

Troubleshooting Guide for Cisco CallManager

Release 3.1(1)

Corporate Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 526-4100

Customer Order Number: DOC-OL1531=
Text Part Number: OL-1531-01

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.

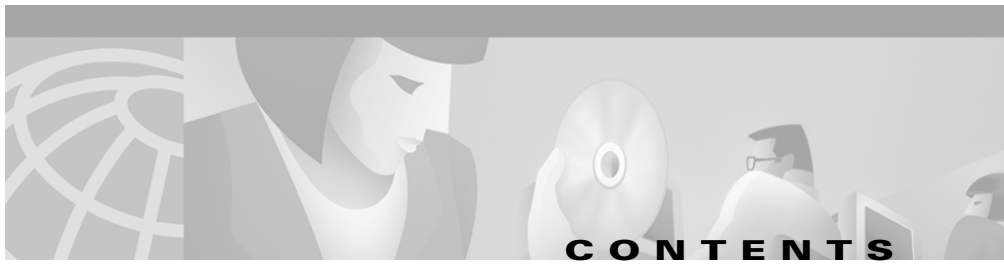
AccessPath, AtmDirector, Browse with Me, CCIP, CCSI, CD-PAC, *CiscoLink*, the Cisco *Powered* Network logo, Cisco Systems Networking Academy, the Cisco Systems Networking Academy logo, Fast Step, Follow Me Browsing, FormShare, FrameShare, GigaStack, IGX, Internet Quotient, IP/VC, iQ Breakthrough, iQ Expertise, iQ FastTrack, the iQ Logo, iQ Net Readiness Scorecard, MGX, the Networkers logo, *Packet*, RateMUX, ScriptBuilder, ScriptShare, SlideCast, SMARTnet, TransPath, Unity, Voice LAN, Wavelength Router, and WebViewer are trademarks of Cisco Systems, Inc.; Changing the Way We Work, Live, Play, and Learn, Discover All That's Possible, and Empowering the Internet Generation, are service marks of Cisco Systems, Inc.; and Aironet, ASIST, BPX, Catalyst, CCDA, CCDP, CCIE, CCNA, CCNP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, the Cisco IOS logo, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Enterprise/Solver, EtherChannel, EtherSwitch, FastHub, FastSwitch, IOS, IP/TV, LightStream, MICA, Network Registrar, PIX, Post-Routing, Pre-Routing, Registrar, StrataView Plus, Stratm, SwitchProbe, TeleRouter, and VCO are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and certain other countries.

All other trademarks mentioned in this document or Web site 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. (0106R)

Troubleshooting Guide for Cisco CallManager

Copyright © 2000, 2001 Cisco Systems, Inc.

All rights reserved.



Preface ix

Purpose **ix**

Audience **x**

Organization **x**

Related Documentation **xi**

Conventions **xii**

Obtaining Documentation **xiii**

World Wide Web **xiv**

Documentation CD-ROM **xiv**

Ordering Documentation **xiv**

Documentation Feedback **xv**

Obtaining Technical Assistance **xv**

Cisco.com **xv**

Technical Assistance Center **xvi**

 Contacting TAC by Using the Cisco TAC Website **xvi**

 Contacting TAC by Telephone **xvii**

CHAPTER 1

Overview 1-1

Understanding Cisco IP Telephony Topology **1-1**

Topology Checklist **1-2**

CHAPTER 2

Troubleshooting and Monitoring Tools and Utilities 2-1

Using Cisco CallManager Administration Details **2-1**

Using Microsoft Performance **2-2**

- Opening Microsoft Performance **2-2**
- Customizing Performance **2-2**
- Using Microsoft Event Viewer **2-3**
 - Opening Event Viewer **2-3**
 - Detailed Information about Events **2-3**
- Using SDI Trace **2-4**
 - SDI Trace Output **2-4**
 - Configuring Traces **2-5**
- Using SDL Trace **2-5**
 - Enabling SDL Trace **2-6**
- Sniffer Trace **2-8**
 - Sniffer Trace Applications **2-9**
- Call Detail Records and Call Management Records **2-9**
 - Enabling or Disabling CDRs **2-10**
 - Using CMRs **2-10**
- Using Admin Serviceability Tool **2-11**

CHAPTER 3

Diagnosing Cisco CallManager Problems 3-1

- Voice Quality **3-2**
 - Diagnosing Lost or Distorted Audio Problems **3-2**
 - Diagnosing Audio Problems from the Cisco IP Phone **3-4**
 - Diagnosing Crackling Sounds **3-5**
 - Checking Your Loads **3-5**
 - Echo **3-5**
 - Testing Speakerphone and Volume Levels **3-6**
 - Testing the Gateway **3-6**
 - Keeping an Echo Log **3-6**
 - Checking Your Loads **3-7**
 - Diagnosing One-Way Audio or No Audio **3-7**

- Checking MTP **3-7**
- Testing Cisco CallManager Configuration **3-7**
- Checking RTP Header Compression **3-8**
- Diagnosing Phone Resets **3-9**
- Diagnosing Dropped Calls **3-10**
- Diagnosing Cisco CallManager Feature Issues **3-11**
 - Diagnosing Codec and Region Mismatches **3-11**
 - Diagnosing Location and Bandwidth Problems **3-11**
 - Diagnosing Conference Bridge Problems **3-12**
 - Transcoding Problems **3-14**
 - MTP Resource Problems **3-16**
 - Diagnosing Dial Plan Problems **3-18**
 - Diagnosing Partition Problems **3-20**
 - Diagnosing Security Problems **3-23**
- Diagnosing Slow Server Response **3-23**
- Determining Gateway Reorder Tone Problems **3-23**
- Diagnosing Gateway Registration Problems **3-24**
- Diagnosing Gatekeeper Problems **3-31**
 - Inter-Cluster Trunks or H.225 Gateway **3-31**
 - Admission Rejects **3-31**
 - Registration Rejects **3-32**

APPENDIX A**Troubleshooting Intra-Cluster Phone Calls A-1**

- Sample Topology **A-2**
- Cisco IP Phone Initialization Process **A-2**
- Skinny Station Registration Process **A-3**
- Cisco IP Phone-to-Cisco IP Phone Call Flow within a Cluster **A-5**
- Cisco IP Phone-to-Cisco IP Phone Exchange of Skinny Station Messages During Call Flow **A-5**

- Cisco CallManager Initialization Process [A-6](#)
- Self-Starting Processes [A-7](#)
- Cisco CallManager Registration Process [A-8](#)
- Cisco CallManager KeepAlive Process [A-10](#)
- Cisco CallManager Intra-Cluster Call Flow Traces [A-11](#)

APPENDIX B

Troubleshooting Cisco IP Phone-to-Cisco IOS Gateway Calls [B-1](#)

- Call Flow Traces [B-2](#)
- Debug Messages and Show Commands on the Cisco IOS Gatekeeper [B-5](#)
- Debug Messages and Show Commands on the Cisco IOS Gateway [B-6](#)
- Cisco IOS Gateway with T1/PRI Interface [B-11](#)
- Cisco IOS Gateway with T1/CAS Interface [B-12](#)

APPENDIX C

Troubleshooting Inter-Cluster Calls [C-1](#)

- Sample Topology [C-1](#)
- Inter-Cluster H.323 Communication [C-1](#)
- Call Flow Traces [C-2](#)
- Failed Call Flow [C-4](#)

APPENDIX D

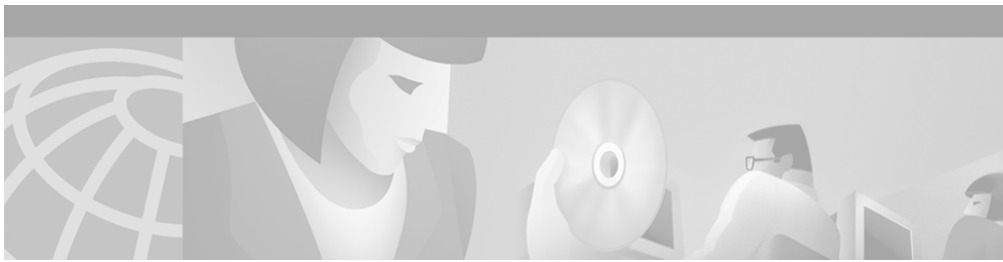
Call Detail and Call Management Records [D-1](#)

- Writing Records [D-2](#)
- Reading Records [D-2](#)
- Removing Records [D-3](#)
- Table Schema [D-4](#)
- Issues [D-4](#)
- Field Data Conversions [D-4](#)
 - Time Values [D-4](#)
 - Deciphering the Time Stamp [D-5](#)

- IP Addresses **D-5**
- Converting IP Addresses **D-5**
- CDR Field Definition **D-6**
- Call Records Logged By Call Type **D-20**
 - Normal Calls (Cisco IP Phone-to-Cisco IP Phone) **D-21**
 - Abandoned Calls **D-21**
 - Forwarded or Redirected Calls **D-21**
 - Calls With Busy or Bad Destinations **D-22**
- Call Management Records Logged By Call Type **D-22**
 - Normal Calls **D-22**
 - Abandoned Calls **D-23**
 - Forwarded Calls **D-23**
 - Calls With Busy or Bad Destinations **D-23**
- Codec Types (Compression / Payload types) **D-23**
- Cause Codes **D-24**
- Alarms **D-28**

GLOSSARY

INDEX



Preface

This preface describes the purpose, audience, organization, and conventions of this guide, and provides information on how to obtain related documentation.

The preface covers these topics:

- [Purpose, page ix](#)
- [Audience, page x](#)
- [Organization, page x](#)
- [Related Documentation, page xi](#)
- [Conventions, page xii](#)
- [Obtaining Documentation, page xiii](#)
- [Obtaining Technical Assistance, page xv](#)

Purpose

The *Troubleshooting Guide for Cisco CallManager* provides descriptions of the tools and utilities used to configure, monitor, and troubleshoot Cisco CallManager Release 3.1(1), Cisco IOS Gateways, and Gatekeeper.

Appendices provide detailed examples of three different call flows. In the first case study, a Cisco IP Phone calls another Cisco IP Phone within a cluster, which is called an intra-cluster call. Once you understand the call flow and debug traces, it will be easier to isolate a problem and determine which component is causing the problem.

Audience

The *Troubleshooting Guide for Cisco CallManager* is written for network administrators responsible for managing the Cisco CallManager system, enterprise managers, and employees. This guide requires knowledge of telephony and IP networking technology.

Organization

Table 1 shows how this guide is organized.

Table 1 *How This Document is Organized*

Chapter and Title	Description
Chapter 1, “Overview”	Provides an overview of the tools available for troubleshooting potential problems you may have with Cisco CallManager.
Chapter 2, “Troubleshooting and Monitoring Tools and Utilities”	Describes the tools and utilities you can use to configure, monitor, and troubleshoot Cisco CallManager.
Chapter 3, “Diagnosing Cisco CallManager Problems”	Describes how to diagnose some of the common problems that you may experience with Cisco CallManager.
Appendix A, “Troubleshooting Intra-Cluster Phone Calls”	Describes a case study on the call flow between two Cisco IP Phones within a cluster.
Appendix B, “Troubleshooting Cisco IP Phone-to-Cisco IOS Gateway Calls”	Describes a case study on a Cisco IP Phone calling through a Cisco IOS Gateway to a phone connected through a local PBX or on the PSTN.

Table 1 *How This Document is Organized (continued)*

Chapter and Title	Description
Appendix C, “Troubleshooting Inter-Cluster Calls”	Describes a case study on a Cisco IP Phone calling another Cisco IP Phone located in a different cluster.
Appendix D, “Call Detail and Call Management Records”	Provides detailed information about CDRs and CMRs.

