



Cisco TelePresence Server Remote Management API Reference Guide

D14373.06

December 2010

Contents

Document revision history	3
Introduction	4
HTTP and HTTPS.....	4
XML-RPC.....	4
Protocol overview	5
Authentication	5
Message flow.....	5
Unicode support.....	7
HTTP Headers.....	7
XML Header	7
Common message elements.....	7
Authentication.....	7
Conference related methods	8
conference.create	8
conference.delete	8
conference.enumerate.....	9
conference.invite.....	10
conference.senddtmf	11
conference.set	11
Deprecated parameter.....	12
conference.status	12
Deprecated parameter.....	14
conference.uninvite.....	15
participant.tidylayout.....	15
Related information sources	16
system.xml.....	16
system.info.....	17
Fault codes	18
References	20
HTTP Keep-alives	21

Document revision history

The latest version of the Cisco TelePresence Server Remote Management API is version 2.1. The following table shows which versions of the Cisco TelePresence Server support this version.

API Version	Cisco TS MSE 8710	Cisco TS 7000 Series
1.0	1.0	Not applicable
1.1	1.1	1.1
1.2	1.2	1.2
2.1	2.1	2.1

API Version 2.1 introduces the following changes.

XML-RPC request	Parameter	Addition/Deprecated	Page
conference.create	numericID	Modified	8
conference.enumerate	conferences->registerwithSipRegistrar	Added	9
conference.invite	participants->address participants->type	Modified	10
conference.status	localAddress coordSize participantList->positionX participantList->positionY participantList->positionZ participantList->width participantList->height participantList->channelList (and all of the channelList fields)	Deprecated	12
	participantList->endpointType	Modified	
	enumerateID enumerateID participantList->callProtocol	Added	
system.xml	clusterType sipRegistrarUsage sipRegistrarAddress RegisterDomain	Added	16

In addition, authentication can use session IDs. See [Authentication](#).

Note: [participant.tidylayout](#) has been added to the document – this call existed in previous versions of the API but was undocumented.

Introduction

This document contains the specification of the Cisco TelePresence Server Remote Management API, by which it is possible to control the following Cisco TelePresence products: Cisco MSE 8710 and the Cisco TS 7000 Series. This is accomplished via messages sent using the XML-RPC protocol.

XML-RPC is a simple protocol for remote procedure calling using HTTP as the transport and XML as the encoding. It is designed to be as simple as possible, whilst allowing complex data structures to be transmitted, processed and returned. XML-RPC has no platform or software dependence and was chosen over SOAP because of its simplicity.

The interface is stateless. Currently, there is no mechanism for the Cisco TelePresence Server (Server) to call back the controlling application and therefore the controlling application must poll the device for status, as required. A future enhancement *may* provide a mechanism for signaling device status changes to the controlling application.

Note: Unless otherwise stated, string lengths are a maximum of 32.

HTTP and HTTPS

The Server expects to receive HTTP communication over TCP/IP connections to port 80. The HTTP messages should be "POST"s to the URL "/RPC2".

HTTPS (a secure, encrypted version of HTTP) is supported if the encryption feature key is installed. By default, HTTPS is provided on TCP port 443, although the Server can be configured to receive HTTP and HTTPS connections on non-standard TCP port numbers if required.

The Server implements HTTP/1.1 as defined by RFC 2616 specification on <http://www.faqs.org/rfcs/rfc2616.html>.

XML-RPC

For the background and details of XML-RPC, please refer to the specification on www.xmlrpc.com.

In this implementation, all parameters and return values are part of a <struct> and are all explicitly named. For example, the "conference.status" method returns the conference ID value as a structure member named 'conferenceID' rather than as a single value of type <integer>.

Protocol overview

Authentication

To manage the Server, the controlling application must authenticate itself as the API user. Accordingly, each message contains a user name and password. It is worth noting that authentication information is sent using plain text and should only be sent over a trusted network.

As an alternative to username / password authentication, the Server now supports authentication by a session ID that has been passed to a T3 via TString at the start of a call. Because a session ID uniquely identifies an endpoint and conference, the conferenceID field is ignored if a session ID has been provided. An endpoint's session ID will be valid until 20 seconds after the end of a call, or at the beginning of a new call, whichever is sooner.

