect all participants including the participant who initiated the operation</td>
</tr>
</tbody>
</table>

### 8.8.3 Retrieving detailed information about an individual dtmfProfile

GET method performed on a “/dtmfProfiles/<dtmfprofile id>” node. If the dtmfProfile ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

### 8.9 IVR Methods

#### 8.9.1 Retrieving IVRs

GET method performed on the “/ivrs” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those IVRs that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those IVRs associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>URI user part</td>
<td>Response is structured as a top-level &lt;ivrs total=&quot;N&quot;&gt; tag with potentially multiple &lt;ivr&gt; elements within it. Each &lt;ivr&gt; tag may include the following elements: The URI to be used for this IVR</td>
</tr>
</tbody>
</table>
8.9.2 Setting up and modifying IVRs

- Creating: POST method to the /ivrs" node
- Modifying: PUT to " /ivrs/<ivr id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri *</td>
<td>URI user part</td>
<td>The URI to be used for this IVR</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If specified, calls to this IVR will only be able to join coSpaces associated with the specified tenant. If no tenant is supplied, calls to this IVR will be able to join any call on the system that has a call ID configured.</td>
</tr>
<tr>
<td>tenantGroup</td>
<td>ID</td>
<td>Calls to this IVR will only be able to join coSpaces associated with tenants within the specified tenant group. If no tenant group is supplied, calls to this IVR will only be able to join coSpaces without a tenant, or associated with a tenant in no tenant group.</td>
</tr>
<tr>
<td>ivrBrandingProfile</td>
<td>ID</td>
<td>If supplied, specifies an IVR branding profile to be used for calls to this IVR - an IVR branding profile supplied here will take precedence over any top-level or tenant-level IVR branding profile.</td>
</tr>
<tr>
<td>resolveCoSpaceCallIds</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>resolveLyncConferenceIds</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

8.9.3 Retrieving detailed information about an individual IVR

GET method performed on a “/ivrs/<ivr id>" node. If the IVR ID supplied is valid, a “200 OK" response is received, with XML content matching the section above.

8.10 IVR Branding Profile Methods

IVR branding profiles can define the experience when dialing into an IVR. For more information see also Section 14.

8.10.1 Retrieving IVR branding profiles

GET method performed on the “/ivrBrandingProfiles" node.
### 8.10.2 Setting up and modifying an IVR branding profile

- **Creating:** POST method to the `/ivrBrandingProfiles` node
- **Modifying:** PUT to `/ivrBrandingProfiles/<ivr branding profile id>`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>resourceLocation</td>
<td>URL</td>
<td>The HTTP or HTTPS URL that the IVR branding files will be retrieved from. This should be the &quot;directory&quot; in which the individual audio and graphic files reside. Details of these files are in the Cisco Meeting Server Customization Guide.</td>
</tr>
</tbody>
</table>

### 8.10.3 Retrieving detailed information about an individual IVR branding profile

GET method performed on a “/ivrBrandingProfiles/<ivr branding profile id>“ node. If the IVR branding profile ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

### 8.11 Participant Related Methods

“participant” should not be confused with “callLeg” objects. A “participant” can be a user’s Lync session in which there might be separate call legs for audio and video, application sharing and IM.
Each Call Bridge involved in a distributed call has an overall picture of the “participant” list for that call, including participants hosted on other Call Bridges. For participants hosted on the Call Bridge being queried, you can enumerate the constituent call legs but for participants hosted on another Call Bridge, querying those participants yields the ID of the Call Bridge on which they are hosted. (You can then query the "owning" Call Bridge using the same participant ID in order to retrieve call leg-level detail.)

**Note:** Also see the section on /call/callLegId/participants Section 8.3.3.

### 8.11.1 Retrieving participants

GET method performed on the "/participants" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those active participants that match the filter</td>
</tr>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2)</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply the tenantFilter to return only those participants belonging to that tenant</td>
</tr>
<tr>
<td>callBridgeFilter</td>
<td>ID</td>
<td>Supply the callBridgeFilter to return only those participants located on that Call Bridge</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The human-readable display name associated with this participant</td>
</tr>
<tr>
<td>call</td>
<td>ID</td>
<td>The call that this participant is part of</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If present, the id of the tenant that this participant is associated with</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>If present, the remote clustered Call Bridge that this participant is connected to</td>
</tr>
<tr>
<td>uri</td>
<td>String</td>
<td>The URI associated with this participant</td>
</tr>
<tr>
<td>numCallLegs</td>
<td>Number</td>
<td>The current number of active call legs associated with this participant. This value will only be present for those participants local to the Call Bridge to which the request is made</td>
</tr>
<tr>
<td>userJid</td>
<td>String</td>
<td>The userJid associated with this participant</td>
</tr>
</tbody>
</table>
### 8.11.2 Making a participant important if they are already in a conference

- Modifying: PUT to "/participants/<participant id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>importance</td>
<td>Number</td>
<td>Set the importance of this participant already in a conference. For example to 1. To remove importance leave the importance parameter as unset (leave value as blank). (From version 2.2)</td>
</tr>
</tbody>
</table>

### 8.11.3 Retrieving detailed information on an individual participant

GET method performed on a “/participants/<participant ID>” node

If the participant ID supplied is valid, a “200 OK” response is received, with XML content of the form:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>The human-readable display name associated with this participant</td>
</tr>
<tr>
<td>call</td>
<td>ID</td>
<td>The call that this participant is part of</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If present, the id of the tenant that this participant is associated with</td>
</tr>
</tbody>
</table>
8.11.4 Retrieving a participant’s call legs

GET method performed on a "/participant/ <participant ID>/callLegs" node retrieves the participant’s active call legs. If successful, the parameters described above for call legs are returned.

Note that if this call leg is part of a distributed meeting (one hosted by more than one Call Bridge) then these details are only returned for local participants. If the participant’s call legs are hosted by another call Bridge the id of that Call Bridge is returned.
8.11.5 Limiting a call’s participants

You can set a limit on the number of participants that are permitted to be in a call. You can set:

- A per-tenant participantLimit value, which imposes a limit on the total number of participants that are allowed to be active for that tenant.
- A "participantLimit" value within a 'callProfile' object; this means that calls (e.g. coSpace instantiations) for whom that "callProfile" is in force will have the limit enforced.

The callProfiles can be attached at the system, tenant or coSpace level, with the most specific taking effect. Therefore a call’s participantLimit depends on a number of factors.

If a call’s "participantLimit" has been reached:

- No new participants can be added to it.

However:

- A participant on a Cisco Meeting App can use any combination of chat, use video, audio and show/receive a presentation. These elements comprise one callLeg, and count as one participant.
  Using a slaved endpoint does not increase the participant count.
- A participant in a meeting on a SIP endpoint can use video, audio and receive a presentation. These elements comprise one callLeg, and count as one participant.
- A participant on a Lync client can use any combination of chat, use video, audio and send a presentation. Any combination of these elements counts as one participant but each element is a separate callLeg. (A received presentation is displayed in the main video stream.)
- New call legs for existing participants can still be added; for example, a Lync presentation call leg to go with a Lync audio/video call leg.

If creation of a call leg or participant via an API method fails because a limit has been reached, you see the appropriate "failureReason". If an incoming connection attempt is unsuccessful because a limit has been reached, you also see an error message (with a separate callLegEnd reason for whether the call’s own limit has been reached or that of its owning tenant).
9 User Related Methods

Users are created by synchronizing against LDAP servers (as discussed later); however, there are a number of methods for retrieving user information. This chapter covers:

- retrieving information on users
- retrieving detailed information on an individual user
- configuring user profiles

9.1 Retrieving Users

GET method performed on the “/users” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those users that match the filter</td>
</tr>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first * page* in the notional list (see above)</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those users associated with the specified tenant</td>
</tr>
<tr>
<td>emailFilter</td>
<td>String</td>
<td>Supply emailFilter to restrict results returned to those users whose email value exactly matches the specified email address (from version 2.1).</td>
</tr>
<tr>
<td>cdrTagFilter</td>
<td>String</td>
<td>Supply cdrTagFilter to restrict results returned to those users whose cdrTag value exactly matches the specified cdrTag (from version 2.1).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>user id</td>
<td>ID</td>
<td>&lt;user&gt; elements follow the general form on the left.</td>
</tr>
<tr>
<td>userJid</td>
<td>String</td>
<td>For example, <a href="mailto:first.last@example.com">first.last@example.com</a></td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>The id of the tenant with which this user is associated, if applicable</td>
</tr>
</tbody>
</table>

9.2 Retrieving Detailed Information on an Individual User

GET method performed on a “/users/<user ID>” node

If the user ID supplied is valid, a “200 OK” response is received, with XML content of the form:
<table>
<thead>
<tr>
<th><strong>Response elements</strong></th>
<th><strong>Type/Value</strong></th>
<th><strong>Description/Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>user id</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>userJid</td>
<td>String</td>
<td>e.g. <a href="mailto:first.last@example.com">first.last@example.com</a></td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>The id of the tenant with which this user is associated, if applicable</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>User’s display name</td>
</tr>
<tr>
<td>email</td>
<td>String</td>
<td>e.g. <a href="mailto:first.last@mail.example.com">first.last@mail.example.com</a></td>
</tr>
<tr>
<td>authenticationId</td>
<td>String</td>
<td>The id used for authentication; this value is checked against values from the certificate that the user presents during certificate-based authentication.</td>
</tr>
<tr>
<td>userProfile</td>
<td>ID</td>
<td>If present, this is the ID of the user profile associated with this user. (from version 2.0)</td>
</tr>
</tbody>
</table>

Retrieving a User’s coSpace Associations

GET method performed on a “/users/<user ID>/usercoSpaces” node retrieves the coSpaces that the user is a member of. (Also see the note on coSpace member permissions for auto-generated members.)

<table>
<thead>
<tr>
<th><strong>Response elements</strong></th>
<th><strong>Type/Value</strong></th>
<th><strong>Description/Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>coSpace</td>
<td>ID</td>
<td>Response is structured as a top-level &lt;usercoSpaces total=&quot;N&quot;&gt; tag with potentially multiple &lt;usercoSpace&gt; elements within it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;usercoSpace&gt; elements follow the general form on the left.</td>
</tr>
</tbody>
</table>

### 9.3 User Profile Methods

User profiles control the facilities provided to the users in the profile, for instance whether they can create new coSpaces, create new calls, make phone calls, slave SIP endpoints, allowed to send and receive chat messages when in a point to point call with another user. For more information see also Section 14.

#### 9.3.1 Retrieving user profiles

GET method performed on the “/userProfiles” node.