Related Documentation

Refer to the following documents for further information about related Cisco IP Telephony applications and products:

- *Cisco CallManager Administration Guide*
- *Cisco CallManager System Guide*
- *Cisco CallManager Serviceability Administration Guide*
- *Installing Cisco CallManager Release 3.1*
- *Release Notes for Cisco CallManager Release 3.1(1)*
- *Hardware Configuration Guide for the Cisco Voice Gateway 200*
- *Software Configuration Guide for the Cisco Voice Gateway 200*
- *Cisco IP Phone Administration Guide for Cisco CallManager*
- *Bulk Administration Tool Guide for Cisco CallManager*
- *Administrative Reporting Tool Guide for Cisco CallManager*

Conventions

This document uses the following conventions:

Convention	Description
boldface font	Commands and keywords are in boldface .
<i>italic font</i>	Arguments for which you supply values are in <i>italics</i> .
[]	Elements in square brackets are optional.
{ x y z }	Alternative keywords are grouped in braces and separated by vertical bars.
[x y z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
screen font	Terminal sessions and information the system displays are in <code>screen font</code> .
boldface screen font	Information you must enter is in boldface screen font .
<i>italic screen font</i>	Arguments for which you supply values are in <i>italic screen font</i> .
< >	Nonprinting characters, such as passwords, are in angle brackets.

Notes use the following conventions:



Note

Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Timesavers use the following conventions:

**Timesaver**

Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Tips use the following conventions:

**Tips**

Means *the information contains useful tips*.

Cautions use the following conventions:

**Caution**

Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Warnings use the following conventions:

**Warning**

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, you must be aware of the hazards involved with electrical circuitry and familiar with standard practices for preventing accidents.

Obtaining Documentation

The following sections provide sources for obtaining documentation from Cisco Systems.

World Wide Web

You can access the most current Cisco documentation on the World Wide Web at the following sites:

- <http://www.cisco.com>
- <http://www-china.cisco.com>
- <http://www-europe.cisco.com>

Documentation CD-ROM

Cisco documentation and additional literature are available in a CD-ROM package, which ships with your product. The Documentation CD-ROM is updated monthly and may be more current than printed documentation. The CD-ROM package is available as a single unit or as an annual subscription.

Ordering Documentation

Cisco documentation is available in the following ways:

- Registered Cisco Direct Customers can order Cisco Product documentation from the Networking Products MarketPlace:
http://www.cisco.com/cgi-bin/order/order_root.pl
- Registered Cisco.com users can order the Documentation CD-ROM through the online Subscription Store:
<http://www.cisco.com/go/subscription>
- Nonregistered Cisco.com users can order documentation through a local account representative by calling Cisco corporate headquarters (California, USA) at 408 526-7208 or, in North America, by calling 800 553-NETS(6387).

Documentation Feedback

If you are reading Cisco product documentation on the World Wide Web, you can submit technical comments electronically. Click **Feedback** in the toolbar and select **Documentation**. After you complete the form, click **Submit** to send it to Cisco.

You can e-mail your comments to bug-doc@cisco.com.

To submit your comments by mail, use the response card behind the front cover of your document, or write to the following address:

Attn Document Resource Connection
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

Obtaining Technical Assistance

Cisco provides Cisco.com as a starting point for all technical assistance. Customers and partners can obtain documentation, troubleshooting tips, and sample configurations from online tools. For Cisco.com registered users, additional troubleshooting tools are available from the TAC website.

Cisco.com

Cisco.com is the foundation of a suite of interactive, networked services that provides immediate, open access to Cisco information and resources at anytime, from anywhere in the world. This highly integrated Internet application is a powerful, easy-to-use tool for doing business with Cisco.

Cisco.com provides a broad range of features and services to help customers and partners streamline business processes and improve productivity. Through Cisco.com, you can find information about Cisco and our networking solutions, services, and programs. In addition, you can resolve technical issues with online technical support, download and test software packages, and order Cisco learning materials and merchandise. Valuable online skill assessment, training, and certification programs are also available.

Customers and partners can self-register on Cisco.com to obtain additional personalized information and services. Registered users can order products, check on the status of an order, access technical support, and view benefits specific to their relationships with Cisco.

To access Cisco.com, go to the following website:

<http://www.cisco.com>

Technical Assistance Center

The Cisco TAC website is available to all customers who need technical assistance with a Cisco product or technology that is under warranty or covered by a maintenance contract.

Contacting TAC by Using the Cisco TAC Website

If you have a priority level 3 (P3) or priority level 4 (P4) problem, contact TAC by going to the TAC website:

<http://www.cisco.com/tac>

P3 and P4 level problems are defined as follows:

- P3—Your network performance is degraded. Network functionality is noticeably impaired, but most business operations continue.
- P4—You need information or assistance on Cisco product capabilities, product installation, or basic product configuration.

In each of the above cases, use the Cisco TAC website to quickly find answers to your questions.

To register for Cisco.com, go to the following website:

<http://www.cisco.com/register/>

If you cannot resolve your technical issue by using the TAC online resources, Cisco.com registered users can open a case online by using the TAC Case Open tool at the following website:

<http://www.cisco.com/tac/caseopen>

Contacting TAC by Telephone

If you have a priority level 1 (P1) or priority level 2 (P2) problem, contact TAC by telephone and immediately open a case. To obtain a directory of toll-free numbers for your country, go to the following website:

<http://www.cisco.com/warp/public/687/Directory/DirTAC.shtml>

P1 and P2 level problems are defined as follows:

- P1—Your production network is down, causing a critical impact to business operations if service is not restored quickly. No workaround is available.
- P2—Your production network is severely degraded, affecting significant aspects of your business operations. No workaround is available.



Overview

This document describes the tools available to troubleshoot potential problems you may have with Cisco CallManager. Also, this document describes call traces and debug outputs and how to use them to diagnose problems with call flow and call events.

This chapter contains the following topics:

- [Understanding Cisco IP Telephony Topology, page 1-1](#)
- [Topology Checklist, page 1-2](#)

Understanding Cisco IP Telephony Topology

It is very important to have an accurate topology, access to all the network devices, and terminal services for management of the Cisco CallManager.

Adding IP telephony to a new or existing network requires significant planning to ensure success. Since real-time voice traffic has different requirements than data traffic, the network must be designed with low latency and Quality-of-Service (QoS) in mind. As with any network that carries mission-critical traffic, it is imperative that the network administrator maintains accurate, detailed diagrams of the network topology. In a crisis situation, it is important to know not just the broad overview of the network, but also which ports are connected to network components, such as routers, switches, Cisco CallManager servers, gateways, and other critical devices. Plan the network with redundancy and scalability in mind.

**Caution**

Cisco does not support using hubs for shared connectivity to the switches as they can interfere with correct operation of the IP telephony system.

Topology Checklist

Use the following checklist to be sure you have properly documented your network topology. You will need:

- A topology diagram that shows all network devices and critical components with the port or interface numbers to which they are attached, and to which VLAN (if applicable) they belong. Special designations should be used for ports that are configured in trunking or channeling mode.
- The root of the spanning-tree should be configured and all normally blocking ports should be identified.
- Any WAN circuits should be identified with the amount of bandwidth (or CIR, in the case of frame-relay) they carry.

Any WAN interface will require special consideration, since this is a potential source of congestion. Cisco IP Phones and gateways set the RTP stream IP precedence field to five; however, this only tags the RTP packet. It is up to the network administrator to ensure that the network is configured for prioritization and call admission control so that the Voice over IP (VoIP) traffic can be serviced with minimal delay and contention for resources.

For additional information on this topic, see:

<http://www.cisco.com/warp/public/793/voip/>



Troubleshooting and Monitoring Tools and Utilities

This chapter addresses the tools and utilities you use to configure, monitor, and troubleshoot Cisco CallManager.

This chapter contains the following topics:

- [Using Cisco CallManager Administration Details, page 2-1](#)
- [Using Microsoft Performance, page 2-2](#)
- [Using Microsoft Event Viewer, page 2-3](#)
- [Using SDI Trace, page 2-4](#)
- [Using SDL Trace, page 2-5](#)
- [Sniffer Trace, page 2-8](#)
- [Call Detail Records and Call Management Records, page 2-9](#)
- [Using Admin Serviceability Tool, page 2-11](#)

Using Cisco CallManager Administration Details

Cisco CallManager Administration provides version information for the system, database, and other components.

Procedure

- Step 1** From the Cisco CallManager Administration window, click **Details**.
- Step 2** Write down the version you are using.
-

Refer to the *Cisco CallManager Administration Guide* for more details.

Using Microsoft Performance

You can use Windows 2000 Performance to collect and display system and device statistics for any Cisco CallManager installation. This administrative tool allows you to gain a full understanding of your system without studying the operation of each of its components.

You can use Performance to monitor a variety of system variables in real time. After adding the Cisco CallManager parameters, you can define the terms that will display statistics generated by the system. For example, you can monitor the number of calls in progress at any time, or the number of calls currently passing through a specific gateway. Performance shows both general and Cisco CallManager-specific status information in real-time.

Opening Microsoft Performance

To open Performance on the server PC running Cisco CallManager:

Click **Start > Settings > Control Panel > Administration Tools > Performance**

Customizing Performance

The Performance monitor must be customized to view the Cisco CallManager-related parameters that you want to monitor. Choose the object, counter, and instance you want to include.

Refer to the *Cisco CallManager Serviceability Administration Guide* for instructions on how to use objects and counters to customize Microsoft Performance for Cisco CallManager operations.

This document is available online at the following location:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm

Using Microsoft Event Viewer

You can use Microsoft Event Viewer to display system, security, and application events (including Cisco CallManager) for the Windows NT Server. If a service (including TFTP) cannot read the database (where it gets the trace configuration), it will add errors to the Event Viewer.

Opening Event Viewer

To open the Event Log on the server PC running Cisco CallManager:

Click **Start > Settings > Control Panel > Administrative Tools > Event Viewer**. The Event Viewer provides error logs for System, Security, and Applications. Cisco CallManager errors are logged under the Application log.

Refer to the Alarms chapter in the *Cisco CallManager Serviceability Administration Guide*, available online at the following location:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm

Detailed Information about Events

You can double-click an event in the log to learn more information about the event.

Using SDI Trace

You can use SDI traces to view local log files. To monitor the occurrence or the disposition of a request, you can trace the IP address, TCP handle, device name or time stamp. This device name could be tracked back to the building of the file, which shows the device pool and model. The device pool and model can be tracked back to the building of the configuration file prototype, which will list the network address of the Cisco CallManager(s) and the TCP connection port.

When observing SDI traces, notice that C++ class and routine names are included with most trace lines. Most routines associated with the serving of a particular request include the thread ID in a standard format.

SDI traces are explained in detail in the case studies in the appendices.

**Note**

Cisco CallManager Release 3.1 Serviceability supports the Trace interface. Refer to the following online location for information on Trace:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm

SDI Trace Output

SDI traces generate files (for example, CCM000000000) that store traces of Cisco CallManager activities. These traces provide information about:

- Cisco CallManager initialization process
- registration process
- KeepAlive process
- call flow
- digit analysis
- related devices such as Cisco IP Phones, Gateways, Gatekeepers, and more.

This information can help you isolate problems when troubleshooting Cisco CallManager. To properly track the information you need—and only the information you need—it is important to understand how to set the options on the trace configuration interface.

If the trace is not configured properly, it will generate a large amount of information, making it very difficult to isolate problems. The following section explains how to properly configure a useful trace.

Configuring Traces

Traces are composed of user mask flags (also known as bits) and trace levels.

-
- Step 1** Open Cisco CallManager Administration.
 - Step 2** Choose **Application > Cisco CallManager Serviceability**.
 - Step 3** Choose **Trace > Configuration**.
-

Refer to the *Cisco CallManager Serviceability Administration Guide* for complete information about configuring Trace.