Message flow

An application can create and manage conferences by sending command messages to the Server. For each command sent (provided the message is correctly formatted according to the XML-RPC specification), the Server responds with a message indicating success or failure. The response message may also contain any data that was requested.

Command messages are sent in XML format. For example, the following message schedules a conference:

```
POST /RPC2 HTTP/1.0
User-Agent: XML-RPC for PHP 2.2.1
Host: 10.2.135.16:80
Accept-Encoding: gzip, deflate
Accept-Charset: UTF-8, ISO-8859-1, US-ASCII
Content-Type: text/xml
Content-Length: 424

<?xml version="1.0"?>
<methodCall>
<methodName>conference.create</methodName>
<params>
<param>
<value><struct>
<member><name>authenticationUser</name>
<value><string>admin</string></value>
</member>
<member><name>conferenceName</name>
<value><string>test conf name</string></value>
</member>
<member><name>numericID</name>
<value><string>11988655</string></value>
</member>
</struct></value>
</param>
</params>
</methodCall>
```

If the command was successful, the Server sends a success response. For example, in response to a successful [conference.create](#) message, the Server returns:

```

HTTP/1.1 200 OK
Connection: close
Content-Type: text/xml
Content-Length: 224

<?xml version="1.0"?>
<methodResponse>
<params>
<param>
<value>
<struct>
<member>
<name>conferenceID</name>
<value>
<int>1037</int>
</value>
</member>
</struct>
</value>
</param>
</params>
</methodResponse>

```

If the command fails, the Server sends a fault response. For example, in response to a [conference.enumerate](#) message where the enumerate ID is not valid, the Server returns:

```

HTTP/1.1 200 OK
Connection: close
Content-Type: text/xml
Content-Length: 407

<?xml version="1.0"?>
<methodResponse>
<fault>
  <value>
    <struct>
      <member>
        <name>faultCode</name>
        <value>
          <int>16</int>
        </value>
      </member>
      <member>
        <name>faultString</name>
        <value>
          <string>invalid enumerate ID</string>
        </value>
      </member>
    </struct>
  </value>
</fault>
</methodResponse>

```

The complete list of command messages, their required and optional parameters, and the expected responses are detailed in this document. The possible [fault codes](#) are listed later in this document.

Unicode support

Parameters in this version of the API can be in ASCII text or unicode (UTF8). There are several methods to distinguish between these encodings. If no method is present, ASCII is used by default.

HTTP Headers

There are two different ways of specifying unicode in the HTTP headers; either using "Accept-Encoding: utf-8", or modifying the Content-Type header to read "Content-Type: text/xml; charset=utf-8".

XML Header

At the top of each XML file, the <?xml> tag is required. This API will accept an additional encoding parameter with value UTF-8 for this tag, i.e. <?xml version="1.0" encoding="UTF-8"?>.

Common message elements

Authentication

All messages must contain a user name and password as follows:

Parameter	Type	Comments
authenticationUser	String	Name of a user with sufficient privilege for the operation being performed. The name is case sensitive.
authenticationPassword	String	The corresponding user's password. This parameter is ignored if the user has no password set - note that this differs from the web interface where a blank password must be blank.

Conference related methods

conference.create

This method creates a named conference. Conferences created through the management API will appear in the list of conferences accessible via the web interface, and vice versa.

Parameter	Type	Comments
conferenceName	String (<32 chars)	Mandatory: Name of the conference to be created. The conference name must be unique.
Optional parameters		
permanent	Boolean	Whether the conference should persist after everyone has left it. Without this option, any conferences will be deleted automatically 30 seconds after the last participant has left the conference.
numericID	String	The ID to be registered with the gatekeeper and the SIP registrar. If no ID is supplied then the conference will not be registered.
tsURI	String	If supplied, this URI will be passed to all T3 systems in the conference. These systems use this URI to connect to the Server's API to retrieve conference information. It should be of the form [protocol://]address[:port], where valid values for protocol, if supplied, are http and https.