<table>
<thead>
<tr>
<th><strong>Parameters</strong></th>
<th><strong>Type/Value</strong></th>
<th><strong>Description/Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>usageFilter</td>
<td>Referenced</td>
<td>unreferenced</td>
</tr>
</tbody>
</table>

### Response elements

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>canCreateCoSpaces</td>
<td>true/false</td>
<td>Whether a user associated with this user profile is permitted to create new coSpaces</td>
</tr>
<tr>
<td>canCreateCalls</td>
<td>true/false</td>
<td>Whether a user associated with this user profile is permitted to create new calls</td>
</tr>
<tr>
<td>canUseExternalDevices</td>
<td>true/false</td>
<td>Whether a user associated with this user profile is permitted to use slave SIP devices</td>
</tr>
<tr>
<td>canMakePhoneCalls</td>
<td>true/false</td>
<td>Whether a user associated with this user profile will be displayed the option to make phone calls in the client</td>
</tr>
<tr>
<td>userToUserMessagingAllowed</td>
<td>true/false</td>
<td>Whether a user associated with this user profile will be allowed to send and receive messages when in a point to point call with another user</td>
</tr>
<tr>
<td>audioParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile and using the Cisco Meeting App will be allowed to send or receive live audio when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. (from version 2.0)</td>
</tr>
<tr>
<td>videoParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile will be allowed to send or receive live video when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. (from version 2.0)</td>
</tr>
<tr>
<td>presentationParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile will be allowed to send or receive presentation media when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. (from version 2.0)</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>hasLicense</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile has a Cisco user license. (from version 2.0)</td>
</tr>
<tr>
<td>canReceiveCalls</td>
<td>true/false</td>
<td>Determines whether or not a user associated with this user profile can receive incoming calls using the Cisco Meeting App. (from version 2.1)</td>
</tr>
</tbody>
</table>

### 9.3.2 Setting up and modifying user profiles
- Creating: POST method to the `/userProfiles` node
- Modifying: PUT to `/userProfiles/<user profile id>`

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>canUseExternalDevices</td>
<td>true/false</td>
<td>Whether a user associated with this user profile is permitted to use slave SIP devices</td>
</tr>
<tr>
<td>canMakePhoneCalls</td>
<td>true/false</td>
<td>Whether a user associated with this user profile will be displayed the option to make phone calls in the client</td>
</tr>
<tr>
<td>userToUserMessagingAllowed</td>
<td>true/false</td>
<td>Whether a user associated with this user profile will be allowed to send and receive messages when in a point to point call with another user</td>
</tr>
<tr>
<td>audioParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile and using the Cisco Meeting App will be allowed to send or receive live audio when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. This parameter is present from version 2.0 onwards.</td>
</tr>
<tr>
<td>videoParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile will be allowed to send or receive live video when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. This parameter is present from version 2.0 onwards.</td>
</tr>
<tr>
<td>presentationParticipationAllowed</td>
<td>true/false</td>
<td>Whether or not a user associated with this user profile will be allowed to send or receive presentation media when in a call. This restriction does not apply to dialing directly into the call via SIP, or slaving to a SIP endpoint. This parameter is present from version 2.0 onwards.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| hasLicense    | true|false     | Whether or not a user associated with this user profile has a Cisco user license.  
This parameter is present from version 2.0 onwards.                                  |
| canReceiveCalls | true|false     | Determines whether or not a user associated with this user profile can receive incoming calls using the Cisco Meeting App.  
This parameter is present from version 2.1 onwards.                                    |

9.3.3 Retrieving detailed information about an individual user profile

GET method performed on a “/userProfiles/<user profile id >" node. If the user profile ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.
10  System Related Methods

This chapter details the API methods related to management of the system. The chapter covers:

- retrieving system status
- retrieving system alarm status
- retrieving system database status
- retrieving and setting the URI of the CDR receivers
- retrieving and setting the global profile
- retrieving licensing information
- configuring the TURN server
- configuring the Web Bridge
- configuring the Call Bridge
- configuring Call Bridge groups
- configuring the XMPP server
- configuring Call Bridge clustering
- configuring the Recorder
- configuring the Streamer
- system load
- system diagnostics

10.1 Retrieving System Status

GET method performed on the “/system/status” node.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>softwareVersion</td>
<td>String</td>
<td>The software version currently running on the Call Bridge</td>
</tr>
<tr>
<td>uptimeSeconds</td>
<td>Number</td>
<td>The length of time that the unit has been running</td>
</tr>
<tr>
<td>activated</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>clusterEnabled</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
### Response elements

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdrTime</td>
<td>Number</td>
<td>The current timestamp as would be written to a CDR generated at the time the request is received. This will be in the same format as the &quot;time&quot; field in CDRs themselves (see <a href="https://tools.ietf.org/html/rfc3339">RFC 3339</a>, for instance &quot;2014-02-11T12:10:47Z&quot;).</td>
</tr>
<tr>
<td>callLegsActive</td>
<td>Number</td>
<td>The number of active call legs at the time of the request</td>
</tr>
<tr>
<td>callLegsMaxActive</td>
<td>Number</td>
<td>The highest number of call legs simultaneously active on this Meeting Server.</td>
</tr>
<tr>
<td>callLegsCompleted</td>
<td>Number</td>
<td>The total number of call legs that have been active but are no longer connected / present</td>
</tr>
<tr>
<td>audioBitRateOutgoing</td>
<td>Number</td>
<td>The current total bit rate (in bits per second) summed over all outgoing audio streams (audio media sent from the Meeting Server to a remote party)</td>
</tr>
<tr>
<td>audioBitRateIncoming</td>
<td>Number</td>
<td>The current total bit rate for incoming audio streams</td>
</tr>
<tr>
<td>videoBitRateOutgoing</td>
<td>Number</td>
<td>The current total bit rate for outgoing video streams</td>
</tr>
<tr>
<td>videoBitRateIncoming</td>
<td>Number</td>
<td>The current total bit rate for incoming video streams</td>
</tr>
<tr>
<td>cdrCorrelatorIndex</td>
<td>Number</td>
<td>The correlator index of the next CDR record that will be sent. When no CDR records have been sent, this will have the value of 0. (From version 2.2).</td>
</tr>
</tbody>
</table>

### 10.2 Retrieving System Alarm Status

GET method performed on the "/system/alarms" node. An offset and limit can be supplied to retrieve alarm conditions other than those in the first page of the notional list. This method returns a table detailing the currently active system-wide alarm conditions.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Returns a list of individual &quot;&lt;alarm&gt;&quot; elements. If there are no currently active alarm conditions, this list will be empty. Each active alarm condition will have an &quot;alarm&quot; tag, which contains:</td>
</tr>
<tr>
<td>Id</td>
<td>ID</td>
<td>A unique ID for this instance of this fault condition</td>
</tr>
<tr>
<td>activeTimeSeconds</td>
<td>Number</td>
<td>The amount of time that this alarm condition has been active.</td>
</tr>
</tbody>
</table>

(see right)
<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>One of:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- callBrandingResourceInvalid—the supplied resource has an invalid format; the call branding profile is specified by the accompanying &quot;callBrandingProfiles&quot; GUID parameter and the problematic file is specified by the accompanying &quot;fileName&quot; text parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- callBridgeConnectionFailure—the Call Bridge has failed to establish a connection to one of its peer clustered Call Bridges which specified by the accompanying &quot;callBridge&quot; GUID parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- callDistributionFailure—the Call Bridge has failed to establish a distributed link for one of its active calls; the Call Bridge that the link should have been to is identified by the accompanying &quot;callBridgeName&quot; text parameter, and the call is present as the &quot;call&quot; GUID parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- cdrConnectionFailure—the Meeting Server has failed to establish a connection to its configured CDR receiver, and therefore may be unable to push out new call detail records</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- databaseClusternodeOutOfSync—a node in the database cluster is out of sync and is not syncing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- databaseConnectionError—Meeting Server has failed to establish a connection to its database</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- guestAccountConnectionFailure—the Meeting Server has been unable to establish a connection to the configured Web Bridge in order to allow guest logins</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- ivrBrandingResourceInvalid—the supplied resource has an invalid format; the IVR branding profile is specified by the accompanying &quot;ivrBrandingProfile&quot; GUID parameter and the problematic file is specified by the accompanying &quot;fileName&quot; text parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- licenseGrace - a feature license has passed the expiry date and will soon be deactivated (from version 2.1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- licenseExpired - a feature license has expired and has been deactivated (from version 2.1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- recorderLowDiskSpace – a recorder has limited disk space; the recorder is specified by the accompanying &quot;recorder&quot; GUID parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- recorderUnavailable – the Call Bridge has not managed to successfully contact a configured recorder; the recorder is specified by the accompanying &quot;recorder&quot; GUID parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- streamerUnavailable – the Call Bridge has not managed to successfully contact a configured streamer; the streamer is specified by the accompanying &quot;streamer&quot; GUID parameter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- turnServerUnavailable—the Call Bridge has not managed to contact a configured TURN server; this TURN server is specified by the</td>
<td></td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>accompanying &quot;turnServer&quot; GUID parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>webBridgeArchivePushFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to push a required customization archive to a Web Bridge</td>
</tr>
<tr>
<td>webBridgeArchiveRetrievalFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to retrieve a required Web Bridge customization archive</td>
</tr>
<tr>
<td>webBridgeBackgroundImagePushFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to push a required customized background image file to a Web Bridge</td>
</tr>
<tr>
<td>webBridgeBackgroundImageRetrievalFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to retrieve a required customized background image file</td>
</tr>
<tr>
<td>webBridgeLoginLogoImagePushFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to push a required customized login logo image to a Web Bridge</td>
</tr>
<tr>
<td>webBridgeLoginLogoImageRetrievalFailure</td>
<td>-</td>
<td>the Call Bridge has not been able to retrieve a required customized login logo image file</td>
</tr>
<tr>
<td>xmppAuthenticationRegistrationFailure</td>
<td>-</td>
<td>the Meeting Server has not been able to register successfully with the named XMPP authentication component</td>
</tr>
<tr>
<td>xmppRegistrationFailure</td>
<td>-</td>
<td>the Meeting Server has not been able to register successfully with its configured XMPP Server</td>
</tr>
</tbody>
</table>
failureReason

For some of the alarm types above, additional information is provided about the cause of that particular failure:

- authenticationFailure
- connectFailure—a failure to connect to a remote destination; for example, a TCP or TLS connection was not able to be established
- dataFormatInvalid—the Call Bridge is configured to use a particular data set (for example, a remotely-hosted resource file) that it then found was not in a useable format
- destinationReadOnly—a critical resource, for example the database, was found by the Call Bridge to be read only when write access was needed
- dnsFailure—a failure to resolve the host name of a remote destination; for example, as part of the process of establishing a connection to a remote system
- error
- fileNotFound—the alarm condition was raised because the Call Bridge failed to load a required file; for instance, if the resourceArchive needed for a web bridge was not able to be retrieved from a remote server
- fileSizeLimitExceeded—the Call Bridge is configured to use a resource, for example, a remotely-hosted resource file, that was not able to be retrieved because it exceeded an internal file size limit
- internalServerError—the Call Bridge was not able to perform an operation, for example the upload or download of a resource file, because the remote party returned "internal server error" when the operation was attempted
- serviceUnavailable

10.3 Retrieving System Database Status

GET method performed on the “/system/database” node.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>clustered</td>
<td>enabled</td>
<td>disabled</td>
</tr>
<tr>
<td>cluster</td>
<td>String</td>
<td>If clustering is enabled then the &lt;cluster&gt; element includes the elements on the left</td>
</tr>
<tr>
<td>Error</td>
<td>String</td>
<td>Error description</td>
</tr>
<tr>
<td>totalNodes</td>
<td>Number</td>
<td>Number of database nodes in the cluster</td>
</tr>
<tr>
<td>nodeInUse</td>
<td>String</td>
<td>Which database node is currently in use (the master)</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>node</td>
<td></td>
<td>The node element is returned for each database in the cluster with the following details:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hostname: the hostname or IP address of the node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• up: if the node is visible from this Call Bridge (true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• syncBehind: approximate number of bytes that this node is behind the current state of the master. 0 means in sync, and -1 means the calculation is unavailable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• master: whether this node is the master database (true</td>
</tr>
</tbody>
</table>

### 10.4 CDR Receiver URI Methods

**Note:** /system/cdrReceiver is deprecated, use the /system/cdrReceivers object which supports multiple CDR receivers.

#### 10.4.1 Retrieving the CDR Receivers URI

You can find the URIs of the CDR receivers through the API (as well as the Web Admin Interface). Issue a GET on the /system/cdrReceivers node to retrieve the URIs that are the full URLs of the configured CDR receivers.