The *Cisco CallManager Serviceability Administration Guide* is located online at:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm

Using SDL Trace

Cisco engineers use SDL traces to find the cause of an error. You are not expected to understand the information contained in an SDL trace. However, while working with the Technical Assistance Center (TAC), you may be asked to enable the SDL trace and provide it to the TAC. SDL trace files can be saved to local directories, the Windows NT Event Viewer, and CiscoWorks 2000. To avoid any performance degradation on the server, be sure that after the trace has been captured, you turn off SDL tracing.

SDL trace provides a C interface to trace and alarms. Alarms are used to inform the administrator of unexpected events, such as being unable to access a file, database, Winsock, or being unable to allocate other operating system resources.

Enabling SDL Trace

SDL traces are enabled in the **Service > Service Parameter** area in the Cisco CallManager Administration. Remember that these traces should be turned on only when requested by a TAC engineer.

Refer to the *Cisco CallManager Serviceability Administration Guide* for complete information about configuring Trace.

The *Cisco CallManager Serviceability Administration Guide* is located online at:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm

Once SDL traces are enabled, collect the traces. If the traces are being sent to the local drive, then you can retrieve them in the Cisco\Trace subdirectory. Alternatively, the trace files can be sent to an event log or to CiscoWorks 2000.

SDL flag bits described in the following table are set in the **Service > Service Parameters** area in Cisco CallManager Administration. Following are two examples of desired values based on the particular problem.

- The recommended value for normal call debugging is
SdlTraceTypeFlags=0x00000b04
- The recommended value for low level debugging or debugging gateways is
SdlTraceTypeFlags=0x00004b05

The following table defines the SdlTraceTypeFlags Definitions.

SDLTraceTypeFlag	Value	Definition
traceLayer1	= 0x00000001	All Layer 1 trace on
TraceDetailLayer1	= 0x00000002	Detail Layer 1 trace on
TraceSdlLinkAdmin	= 0x00000004	Trace inter-Cisco CallManager links within a cluster
traceUnused	= 0x00000008	Not used
traceLayer2	= 0x00000010	All Layer 2 trace on
traceLayer2Interface	= 0x00000020	Layer 2 interface trace on
traceLayer2TCP	= 0x00000040	Layer 2 TCP trace on

SDLTraceTypeFlag	Value	Definition
TraceDetailLayer2	= 0x00000080	More detail dump of Layer 2 Frames
traceLayer3	= 0x00000100	All Layer 3 trace on
traceCc	= 0x00000200	All call control trace on
traceMiscPolls	= 0x00000400	Trace miscellaneous polls
traceMisc	= 0x00000800	Miscellaneous trace on (Database signals)
traceMsgtrans	= 0x00001000	Message Translation signals (TranslateIsdnToSdlReq, TranslateIsdnToSdlRes, TranslateSdlToIsdnReq, TranslateSdlToIsdnRes)
traceUuie	= 0x00002000	UUIE output trace on
traceGateway	= 0x00004000	Gateway signals

Data bits described in the following table are set in the **Service > Service Parameters** area in Cisco CallManager Administration. Following are two examples of desired values based on the particular problem.

- The recommended value for normal system debugging is SdlTraceDataFlags=0x110
- The recommended value when tracking problems with SDL links is 0x13D (non-compacted trace; if a compact trace is desired, bit 0x200 must be set. It can be set in combination with any other bits)

The following table shows the SDLTraceDataFlags Definitions.

SDLTraceDataFlag	Value	Definition
TraceSdlLinkState	= 0x001	Enable trace of SDL Link Initialization
TraceSdlLowLevel	= 0x002	Enable tracing of low-level SDL events, for example, fileOpen, socket events, and so on
TraceSdlLinkPoll	= 0x004	Enable tracing of SDL Link Poll message
TraceSdlLinkMsg	= 0x008	Enable tracing of SDL Link message
traceRawData	= 0x010	Enable raw signal data trace on all signals
TraceSdlTagMap	= 0x020	Enable tag mapping
traceCreate	= 0x100	Enable process create and stop traces
TraceNoPrettyPrint	= 0x200	Disable pretty printing of trace files



Tips

Be advised that information obtained from this interface could be very detailed, and therefore consume a large amount of disk space. For this reason, we advise you to turn on the trace file for a specific amount of time, then review the information and turn off the trace.

Sniffer Trace

A Sniffer is a software application that monitors IP traffic on a network and provides information in the form of a trace. Sniffer traces provide information about the quantity and type of network traffic on your network. TCP/IP or UDP

packets are protocols utilized by Cisco CallManager and endpoint devices such as phones and gateways. Sniffer traces can also help you identify high levels of broadcast traffic that could result in voice audio problems or dropped calls.

Common Sniffer applications include Network Associates SnifferPro, Hewlett Packard Internet Advisor, and Acterna Domino. Domino offers sniffing hardware and software solutions and a network analyzer. If you want to use Domino, we recommend using the analysis software to evaluate a captured sniffer file (such as from the SnifferPro application).

Sniffer Trace Applications

Use the following links to learn more about some available sniffer trace applications. Any sniffer application will work with Cisco CallManager.

- Network Associates SnifferPro: <http://www.sniffer.com/>
- Hewlett Packard Network Analyzer:
http://www.hp.com/rnd/products/top_tools/summary.htm
- Acterna Domino Analyzer:
<http://www.acterna.com/products/domino/index.html>
- Shomiti Surveyor Protocol Analyzer:
<http://www.shomiti.com>

Call Detail Records and Call Management Records

Call Detail Records (CDR) is a reporting option that logs every call made (or attempted) from any Cisco IP Phone. There are two kinds of CDRs:

- basic CDRs
- Diagnostic CDRs, also known as CMRs

Once enabled, you can open CDRs or CMRs in the SQL Server Enterprise Manager. CDR files are saved in a SQL database that can be exported to nearly any application, including Microsoft Access or Excel.

Enabling or Disabling CDRs

CDR record creation is disabled by default when the system is installed. If you wish to have CDR data, you must enable CDRs in the **Service > Service Parameters** area of Cisco CallManager Administration. CDR processing can be enabled and disabled at any time while the system is in operation. You do not need to restart Cisco CallManager for the enabling or disabling of CDRs to take effect. The system will respond to all changes within a few seconds.

Using CMRs

Diagnostic CDRs provide detailed call information, such as the number of packets sent, received, and lost, the amount of jitter and latency, and so on. This level of detail can provide explanations for some problems, such as one-way audio. For example, a one-way audio problem is indicated if a packet size of 10,000 is sent, but the received size is only 10.

CMR or diagnostic data is enabled separately from CDR data. CMR data will not be generated unless both CDRs and Call Diagnostics are enabled, but CDR data may be generated and logged without CMR data.

Perform the following procedure to enable CDRs.

**Caution**

Tracing voice connectivity requires that CDR logging be enabled on every Cisco CallManager installation in a cluster.

Procedure

-
- Step 1** Open Cisco CallManager Administration.
 - Step 2** Choose **Service > Service Parameters**.
 - Step 3** Choose the IP address of your Cisco CallManager installation.
 - Step 4** Choose the Cisco CallManager service.
 - Step 5** From the list of Parameters, choose **CDREnabled**.
 - Step 6** Select **T** for True.

- Step 7** Click **Update**.
Result: Call Detail Records will start logging immediately.
-

Using Admin Serviceability Tool

You can use the Admin Serviceability Tool (AST) in Cisco CallManager Serviceability to monitor the real-time behavior of the components in a Cisco CallManager cluster. The AST uses HTTP and TCP to monitor device status, system performance, and device discovery. It also connects directly to devices using HTTP for troubleshooting system problems.

The *Cisco CallManager Serviceability Administration Guide* is located online at:
http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/service/index.htm



Diagnosing Cisco CallManager Problems

This chapter addresses some common problems that you may experience with Cisco CallManager and related devices. Each problem category provides suggestions for the troubleshooting tools you should use to help isolate the problem. This chapter provides general categories of potential problems and suggestions about how to troubleshoot those problems. It does not provide an exhaustive list of problems and resolutions.

If you encounter a problem that cannot be resolved using the tools and utilities described in this document, consult your Cisco Technical Assistance Center (TAC) for assistance. Be sure to have available the Cisco CallManager Administration Details, plus any diagnostic information (for example, traces) you have gathered before calling the TAC.

This chapter contains the following topics:

- [Voice Quality, page 3-2](#)
- [Diagnosing Phone Resets, page 3-9](#)
- [Diagnosing Dropped Calls, page 3-10](#)
- [Diagnosing Cisco CallManager Feature Issues, page 3-11](#)
- [Diagnosing Slow Server Response, page 3-23](#)
- [Determining Gateway Reorder Tone Problems, page 3-23](#)
- [Diagnosing Gateway Registration Problems, page 3-24](#)
- [Diagnosing Gatekeeper Problems, page 3-31](#)

Voice Quality

You may experience voice quality issues including lost or distorted audio signal during phone calls. Common problems include audio breaks (like broken words), or the presence of odd noises and audio distortion, such as echo, watery or robotic voice quality. One-way audio, that is, a conversation between two people where only one person can hear anything, is not actually a voice quality issue, but will be discussed later in this section.

You may experience audio problems with one or more of the following:

- Gateways
- Phones
- Networks

Diagnosing Lost or Distorted Audio Problems

One of the most common problems you may encounter is broken audio signal (often described as garbled speech, or lost syllables within a word or sentence). There are two common causes for this: packet loss and/or jitter. Packet loss means that audio packets do not arrive at their destination because they were dropped or arrived too late to be useful. Jitter is the variation in the arrival times of packets. In the ideal situation, all Voice over IP (VoIP) packets would arrive exactly at a rate of 1 every 20 microseconds (ms). Notice that this is not the length of time it takes for a packet to get from point A to point B, but is simply the variation in packet arrival times.

There are many sources of variable delay in a network. Some of these cannot be controlled, and some can. Variable delay cannot be eliminated entirely in a packetized voice network. Digital Signal Processors (DSP) on phones and other voice-capable devices are designed to buffer some of the audio, in anticipation of variable delay. This dejittering is done only when the audio packet has reached its destination and is now ready to be put into a conventional audio stream.

The Cisco IP Phone 7960 can buffer as much as one second of voice samples. The jitter buffer is adaptive, meaning if a burst of packets is received, the Cisco IP Phone 7960 can play them out in an attempt to control the jitter. The network administrator needs to minimize the variation between packet arrival times by applying Quality-of-Service (QoS) and other measures in advance (especially if calls cross a wide-area network).

When faced with a lost or distorted audio problem, the first thing to do is to try to isolate the path of the audio. Try to identify each network device (switches and routers) in the path of the call's audio stream. Keep in mind that the audio may be between two phones, or between a phone and a gateway, or it could have multiple legs (from a phone to a transcoding device and from there to another phone). Try to identify if the problem occurs only between two sites, only through a certain gateway, on a certain subnet, and so on. This will help narrow down which devices you need to look at more carefully.

Next, it is often best to disable silence suppression (also known as Voice Activation Detection or VAD) if this has not been done already. This mechanism does save bandwidth by not transmitting any audio when there is silence, but may cause noticeable or unacceptable clipping at the beginning of words.

You can disable this in Cisco CallManager Administration, under **Service > Service Parameters**. From there, select the server and the Cisco CallManager service. Then set `SilenceSuppressionSystemWide` to F (alternatively you can set `SilenceSuppressionWithGateways` to F, but this does not apply to H.323 gateways or MGCP gateways). When in doubt, turn both off by selecting the Value F for each.

Using a network analyzer, if a network analyzer is available, then a monitored call between two phones should have 50 packets per second (or 1 packet every 20 ms) when silence suppression is disabled. With proper filtering, it should be possible to identify if packets are being lost or delayed excessively.

Remember that delay by itself will not cause clipping, only variable delay will. Notice in the table below, which represents a perfect trace, the arrival times between the audio packets (which will have an RTP header), will be 20 ms. In a poor quality call (such as a call with a lot of jitter), the arrival times would vary greatly.