The following is returned.

Parameter	Type	Comments
conferenceID	Integer	The new conference's unique ID. Used to identify the conference in other requests.

conference.delete

This method deletes a conference. A conference can be deleted at any time; that is, before the conference has begun, during the conference or after the conference has ended. Deleted conferences are removed entirely from the system.

Parameter	Type	Comments
conferenceID	Integer	Unique ID of the conference to be destroyed. This is returned by conference.create .

conference.enumerate

The conference enumerate method is used to return some or all conferences scheduled, running or completed.

Parameter	Type	Comments
Optional parameters		
enumeratelD	Integer	The value returned by the last enumeration call. If it is omitted, a new enumeration is started.
activeFilter	Boolean	true to request only active conferences; false or omit for all conferences.

The following is returned.

Parameter	Type	Comments
enumeratelD	Integer	The value which should be used in the next call to get the next set of data. If this is omitted, no further data is available from the Server.
conferences	Array	See below for details.

The array “conferences” contains structures with the following fields:

Parameter	Type	Comments
enumeratelD	String	The value which should be used in the next call to get the next set of data. If this is omitted, no further data is available from the Server.
conferences->conferenceName	String	The conference name.
conferences->conferenceID	Integer	The unique ID of the conference.
conferences->active	Boolean	True for an active conference.
conferences->numericID	Integer	The numericID for the conference.
conferences->registerWithGatekeeper	Boolean	True if the conference's numeric ID is registered with a gatekeeper.
conferences->registerWithSIPRegistrar	Boolean	True if the conference's numeric ID is registered with a SIP registrar.
conferences->h239ContributionEnabled	Boolean	True if H.239 is allowed for this conference.

conference.invite

This method adds one or more participants to a conference. Participants can be added before or during a conference.

Parameter	Type	Comments
conferenceID	Integer	Required: the unique ID of the conference to which to add the participant. This is returned by conference.create .
Either		
participants	Array	Array of participants, each with the following parameters.
participants->address	String	Required: address of the participant to invite. For example, 10.2.171.232, or h323:john.smith@cisco.com or sip:jonesdav@cisco.com
participants->type	String	Optional: Specify the type of endpoint as one of: t3 = CiscoT3 or exp = Cisco Experia cts = Cisco CTS. The Server determines the number of screens in this endpoint. cts1 = forces the endpoint to be treated as a one-screen Cisco CTS system cts3 = forces the endpoint to be treated as a three-screen Cisco CTS system
participants->master	Boolean	Optional: Specify whether this endpoint is a master or slave endpoint in this conference. True for master, false or omit for slave.
participants->oneTableIndex	Integer	Optional: Applies to T3 or Experia only. This endpoint's position in a conference using OneTable mode. Takes 1 , 2 , 3 or 4 .
participants->maxBitRate	Integer	Optional: The maximum bit rate in Kbit/s both to and from the Server for this participant. If omitted, the box-wide default will be used.
Participants->recordingDevice	Boolean	Optional: If true , this participant will be treated as a recording device and will not be shown as a participant in the layout. A red dot icon appears as appropriate.
Or		
participantList	String	Deprecated: Comma separated list of participants to invite, For example, '10.2.171.232, 10.47.2.246, h323:john.smith@cisco.com'

The following is returned.

Parameter	Type	Comments
participantList	Array	See below for details.

The array "participantList" contains structures with the following fields:

Parameter	Type	Comments
participantList->participantID	Integer	Participant ID.
participantList->address	String	Address of participant.

conference.senddtmf

This method sends DTMF tones to the Server.

Parameter	Type	Comments
conferenceID	Integer	Conference ID returned by conference.create .
dtmf	String	The DTMF string to send.
Optional parameters		
participantID	Integer	A participantID as returned by conference.status . If supplied, only send DTMF to this participant.
omitID	Integer	A participantID as returned by conference.status . If supplied, do not send DTMF to this participant. Only used if participantID is absent.

conference.set

This method updates an existing conference. Because conferences created through the management API appear in the list of conferences accessible via the web interface, the API can be used to modify conferences scheduled via the web interface, and vice versa.