This method accesses the URIs of the CDR receivers using the Web Admin Interface **Configuration > CDR settings** page.

GET method performed on the "/system/cdrReceivers" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to CDR receivers other than the first &quot;page&quot; in the notional list</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>String</td>
<td>Full URL of the configured CDR receiver address</td>
</tr>
</tbody>
</table>

Response is structured as a top-level <cdrReceivers total="N" > tag with potentially multiple <cdrReceiver> elements within it. Each <cdrReceiver> tag may include the following element:
Note: GET of /system/cdrReceivers/<cdr receiver id> allows you to retrieve the configuration for a single specified CDR receiver.

10.4.2 Setting the CDR Receivers URI

Set the CDR receiver URI through the API (as well as the Web Admin Interface). You can issue a PUT or a POST on the /system/cdrReceivers node.

Use POST and specify a "url" value to create and configure the CDR receiver in one operation, or use PUT to initially create the CDR receiver but configure the “url” separately later.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>Full URL to which CDRs will be sent.</td>
</tr>
</tbody>
</table>

If the creation is successful, you should receive a “200 OK” response and a "Location: /api/v1/system/cdrReceivers/<cdr receiver id>" object reference; if too many CDR receivers are already configured, you will receive a "tooManyCdrReceivers" error (in a "failureDetails" section).

Note: If you perform a PUT with an empty "url" to the legacy /system/cdrReceiver node, any GUID associated with that CDR receiver is removed, and effectively that CDR receiver is no longer present. If you later PUT a non-empty "url" value to the same (legacy) node, a new GUID will be generated for that CDR receiver.

If you perform a PUT with an empty "url" to a non-legacy CDR receiver (/system/cdrReceivers/<cdr receiver id>) then that CDR receiver remains with the same GUID, but no "url" value. It will continue to show up in GET operations. This is because there is an explicit "DELETE" method (section 4.4) for the new CDR receiver objects, whereas for the legacy CDR receiver the only deconfiguration method is to set its location to the empty value.

To set or update the URIs of the CDR receivers via the Web Admin Interface use the Configuration > CDR settings page.

10.5 Global Profile Methods

10.5.1 Retrieving the Global Profile

GET to /system/profiles returns the values described in the following section.

10.5.2 Setting the Global Profile

You can set (or unset) the callLegProfile ID value under api/v1/system/profiles to impose (or remove) a top-level profile.
PUT or POST to /api/v1/system/profiles. Supplying an empty value unsets the top-level profile.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>Sets the top level call leg profile to the one with the specified ID.</td>
</tr>
<tr>
<td>callProfile</td>
<td>ID</td>
<td>Sets the top level call profile to the one specified</td>
</tr>
<tr>
<td>dtmfProfile</td>
<td>ID</td>
<td>Sets the top level DTMF profile to the one specified</td>
</tr>
<tr>
<td>userProfile</td>
<td>ID</td>
<td>Sets the top level user profile to the one specified</td>
</tr>
<tr>
<td>ivrBrandingProfile</td>
<td>ID</td>
<td>Sets the top level IVR branding profile to the one specified</td>
</tr>
<tr>
<td>callBrandingProfile</td>
<td>ID</td>
<td>Sets the top level call branding profile to the one specified</td>
</tr>
<tr>
<td>compatibilityProfile</td>
<td>ID</td>
<td>Sets the top level compatibility profile to the one specified (2.1 onwards)</td>
</tr>
</tbody>
</table>

10.6 Licensing Methods

10.6.1 Retrieving information on licensing

GET method performed on the “/system/licensing” node. See Appendix Appendix B.1 for an example of the structure for the information.

<table>
<thead>
<tr>
<th></th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>features</td>
<td></td>
<td>If licensing is enabled then the &lt;features&gt; element includes the elements on the left</td>
</tr>
<tr>
<td>callBridge status</td>
<td>noLicense</td>
<td>activated</td>
</tr>
<tr>
<td></td>
<td>String</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>webBridge status</td>
<td>noLicense</td>
<td>activated</td>
</tr>
<tr>
<td></td>
<td>String</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>turn noLicense</td>
<td>activated</td>
<td>expired</td>
</tr>
<tr>
<td></td>
<td>String</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>ldap</td>
<td>status</td>
<td>No license applied to the LDAP server</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License applied and LDAP server activated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License expired, now in grace period for license renewal</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>branding</td>
<td>status</td>
<td>No license applied for branding</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License applied and branding activated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License expired, now in grace period for license renewal</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>recording</td>
<td>status</td>
<td>No license applied to the Recorder</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License applied and Recorder activated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License expired, now in grace period for license renewal</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>streaming</td>
<td>status</td>
<td>No license applied to the Streamer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License applied and Streamer activated</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>personal</td>
<td>status</td>
<td>No Personal Multiparty license applied</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Personal Multiparty license activated</td>
</tr>
<tr>
<td></td>
<td>expired</td>
<td>Personal Multiparty license expired</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>shared</td>
<td>status</td>
<td>No Shared Multiparty license applied</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shared Multiparty license activated</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Shared Multiparty license expired</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>capacityUnits</td>
<td>status</td>
<td>No license for Capacity Units applied</td>
</tr>
<tr>
<td></td>
<td></td>
<td>License for Capacity Units activated</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>License expired, now in grace period for license renewal</td>
</tr>
<tr>
<td></td>
<td>expiry</td>
<td>License for Capacity Units expired</td>
</tr>
</tbody>
</table>
### Response elements

<table>
<thead>
<tr>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>expiry</td>
<td>Date of expiry</td>
</tr>
<tr>
<td>limit</td>
<td></td>
</tr>
</tbody>
</table>

GET method performed on the “/system/multipartyLicensing” node. See Appendix B.2 for an example of the structure for the information.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>string</td>
<td>UTC time at which the report was generated</td>
</tr>
<tr>
<td>personalLicenseLimit</td>
<td>Number</td>
<td>Number of available personal licenses</td>
</tr>
<tr>
<td>sharedLicenseLimit</td>
<td>Number</td>
<td>Number of available shared licenses</td>
</tr>
<tr>
<td>capacityUnitLimit</td>
<td>Number</td>
<td>Number of available capacity units</td>
</tr>
<tr>
<td>users</td>
<td>Number</td>
<td>Number of non-guest users on the system</td>
</tr>
<tr>
<td>personalLicenses</td>
<td>Number</td>
<td>Number of personal licenses assigned to users</td>
</tr>
<tr>
<td>participantsActive</td>
<td>Number</td>
<td>Number of active participants</td>
</tr>
<tr>
<td>callsActive</td>
<td>Number</td>
<td>Number of active calls</td>
</tr>
<tr>
<td>weightedCallsActive</td>
<td>Number</td>
<td>Number of weighted active calls (see note below).</td>
</tr>
<tr>
<td>capacityUnitUsage</td>
<td>Number</td>
<td>Number of capacity units in use</td>
</tr>
<tr>
<td>callsWithoutPersonalLicense</td>
<td>Number</td>
<td>Number of calls without a personal license</td>
</tr>
<tr>
<td>weightedCallsWithoutPersonalLicense</td>
<td>Number</td>
<td>Number of weighted calls without a personal license (see note below).</td>
</tr>
<tr>
<td>capacityUnitUsageWithoutPersonalLicense</td>
<td>Number</td>
<td>Number of capacity units in use in calls without a personal license</td>
</tr>
</tbody>
</table>

GET method performed on the “/system/multipartyLicensing/activePersonalLicenses”.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve active personal license entries other than those in the first &quot;page&quot; of the notional list</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>
### 10.7 TURN Server Methods

**Note:** The TURN Server is not available on the Cisco Meeting Server 2000. It is more suited to the lower capacity Cisco Meeting Server 1000 and specification-based VM servers.

### 10.7.1 Retrieving Information on TURN Servers

GET method performed on the "/turnServers" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those TURN servers that match the filter.</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

### Response elements

| Description/Notes | Type/Value | |
|-------------------|------------| |
| Response is structured as a top-level <activePersonalLicenses total="N" > tag with potentially multiple <user> elements within it. Each <user> tag may include the following elements: | callsActive | Number |
| Number of active calls that use the license of this user | weightedCallsActive | Number |
| Number of weighted active calls that use the license of this user (see note below). | |

**Note:** the sum of weighted calls across a cluster matches the number of distinct calls on the cluster. For example, if CMS1 shows 3 callsActive and 2 weightedCallsActive, and CMS2 shows 2 callsActive and 1 weightedCallsActive, then there are 3 conferences in total on the cluster and 3 licenses are required.
10.7.2 Setting up and modifying TURN servers

- Creating: POST method to the /turnServers" node
- Modifying: PUT to " /turnServers/<turn server id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverAddress</td>
<td>String</td>
<td>The address for the Call Bridge to use to reach this TURN server</td>
</tr>
<tr>
<td>clientAddress</td>
<td>String</td>
<td>The address that Cisco Meeting Apps should use to reach this TURN Server</td>
</tr>
<tr>
<td>username</td>
<td>String</td>
<td>The username to use when making allocations on this TURN Server</td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td>The password to use when making allocations on this TURN Server</td>
</tr>
<tr>
<td>type</td>
<td>acano</td>
<td>cms</td>
</tr>
<tr>
<td>numRegistrations</td>
<td>Number</td>
<td>The number of registrations that should be made to this TURN Server. This parameter is only meaningful for configured Lync Edge servers</td>
</tr>
<tr>
<td>tcpPortNumberOverride</td>
<td>Number</td>
<td>An optional override for the port number to use when using this TURN Server for TCP media (for example, Lync presentation call legs). This parameter is not needed for configured Lync Edge servers, where the TCP port number can always be determined automatically.</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>If specified, associate this TURN server with the supplied Call Bridge (from version 2.1)</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>If specified, associate this TURN server with the supplied Call Bridge group (from version 2.1)</td>
</tr>
</tbody>
</table>