The following table illustrates a perfect trace.

Packet Number	Time - absolute (ms)	Time - delta (ms)
1	0	
2	0.02	20
3	0.04	20
4	0.06	20
5	0.08	20

Placing the packet analyzer into various points in the network will help narrow down where the delay is coming from. If no analyzer is available, other methods will be required. It is important to examine interface statistics of each device in the path of the audio. Another tool for tracking calls with poor voice quality is the Diagnostic Call Detail Records (CDR). See [Appendix D, “Call Detail and Call Management Records”](#) for more information about CDRs.

Diagnosing Audio Problems from the Cisco IP Phone

The Cisco IP Phone 7960 provides another tool for diagnosing possible audio problems. On an active call, you can press the “i” button twice rapidly and the phone will display an information screen that contains packet receive and transmit statistics, as well as average and maximum jitter counters.

**Note**

On this screen, jitter is the average of the last 5 packets that arrived; the maximum jitter is the high-water mark for the average jitter.

The most common sources for delay and packet loss are devices where a higher speed interface feeds into a lower speed interface. For example, a router may have a 100 Megabyte (MB) fast Ethernet interface connected to the LAN and a slow frame-relay interface, connected to the WAN. If the poor audio quality occurs only when communicating to the remote site, then the most likely causes of the problem include:

- The router has not been properly configured to give voice traffic priority over data traffic
- There are too many active calls for the WAN to support (that is, there is no call admission control to restrict the number of calls that can be placed)
- There are physical port errors
- There is congestion in the WAN itself

On the LAN, the most common problems are physical-level errors (such as CRC errors) caused by faulty cables, interfaces, or by incorrectly configured devices (such as a port speed or duplex mismatch). Make sure that the traffic is not crossing any shared-media device, such as a hub.

There could also be situations where the traffic is taking a slower path through the network than expected. If QoS has been configured correctly, then the possibility exists that there is no call admission control. Depending on your topology, this can be accomplished through the use of Locations in Cisco CallManager Administration configuration, or by using a Cisco IOS router as a gatekeeper. In any case, you should always know how many calls could be supported across your WAN.

If possible, test this by disabling silence suppression as described in the preceding sections, then place calls between the two sites. Do not place the calls on hold or on mute, because this will stop packets from being transmitted. With the maximum number of calls across the WAN, the calls should all have acceptable quality. Test to make sure that a fast busy is returned when trying to make one more call.

Diagnosing Crackling Sounds

Another poor quality symptom may be a crackling, which is sometimes caused by a defective power supply or some kind of strong electrical interference close to the phone. Try swapping the power supply and moving the phone.

Checking Your Loads

Gateway and phone loads should be verified. Check Cisco Connection Online (CCO) at www.cisco.com for the latest software loads, new patches, or release notes relating to the problem.

Echo

Echo occurs when the speech energy being generated and transmitted down the primary signal path is coupled into the receive path from the far end. The speaker then hears his or her own voice, delayed by the total echo path delay time.

Voice can reflect back. This can happen but go unnoticed in a traditional voice network because the delay is so low. To the user, it sounds more like a side-tone than an echo. In a VoIP network, it will always be noticeable, since packetization and compression contributes to the delay.

The important thing to remember is that the cause of the echo is always with analog components and wiring. For instance, IP packets cannot simply turn around and go back to the source at a lower audio level or on digital T1/E1 circuits. So, on a call from one Cisco IP Phone to another, there should never be any problem. The only exception may be if one party is using a speakerphone that has the volume set too high or some other situation where an audio loop is created.

Testing Speakerphone and Volume Levels

Make sure that the problem phones are not using the speakerphone, and that they have the headset volume set to reasonable levels (start with 50% of the maximum audio level). Most of the time, the problems will occur when attaching to the PSTN by way of a digital or analog gateway.

Testing the Gateway

Determine which gateway is being used. If a digital gateway is in use, then it may be possible to add additional padding in the transmit direction (towards the PSTN). Since lower signal strength will yield less reflected energy, this should clear up the problem.

Additionally, you can adjust the receive level so that any reflected audio is reduced even further. It is very important to remember to make small adjustments at a time. Too much attenuation of the signal will make the audio impossible to hear on both sides. Alternatively, you can contact the carrier and request to have the lines checked. On a typical T1/PRI circuit in North America, the input signal should be -15 dB. If the signal level is much higher (-5 dB, for example), then echo will be the likely result.

Keeping an Echo Log

You should keep a log of all calls that experience echo. Record the time of the problem, the source phone number, and the number called. Gateways have a fixed time of 16 ms of echo cancellation. If the delay in the reflected audio is longer than this, the echo cancellor will be unable to work properly. This should not be an issue for local calls, and long distance calls should have external echo cancellers built into the network at the Central Office. This is one of the reasons why it is important to note the external phone number of a call that experiences echo.

Checking Your Loads

Verify your gateway and phone loads. Check Cisco Connection Online at www.cisco.com for the latest software loads, new patches, or release notes that may relate to the problem.

Diagnosing One-Way Audio or No Audio

When one person cannot hear another person during a call, there is one-way audio. This can be caused by an improperly configured Cisco IOS Gateway, a firewall, or a routing or default gateway problem, among other things.

Diagnostic Call Detail Records (CDR) are useful for determining if a call is experiencing one-way audio because they log transmitted and received packets (see [Diagnosing Lost or Distorted Audio Problems, page 3-2](#)). You can also press the *i* button twice quickly on a Cisco IP Phone 7960 during an active call to view details about transmitted and received packets.

Checking MTP

If you are using Media Termination Point (MTP) in a call (to support supplementary services such as hold and transfer with H.323 devices that do not support H.323 version 2), check to see if the MTP allocated is working correctly. Cisco IOS routers support H.323 version 2 beginning in Cisco IOS releases 11.3(9)NA and 12.0(3)T. Starting with Cisco IOS release 12.0(7)T, the optional H.323 Open/Close LogicalChannel is supported, so that software-based MTP is no longer required for supplementary services.

You can use the MTP device, as well as Conference Bridge and Transcoder features to bridge two or more audio streams. If the MTP, Conference Bridge, or Transcoder is not working properly, one-way audio or audio loss might be experienced. Shut down MTP to find out if MTP is causing the problem.

Testing Cisco CallManager Configuration

There are a number of causes for one-way audio or loss of audio during a call. The most common cause is an improperly configured device. For instance, Cisco CallManager handles the call setup for a Cisco IP Phone. The actual audio stream occurs between the two Cisco IP Phones (or between the Cisco IP Phone

and a gateway). It is entirely possible that the Cisco CallManager is able to signal to a destination phone (making it ring) when the phone originating the call does not have an IP route to the destination phone. A common cause for this is when the default gateway in the phone is improperly configured either manually or on the DHCP server.

If a call consistently has one-way audio, use a PC that is configured on the same subnet as the phone and has the same default gateway and try to ping the destination Cisco IP Phone. Then using a PC that is configured on the same subnet as the destination phone (with the same default gateway as the destination phone) ping the source phone. Both of those tests should work. Other things that can affect the audio traffic include a firewall or a packet filter (such as access lists on a router) that may be blocking the audio in one or both directions. If the one-way audio occurs only through a voice-enabled Cisco IOS Gateway, then check the configuration carefully. IP routing must be enabled (look at the configuration to make sure that no ip routing is not found near the beginning of the configuration).

Checking RTP Header Compression

Make sure that if you are using RTP header compression to save bandwidth across the WAN, that it is enabled on each router carrying voice traffic that attaches to the WAN circuit. There should not be a situation where the RTP header is compressed on one end but cannot be decompressed on the other side of the WAN. A Sniffer is a very useful tool when troubleshooting one-way audio problems, because you can verify whether the phone or gateway is actually sending or receiving packets.



Note

When a call is muted, no packets will be transmitted from the phone that has pressed the mute button. The Hold button stops the audio stream, so no packets are sent in either direction. When the Hold button is released, all the packet counters are reset. Remember that Silence Suppression must be disabled on both devices for the TX and RX counters to stay equal. Disabling Silence Suppression system-wide will not affect Cisco IOS Gateways.

Diagnosing Phone Resets

Phones will power cycle or reset for two reasons: 1) TCP failure connecting to Cisco CallManager, or 2) failure to receive an acknowledgement to the phone's KeepAlive messages.

The following procedure shows the steps for troubleshooting phone resets.

Procedure

- Step 1** Check the phones and gateways to ensure that you are using the latest software loads.
 - Step 2** Check Cisco Connection Online at www.cisco.com for the latest software loads, new patches, or release notes that may relate to the problem.
 - Step 3** Check the Event Viewer for instances of phone(s) resetting. Phone resets are considered Information events.
 - Step 4** Look for these and any errors that may have occurred around the time that the phone(s) reset.
 - Step 5** Start an SDI trace and try to isolate the problem by identifying any common characteristics in the phones that are resetting. For example, check whether they are all located on the same subnet, same VLAN, and so on. Look at the trace and determine:
 - If the resets occur during a call or happen intermittently
 - If there are any similarities of phone model (such as Cisco IP Phone 7960 or Cisco IP Phone 30VIP)
 - Step 6** Start a Sniffer trace on a phone that frequently resets. After it has reset, look at the trace to determine if there are any TCP retries occurring. If so, this indicates a network problem. The trace may show some consistencies in the resets, such as the phone resetting every seven days. This might indicate DHCP lease expiration every seven days (this value is user-configurable; for example, it could be every two minutes).
-

Diagnosing Dropped Calls

Premature termination of dropped calls can be the result of a phone or gateway resetting (see [Diagnosing Phone Resets, page 3-9](#)) or a circuit problem, such as incorrect PRI configuration.

The first step is to determine whether this problem is isolated to one phone or to a group of phones. Perhaps the affected phones are all on a particular subnet or location. The next step is to check the Event Viewer for phone or gateway resets.

You will see one Warning and one Error message for each phone that resets. This indicates that the phone cannot keep its TCP connection to the Cisco CallManager alive, so the Cisco CallManager resets the connection. This may be because a phone was turned off or there may be a problem in the network. If this is an intermittent problem, it may be useful to use Microsoft Performance to record phone registrations.

If the problem seems to be occurring only through a certain gateway, such as a Cisco Access DT-24+, then the best course of action is to enable tracing and/or view the Call Detail Records (CDR). The CDR files will give a Cause Of Termination (CoT) that may help determine the cause of the problem. See [Appendix D, “Call Detail and Call Management Records”](#) for more information about CDRs.

The disconnect cause values (origCause_value and destCause_value— depending on which side hung up the call), map to Q.931 disconnect cause codes (in decimal) and can be found at the following location:

<http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/dbook/disdn.htm>.

If the call is going out of a gateway to the PSTN, then the CDR can be used to determine which side is hanging up the call. Much of the same information can be obtained by enabling tracing on the Cisco CallManager. Because the trace tool can affect Cisco CallManager performance, you will want to use this option only as a last resort or if your network is not yet in production.

Diagnosing Cisco CallManager Feature Issues

Problems may occur with features that are used in conjunction with Cisco CallManager, such as Conference Bridge or Media Termination Point. Some feature problems can be caused by configuration errors or lack of server resources.

Diagnosing Codec and Region Mismatches

If a user gets a reorder tone (busy signal) when going off-hook, it could be the result of codec disagreement between regions. Verify that both call ends support at least one common codec (for example, G.711). If they do not, you will need to use transcoders.

A region specifies the range of supported codecs that can be used with each of the other regions. Every device belongs to a region.

**Note**

Codec negotiation with a Cisco IOS router is not supported.

For example, Region1<->Region2 = G.711, means that a call between a device in Region1 and a device in Region2 can use G.711 or any other supported codec that requires the same or less bandwidth as G.711 (any supported codecs within G.711, G.729, G.723, and so on).