Parameter	Type	Comments
conferenceID	Integer	Mandatory: unique ID of the conference returned by conference.create .
Optional parameters		
oneTableMode	Integer	Whether the conference is in oneTable mode. 0 = not in oneTable mode 1 = 4-person mode 2 = 2-person mode
h239ContributionEnabled	Boolean	Whether to allow H.239 contributions.

Deprecated parameter

The following parameters are deprecated as of Server version 1.1.

Parameter	Type	Comments
roundTableEnable	Boolean	Whether the conference is in round table mode.

conference.status

This method returns the current status of a conference and its participants.

Parameter	Type	Comments
conferenceID	Integer	Mandatory: unique ID of the conference whose status is required. This is returned by conference.create .
Optional parameters		
enumerateID	String	Supply the value returned by the last call in order to receive the next set of data in participantList. If omitted, a new enumeration starts.

The following is returned.

Parameter	Type	Comments
enumerateID	String	The value which should be used in the next call to get the next set of data. If this is omitted, no further data is available from the Server.
conferenceID	Integer	The conference's unique ID. This is returned by conference.create .
active	Boolean	Whether the conference is currently active.
oneTableMode	Integer	Whether the conference is on oneTable mode. 0 = not in oneTable mode 1 = 4-person mode 2 = 2-person mode
h239ContributionID	Integer	If H.239 is active, then this is the participant ID of the contributing endpoint. If there is no H.239 in the conference, then zero is returned.
portsVideoFree	Integer	Total number of free video ports on the hosting Server (or zero if the conference is inactive).

Parameter	Type	Comments
portsAudioFree	Integer	Total number of free audio ports on the hosting Server (or zero if the conference is inactive).
portsContentFree	Integer	Total number of free content ports on the hosting Server (or zero if the conference is inactive).
numericID	String	The numericID of this conference registered with the gatekeeper.
recording	Boolean	Returns true if the conference is being recorded by a recording device specified by Participants->recordingDevice in conference.invite .
participantList	Array	See below for details.

The array “participantList” contains structures with the following fields:

Parameter	Type	Comments
participantList->participantID	Integer	Participant ID.
participantList->callState	Integer	State: 0 = Not connected 1 = Calling in, not in conference yet 2 = Called in 3 = Calling out e.g. alerting 4 = Called out and in conference
participantList->endpointType	Integer	Type of endpoint: 1 = Normal endpoint 2 = Cisco Experia 3 = Grouped endpoints 4 = Cisco T3 5 = Cisco CTS
participantList->address	String	Address of participant.
participantList->disconnectReason	String	Disconnection reason if the endpoint has been disconnected. unspecified = unspecified error localTeardown = requested by administrator noAnswer = no answer rejected = call was rejected busy = busy gatekeeperError = gatekeeper error remoteTeardown = left conference
participantList->callStartMute	Boolean	Returns true if this endpoint is being sent black video for call setup.

Parameter	Type	Comments
participantList->master	Boolean	Returns true if this endpoint is a master or slave endpoint for this conference.
participantList->rxPreviewURL	String	The URL to retrieve a jpeg snapshot of video received from this participant.
participantList->txPreviewURL	String	The URL to retrieve a jpeg snapshot of video sent to this participant.
participantList->callType	String	Either Audio or Video
participantList->callDirection	String	Either Incoming or Outgoing
participantList->callBandwidth	Integer	Call bandwidth in kbps.
participantList->micMute	Boolean	Returns true if far end microphone is muted.
participantList->recordingDevice	Boolean	Returns true if the endpoint is a recording device.
participantList->callProtocol	String	Either sip or h323 .

Deprecated parameter

The following parameter is deprecated as of Server version 1.1.

Parameter	Type	Comments
roundTableEnable	Boolean	Whether the conference is in round table mode.

conference.uninvite

This method removes one or more participants from a conference. Participants can be removed before or during a conference.