10.7.3 Retrieving detailed information about an individual TURN server

GET method performed on a “/turnServers/<turn server id>" node. If the turn server ID supplied is valid, a “200 OK” response is received, with XML content:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverAddress</td>
<td>String</td>
<td>The address on which the Call Bridge reaches this TURN server</td>
</tr>
<tr>
<td>Response elements</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>clientAddress</td>
<td>String</td>
<td>The address on which Cisco Meeting Apps reach TURN Server</td>
</tr>
<tr>
<td>username</td>
<td>String</td>
<td>The username to use when making allocations on this TURN Server</td>
</tr>
<tr>
<td>type</td>
<td>acano</td>
<td>cms</td>
</tr>
<tr>
<td>numRegistrations</td>
<td>Number</td>
<td>For configured Lync Edge servers only the number of registrations made to this TURN Server.</td>
</tr>
<tr>
<td>tcpPortNumberOverride</td>
<td>Number</td>
<td>The port number used when using TURN server for TCP media.</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>If specified, this is the Call Bridge associated with this TURN server.</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>If specified, this is the Call Bridge Group associated with this TURN server.</td>
</tr>
</tbody>
</table>

10.7.4 Retrieving individual TURN Server status

GET method performed on a “/turnServers/<turn server id>/status” node. If the turn server ID supplied is valid, a “200 OK” response is received, with XML content:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Zero, one or more &lt;host&gt; child nodes, with each including the following elements:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>portNumber</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>reachable</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>roundTripTimeMs</td>
<td>Number</td>
<td>If this TURN server is reachable, the round trip time (in milliseconds) of the Call Bridge’s path to it</td>
</tr>
<tr>
<td>mappedAddress</td>
<td>String</td>
<td>If populated, indicates the source IP and source port that the TURN server saw the STUN binding request coming from when the Call Bridge performed TURN Server reachability checks. This can be different to the IP address of the Call Bridge in deployments where there is a NAT between the Call Bridge and the TURN Server.</td>
</tr>
<tr>
<td>mappedPortNumber</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>
## 10.8 Web Bridge Methods

### 10.8.1 Retrieving Information on Web Bridges

GET method performed on the "/webBridges" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those Web Bridges that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those Web Bridges associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>URL</td>
<td>Response is structured as a top-level &lt;webBridges total=&quot;N&quot;&gt; tag with potentially multiple &lt;webBridge&gt; elements within it. Each &lt;webBridge&gt; tag may include the following elements. See the next section</td>
</tr>
<tr>
<td>resourceArchive</td>
<td>URL</td>
<td></td>
</tr>
</tbody>
</table>

### 10.8.2 Setting Up and Modifying a Web Bridge

- Creating: POST method to the "/webBridges" node
- Modifying: PUT to "/webBridges/<web bridge id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>URL</td>
<td>The address for the Call Bridge to use to reach this Web Bridge</td>
</tr>
<tr>
<td>resourceArchive</td>
<td>URL</td>
<td>The address of any customization archive file for the Call Bridge to use for the background image and logo branding on the login page for this Web Bridge Note: When specifying the path any port value other than :80 for http and :443 for https is considered invalid.</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If you supply the ID for a tenant to associate with this Web Bridge, only call IDs for coSpaces owned by that tenant can be joined through it</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tenantGroup</td>
<td>ID</td>
<td>Only coSpaces associated with tenants within the specified tenant group can be accessed by call ID through this web bridge. If no tenant group is supplied, only coSpaces without a tenant, or associated with a tenant in no tenant group, can be accessed by call ID.</td>
</tr>
</tbody>
</table>
| idEntryMode         | disabled|secure|legacy | Controls coSpace access via call ID and passcode;  
  - when set to ‘disabled’, access via call ID and passcode is disabled;  
  - when set to ‘secure’ both the call ID and the passcode must be specified to look up and join a coSpace.  
  - when set to ‘legacy’ only a call ID need be specified to look up a coSpace.  
  If this parameter is not supplied in a create (POST) operation, it defaults to secure. |
| allowWeblinkAccess  | true|false | Whether this Web Bridge will allow users to access coSpace (and coSpace access methods) as guests by following a weblink. If this parameter is not supplied in a create (POST) operation, it defaults to true. |
| showSignIn          | true|false | Whether this Web Bridge will display the sign in tab on the index page. If this parameter is not supplied in a create (POST) operation, it defaults to true. |
| resolveCoSpaceCallIds | true|false | Whether this Web Bridge should accept coSpace and coSpace access method call IDs for the purpose of allowing visitors to join coSpaces. If this parameter is not supplied in a create (POST) operation, it defaults to true. |
| resolveLyncConferenceIds | true|false | Whether this Web Bridge will accept IDs to be resolved to Lync scheduled conference IDs. If this parameter is not supplied in a create (POST) operation, it defaults to false. |
| callBridge          | ID         | If specified, associate this Web bridge with the supplied Call Bridge (from version 2.1)                                                      |
| callBridgeGroup     | ID         | If specified, associate this Web bridge with the supplied Call Bridge group (from version 2.1)                                                |

### 10.8.3 Retrieving detailed information about an individual Web Bridge

GET method performed on a “/webBridges/<web bridge id>” node. If the web bridge ID supplied is valid, a “200 OK” response is received, with XML content described in the previous section.
10.8.4 Updating the Web Bridge customization

A POST to "/webBridges/<web bridge id>/updateCustomization" node causes any configured customization archive for the specified Web Bridge to be re-retrieved and pushed to that Web Bridge. For example, this allows the contents of a customization archive to be changed, and for those changes to take effect without needing to restart either the Call Bridge or the Web Bridge.

10.8.5 Retrieving diagnostics on a Web Bridge (from 2.2)

GET method performed on a "/webBridges/<web bridge id>/status" node. If the web bridge ID supplied is valid, a "200 OK" response is received, with XML content matching the table below.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>unused</td>
<td>success</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10.9 Call Bridge Methods

10.9.1 Retrieving Information on Call Bridges

GET method performed on the "/callBridges" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>See the next section</td>
</tr>
</tbody>
</table>

10.9.2 Setting Up and Modifying a Call Bridge

- Creating: POST method to the "/callBridges" node
- Modifying: PUT to "/callBridges/<call bridge id>"
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name *</td>
<td>String</td>
<td>The unique name for this configured clustered Call Bridge</td>
</tr>
<tr>
<td>address</td>
<td>String</td>
<td>The address at which this Call Bridge in the cluster can be reached</td>
</tr>
<tr>
<td>sipDomain</td>
<td>String</td>
<td>The SIP domain to use for establishing peer-to-peer links with this clustered Call Bridge</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>If specified, associate this Call Bridge with the supplied Call Bridge group (from version 2.1)</td>
</tr>
</tbody>
</table>

### 10.9.3 Retrieving detailed information about an individual Call Bridge

GET method performed on a “/callBridges/<call bridge id>” node. If the call bridge ID supplied is valid, a “200 OK” response is received, with XML content described in the previous section.

### 10.10 Call Bridge Group Methods

#### 10.10.1 Retrieving Information on Call Bridge Groups

GET method performed on the “/callBridgeGroups” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Name of Call Bridge group</td>
</tr>
</tbody>
</table>

#### 10.10.2 Setting Up and Modifying a Call Bridge Group

- Creating: POST method to the " /callBridgeGroups" node
- Modifying: PUT to " /callBridgeGroups/<call bridge group id>"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Optional name of the Call Bridge group</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>loadBalancingEnabled</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>loadBalanceLyncCalls</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>loadBalanceOutgoingCalls</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

### 10.10.3 Retrieving detailed information about an individual Call Bridge Group

GET method performed on a "/callBridgeGroups/<call bridge group id>" node. If the Call Bridge group ID supplied is valid, a “200 OK” response is received, with XML content described in the previous section.

### 10.11 XMPP Methods

#### 10.11.1 Retrieving the XMPP server details

Issue a GET on the new /system/configuration/xmpp node to retrieve the information below.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uniqueName</td>
<td>String</td>
<td>The name by which this Call Bridge is known by the XMPP server</td>
</tr>
<tr>
<td>domain</td>
<td>String</td>
<td>The domain that the Call Bridge is using for XMPP</td>
</tr>
<tr>
<td>serverAddressOverride</td>
<td>String</td>
<td>If supplied, the Call Bridge is connecting to an XMPP server at the specified address rather than using the &quot;domain&quot; to discover it (via DNS).</td>
</tr>
</tbody>
</table>

#### 10.11.2 Setting Up and Modifying an XMPP server

Issue a PUT or a POST on the new /system/configuration/xmpp node

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uniqueName</td>
<td>String</td>
<td>The name by which this Call Bridge should be known by the XMPP server</td>
</tr>
<tr>
<td>domain</td>
<td>String</td>
<td>The domain that the Call Bridge should use for XMPP</td>
</tr>
</tbody>
</table>
### 10.12 Call Bridge Cluster Methods

#### 10.12.1 Retrieving the Call Bridge Cluster details

Issue a GET on the `/system/configuration/cluster` node to retrieve the information below.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uniqueName</td>
<td>String</td>
<td>The name by which this call bridge is known within the call bridge cluster; this should match the &quot;name&quot; value for its entry in the /callBridges table</td>
</tr>
<tr>
<td>peerLinkBitRate</td>
<td>Number</td>
<td>The maximum media bit rate specified to use for call distribution connections between call bridges</td>
</tr>
<tr>
<td>participantLimit</td>
<td>Number</td>
<td>If supplied, the maximum number of participants allowed to be active on this Call Bridge; when this limit is reached, new incoming SIP calls will be rejected</td>
</tr>
<tr>
<td>loadLimit</td>
<td>Number</td>
<td>If supplied, the maximum number of load units to be used on this Call Bridge (from version 2.1)</td>
</tr>
<tr>
<td>newConferenceLoadLimitBasisPoints</td>
<td>Number</td>
<td>Basis points (1 in 10,000) of the load limit at which incoming calls to non-active conferences will be disfavored, ranges from 0 to 10000, defaults to 5000 (50% load). Value is scaled relative to load limit. (From version 2.1)</td>
</tr>
<tr>
<td>existingConferenceLoadLimitBasisPoints</td>
<td>Number</td>
<td>Basis points of load limit at which incoming calls to this Call Bridge will be rejected, ranges from 0 to 10000, defaults to 8000 (from version 2.1)</td>
</tr>
<tr>
<td>maxPeerVideoStreams</td>
<td>Number</td>
<td>The maximum number of streams sent over a call distribution connection between Call Bridges, defaults to 4 if not supplied. (from version 2.3.3)</td>
</tr>
</tbody>
</table>

#### 10.12.2 Setting Up and Modifying the Call Bridge Cluster

Issue a PUT or a POST on the `/system/configuration/cluster` node
### Response elements

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uniqueName</td>
<td>String</td>
<td>The name by which this call bridge is known within the call bridge cluster; this should match the &quot;name&quot; value for its entry in the /callBridges table</td>
</tr>
<tr>
<td>peerLinkBitRate</td>
<td>Number</td>
<td>If supplied, the maximum media bit rate to use for call distribution connections between call bridges</td>
</tr>
<tr>
<td>participantLimit</td>
<td>Number</td>
<td>If supplied, the maximum number of participants allowed to be active on this Call Bridge; when this limit is reached, new incoming SIP calls will be rejected</td>
</tr>
<tr>
<td>loadLimit</td>
<td>Number</td>
<td>If supplied, the maximum number of load units to be used on this Call Bridge (from version 2.1)</td>
</tr>
<tr>
<td>newConferenceLoadLimitBasisPoints</td>
<td>Number</td>
<td>Basis points (1 in 10,000) of the load limit at which incoming calls to non-active conferences will be disfavored, ranges from 0 to 10000, defaults to 5000 (50% load). Value is scaled relative to load limit. (From version 2.1).</td>
</tr>
<tr>
<td>existingConferenceLoadLimitBasisPoints</td>
<td>Number</td>
<td>Basis points of the load limit at which incoming calls to non-active conferences will be rejected, ranges from 0 to 10000, defaults to 5000 (from version 2.1).</td>
</tr>
</tbody>
</table>

### 10.13 Recorder methods

**Note:** The Recorder is not available on the Cisco Meeting Server 2000. It is more suited to the lower capacity Cisco Meeting Server 1000 and specification-based VM servers.