**Note**

The following codecs are supported for each device:
Cisco IP Phone 7960—G.711A-law/μ-law, G.729AnnexB
Cisco IP Phone SP12 series and VIP 30—G.711A-law/μ-law, G.723.1
Cisco Access Gateway DE30 and DT-24+—G.711A-law/μ-law, G.723.1

Diagnosing Location and Bandwidth Problems

If a user gets a reorder tone after dialing a number, it could be because the Cisco CallManager bandwidth allocation for the location of one of the call end devices has been exceeded. Cisco CallManager checks for 24 kilobit available

bandwidth for each device before making a call. If less than 24k bandwidth is available, Cisco CallManager will not set up the call and the user will hear a reorder tone.

```
12:42:09.017 Cisco CallManager|Locations:Orig=1 BW=12Dest=0 BW=-1(-1
implies infinite bw available)
12:42:09.017 Cisco CallManager|StationD - stationOutputCallState
tcpHandle=0x4f1ad98
12:42:09.017 Cisco CallManager|StationD - stationOutputCallInfo
CallingPartyName=, CallingParty=5003, CalledPartyName=,
CalledParty=5005, tcpHandle=0x4f1ad98
12:42:09.017 Cisco CallManager|StationD - stationOutputStartTone:
37=ReorderTone tcpHandle=0x4f1ad98
```

Once the call is established, the Cisco CallManager will subtract bandwidth from the locations depending on the codec used in that call. If the call is using G.711, Cisco CallManager will subtract 80k; if the call is using G.723, Cisco CallManager will subtract 24k; if the call is using G.729, Cisco CallManager will subtract 24k.

Diagnosing Conference Bridge Problems

Use the following information to troubleshoot a No Conference Bridge available error message. This could indicate either a software or a hardware problem.

First, check to see if you have any available software or hardware Conference Bridge resources registered with Cisco CallManager. Use either Microsoft Performance or the Admin Serviceability Tool to check the number of Unicast AvailableConferences.



Note

Cisco CallManager Release 3.1 uses different names for counters and objects. Refer to the *Cisco CallManager Serviceability Administration Guide* for more information.

The Cisco IP Voice Media Streaming application performs the conference bridge function. One software installation of Cisco IP Voice Media Streaming will support 16 Unicast Available Conferences (3 people/conference), as shown in the following trace.

**Note**

The number of supported devices may vary with different Cisco CallManager releases. Refer to the release 3.1 documentation at the following location:
http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm

```
10:59:29.951 Cisco CallManager|UnicastBridgeControl -
wait_capabilities_StationCapRes - Device= CFB_kirribilli - Registered
- ConfBridges= 16, Streams= 48, tcpHandle=4f12738
10:59:29.951 Cisco CallManager|UnicastBridgeManager -
UnicastBridgeRegistrationReq - Device Registration Complete for Name=
Xoø ô%ø - DeviceType= 50, ResourcesAvailable= 16, deviceTblIndex= 0
```

One E1 port (WS-X6608-E1 card contains 8x E1 ports) provides five Unicast Available Conferences (max conference size = 6), as shown in the following trace.

```
11:14:05.390 Cisco CallManager|UnicastBridgeControl -
wait_capabilities_StationCapRes - Device= CFB00107B000FB0 - Registered
- ConfBridges= 5, Streams= 16, tcpHandle=4f19d64
11:14:05.480 Cisco CallManager|UnicastBridgeManager -
UnicastBridgeRegistrationReq - Device Registration Complete for Name=
Xoø ô%ø - DeviceType= 51, ResourcesAvailable= 5, deviceTblIndex= 0
```

The following hardware trace on the Cisco Catalyst 6000 8 Port Voice T1/E1 and Services Module, indicates that the E1 port 4/1 in the card has registered as a Conference Bridge with Cisco CallManager.

```
greece-sup (enable) sh port 4/1
Port Name Status Vlan Duplex Speed Type
-----
4/1 enabled 1 full -Conf Bridge

Port DHCP MAC-Address IP-Address Subnet-Mask
-----
4/1 disable 00-10-7b-00-0f-b0 10.200.72.31 255.255.255.0

Port Call-Manager(s) DHCP-Server TFTP-Server Gateway
-----
4/1 10.200.72.25 - 10.200.72.25 -

Port DNS-Server(s) Domain
-----
4/1 - 0.0.0.0

Port CallManagerState DSP-Type
```

```

-----
4/1      registered      C549

Port  NoiseRegen NonLinearProcessing
-----
4/1  disabled    disabled

```

Second, check the maximum number of users configured in your Ad Hoc or Meet-Me conference to determine if the problem occurred because this number was exceeded.

Transcoding Problems

If you have installed a hardware transcoder in the Cisco Catalyst 6000 8 Port Voice T1/E1 and Services Module, and it does not work as expected (you cannot make calls between two users with no common codec), check to see if you have any available Transcoder resources registered with Cisco CallManager (must be hardware). Use Microsoft Performance or the Admin Serviceability Tool to check the number of MediaTermPointsAvailable available.



Note

Cisco CallManager Release 3.1 uses different names for counters and objects. Refer to the *Cisco CallManager Serviceability Administration Guide* for more information.

One E1 port (WS-X6608-E1 card contains 8x E1 ports) provides Transcoder/MTP resources for 16 calls, as shown in the following trace.



Note

The number of supported devices may vary with different Cisco CallManager releases. Refer to the release 3.1 documentation at the following location: http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm

```

11:51:09.939 Cisco CallManager|MediaTerminationPointControl -
Capabilities Received - Device= MTP00107B000FB1 - Registered -
Supports 16 calls

```

The following hardware trace on the Cisco Catalyst 6000 8 Port Voice T1/E1 and Services Module, indicates that the E1 port 4/2 in the card has registered as an MTP/transcoder with Cisco CallManager.

```

greece-sup (enable) sh port 4/2
Port Name Status Vlan Duplex Speed Type
-----
4/2 enabled 1 full - MTP

Port DHCP MAC-Address IP-Address Subnet-Mask
-----
4/2 disable 00-10-7b-00-0f-b1 10.200.72.32 255.255.255.0

Port Call-Manager(s) DHCP-Server TFTP-Server Gateway
-----
4/2 10.200.72.25 - 10.200.72.25 -

Port DNS-Server(s) Domain
-----
4/2 - 0.0.0.0

Port CallManagerState DSP-Type
-----
4/2 registered C549

Port NoiseRegen NonLinearProcessing
-----
4/2 disabled disabled

```

**Note**

The same E1 port cannot be configured for both Conference Bridge and Transcoder/MTP

To make a call between two devices using a low bit rate code (such as G.729 and G.723) that do not support the same codec, a transcoder resource is required.

Assume Cisco CallManager has been configured such that the codec between Region1 and Region2 is G.729. The following scenarios are possible:

- If caller on Phone A initiates a call, Cisco CallManager realizes it is a Cisco IP Phone 7960, which supports G.729. After the digits have been collected, the Cisco CallManager determines that the call is destined for User D who is in Region2. Since the destination device also supports G.729, the call is set up and the audio flows directly between Phone A and Phone D.
- If a caller on Phone B, who has a Cisco IP Phone 12SP+, were to initiate a call to Phone D, then this time the Cisco CallManager would realize that the originating phone only supports G.723 or G.711. Cisco CallManager would need to allocate a transcoding resource so that audio would flow as G.711

between Phone B and the transcoder, but as G.729 between the transcoder and Phone D. If no transcoder were available, Phone D's phone would ring, but as soon as the call was answered, the call would disconnect.

- If a user on Phone B were to call Phone F, which is a Cisco IP Phone 12SP+, the two phones would actually use G.723, even though G.729 is configured as the codec to use between the regions. G.723 is used because both endpoints support it and it uses less bandwidth than G.729.
- If a Cisco uOne voice mail system is added (which only supports G.711) or a Cisco IOS router configured for G.711 to Region1, then a transcoding device must be used if calling from Region2. If none is available, then the call will fail.

MTP Resource Problems

An MTP resource problem could be the source of the transcoding problem if a call is established, but supplementary services are not available on an H.323 device that does not support H323v2. First, determine whether you have any available software or hardware MTP resources registered with Cisco CallManager. Use Microsoft Performance or the Admin Serviceability Tool to check the number of MediaTermPointsAvailable.



Note

Cisco CallManager Release 3.1 uses different names for counters and objects. Refer to the *Cisco CallManager Serviceability Administration Guide* for more information.

Using MTP to support supplementary services with H.323 devices that do not support H.323v2 allows one MTP software application to support 24 calls as shown in the following trace.



Note

The number of supported devices may vary with different Cisco CallManager releases. Refer to the release 3.1 documentation at the following location: http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm

```
10:12:19.161 Cisco CallManager|MediaTerminationPointControl -
Capabilities Received - Device= MTP_kirribilli. - Registered -
Supports 24 calls
```

One E1 port (WS-X6608-E1 card contains 8x E1 ports) provides MTP resources for 16 calls, as shown in the following trace.

```
11:51:09.939 Cisco CallManager|MediaTerminationPointControl -
Capabilities Received - Device= MTP00107B000FB1 - Registered -
Supports 16 calls
```

The following hardware trace from the Cisco Catalyst 6000 8 Port Voice T1/E1 and Services Module, indicates that the E1 port 4/2 in the card has registered as an MTP/transcoder with Cisco CallManager.

```
greece-sup (enable) sh port 4/2
Port  Name                Status      Vlan      Duplex Speed Type
-----
4/2                enabled      1          full    - MTP

Port      DHCP      MAC-Address      IP-Address      Subnet-Mask
-----
4/2      disable  00-10-7b-00-0f-b1  10.200.72.32    255.255.255.0

Port      Call-Manager(s)  DHCP-Server      TFTP-Server      Gateway
-----
4/2      10.200.72.25    -                  10.200.72.25    -

Port      DNS-Server(s)    Domain
-----
4/2      -                0.0.0.0

Port      CallManagerState DSP-Type
-----
4/2      registered      C549

Port      NoiseRegen NonLinearProcessing
-----
4/2      disabled      disabled
```

Second, in the Gateway Configuration screen of Cisco CallManager Administration, check to see if the Media Termination Point Required box is checked.

Third, verify that Cisco CallManager has allocated the required number of MTP devices.

From the SDI file:

```

15:22:23.848 Cisco CallManager|MediaManager(40) started
15:22:23.848 Cisco CallManager|MediaManager - wait_AuConnectRequest
15:22:23.848 Cisco CallManager|MediaManager - wait_AuConnectRequest -
Transcoder Enabled
15:22:23.848 Cisco CallManager|MediaManager - wait_AuConnectRequest -
party1(16777357), party2(16777358), proxies=1, connections=2, current
proxies=0
15:22:23.848 Cisco CallManager|MediaManager - wait_AuConnectRequest -
proxy connections
15:22:23.848 Cisco CallManager|MediaManager - wait_AuConnectRequest -
allocating MTP(ci=16777359)
15:22:23.848 Cisco CallManager|MediaManager -
wait_AllocateMtpResourceRes
15:22:23.848 Cisco CallManager|MediaManager -
wait_AllocateMtpResourceRes - start 2 connections
15:22:23.848 Cisco CallManager|MediaManager -
wait_AllocateMtpResourceRes - creating connection between
party1(16777357) and party2(16777359)
15:22:23.848 Cisco CallManager|MediaManager -
wait_AllocateMtpResourceRes - creating connection between
party1(16777358) and party2(16777359)
15:22:23.848 Cisco CallManager|MediaCoordinator -
wait_MediaCoordinatorAddResource - CI=16777359 count=1
15:22:23.848 Cisco CallManager|MediaCoordinator -
wait_MediaCoordinatorAddResource - CI=16777359 count=2

```

Diagnosing Dial Plan Problems

A Dial Plan is a list of numbers and groups of numbers that tells the Cisco CallManager what devices (such as phones and gateways) to send calls to when a certain string of digits is collected. It is analogous to a static routing table in a router. Please be certain your dial plan concepts, basic call routing, and planning have been carefully considered and properly configured before trying to troubleshoot a potential dial plan issue. Very often, the problem lies with planning and configuration. Refer to the route plan configuration chapters in the *Cisco CallManager Administration Guide* for more information.