Notes:

- ▶ Either participantList or participantListID must be present
- ▶ A fault is returned only if no participants are found, e.g. if two endpoints are uninvited and one is found and one is not, then success is returned

Parameter	Type	Comments
conferenceID	Integer	Mandatory: unique ID of the conference from which the participant will be removed. This is returned by conference.create .
Optional parameters		
participantList	String	Comma separated list of participants to remove e.g. '10.2.171.232, 10.47.2.246, h323:john.smith@cisco.com'
participantListID	String	Comma separated list of participant IDs to remove e.g. '1024, 1056'

participant.tidylayout

This method forces the Server to reconsider the conference layout and can be useful when there had previously been several single-screen endpoints in the conference that have now left, for example. It should produce a more satisfactory grouping of these participants.

Parameter	Type	Comments
participantID	Integer	Participant ID returned by conference.invite (optional if authentication is performed using a sessionID).

Related information sources

system.xml

While not strictly part of the XML-RPC API, some information can be retrieved from the system.xml file. This can be downloaded via HTTP as the file system.xml in the root of the unit, for example <http://TPS/system.xml>.

The meaning of the fields is:

Field	Comments
manufacturer	The manufacturer of this TelePresence Server, e.g. Cisco.
model	The model of this particular Server, e.g. TS 8710.
serial	The serial number of this Server.
softwareVersion	The software version currently running.
buildVersion	The build version of the software currently running.
hostName	The host name of the system.
ipAddress	IP address of system.
macAddress	MAC address of system.
gatekeeperUsage	Whether a gatekeeper is configured/enabled (Yes) or not (No).
gatekeeperAddress	Gatekeeper host name or IP address.
gatekeeperIds	Comma separated list of registered IDs associated with this Server and those that it controls. (Omitted if this Server is not a Conference controller).
isMaster	Yes = this Server is a Conference controller, No = this Server is controlled by another Server.
totalVideoPorts	The total number of video ports on the Server.
totalContentPorts	The total number of video content ports on the Server.
totalAudioOnlyPorts	The total number of additional audio only ports on the Server.
uptimeSeconds	The number of seconds since boot.
clusterType	The role of the Server when it is part of a cluster. One of: unclustered , master or slave .
sipRegistrarUsage	Whether a SIP registrar is configured/enabled (Yes) or not (No).
sipRegistrarAddress	SIP registrar host name or IP address.

Field	Comments
sipRegistrarDomain	SIP registrar domain.

system.info

This function retrieves the system status. It has no mandatory or optional parameters. The returned information is:

Parameter	Type	Comments
gateKeeperOK	Boolean	Gatekeeper is configured and the Server is registered.
makeCallsOK	Boolean	Server has enough resources to make at least one call.
tpsNumberOK	Integer	Number of configured and active Servers. If the system is a Conference controller, then this number includes your system plus any others that it controls.
tpdVersion	String	Server version number.
tpdName	String	Server system name.
tpdUptime	Integer	Server in seconds.
tpdSerial	String	Server serial number.
portsVideoTotal	Integer	Total number of video ports on the Server.
portsVideoFree	Integer	Total number of free video ports on the Server.
portsAudioTotal	Integer	Total number of audio ports on the Server.
portsAudioFree	Integer	Total number of free audio ports on the Server.
portsContentTotal	Integer	Total number of content ports on the Server.
portsContentFree	Integer	Total number of free content ports on the Server.
maxConferenceSizeVideo	Integer	Number of video ports available on the least used media blade (i.e. max video participants in a single conference).
maxConferenceSizeAudio	Integer	Number of audio ports available on the least used media blade.
maxConferenceSizeContent	Integer	Number of content ports available on the least used media blade.

Fault codes

The Server has a series of fault codes which are given when a fault occurs during the processing of an XML-RPC request. The codes are shared with Cisco TelePresence MCUs, gateways and VCRs and all the codes are listed here; however **only a subset are returned** by the Cisco TelePresence Server. While individual method descriptions above give some indication of which faults may occur, below is a description of all possible fault codes used within this specification, and the usual interpretations.