**Note:** At the end of recording a meeting, the recording is automatically converted to MP4. The converted file is suitable for placing within a document storage/distribution system, for example, in a network file system (NFS) they are stored in the NFS folder spaces/ &lt;space ID&gt;; tenant spaces are stored in tenants/ &lt;tenant ID&gt;/spaces/ &lt;space ID&gt;.

#### 10.13.1 Retrieving recorder operations

GET method performed on the “/recorders” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve recorders other than those in the first &quot;page&quot; of the notional list.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>
Response elements | Type/Value | Description/Notes
--- | --- | ---
url | URL | The address that the Call Bridge uses to reach this recorder
CallBridge | ID | The ID of the Call Bridge associated with this recorder (from version 2.1)
CallBridgeGroup | ID | The ID of the Call Bridge group associated with this recorder (from version 2.1)

10.13.2 Setting up and modifying recording operations
- Creating: POST method to the /recorders node
- Modifying: PUT to "recorders/<recorder id>"

Parameters | Type/Value | Description/Notes
--- | --- | ---
url | URL | The HTTP or HTTPS URL address that the Call Bridge should use to reach this recorder.
CallBridge | ID | If specified, associate this recorder with the supplied Call Bridge (from version 2.1)
CallBridgeGroup | ID | If specified, associate this recorder with the supplied Call Bridge group (from version 2.1)

10.13.3 Retrieving detailed information about an individual recording node
GET method performed on a "recorders/<recorder id>" node. If the recorder ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

10.13.4 Retrieving diagnostics on a Recorder (from 2.2)
GET method performed on a "recorders/<recorder id>/status" node. If the recorder ID supplied is valid, a “200 OK” response is received, with XML content matching the table below.

Response elements | Type/Value | Description/Notes
--- | --- | ---
status | unused|success|invalidAddress|dnsFailure|connectionFailure| The Recorder is not used by the queried Call Bridge
The Recorder is connected to the queried Call Bridge
The configured Recorder URL is invalid
The configured Recorder URL could not be resolved
The Recorder could not connect to the queried Call Bridge
### 10.14 Streamer methods

**Note:** The Streamer is not available on the Cisco Meeting Server 2000. It is more suited to the lower capacity Cisco Meeting Server 1000 and specification–based VM servers.

#### 10.14.1 Retrieving streamer operations

GET method performed on the “/streamers” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve streamers other than those in the first &quot;page&quot; of the notional list.</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>URL</td>
<td>The address that the Call Bridge uses to reach this streamer</td>
</tr>
<tr>
<td>callBridge</td>
<td>ID</td>
<td>The ID of the Call Bridge associated with this streamer</td>
</tr>
<tr>
<td>callBridgeGroup</td>
<td>ID</td>
<td>The ID of the Call Bridge group associated with this streamer</td>
</tr>
</tbody>
</table>

**Response elements**

- **remoteFailure**: A connection was established with the Recorder but received a failure response
- **unknownFailure**: An unknown failure occurred
- **lowDiskSpace**: The Recorder has limited disk space available
- **activeRecordings**: Total number of active recordings on this Recorder

#### 10.14.2 Setting up and modifying streaming operations

- Creating: POST method to the /streamers " node
- Modifying: PUT to " /streamers/<streamer id>"
Parameters | Type/Value | Description/Notes
--- | --- | ---
url | URL | The HTTP or HTTPS URL address that the Call Bridge should use to reach this streamer.
callBridge | ID | If specified, associate this streamer with the supplied Call Bridge
callBridgeGroup | ID | If specified, associate this streamer with the supplied Call Bridge group

10.14.3 Retrieving detailed information about an individual streaming node

GET method performed on a “/streamers/<streamer id>” node. If the streamer ID supplied is valid, a “200 OK” response is received, with XML content matching the section above.

10.14.4 Retrieving diagnostics on a Streamer (from 2.2)

GET method performed on a “/streamers/<streamer id>/status” node. If the streamer ID supplied is valid, a “200 OK” response is received, with XML content matching the table below.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>unused</td>
<td>The Streamer is not used by the queried Call Bridge</td>
</tr>
<tr>
<td></td>
<td>success</td>
<td>The Streamer is connected to the queried Call Bridge</td>
</tr>
<tr>
<td></td>
<td>invalidAddress</td>
<td>The configured Streamer URL is invalid</td>
</tr>
<tr>
<td></td>
<td>dnsFailure</td>
<td>The configured Streamer URL could not be resolved</td>
</tr>
<tr>
<td></td>
<td>connectionFailure</td>
<td>The Streamer could not connect to the queried Call Bridge</td>
</tr>
<tr>
<td></td>
<td>remoteFailure</td>
<td>A connection was established with the Streamer but received a failure response</td>
</tr>
<tr>
<td></td>
<td>unknownFailure</td>
<td>An unknown failure occurred</td>
</tr>
<tr>
<td>activeStreams</td>
<td>Number</td>
<td>Total number of active streams on this Streamer</td>
</tr>
</tbody>
</table>

10.15 System Load Method

GET method performed on the “/system/load” node.

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>mediaProcessingLoad</td>
<td>Number</td>
<td>Current media processing load on the Call Bridge</td>
</tr>
</tbody>
</table>
10.16 Compatibility Profile Methods

10.16.1 Retrieving compatibility profile operations

GET method performed on the “/compatibilityProfiles” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An “offset” and “limit” can be supplied to retrieve compatibility profiles other</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td>than those in the first “page” of the notional list.</td>
</tr>
<tr>
<td>usageFilter</td>
<td>unreferenced</td>
<td>Supply “usageFilter=unreferenced” in the request to retrieve only those comp-</td>
</tr>
<tr>
<td></td>
<td>referenced</td>
<td>atibility profiles that are not referenced by another object. This is a useful</td>
</tr>
<tr>
<td></td>
<td></td>
<td>check before deleting the profile. To retrieve just those compatibility profiles</td>
</tr>
<tr>
<td></td>
<td></td>
<td>which are referenced in at least one place, you can supply “usageFilter=referred”</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Response is structured as a top-level &lt;compatibilityProfiles total=&quot;N&quot; &gt; tag with</td>
</tr>
<tr>
<td>sipUdt</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipMultistream</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>sipMediaPayloadTypeMode</td>
<td>auto</td>
<td>broadsoft</td>
</tr>
</tbody>
</table>

10.16.2 Setting up and modifying compatibility profile operations

- Creating: POST method to the " /compatibilityProfiles" node
- Modifying: PUT to " /compatibilityProfiles/<compatibility profile id>"
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
</table>
| sipUdt              | true/false | Controls whether use of UDT is allowed within SIP calls. Active Control uses UDT transport for certain features, for example sending roster lists to endpoints, allowing users to disconnect other participants while in call, and inter-deployment participant lists. (From version 2.1)  
  true - UDT is allowed within SIP calls  
  false - UDT is not allowed within SIP calls  |
| sipMultistream      | true/false | Controls whether use of Cisco multistream protocols is allowed within SIP calls. The dual video feature for Cisco dual endpoints uses this protocol. If this is disabled, then no calls will be able to use dual screen video. (From version 2.2.3)  
  true - Cisco multistream signalling is allowed within SIP calls  
  false - Cisco multistream signalling is not allowed within SIP calls  |
| sipMediaPayloadTypeMode | auto/broadsoft | Controls whether the default codec media payload types are used, or a special variant. (From version 2.2)  
  auto - the default mode, where the normal media payload type values are used  
  broadsoft - a special exception mode, where the H.264 video codec is advertised with payload type 109. |

### 10.17 System Diagnostics Methods

#### 10.17.1 Retrieving system diagnostics

Issue a GET on the new `/system/diagnostics` node to retrieve the information below.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number/Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list.</td>
</tr>
<tr>
<td>coSpaceFilter</td>
<td>ID</td>
<td>If supplied, this filter restricts results returned to those diagnostics that correspond to the specified coSpace</td>
</tr>
<tr>
<td>callCorrelatorFilter</td>
<td>ID</td>
<td>If supplied, this filter restricts results returned to those diagnostics that correspond to the specified callCorrelator</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>Text description associated with the specified diagnostics log</td>
</tr>
</tbody>
</table>
10.17.2 Retrieving an individual system diagnostic

Issue a GET on the /system/diagnostics/<diagnostics ID> node

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String</td>
<td>Text description associated with the specific diagnostics log</td>
</tr>
<tr>
<td>coSpace</td>
<td>ID</td>
<td>If the diagnostics log is associated with a specific coSpace, this parameter holds the ID of that coSpace</td>
</tr>
<tr>
<td>callCorrelator</td>
<td>ID</td>
<td>An ID that is the same across all distributed instances of the active call - this value will be the same for other Call Bridge peers' diagnostics file for the same call</td>
</tr>
<tr>
<td>timestamp</td>
<td>String</td>
<td>The time at which the diagnostics log was generated</td>
</tr>
<tr>
<td>contentsSize</td>
<td>Number</td>
<td>The size of the diagnostics data for this log entry</td>
</tr>
</tbody>
</table>

10.17.3 Retrieving the content of an individual system diagnostic

Issue a GET method on the /system/diagnostics/<diagnostics id>/contents node to retrieve the data stored in the system diagnostic.
11 LDAP Methods

Objects in the hierarchy that reside in the “/ldapMappings”, “/ldapServers” and “/ldapSources” nodes in the object tree relate to the Meeting Server’s interaction with one or more LDAP servers (for instance, Active Directory) which are used to import user accounts to the Meeting Server.

- One or more LDAP servers should be configured, with each one having associated username and password information for the Meeting Server to use to connect to it for the purpose of retrieving user account information from it.

- One or more LDAP mappings are also required, which define the form of the user account names which will be added to the system when users are imported from configured LDAP servers.

- A set of LDAP sources then need to be configured, which tie together configured LDAP servers and LDAP mappings, along with parameters of its own, which correspond to the actual import of a set of users.

An LDAP source takes an LDAP server / LDAP mapping combination and imports a filtered set of users from that LDAP server. This filter is determined by the LDAP source’s "baseDn" (the node of the LDAP server’s tree under which the users can be found) and a filter to ensure that user accounts are only created for LDAP objects that match a specific pattern.

The API LDAP methods allow multiple additional sets of “Active Directory Configuration” as per the Web Admin Interface Configuration > Active Directory page. On this page, the Active Directory Server Settings section corresponds to an API-configured LDAP server, the Import Settings to an LDAP source, and the Field Mapping Expressions to an LDAP mapping.