Use the following tips to help troubleshoot dial plans problems:

- Identify the Directory Number (DN) that is originating the call.
- Identify the Calling Search Space for this DN.

- If applicable, identify which device the Calling Search Space is associated with this DN. Make sure that you identify the correct device; because multiple line appearances are supported, it is possible to have the same DN on multiple devices. Note the device's Calling Search Space. If this is a Cisco IP Phone originating the call, remember that a particular line (DN) and the device that line is associated with have Calling Search Spaces. They will be combined when making a call. For example, if line instance 1000 has a Calling Search Space of AccessLevelX and the Cisco IP Phone that has extension 1000 configured on it has AccessLevelY as its Calling Search Space, then when making a call from that line appearance, Cisco CallManager will search through partitions contained in Calling Search Space AccessLevelX and AccessLevelY.
- Identify which Partitions are associated with the Calling Search Space(s).
- Identify which Partition of the device to which the call should (or should not) go.
- Identify which number is being dialed. Note that if and when the user is getting a secondary dial tone. Also note what they hear after all the digits have been entered (re-order, fast-busy). Does the user get the progress tones before expecting to hear anything? Make sure callers wait at least 10 seconds after typing the last digit because they may have to wait for the inter-digit timer to expire.
- Generate a Route Plan Report in Cisco CallManager Administration, and use it to examine all the route patterns for the partitions that are in the Calling Search Space for the problem call.
- If necessary, add or modify the Route Patterns or Route Filters.
- If you can find the Route Pattern to which the call is being sent, note the Route List or Gateway to which the pattern points.
- If it is a Route List, check which Route Groups are part of the list and which Gateway(s) is part of the Route Groups.
- Verify that the applicable devices are registered with Cisco CallManager.
- If there is no access to Cisco CallManager, use the show tech command to capture and verify this information.
- Pay attention to the @ sign. This is a macro that can expand to include many different things. It is often used in combination with filtering options.

- If a device is not part of a partition, it is said to be part of the Null or default partition. Every user should be able to call that device. The Null partition is always searched last.
- If you dial an outside number that is matching a 9.@ pattern and it takes 10 seconds before the call goes through, check the filtering options. By default, with a 9.@ pattern, when dialing a 7-digit number, the Cisco IP Phone will wait 10 seconds before placing the call. You need to apply a Route Filter to the pattern that says LOCAL-AREA-CODE DOES-NOT- EXIST and END-OF-DIALING DOES-NOT-EXIST.

Diagnosing Partition Problems

Route partitions inherit the error handling capabilities for the Cisco CallManager software. That is, a console and SDI file trace are provided for logging information and error messages. These messages will be part of the digit analysis component of the traces. It is vital that you know how the Partitions and Calling Search Spaces are configured and what devices are in each partition and its associated calling search space to determine the source of the problem. Refer to the route plan chapters in the *Cisco CallManager Administration Guide* and the *Cisco CallManager System Guide* for more information.

The following trace is an example of a number dialed that is in the device's Calling Search Space. For more detailed explanations about SDI traces, review the case studies in this document.

```
08:38:54.968 Cisco CallManager|StationInit - InboundStim -
OffHookMessageID tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|StationD - stationOutputDisplayText
tcpHandle=0x6b88028, Display= 5000
08:38:54.968 Cisco CallManager|StationD - stationOutputSetLamp stim:
9=Line instance=1 lampMode=LampOn tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|StationD - stationOutputCallState
tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|StationD -
stationOutputDisplayPromptStatus tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|StationD - stationOutputSelectSoftKeys
tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|StationD -
stationOutputActivateCallPlane tcpHandle=0x6b88028
08:38:54.968 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood:RTP_NC_Woodland:Local RTP", dd="")
```

In the Digit Analysis component of the previous trace, the pss (Partition Search Space, also known as Calling Search Space) is listed for the device placing the call.

In the following trace, RTP_NC_Hardwood;RTP_NC_Woodland;Local_RTP are the partitions this device is allowed to call.

```
08:38:54.968 Cisco CallManager|Digit analysis:
potentialMatches=PotentialMatchesExist
08:38:54.968 Cisco CallManager|StationD - stationOutputStartTone:
33=InsideDialTone tcpHandle=0x6b88028
08:38:55.671 Cisco CallManager|StationInit - InboundStim -
KeypadButtonMessageID kpButton: 5 tcpHandle=0x6b88028
08:38:55.671 Cisco CallManager|StationD - stationOutputStopTone
tcpHandle=0x6b88028
08:38:55.671 Cisco CallManager|StationD - stationOutputSelectSoftKeys
tcpHandle=0x6b88028
08:38:55.671 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood;RTP_NC_Woodland;Local RTP", dd="5")
08:38:55.671 Cisco CallManager|Digit analysis:
potentialMatches=PotentialMatchesExist
08:38:56.015 Cisco CallManager|StationInit - InboundStim -
KeypadButtonMessageID kpButton: 0 tcpHandle=0x6b88028
08:38:56.015 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood;RTP_NC_Woodland;Local RTP", dd="50")
08:38:56.015 Cisco CallManager|Digit analysis:
potentialMatches=PotentialMatchesExist
08:38:56.187 Cisco CallManager|StationInit - InboundStim -
KeypadButtonMessageID kpButton: 0 tcpHandle=0x6b88028
08:38:56.187 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood;RTP_NC_Woodland;Local RTP", dd="500")
08:38:56.187 Cisco CallManager|Digit analysis:
potentialMatches=PotentialMatchesExist
08:38:56.515 Cisco CallManager|StationInit - InboundStim -
KeypadButtonMessageID kpButton: 3 tcpHandle=0x6b88028
08:38:56.515 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood;RTP_NC_Woodland;Local RTP", dd="5003")
08:38:56.515 Cisco CallManager|Digit analysis: analysis results
08:38:56.515 Cisco CallManager||PretransformCallingPartyNumber=5000
```

The key thing to note is that PotentialMatchesExist is the result of Digit Analysis of the numbers that were dialed until the exact match is found and the call is routed accordingly.

The following trace describes what happens when the Cisco CallManager is attempting to dial the directory number 1001 and it is not in the Calling Search Space for that device. Again, the key thing to note is the digit analysis routine had

potential matches until only the first digit was dialed. The route pattern associated with the digit 1 is in a partition that is not in the device's calling search space, RTP_NC_Hardwood;RTP_NC_Woodland;Local_RTP. Therefore, the phone received a reorder tone (busy signal).

```
08:38:58.734 Cisco CallManager|StationInit - InboundStim -
OffHookMessageID tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|StationD - stationOutputDisplayText
tcpHandle=0x6b88028, Display= 5000
08:38:58.734 Cisco CallManager|StationD - stationOutputSetLamp stim:
9=Line instance=1 lampMode=LampOn tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|StationD - stationOutputCallState
tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|StationD -
stationOutputDisplayPromptStatus tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|StationD - stationOutputSelectSoftKeys
tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|StationD -
stationOutputActivateCallPlane tcpHandle=0x6b88028
08:38:58.734 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood:RTP_NC_Woodland:Local RTP", dd="")
08:38:58.734 Cisco CallManager|Digit analysis:
potentialMatches=PotentialMatchesExist
08:38:58.734 Cisco CallManager|StationD - stationOutputStartTone:
33=InsideDialTone tcpHandle=0x6b88028
08:38:59.703 Cisco CallManager|StationInit - InboundStim -
KeypadButtonMessageID kpButton: 1 tcpHandle=0x6b88028
08:38:59.703 Cisco CallManager|StationD - stationOutputStopTone
tcpHandle=0x6b88028
08:38:59.703 Cisco CallManager|StationD - stationOutputSelectSoftKeys
tcpHandle=0x6b88028
08:38:59.703 Cisco CallManager|Digit analysis: match(fqcn="5000",
cn="5000", pss="RTP_NC_Hardwood:RTP_NC_Woodland:Local RTP", dd="1")
08:38:59.703 Cisco CallManager|Digit analysis:
potentialMatches=NoPotentialMatchesExist
08:38:59.703 Cisco CallManager|StationD - stationOutputStartTone:
37=ReorderTone tcpHandle=0x6b88028
```

Route partitions work by associating a partition name with every directory number in the system. The directory number can be called only if the calling device contains the partition within a list of partitions to which it is permitted to place calls—its partition search space. This provides for extremely powerful control over routing.

When a call is being placed, Digit Analysis attempts to resolve the dialed address only in those partitions that the partition search space specifies. Each partition name comprises a discrete subset of the global dial-able address space. From each

listed partition, Digit Analysis retrieves the pattern that best matches the sequence of dialed digits. Then, from among the matching patterns, Digit Analysis chooses the best match. If two patterns equally match the sequence of dialed digits, Digit Analysis breaks the tie by selecting the pattern associated with the partition listed first in the partition search space.

Diagnosing Security Problems

Using partitions and calling search spaces, in addition to more common filtering based on sections of the @ macro (which stands for the North American Numbering Plan) in a route pattern, you can configure Cisco CallManager to create a secure dialing plan for users. Partitions and Calling Search Spaces are an integral part of security and are especially useful for multi-tenant environments and for creating an individual user level. Filtering, a subset of the Calling Search Space/Partition concept, can add additional granularity to the security plan.

This is an extension to the [Diagnosing Dial Plan Problems, page 3-18](#). Be advised, usually the last thing you want to do when trying to fix a filtering problem is to run an SDI trace. There is simply not enough information and the potential for causing more harm is too great.

Run a show tech command on Cisco CallManager to diagnose Slow Server Response.

Diagnosing Slow Server Response

Slow response from the server could result if the duplex of the switch does not match the duplex port setting on the Cisco CallManager server. For optimal performance set both switch and server to 100/Full. We do not recommend using the Auto setting on either the switch or the server. You must restart the Cisco CallManager server for this change to take effect.

Determining Gateway Reorder Tone Problems

Users placing a call through the gateway might get a reorder tone if they are attempting to make a restricted call or to call a number that has been blocked. A reorder tone may occur if the dialed number is out of service or if the PSTN has

an equipment or service problem. Check to be sure the device giving the reorder tone has registered. Also, check your dial plan configuration to ensure that the call can be successfully routed.

The following procedure shows the steps for troubleshooting reorder tones through gateways.

Procedure

- Step 1** Check the gateways to ensure that you are using the latest software loads.
 - Step 2** Check Cisco Connection Online at www.cisco.com for the latest software loads, new patches, or release notes relating to the problem.
 - Step 3** Start an SDI trace and recreate the problem. Reorder tones could be the result of a configuration issue with location-based admission control or gatekeeper-based admission control where the Cisco CallManager might limit the number of allowable calls. In the SDI trace, locate the call to determine if it was blocked intentionally by a route pattern or the calling search space, or by any other configuration setting.
 - Step 4** Reorder tones can also occur when calling through the PSTN. Check the SDI trace for Q.931 messages, in particular, for disconnect messages. If a Q.931 disconnect message is present, it means the other party caused the disconnect and you cannot correct for that.
-

Diagnosing Gateway Registration Problems

One of the most common issues encountered with gateways on a Cisco CallManager is a registration problem. Registration can fail for a variety of reasons.

This section describes two similar but different categories of gateways. The Cisco Access AS-X, AT-X and Cisco Access DT-24+ and DE-30+ belong to one category. These gateways are stand-alone units that are not directly connected to a Network Management Processor (NMP). The second category includes the

Analog Access WS-X6624 and Digital Access WS-X6608. These gateways are blades installed in a Catalyst 6000 chassis with direct connectivity to the NMP for control and statusing.