Fault Code	Description
1	Method not supported. This method is not supported on this device.
2	Duplicate conference name. A conference name was specified, but is already in use.
3	Duplicate participant name. A participant name was specified, but is already in use.
4	No such conference or auto attendant. The conference or auto attendant identification given does not match any conference or auto attendant.
5	No such participant. The participant identification given does not match any participants.
6	Too many conferences. The device has reached the limit of the number of conferences that can be configured.
7	Too many participants. There are already too many participants configured and no more can be created.
8	No conference name or auto attendant id supplied. A conference name or auto attendant identifier was required, but was not present.
9	No participant name supplied. A participant name is required but was not present.
10	No participant address supplied. A participant address is required but was not present.
11	Invalid start time specified. A conference start time is not valid.
12	Invalid end time specified. A conference end time is not valid.
13	Invalid PIN specified. A PIN specified is not a valid series of digits.
14	Unauthorised. The requested operation is not permitted on this device.
15	Insufficient privileges. The specified user id and password combination is not valid for the attempted operation.
16	Invalid enumerateID value. An enumerate ID passed to an enumerate method invocation was invalid. Only values returned by the device should be used in enumerate methods.
17	Port reservation failure. The reservedAudioPorts or reservedVideoPorts value is set too high, and the device cannot support this.

Fault Code	Description
18	Duplicate numeric ID. A numeric ID was given, but this ID is already in use.
19	Unsupported protocol. A protocol was used which does not correspond to any valid protocol for this method. In particular, this is used for participant identification where an invalid protocol is specified.
20	Unsupported participant type. A participant type was used which does not correspond to any participant type known to the device.
21	No such folder. A folder identifier was present, but does not refer to a valid folder.
22	No such recording. A recording identifier was present, but does not refer to a valid recording.
23	No changes requested. A method for changing something correctly identifies an object, but no changes to that object are specified.
24	No such port. An ISDN port is given as a parameter which does not exist on an ISDN gateway.
101	Missing parameter. A required parameter is absent. The parameter in question is given in the fault string in the format "missing parameter - <i>parameter_name</i> ".
102	Invalid parameter. A parameter was successfully parsed, is of the correct type, but falls outside the valid values; for example an integer is too high or a string value for a protocol contains an invalid protocol. The parameter in question is given in the fault string in the format "invalid parameter - <i>parameter_name</i> ".
103	Malformed parameter. A parameter of the correct name is present, but cannot be read for some reason; for example the parameter is supposed to be an integer, but is given as a string. The parameter in question is given in the fault string in the format "malformed parameter - <i>parameter_name</i> ".
201	Operation failed. This is a generic fault for when an operation does not succeed as required.

References

The following table lists documents and web sites referenced in this document. All product documentation can be found on our [web site](#).

Title	Link
XML-RPC	http://www.xmlrpc.com/
Hypertext Transfer Protocol (HTTP/1.1)	http://www.faqs.org/rfcs/rfc2616.html

HTTP Keep-alives

The amount of TCP traffic can be reduced when polling the Server via the API by using HTTP keep-alives.

Any client which supports HTTP keep-alives may include the following line in the HTTP header of an API request:

```
Connection: Keep-Alive
```

This indicates to the product that the client supports HTTP keep-alives. The device then *may* choose to not close the TCP connection after returning its response to the request. If the connection will be closed, the device returns the following line in the HTTP header of its response:

```
Connection: close
```

The absence of this line indicates that the device will keep the TCP connection open and that the client may use the same connection for a subsequent request.

The device will not allow a connection to be kept alive if:

- ▶ the current connection has already serviced a set number of requests
- ▶ the current connection has already been open for a certain amount of time
- ▶ there are already more than a certain number of connections in a “kept alive” state

These restrictions are in place to limit the resources associated with kept-alive connections. If a connection is terminated for either of the first two reasons, the client will probably find that the connection is back in a keep-alive state following the next request.

The client should never assume a connection will be kept alive.

Also note that, even after a response not containing the “connection: close” header, the connection will still be closed if no further requests are made within one minute. If requests from the client are likely to be this far apart then there is little to be gained by using HTTP keep-alives.

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, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at 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. (1005R)

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.

© December 2010 Cisco Systems, Inc. All rights reserved.