**Note:** The LDAP server credentials are used to read the following fields (for security reasons you may want to restrict the fields and permissions available using those credentials):

- mail
- objectGUID
- entryUUID
- nsuniqueid
- telephoneNumber
- mobile
- sn
- givenName
11.1 LDAP Server Methods

Figure 2: Outline LDAP process

![Outline LDAP process diagram](image)

11.1.1 Retrieving Information on LDAP Servers

GET method performed on the “/ldapServers” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those LDAP servers that match the filter.</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response elements</td>
<td></td>
<td>Response is structured as a top-level &lt;ldapServers total=&quot;N&quot;&gt; tag with potentially multiple &quot;&lt;ldapServer&gt;&quot; elements within it. &quot;&lt;ldapServer&gt;&quot; elements returned follow the general form on the left.</td>
</tr>
</tbody>
</table>
11.1.2 Adding and modifying an LDAP Server

- Create: POST method performed on the “/ldapServers” node. If the LDAP server is configured on the system successfully, its ID is returned in the “Location” field of the response header.
- Modifying an LDAP server is a PUT method on a “/ldapServers/<ldapServer id>” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>address *</td>
<td>String</td>
<td>The address of the LDAP server to connect to.</td>
</tr>
<tr>
<td>portNumber *</td>
<td>Number</td>
<td>The TCP or TLS port number to connect to on the remote LDAP server.</td>
</tr>
<tr>
<td>username</td>
<td>String</td>
<td>The username to use when retrieving information from the LDAP server.</td>
</tr>
<tr>
<td>password</td>
<td>String</td>
<td>The password of the account associated with username.</td>
</tr>
<tr>
<td>secure *</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>usePagedResults</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

11.1.3 Retrieving detailed information about an individual LDAP Server

GET method performed on a “/ldapServers/<ldapServer ID>” node. If the ldapServer ID supplied is valid, a “200 OK” response is received, with XML content:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldapServer id</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>address</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>portNumber</td>
<td>Number</td>
<td>Domain name</td>
</tr>
<tr>
<td>username</td>
<td>String</td>
<td>directoryUser</td>
</tr>
<tr>
<td>secure</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>
11.2 LDAP Mapping Methods

11.2.1 Adding and modifying an LDAP Mapping

- Creating: POST method to the “/ldapMappings” If the LDAP mapping is configured on the system successfully, its ID is returned in the “Location” field of the response header.
- Modifying: PUT method on a “/ldapMappings/ <ldapMapping id>” node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>jidMapping</td>
<td>String</td>
<td>The template for generating user JIDs from the associated LDAP server's entries, for instance $sAMAccountName$@example.com. Note: user JIDs generated by jidMapping are also used as URIs so must be unique and not the same as any URI or call ID.</td>
</tr>
<tr>
<td>nameMapping</td>
<td>String</td>
<td>The template for generating user names from the associated LDAP server's entries; for instance &quot;$cn$&quot; to use the common name.</td>
</tr>
<tr>
<td>cdrTagMapping</td>
<td>String</td>
<td>The template for generating a users' cdrTag value. Can be set either to a fixed value or be constructed from other LDAP fields for that user. The user's cdrTag is used in callLegStart CDRs. See the Cisco Meeting Server CDR Reference for details.</td>
</tr>
<tr>
<td>authenticationIdMapping</td>
<td>String</td>
<td>The template for generating authentication IDs from the associated LDAP server’s entries, for instance &quot;$userPrincipalName$&quot;.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Type/Value</td>
<td>Description/Notes</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>coSpaceUriMapping</td>
<td>String</td>
<td>If these parameters are supplied, they ensure that each user account generated by this LDAP mapping has an associated personal coSpace. The user is automatically added as a member of the coSpace, with permissions defined above. In order for that coSpace to be set up as required, these parameters provide the template for setting the coSpaces’ URI, displayed name and configured Call ID. For example, setting coSpaceNameMapping to “$cn$ personal coSpace” ensures that each user’s coSpace is labelled with their name followed by “personal coSpace”.</td>
</tr>
<tr>
<td>coSpaceSecondaryUriMapping</td>
<td>String</td>
<td>Note that the generated coSpace will have its own cdrTag – and it will be the same as the user’s cdrTag and cannot be changed other than by changing the cdrTagMapping above and re-syncing. (The coSpace's cdrTag is used in the callStart CDR. See the Cisco Meeting Server CDR Reference for details.)</td>
</tr>
<tr>
<td>coSpaceNameMapping</td>
<td>String</td>
<td>Note that the normal uniqueness rules apply to the URI and Call IDs of coSpaces set up in this way: it is not valid to have the same URI or Call ID for more than one coSpace set up by a given LDAP mapping, nor is it valid for such a coSpace URI or Call ID to be the same as one currently in use elsewhere on the Meeting Server. Note: user JIDs generated by jidMapping are also used as URIs so must be unique and not the same as any URI or call ID.</td>
</tr>
<tr>
<td>coSpaceCallIdMapping</td>
<td>String</td>
<td></td>
</tr>
</tbody>
</table>

### 11.2.2 Secondary LDAP Mapping parameter

Per LDAP mapping, there is a new optional coSpaceSecondaryUriMapping parameter so that the coSpaces that are created automatically have a secondary URI.

- When creating an LDAP mapping (see the previous section) or modifying the configuration of an existing LDAP mapping you can supply a " coSpaceSecondaryUriMapping" parameter
- When retrieving information on an individual LDAP mapping (a GET method on a " /ldapMappings/<LDAP mapping ID>" node) the coSpaceSecondaryUriMapping value will be returned if it is defined for that LDAP mapping

### 11.2.3 Retrieving information on LDAP Mappings

GET method performed on the “/ldapMappings” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those LDAP mappings that match the filter</td>
</tr>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
</tbody>
</table>
11.2.4 Retrieving detailed Information about an individual LDAP Mapping

GET method performed on a “/IdapMappings/<ldapMapping ID>” node. If the ldapMapping ID supplied is valid, a “200 OK” response is received, with XML content described in the creating

11.3 LDAP Source Methods

11.3.1 Retrieving Information on LDAP Sources

GET method performed on the “/IdapSources” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>String</td>
<td>Supply filter=&lt;string&gt; in the URI to return just those LDAP sources that match the filter.</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those LDAP sources associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldapSources</td>
<td></td>
<td>Response is structured as a top-level &lt;ldapSources total=&quot;N&quot;&gt; tag with potentially multiple &quot;&lt;ldapSource&gt;&quot; elements within it. &quot;&lt;ldapSource&gt;&quot; elements returned follow the general form on the left.</td>
</tr>
</tbody>
</table>
## 11.3.2 Adding and modifying an LDAP Source

- **Creating:** POST method to the “/ldapSources” node. If the LDAP source is configured on the system successfully, its ID is returned in the “Location” field of the response header.
- **Modifying:** PUT method on a “/ldapSources/<ldapSource id>” node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>server *</td>
<td>ID</td>
<td>The ID of a previously-configured LDAP server (see above)</td>
</tr>
<tr>
<td>mapping *</td>
<td>ID</td>
<td>The ID of a previously-configured LDAP mapping (see above)</td>
</tr>
<tr>
<td>baseDn *</td>
<td>String</td>
<td>The distinguished name of the node in the LDAP server’s tree from which users should be imported, for instance “cn=Users,dc=&lt;companyname&gt;,dc=com”</td>
</tr>
<tr>
<td>filter</td>
<td>String</td>
<td>An LDAP filter string that records must satisfy in order to be imported as users, for instance “(objectClass=person)”</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If supplied, the ID for the tenant to which the LDAP source should be associated. Users imported with this LDAP source will be associated with that tenant</td>
</tr>
<tr>
<td>userProfile</td>
<td>ID</td>
<td>If supplied, this is the ID of the user profile to associate with users imported via this LDAP source. This parameter is present from version 2.0 onwards.</td>
</tr>
</tbody>
</table>

## 11.3.3 Retrieving detailed information on a LDAP Source

GET method performed on a “/ldapSources/<ldapSource ID>” node. If the ldapSource ID supplied is valid, a “200 OK” response is received, with XML content as per LDAP source creation described above.

## 11.4 LDAP Sync Methods

API support for LDAP synchronization comprises the ability to:

- Trigger a new sync via the API
- Monitor pending and in-progress LDAP syncs
There is a top-level /ldapSyncs node in the object tree, and associated GET, DELETE, POST methods to use on objects underneath it.

11.4.1 Retrieving scheduled LDAP sync methods

GET method on the " /ldapSyncs" node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see Section 4.1.2).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldapSyncid</td>
<td>ID</td>
<td>The current status of this LDAP sync operation: inProgress - this LDAP sync operation is happening now pending - this LDAP sync operation has yet to start complete - this LDAP sync operation has completed successfully failed - this LDAP sync operation has failed</td>
</tr>
<tr>
<td>state</td>
<td>inProgress</td>
<td>pending</td>
</tr>
<tr>
<td>failureReason</td>
<td>tenantDoesNotExist</td>
<td>ldapSourceDoesNotExist</td>
</tr>
<tr>
<td>numUsersImported</td>
<td>Number</td>
<td>The number of users imported so far for an in-progress LDAP sync</td>
</tr>
<tr>
<td>numLdapSourcesComplete</td>
<td>Number</td>
<td>The number of LDAP sources for which the sync method has been completed for an in-progress LDAP sync of multiple LDAP sources. However if the first LDAP source synchronization is still in progress so that numLdapSourcesComplete=0, then the parameter is omitted</td>
</tr>
</tbody>
</table>

11.4.2 Initiating a new LDAP sync

POST method on the " /ldapSyncs" node. If neither parameter in the following table is included, the sync is equivalent to the Sync now button on the Web Admin Interface Configuration > Active Directory page.
11.4.3 Cancelling a scheduled LDAP sync

DELETE method on a " /ldapSyncs/<LDAP sync ID>" node. This method cancels a scheduled LDAP sync. This method will fail if the sync method has already started (or started and completed).

11.4.4 Retrieving information on a single LDAP sync method

GET method on an " /ldapSyncs/<LDAP sync ID>" node.

If the LDAP sync ID provided is valid, the result is of the form <ldapSync id=LDAP sync ID> ... </ldapSync> with values as described above plus the following if an LDAP sync operation has failed with a failureReason of clashOccurred:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>clashingUserJid</td>
<td></td>
<td>If present, these fields include the ID(s) that clashed</td>
</tr>
<tr>
<td>clashingUri</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clashingCallId</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

11.5 External Directory Search Locations

Via the API you can add to the Call Bridge, additional directory search locations to be consulted when users of Cisco Meeting Apps perform searches. External directory search locations can be added on a per-tenant level. Results from these locations will be added to the "normal" results (e.g. those from our LDAP-sourced user lists) and presented in the Cisco Meeting App.
11.5.1 Retrieving Information on external directory search locations

GET method performed on the “/directorySearchLocations” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset limit</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>tenantFilter</td>
<td>ID</td>
<td>Supply tenantFilter to return only those external directory search locations associated with the specified tenant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldapServer tenant</td>
<td>ID ID</td>
<td>All as per external directory search location creation described below</td>
</tr>
<tr>
<td>baseDn</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>filterFormat</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td></td>
</tr>
<tr>
<td>priority</td>
<td>Number</td>
<td></td>
</tr>
</tbody>
</table>

11.5.2 Adding and modifying external directory search locations

- Creating: POST method to the “/directorySearchLocations” node. If the LDAP source is configured on the system successfully, its ID is returned in the “Location” field of the response header
- Modifying: PUT method on a “/directorySearchLocations/<directory search location id>” node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldapServer *</td>
<td>ID</td>
<td>The ID of a previously-configured LDAP server (see above)</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If supplied, the tenant to which this external directory applies; entries from the remote directory will only be supplied to users associated with this tenant</td>
</tr>
<tr>
<td>baseDn</td>
<td>String</td>
<td>The distinguished name of the node in the LDAP server's tree within which to search</td>
</tr>
</tbody>
</table>
### 11.5.3 Retrieving detailed information on external directory search locations

GET method performed on a “/directorySearchLocations/<directory search location id>” node. If the directory search location ID supplied is valid, a “200 OK” response is received, with XML content as per directory search location creation described above.