First, check that the gateway is up and running. All of the gateways have a heartbeat LED that blinks 1-second-on, 1-second-off when the gateway software is running normally. If this LED is not blinking at all, or blinking very rapidly, then the gateway software is not running. Normally this will result in an automatic reset of the gateway. Also, it is normal for the gateway to reset itself if it cannot complete the registration process after about 2 to 3 minutes. So you may happen to look at the heartbeat LED while the device is resetting, but if the normal blinking pattern does not appear in 10 to 15 seconds, then the gateway has suffered a serious failure. On the Cisco Access Analog gateways, the green heartbeat LED is on the far right of the front panel. On the Cisco Access Digital gateways, it is the far left red LED on the top edge of the card. On the Cisco Analog Access WS-X6624, it is a green LED inside the blade (not visible from the front panel) on the far right card edge near the front. Finally, on the Digital Access WS-X6608 there is a separate heartbeat LED for each of the 8 spans on the blade. There are 8 red LEDs across the card (not visible from the front panel) about two thirds of the way towards the back.

The second thing to check is that the gateway has received its IP address. A standalone gateway must receive its IP address via DHCP or BOOTP. A Catalyst gateway may receive its IP address by DHCP, BOOTP, or by manual configuration through the NMP. If you have access to the DHCP server, the best way to check a stand-alone gateway is to verify that the device has an outstanding lease on an IP address. If the gateway shows up on your server, this is a good indication, but not definitive. Delete the lease at the DHCP server, and then reset the gateway. If the gateway reappears on the server with a lease within a couple of minutes, then everything is working fine in this area. If not, then either the gateway cannot contact the DHCP server (Is a router improperly configured and not forwarding DHCP broadcasts? Is the server running?), or cannot get a positive response (Is the IP address pool depleted?). If checking these suggestions does not yield the answer, use a sniffer trace to determine the specific problem.

For a Catalyst 6000 gateway, you should check to make sure the NMP can communicate with the gateway. You can check this by trying to ping its internal IP address from the NMP. The IP address is in the format:

```
127.1.module.port
```

In our example, we would enter:

```
Console (enable) ping 127.1.7.1
```

```
127.1.7.1 is alive
```

If pinging works, then the `sh-port` command will show IP address information. Make sure the IP address information and the TFTP IP address is correct as well. If the gateway is failing to obtain valid DHCP information, the `tracy` utility (supplied by Cisco TAC) can be used to determine the problem. After obtaining this utility from TAC, issue the following command from the Cat6000 Command Line Interface (CLI):

```
tracy_start mod port
```

In this example, the WS-X6624 is module 7 and it only has a single 860 processor, so it is port 1. The command we would issue is **tracy_start 7 1**.

The following output is actually from the 860-console port on the gateway board itself, however, the output of the `tracy` command is nothing more than a remote copy of the 860-console port.

```

      |
      |
    | |
    | |
  | | | |
  | | | | | :..|
C i s c o   S y s t e m s
CAT6K Analog Gateway (ELVIS)
APP Version : A0020300, DSP Version : A0030300, Built Jun  1 2000
16:33:01

ELVIS>> 00:00:00.020 (XA) MAC Addr : 00-10-7B-00-13-DE
00:00:00.050 NMPTask:got message from XA Task
00:00:00.050 (NMP) Open TCP Connection ip:7f010101
00:00:00.050 NMPTask:Send Module Slot Info
00:00:00.060 NMPTask:get DIAGCMD
00:00:00.160 (DSP) Test Begin -> Mask<0x00FFFFFF>
00:00:01.260 (DSP) Test Complete -> Results<0x00FFFFFF/0x00FFFFFF>
00:00:01.260 NMPTask:get VLANCONFIG
00:00:02.870 (CFG) Starting DHCP
00:00:02.870 (CFG) Booting DHCP for dynamic configuration.
00:00:06.570 (CFG) DHCP Request or Discovery Sent, DHCPState =
INIT_REBOOT
00:00:06.570 (CFG) DHCP Server Response Processed, DHCPState =
INIT_REBOOT
00:00:06.780 (CFG) IP Configuration Change!  Restarting now...
00:00:10.480 (CFG) DHCP Request or Discovery Sent, DHCPState = INIT
00:00:14.480 (CFG) DHCP Timeout Waiting on Server, DHCPState = INIT
00:00:22.480 (CFG) DHCP Timeout Waiting on Server, DHCPState = INIT
00:00:38.480 (CFG) DHCP Timeout Waiting on Server, DHCPState = INIT

```

If this timeout message continues to scroll by, then there is a problem contacting the DHCP server. First thing to check is that the Catalyst 6000 gateway port is in the correct VLAN.

You will find this information in the information you retrieved using the **sh port** command. If the DHCP server is not on the same VLAN as the Catalyst 6000 gateway, then make sure the appropriate IP helper addresses have been configured to forward the DHCP requests to the DHCP server. It is possible for the gateway to get stuck in the INIT state after a VLAN number change until the gateway resets. When in this state, it would not hurt to try resetting the gateway. Every time the 860 gets reset, your tracy session will be lost, so you must close your existing session and re-establish a new one by issuing the following commands:

```
tracy_close mod port
```

```
tracy_start mod port
```

If all this checks out and you are still seeing the DHCPState = INIT messages, you should check to see if the DHCP server is functioning correctly. If so, start a sniffer trace to see whether the requests are being sent and the server is responding.

Once DHCP is working correctly, the gateway will have an IP address that will allow the use of the tracy debugging utility. This utility is a built-in feature of the NMP command set for the Catalyst gateways, and available as a helper application that runs on Windows 98/NT/2000 for the stand-alone gateways. To use the helper application tracy utility, you need to connect to the gateway by using the IP address to which it is assigned. This tracy application works on all the gateways, provides a separate trace window for each gateway (up to eight may be traced at once), and allows traces to be logged directly to a file you specify.

The next step is to verify that the TFTP server IP address was correctly provided to the gateway. This is normally provided by DHCP in either Option 66 (by name or IP address), Option 150 (IP address only), or si_addr (IP address only). If your server has multiple Options configured, si_addr will take precedence over Option 150, which will take precedence over Option 66. If Option 66 provides the DNS_NAME of the TFTP server, then the DNS server(s) IP address(es) must have been specified by DHCP, and the name entered in Option 66 must resolve to the correct TFTP server IP address. A Catalyst gateway could be configured by the NMP to disable DHCP, and the NMP operator must then enter all configuration parameters by hand at the console, including the TFTP server address.

Additionally, the gateways will always attempt to resolve the name CiscoCM1 via DNS. If successful, the CiscoCM1 IP address will take precedence over anything the DHCP server or NMP tells it for the TFTP server address, even if the NMP has DHCP disabled.

You can check the current TFTP server IP address in a gateway by using the `tracy` utility. Enter the following command to get the configuration task number:

```
TaskID: 0
Cmd:    show t1
```

Look for a line with `config` or `CFG` and use the corresponding number as the `taskID` for the next line. For example, for the Cisco Access Digital gateway. In the examples that follow, we have bolded lines of text to make it easier for you to see the messages being explained. In the actual display output, text is not bolded. The examples are from an WS-X6624 model, the command to dump the DHCP information is:

```
TaskID: 6
Cmd:    show dhcp
```

The TFTP server IP address is then clearly shown. If it is not correct, verify that your DHCP options and other information it provides are correct.

Once the TFTP address is correct, the next step would be to ensure that the gateway is getting its configuration file from the TFTP server. If you see the following in the `tracy` output, your TFTP service may not be working correctly or the gateway might not be configured on the Cisco CallManager:

```
00:09:05.620 (CFG) Requesting SAA00107B0013DE.cnf File From TFTP
Server
00:09:18.620 (CFG) TFTP Error: Timeout Awaiting Server Response for
.cnf File!
```

The gateway will attempt to connect to the same IP address as the TFTP server if it does not get a configuration file. This is fine unless you are in a clustered environment in which the gateway needs to receive its list of redundant Cisco CallManagers. If the card is not getting its TFTP information correctly, check the TFTP service on the Cisco CallManager and make sure it is running. Also, check the TFTP trace on the Cisco CallManager.

Another common problem is that the gateway is not configured correctly on the Cisco CallManager. A typical error is entering an incorrect MAC address for the gateway. If this is the case, for a Catalyst 6000 gateway, you will probably get the following messages on the NMP console every two minutes:

```

2000 Apr 14 19:24:08 %SYS-4-MODHPRESET:Host process (860) 7/1 got
reset asynchronously
2000 Apr 14 19:26:05 %SYS-4-MODHPRESET:Host process (860) 7/1 got
reset asynchronously
2000 Apr 14 19:28:02 %SYS-4-MODHPRESET:Host process (860) 7/1 got
reset asynchronously

```

This is what the Tracy output would look like if the gateway is not in the Cisco CallManager database:

```

00:00:01.670 (CFG) Booting DHCP for dynamic configuration.
00:00:05.370 (CFG) DHCP Request or Discovery Sent, DHCPState =
INIT_REBOOT
00:00:05.370 (CFG) DHCP Server Response Processed, DHCPState = BOUND
00:00:05.370 (CFG) Requesting DNS Resolution of CiscoCM1
00:00:05.370 (CFG) DNS Error on Resolving TFTP Server Name.
00:00:05.370 (CFG) TFTP Server IP Set by DHCP Option 150 = 10.123.9.2
00:00:05.370 (CFG) Requesting SAA00107B0013DE.cnf File From TFTP
Server
00:00:05.370 (CFG) TFTP Error: .cnf File Not Found!
00:00:05.370 (CFG) Requesting SAADefault.cnf File From TFTP Server
00:00:05.380 (CFG) .cnf File Received and Parsed Successfully.
00:00:05.380 (CFG) Updating Configuration ROM...
00:00:05.610 GMSG: GWEvent = CFG_DONE --> GWState = SrchActive
00:00:05.610 GMSG: CCM#0 CPEvent = CONNECT_REQ --> CPState =
AttemptingSocket
00:00:05.610 GMSG: Attempting TCP socket with CCM 10.123.9.2
00:00:05.610 GMSG: CCM#0 CPEvent = SOCKET_ACK --> CPState = BackupCCM
00:00:05.610 GMSG: GWEvent = SOCKET_ACK --> GWState = RegActive
00:00:05.610 GMSG: CCM#0 CPEvent = REGISTER_REQ --> CPState =
SentRegister
00:00:05.680 GMSG: CCM#0 CPEvent = CLOSED --> CPState = NoTCPSTocket
00:00:05.680 GMSG: GWEvent = DISCONNECT --> GWState = Rollover
00:00:20.600 GMSG: GWEvent = TIMEOUT --> GWState = SrchActive
00:00:20.600 GMSG: CCM#0 CPEvent = CONNECT_REQ --> CPState =
AttemptingSocket
00:00:20.600 GMSG: Attempting TCP socket with CCM 10.123.9.2
00:00:20.600 GMSG: CCM#0 CPEvent = SOCKET_ACK --> CPState = BackupCCM

```

Another possible registration problem could be that the load information is incorrect or the load file is corrupt. The problem could also occur if the TFTP server is not working. In this case, Tracy clearly shows that the TFTP server reported the file is not found:

```

00:00:07.390 GMSG: CCM#0 CPEvent = REGISTER_REQ --> CPState =
SentRegister
00:00:08.010 GMSG: TFTP Request for application load A0021300
00:00:08.010 GMSG: CCM#0 CPEvent = LOADID --> CPState = AppLoadRequest

```

```
00:00:08.010 GMSG: ***TFTP Error: File Not Found***
00:00:08.010 GMSG: CCM#0 CPEvent = LOAD_UPDATE --> CPState =
LoadResponse
```