### 11.5.4 Example of adding external directory search locations

This section provides an example of adding additional directory search locations that the Call Bridge will consult when users of Cisco Meeting Apps perform searches.

Follow these steps:

1. Use an app such as Chrome Postman to login to the API of the Meeting Server.

2. Create an LDAP server entry in the Meeting Server. The figure below shows an Idapsserver entry being POSTed to the Meeting Server at URL 192.168.6.25. The entry is for the LDAP server at URL 192.168.1.10, authorization information is provided.

---

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterFormat</td>
<td>String</td>
<td>The LDAP filter used to select directory search results; $1 should be used to represent the user-supplied search string</td>
</tr>
<tr>
<td>label</td>
<td>String</td>
<td>The human-readable name that should be associated with search results from this directory when displayed by requesting clients.</td>
</tr>
<tr>
<td>priority</td>
<td>Number</td>
<td>Controls the order in which directorySearchLocations should be used in searching; entries with higher priorities will be used first</td>
</tr>
<tr>
<td>firstName, lastName, displayName, phone, mobile, email, sip, organisation</td>
<td></td>
<td>These fields specify the name of a field from LDAP to be used when populating the contents of the search result. For example, displayName might be set to &quot;cn&quot; to use the canonical name.</td>
</tr>
</tbody>
</table>
Step 2 provides the necessary information to authorize Call Bridge to access the LDAP server.

3. Use GET to obtain the ID of the LDAP server entry created in step 2.

4. Create a DirectorySearchLocation by POSTing the LDAP server ID from step 3. The DirectorySearchLocation settings define the behavior of searching the directory. Ensure you set the data format to raw.
5. Use GET to obtain the ID of the Directory Search Location created in step 4.

6. Use PUT to edit the DirectorySearchLocation. Add the DirectorySearchLocation ID in the PUT URL and provide the detailed information about baseDN and filterFormat etc. Make sure that you set the data format to raw. For example:

PUT:

```plaintext

baseDn=OU=contacts,DC=example,DC=com&filterformat=cn=*$1*&firstname=givenna me&lastname=sn&displayname=cn&sip=mail
```
7. Check that the configuration of the DirectorySearchLocation is as you expected.
12 Multi-tenancy

The Meeting Server supports multi-tenancy; this refers to sub-dividing its capacity into a set of “islands” where each island has all of the functionality of the unit as a whole, but has no access to the resources (for instance users, coSpaces, or active calls) of other tenants.

There are two main implications of multi-tenancy on the API:

- The API allows tenants to be created, modified and removed, and is the primary means by which tenants are managed.
- The API can return results to a specific tenant. In multi-tenancy mode, typically each coSpace, call and user is keyed to (“owned by”) a tenant; when retrieving information on a specific user, coSpace or call object, the API includes which tenant owns that object, and enumeration-based retrievals include the tenant information for each object in the returned list.

Equally GET methods can be filtered to only include information for a specific tenant.

The majority of the API methods detailed earlier in this document also work in multi-tenancy mode. In most cases, this equates to supplying a tenant ID in creation (POST) methods via the form parameters, such that the system knows which tenant a new object is to be associated with. For instance, creating a coSpace when not in multi-tenant mode would involve a POST method to “/api/v1/coSpaces” with parameters such as “name” as form parameters. To create a coSpace for a specific tenant, the POST would again be to “/api/v1/coSpaces”, but additionally include a tenant=<tenant id>” in the form parameters, where “<tenant id>” would have been learnt either as the result of a previous tenant creation or via an earlier enumeration.

In any initial POST method, an absent “tenant” parameter or a zero-length value are treated as equivalent. The effect is that there is that the object is not associated with any tenant.

According to the rules above, <tenant id> values are valid in creation and modification (POST and PUT) methods for the following objects (see Figure 3):

- coSpaces (“/coSpaces” or “/coSpaces/<coSpace ID>”)
- LDAP sources (“/ldapSources” or “/ldapSources/<ldap source id>”)
- forwarding dial plan rules (“/forwardingDialPlanRules” or “/forwardingDialPlanRules/<forwarding dial plan rule ID>”)
- outbound dial plan rules (“/outboundDialPlanRules” or “/outboundDialPlanRules/<outbound dial plan rule ID>”)

<tenant id> values are returned by the Meeting Server in retrieval (GET) methods for the following objects:
coSpaces ("/coSpaces" or "/coSpaces/<coSpace ID>")
users ("/users" or "/users/<user ID>")
callLegs ("/callLegs", "/callLegs/<call leg id>", or "/calls/<call id>/callLegs")
LDAP sources ("/ldapSources" or "/ldapSources/<ldap source id>")
forwarding dial plan rules ("/forwardingDialPlanRules" or "/forwardingDialPlanRules/<forwarding dial plan rule ID>")
outbound dial plan rules ("/outboundDialPlanRules" or "/outboundDialPlanRules/<outbound dial plan rule ID>"")

For enumerations of all of these objects a “tenantFilter” value can be supplied in the requested URI in order to retrieve only those objects associated with the specified tenant.

Figure 3: Outline multi-tenancy process

12.1 Tenants

12.1.1 Retrieving Tenants
GET method performed on a “/tenants” node.
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>Text</td>
<td>Supply filter=&lt;tenant&gt; in the URI to return just those tenants that match the filter</td>
</tr>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first &quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>limit</td>
<td>Number</td>
<td></td>
</tr>
<tr>
<td>callLegProfileFilter</td>
<td>ID</td>
<td>Supply callLegProfileFilter=&lt;call leg profile id&gt; to return just those coSpaces using that call leg profile</td>
</tr>
</tbody>
</table>

### Response elements

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>tenant id</td>
<td>ID</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>Text</td>
<td>&lt;tenant&gt; elements follow the general form on the left.</td>
</tr>
<tr>
<td>tenantGroup</td>
<td>ID</td>
<td>If specified, associate this tenant with the supplied tenant group; the IDs of coSpaces in tenants within the same tenant group must be unique.</td>
</tr>
</tbody>
</table>

### 12.1.2 Creating and Modifying a Tenant

- **Creating:** POST method to the “/tenants” node. If the tenant is created successfully, an ID for the new tenant is returned in the “Location” field of the response header.
- **Modifying:** PUT method performed on a “/tenants/<tenant id>” node

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Text</td>
<td>A label for the tenant</td>
</tr>
<tr>
<td>tenantGroup</td>
<td>ID</td>
<td>If specified, associate this tenant with the supplied tenant group; the IDs of coSpaces in tenants within the same tenant group must be unique.</td>
</tr>
<tr>
<td>callLegProfile</td>
<td>ID</td>
<td>If specified, associates the specified call leg profile with this tenant</td>
</tr>
<tr>
<td>callProfile</td>
<td>ID</td>
<td>If specified, associates the specified call profile with this tenant</td>
</tr>
<tr>
<td>dtmfProfile</td>
<td>ID</td>
<td>If specified, associates the specified DTMF profile with this tenant</td>
</tr>
<tr>
<td>ivrBrandingProfile</td>
<td>ID</td>
<td>If specified, associates the specified IVR branding profile with this tenant</td>
</tr>
<tr>
<td>callBrandingProfile</td>
<td>ID</td>
<td>If specified, associates the specified call branding profile with this tenant</td>
</tr>
<tr>
<td>participantLimit</td>
<td>Number</td>
<td>If specified, sets a limit on the number of participants associated with this tenant that can be simultaneously active; new participants beyond this limit will not be permitted.</td>
</tr>
<tr>
<td>userProfile</td>
<td>ID</td>
<td>If supplied, a user profile to associate with this tenant; unless otherwise overridden, all users associated with this tenant will use this user profile</td>
</tr>
</tbody>
</table>
12.1.3 Retrieving Detailed Information about an Individual Tenant

GET method performed on a “/tenants/<tenant ID>” node. If the tenant ID supplied is valid, a “200 OK” response is received, with XML content described in the previous section.

12.2 Tenant group operations

12.2.1 Retrieving Tenant Groups

GET method performed on a “/tenantGroups” node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>Number</td>
<td>An &quot;offset&quot; and &quot;limit&quot; can be supplied to retrieve elements other than the first</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>&quot;page&quot; in the notional list (see above).</td>
</tr>
<tr>
<td>tenantGroups</td>
<td>Number</td>
<td>Number of tenant groups</td>
</tr>
<tr>
<td>tenantGroup id</td>
<td>ID</td>
<td>Id for each tenant group</td>
</tr>
</tbody>
</table>

12.2.2 Creating and Modifying a Tenant Group

- Creating: POST method to the “/tenantGroups” node. If the tenant group is created successfully, an ID for the new tenant group is returned in the “Location” field of the response header
- Modifying: PUT method performed on a “/tenantGroups/<tenant group id>” node

12.2.3 Retrieving Detailed Information about an Individual Tenant Group

GET method performed on a “/tenantGroups/<tenant group id>” node. If the tenant ID supplied is valid, a “200 OK” response is received.
13 Query Methods

13.1 accessQuery Method

The accessQuery method finds details of how a given URI or call ID (for example, one that could be associated with a coSpace) might be reached. One use is an external system discovering that a coSpace with URI "sales.meeting" would be reached via the SIP URI "sales.meeting@example.com".

POST performed on the /api/v1/accessQuery node.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>Text</td>
<td>The URI &quot;user part&quot; is the part before any '@' character in a full URI.</td>
</tr>
<tr>
<td>callId</td>
<td>String</td>
<td>A numeric ID (typically 9 digits long)</td>
</tr>
<tr>
<td>tenant</td>
<td>ID</td>
<td>If supplied, limits the search to the specific tenant</td>
</tr>
</tbody>
</table>

None of the parameters above are mandatory but the query is only meaningful if the uri or callId is supplied.

Response format:

<table>
<thead>
<tr>
<th>Response elements</th>
<th>Type/Value</th>
<th>Description/Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>String</td>
<td>The full URI corresponding to the uri supplied in the request</td>
</tr>
<tr>
<td>webAddress</td>
<td>String</td>
<td>An HTTPS URI for web access to the callId supplied in the request</td>
</tr>
<tr>
<td>ivr</td>
<td>String</td>
<td>A telephone number to reach an IVR that can be supplied with the callId supplied in the request</td>
</tr>
</tbody>
</table>
14 Using Profiles

There are a number of profiles you can use:

- /system/profiles (this is the top-level profile)
- /callProfiles
- /callLegProfiles
- /callBrandingProfiles
- /dtmfProfiles
- /ivrBrandingProfiles
- /userProfiles
- /compatibilityProfiles (2.1 onwards)

The top-level profile can include all the other profiles.

Equally, all the other profiles can be set, modified or retrieved for a tenant. For example on a per-tenant basis you can set a different callBrandingProfile to the top-level system profile so that calls for tenant A have different branding from calls to users who do not belong to a tenant and from calls for tenant B.

coS, access methods, users and IVRs can be created with some of the profiles – see the diagrams over page. The values set in the profiles in these definitions would override any values set at the top level or tenant level. For example, the values in the callLegProfile set for a coSpace would be used for call legs in that coSpace – overriding any values set in the callLegProfile specified for the tenant that the coSpace is associated with, or the top-level profile.

Finally you can specify a value for one of the parameters that also appears in a profile in each individual call leg. Then you can use a profile for most parameters but override just one such as the defaultLayout for a call leg.

Using profiles is optional at all levels.

The following diagrams show the relationship, inheritance and overrides between profiles.
Figure 4: Value inheritance with profiles

Values in lower profile override those set above

If no profile is set at one level, the object inherits values from the profile above

Figure 5: Value inheritance with profiles when using tenants

Values in lower profile override those set above

If no profile is set at one level, the object inherits values from the profile above
Appendix A  Additional Call Leg Information

A.1  Call Leg Information

When information is retrieved on a specific individual call leg, its structure follows the form:

```
<callLeg id="386621ab-927b-4624-a77d-0288913c92ac">

call leg response values (see section 8.3.3)

<configuration>

call leg configuration (see below)

</configuration>

<status>

call leg status (see below)

</status>

</callLeg>
```

A.1.1  Call leg configuration

The configuration section returned includes the same values as those that can be modified in a PUT method on that call leg.

An example configuration section is shown below:

```
<configuration>
  <ownerId>XXXXX</ownerId>
  <chosenLayout></chosenLayout>
  <needsActivation>false</needsActivation>
  <defaultLayout>speakerOnly</defaultLayout>
  <participantLabels>false</participantLabels>
  <presentationDisplayMode>dualStream</presentationDisplayMode>
  <presentationContributionAllowed>false</presentationContributionAllowed>
  <presentationViewingAllowed>true</presentationViewingAllowed>
  <endCallAllowed>true</endCallAllowed>
  <muteOthersAllowed>true</muteOthersAllowed>
  <videoMuteOthersAllowed>true</videoMuteOthersAllowed>
  <muteSelfAllowed>true</muteSelfAllowed>
  <videoMuteSelfAllowed>true</videoMuteSelfAllowed>
  <changeLayoutAllowed>true</changeLayoutAllowed>
  <joinToneParticipantThreshold>0</joinToneParticipantThreshold>
  <leaveToneParticipantThreshold>0</leaveToneParticipantThreshold>
  <videoMode>false</videoMode>
  <rxAudioMute>false</rxAudioMute>
  <txAudioMute>false</txAudioMute>
  <rxVideoMute>false</rxVideoMute>
  <txVideoMute>false</txVideoMute>
```
A.1.2 Call leg status

The status information returned contains live values relating to the call leg’s active state. Media information is contained within one or more rxAudio, txAudio, rxVideo and txVideo sub-sections; multiple video or audio streams may be identified and distinguished via their “role” attribute, which may be either “main” or “presentation”.

Each media section includes packet loss percentage and jitter.

An example set of status data is shown below:

```xml
<status>
  <state>connected</state>
  <durationSeconds>349</durationSeconds>
  <direction>incoming</direction>
  <sipCallId>6ff25ec40843df0b5153cab4a8601ee</sipCallId>
  <groupId>8b29e92d-27c2-421a-8d73-47daf09d7fe7</groupId>
  <recording>true</recording>
  <streaming>false</streaming>
  <deactivated>false</deactivated>
  <encryptedMedia>true</encryptedMedia>
  <unencryptedMedia>false</unencryptedMedia>
  <layout>telepresence</layout>
  <activeLayout/>
  <availableVideoStreams/>
  <rxAudio>
    <codec>aac</codec>
    <packetLossPercentage>0.0</packetLossPercentage>
    <jitter>2</jitter>
    <bitRate>64000</bitRate>
  </rxAudio>
  <txAudio>
    <codec>aac</codec>
    <packetLossPercentage>0.0</packetLossPercentage>
    <jitter>0</jitter>
    <bitRate>64000</bitRate>
    <roundTripTime>66</roundTripTime>
  </txAudio>
  <rxVideo role="main">
    <codec>h264</codec>
    <width>768</width>
    <height>448</height>
    <frameRate>29.7</frameRate>
    <bitRate>544603</bitRate>
  </rxVideo>
</status>
```
Additionally, a “<sipCallId>” value may be included; this will be present if the call leg corresponds to a SIP connection, and will be the global unique “Call-ID” value from the SIP protocol headers.

If there is an active video stream from the Call Bridge to the remote party, there will be a “layout” value showing the actual layout currently in use for that call leg, either because of a specific choice on the part of the user (for Cisco Meeting App call legs) or because of a coSpace default or call leg override.
Appendix B  Additional Multiparty Licensing Information

B.1  /system/licensing Information

When information is retrieved from the /system/licensing node, its structure follows the form:

<licensing>
  <features>
    <callBridge>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
    </callBridge>
    <webBridge>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
    </webBridge>
    <turn>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
    </turn>
    <ldap>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
    </ldap>
    <branding>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
      <level>whiteLabel</level>
    </branding>
    <recording>
      <status>activated</status>
      <expiry>2100-Jan-01</expiry>
      <limit>30</limit>
    </recording>
    <personal>
      <status>noLicense</status>
    </personal>
    <shared>
      <status>noLicense</status>
    </shared>
    <capacityUnits>
      <status>noLicense</status>
    </capacityUnits>
  </features>
</licensing>
Appendix B  Additional Multiparty Licensing Information

B.2 /system/multipartyLicensing Information

When information is retrieved from the /system/multipartyLicensing node, its structure follows the form:

<multipartylicensing>
  <timestamp>2016-07-20T14:22:17Z</timestamp>
  <personalLicenseLimit>0</personalLicenseLimit>
  <sharedLicenseLimit>0</sharedLicenseLimit>
  <capacityUnitLimit>0</capacityUnitLimit>
  <users>545</users>
  <personalLicenses>0</personalLicenses>
  <participantsActive>0</participantsActive>
  <callsActive>0</callsActive>
  <weightedCallsActive>0.000</weightedCallsActive>
  <callsWithoutPersonalLicense>0</callsWithoutPersonalLicense>
  <weightedCallsWithoutPersonalLicense>0.000</weightedCallsWithoutPersonalLicense>
  <capacityUnitUsage>0.000</capacityUnitUsage>
  <capacityUnitUsageWithoutPersonalLicense>0.000</capacityUnitUsageWithoutPersonalLicense>
</multipartylicensing>
Appendix C  Installing API tools

This appendix covers installing:

- Chrome Postman
- Firefox Poster
- Chrome Advanced REST

but first ensure you have an account on the Meeting Server with API access, as described in Section 3.1.

C.1 Installing Chrome Postman

1. Install Postman into Chrome by opening the Apps button in the upper left corner.

2. Click on Web Store to open the Chrome App Store

3. Enter Postman in the Search field and press Enter.

4. Install Postman – REST Client by clicking on the + FREE button to the right
5. Click Add to continue with the installation

Once installed, Postman – REST Client will appear on the Apps page

6. Open Postman by clicking on the Icon from within the Apps page of Chrome

7. Select the Basic Auth tab and enter the Username and Password for the API of the Meeting Server. Click Refresh headers. Enter the URL of the Meeting Server and press SEND
A 200 OK Response should be returned.

Note: if you have difficulty logging in, then try to log into the Web Admin interface, accept any certificate warnings and add them as a permanent exception. (If you do not do this and there is a certificate warning, Poster will not work.)

8. To verify API connectivity is working, perform a GET operation on the /system/status node of your server.

If it fails, verify that the login is correct and that you can access the Web URL from this PC.

C.2 Installing Firefox Poster

Install Poster into Firefox by opening the Menu button in the upper right corner and choosing Add-ons.
2. Enter Poster in the Search field and press Enter.

3. Click Install and follow directions to restart Firefox once complete.

4. Enable the Menu Bar by right clicking just to the right of the + sign that opens a new tab and selecting Menu Bar.

5. Once enabled, access Poster through Tools > Poster
6. To verify that API connectivity is working, enter the URL of the Meeting Server, and the Username and Password for the API. Press GET.

A 200OK Response should be returned with the Status Data, for example:
If the GET fails, verify that:

- the login is correct,
- you can access the Web URL from this PC
- Firefox trusts the certificate assigned to Web Admin, or make a Permanent exception by adding this Trust to the browser’s Trust store. Do this by opening a standard browser session to Web Admin using Firefox, and accept the warnings and add the exception.

### C.3 Installing Chrome Advanced Rest Client

1. Install the Advanced REST Client into Chrome by opening the Apps button in the upper left corner.
2. Click on Web Store to open the Chrome App Store
3. Enter Advanced REST Client in the Search field and press Enter. Install Advanced REST Client by clicking on the + FREE button to the right.

4. Click Add to continue with the installation

5. Once installed, the Advanced REST Client will appear on the Apps page
6. Open the Advanced REST Client by clicking on the icon from within the Apps page of Chrome. To verify API connectivity is working, enter the URL of the Meeting Server and press **SEND**.

7. Enter the API login when prompted and press **Log In**.

A 200OK Response should be returned with the Status Data, for example:
If it fails, verify that the login is correct and that you can access the Web URL from this PC.
Appendix D  Using Firefox Poster with the API

This appendix provides an example of using the API tool, Poster, to create a Call Leg Profile. Other tools are available, see Appendix Appendix C.

1. Ensure you have an account with API access as described earlier.
2. Install Poster, see Appendix Appendix C.2.
3. To create a Call Leg Profile, change the URL to include `/callLegProfiles` and specify the associated parameters to send, for example:

   ![Image](image.png)

   Add the API object to the URL, for instance `/callLegProfiles`

   Specify the parameters of the API object to be changed, and the new values

   **Note:** Setting `joinToneParticipantThreshold` to 100 will result in tones only being played for the first 100 sites joining the call. Similarly, setting `leaveToneParticipantThreshold` to 100 will result in tones only being played when a participant leaves the call and there are 100 or fewer participants in the call. (These values are just examples)

4. Click POST.

   A pop up similar to the following appears:
5. Verify it says 200 OK at the top.

6. Copy the Call Leg Profile ID from the Location field; the long ID string after /callLegProfiles/

7. To set this Call Leg Profile as the default Global Profile to be used on all calls, create a PUT similar to the example below:
8. Click **PUT**.

You should receive a popup with 200 OK again.

Bleep notifications will now be played for anyone joining and leaving SIP calls (within the 100 limit).

**Note:** Cisco Meeting Apps do not get these audio indications because the Roster List provides a visual indication of sites joining and leaving.
Cisco Legal Information

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

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

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

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

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

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

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

© 2018 Cisco Systems, Inc. All rights reserved.
Cisco Trademark

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)