In this case, you can see that the gateway is requesting application load A0021300, although the correct load name would be A0020300. For a Catalyst 6000 gateway, the same problem can occur when a new application load needs to get its corresponding DSP load as well. If the new DSP load is not found, a similar message will appear.

```
ELVIS>> 00:00:00.020 (XA) MAC Addr : 00-10-7B-00-13-DE
00:00:00.050 NMPTask:got message from XA Task
00:00:00.050 (NMP) Open TCP Connection ip:7f010101
00:00:00.050 NMPTask:Send Module Slot Info
00:00:00.060 NMPTask:get DIAGCMD
00:00:00.160 (DSP) Test Begin -> Mask<0x00FFFFFF>
00:00:01.260 (DSP) Test Complete -> Results<0x00FFFFFF/0x00FFFFFF>
00:00:01.260 NMPTask:get VLANCONFIG
00:00:02.030 (CFG) Starting DHCP
00:00:02.030 (CFG) Booting DHCP for dynamic configuration.
00:00:05.730 (CFG) DHCP Request or Discovery Sent, DHCPState =
INIT_REBOOT
00:00:05.730 (CFG) DHCP Server Response Processed, DHCPState = BOUND
00:00:05.730 (CFG) Requesting DNS Resolution of CiscoCM1
00:00:05.730 (CFG) DNS Error on Resolving TFTP Server Name.
00:00:05.730 (CFG) TFTP Server IP Set by DHCP Option 150 = 10.123.9.2
00:00:05.730 (CFG) Requesting SAA00107B0013DE.cnf File From TFTP
Server
00:00:05.730 (CFG) .cnf File Received and Parsed Successfully.
00:00:05.730 GMSG: GWEvent = CFG_DONE --> GWState = SrchActive
00:00:05.730 GMSG: CCM#0 CPEvent = CONNECT_REQ --> CPState =
AttemptingSocket
00:00:05.730 GMSG: Attempting TCP socket with CCM 10.123.9.2
00:00:05.730 GMSG: CCM#0 CPEvent = SOCKET_ACK --> CPState = BackupCCM
00:00:05.730 GMSG: GWEvent = SOCKET_ACK --> GWState = RegActive
00:00:05.730 GMSG: CCM#0 CPEvent = REGISTER_REQ --> CPState =
SentRegister
00:00:06.320 GMSG: CCM#0 CPEvent = LOADID --> CPState = LoadResponse
00:01:36.300 GMSG: CCM#0 CPEvent = TIMEOUT --> CPState = BadCCM
00:01:36.300 GMSG: GWEvent = DISCONNECT --> GWState = Rollover
00:01:46.870 GMSG: CCM#0 CPEvent = CLOSED --> CPState = NoTCPsocket
00:01:51.300 GMSG: GWEvent = TIMEOUT --> GWState = SrchActive
00:01:51.300 GMSG: CCM#0 CPEvent = CONNECT_REQ --> CPState =
AttemptingSocket
00:01:51.300 GMSG: Attempting TCP socket with CCM 10.123.9.2
00:01:51.300 GMSG: CCM#0 CPEvent = SOCKET_ACK --> CPState = BackupCCM
00:01:51.300 GMSG: GWEvent = SOCKET_ACK --> GWState = RegActive
```

```
00:01:51.300 MSG: CCM#0 CPEvent = REGISTER_REQ --> CPState =  
SentRegister  
00:01:51.890 MSG: CCM#0 CPEvent = LOADID --> CPState = LoadResponse
```

The difference here is that the gateway gets stuck in the LoadResponse stage and eventually times out. This problem can be resolved by correcting the load file name in the Device Defaults area of Cisco CallManager Administration.

Diagnosing Gatekeeper Problems

Before starting any gateway-to-gatekeeper troubleshooting, verify that there is IP connectivity within the network. Assuming that there is IP connectivity, use the information in this section to troubleshoot your gateway.

Inter-Cluster Trunks or H.225 Gateway

Refer to the *Cisco CallManager Administration Guide* and *Cisco CallManager System Guide* at the following location:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm

Admission Rejects

Admission Rejects (ARJ) are issued when Cisco CallManager has registered with the Gatekeeper but cannot send a phone call. Configuration issues on the gatekeeper should be the primary focus when the gatekeeper is issuing an ARJ. The following procedure is a general guideline for troubleshooting.

Procedure

-
- Step 1** Verify IP connectivity from the gateway to the gatekeeper.
 - Step 2** Show gatekeeper status—verify the gatekeeper state is up.

- Step 3** Is there a zone subnet defined on the gatekeeper? If so, verify that the subnet of the gateway is in the allowed subnets.
-

Registration Rejects

Registration Rejects (RRJ) are issued when Cisco CallManager cannot register with the Gatekeeper. Configuration issues on the gatekeeper should be the primary focus when the gatekeeper is issuing a RRJ.

The following procedure is a general guideline for troubleshooting.

Procedure

- Step 1** Verify IP connectivity from the gateway to the gatekeeper.
- Step 2** Show gatekeeper status—verify the gatekeeper state is up.
- Step 3** Is there a zone subnet defined on the gatekeeper? If so, verify that the subnet of the gateway is in the allowed subnets.
-



Troubleshooting Intra-Cluster Phone Calls

The case study in this appendix discusses in detail the call flow between two Cisco IP Phones within a cluster, called an intra-cluster call. This case study also focuses on Cisco CallManager and Cisco IP Phone initialization, registration, and KeepAlive processes. That is followed by a detailed explanation of an intra-cluster call flow. These processes are explained using the trace utilities and tools discussed in [Chapter 2, “Troubleshooting and Monitoring Tools and Utilities.”](#)

This appendix contains the following topics:

- [Sample Topology, page A-2](#)
- [Cisco IP Phone Initialization Process, page A-2](#)
- [Skinny Station Registration Process, page A-3](#)
- [Cisco IP Phone-to-Cisco IP Phone Call Flow within a Cluster, page A-5](#)
- [Cisco IP Phone-to-Cisco IP Phone Exchange of Skinny Station Messages During Call Flow, page A-5](#)
- [Cisco CallManager Initialization Process, page A-6](#)
- [Self-Starting Processes, page A-7](#)
- [Cisco CallManager Registration Process, page A-8](#)
- [Cisco CallManager KeepAlive Process, page A-10](#)
- [Cisco CallManager Intra-Cluster Call Flow Traces, page A-11](#)

Sample Topology

Given that you have two clusters named Cluster 1 and Cluster 2, the two Cisco CallManagers in Cluster 1 are called CCM3 and CCM4, while the two Cisco CallManagers in Cluster 2 are called CCM1 and CCM2.

The traces collected for this case study are from CCM1, which is located in Cluster 2. The call flow is based on the two Cisco IP Phones in Cluster 2. The IP addresses of these two Cisco IP Phones are 172.16.70.230 (directory number 1000), and 172.16.70.231 (directory number 1001), respectively.

Cisco IP Phone Initialization Process

The Cisco IP Phone initialization (or boot up) process is explained in detail in the following procedure.

Procedure

- Step 1** If you have set the appropriate options in DHCP server (such as Option 066 or Option 150), the Cisco IP Phone sends a request, at initialization, to the DHCP server to get an IP address, Domain Name System (DNS) server address, and TFTP server name or address. It also gets a default gateway address if you have set these options in the DHCP server (Option 003).
- Step 2** If a DNS name of the TFTP sever is sent by DHCP, then a DNS sever IP address is required to map the name to an IP address. This step is bypassed if the DHCP server sends the IP address of the TFTP server. In this case study, the DHCP server sent the IP address of TFTP because DNS was not configured.
- Step 3** If a TFTP server name is not included in the DHCP reply, then the Cisco IP Phone uses the default server name.
- Step 4** The configuration file (.cnf) file is retrieved from the TFTP server. All .cnf files have the name SEP<mac_address>.cnf. If this is the first time the phone is registering with the Cisco CallManager, then a default file, SEPdefault.cnf, is downloaded to the Cisco IP Phone. In this case study, the first Cisco IP Phone uses the IP address 172.16.70.230 (its MAC address is SEP0010EB001720), and the second Cisco IP Phone uses the IP address 172.16.70.231 (its MAC address is SEP003094C26105).

- Step 5** All .cnf files include the IP address(es) for the primary and secondary Cisco CallManager(s). The Cisco IP Phone uses this IP address to contact the primary Cisco CallManager and to register.
- Step 6** Once the Cisco IP Phone has connected and registered with Cisco CallManager, the Cisco CallManager tells the Cisco IP Phone which executable version (called a load ID) to run. If the specified version does not match the executing version on the Cisco IP Phone, the Cisco IP Phone will request the new executable from the TFTP server and reset automatically
-

Skinny Station Registration Process

Cisco IP Phones communicate with the Cisco CallManager using the Cisco Skinny Station Protocol. The registration process allows a Skinny Station, such as the Cisco IP Phone, to inform the Cisco CallManager of its existence and to make calling possible.

[Table A-1](#) describes the primary messages in the Skinny Station registration process.

Table A-1 *Skinny Station Registration Process Descriptions*

Message	Description
Station Register	The station sends this message to announce its existence to the controlling Cisco CallManager.
Station Reset	Cisco CallManager sends this message to command the station to reset its processes.
Station IP Port	The station sends this message to provide Cisco CallManager with the User Datagram Protocol (UDP) port to be used with the Real Time Protocol (RTP) stream.
Station Register Acknowledge	Cisco CallManager sends this message to acknowledge the registration of a station.

Table A-1 *Skinny Station Registration Process Descriptions (continued)*

Message	Description
Station Register Reject	<p>Cisco CallManager sends this message to reject a registration attempt from the indicated phone.</p> <pre>char text [StationMaxDisplayTextSize];</pre> <p>};</p> <p>Where:</p> <p>text is a character string, maximum length of 33 bytes, containing a textual description of the reason that registration is rejected.</p>
Station Capabilities Request	Cisco CallManager sends this message to request the current capabilities of the station. Station capabilities may include compression standard and other H.323 capabilities.
Station Version Request	The station sends this message to request the version number of the software load for the station.
Station Version Response	Cisco CallManager sends this message to inform the station of the appropriate software version number.
Station Capabilities Response	The station sends this message to the Cisco CallManager in response to a Station Capabilities Request. The station's capabilities are cached in the Cisco CallManager and used to negotiate terminal capabilities with an H.323 compliant terminal.
Station Button Template Request	The station sends this message to request the button template definition for that specific terminal or Cisco IP Phone.
Station Button Template Response	Cisco CallManager sends this message to update the button template information contained in the station.
Station Time Date Request	The station sends this message to request the current date and time for internal usage and for displaying as a text string.
Station Define Time and Date	Cisco CallManager uses this message to provide the date and time information to the station. It provides time synchronization for the stations.

Cisco IP Phone-to-Cisco IP Phone Call Flow within a Cluster

This section describes a Cisco IP Phone (directory number 1000) calling another Cisco IP Phone (directory number 1001) within the same cluster. The cluster is a group of Cisco CallManagers having one common Publisher Structured Query Language (SQL) database and many Subscriber SQL databases.

In this cluster scenario, CCM1 is the publisher and CCM2 is a subscriber. The two Cisco IP Phones (1000 and 1001) are registered to CCM1 and CCM2, respectively. The call flow is shown in the diagram below. The two Cisco CallManagers within a cluster communicate with each other using Intra-Cluster Control Protocol (ICCP).

When a Cisco IP Phone goes off-hook, it opens a control Skinny Station session (with TCP as the underlying protocol) with the Cisco CallManager. After call control signaling is established between the two Cisco IP Phones and their respective Cisco CallManagers, the RTP stream starts flowing directly between the two phones, as shown in the diagram below. The Skinny Station call flow messages for this intra-cluster call are explained in next section.

Cisco IP Phone-to-Cisco IP Phone Exchange of Skinny Station Messages During Call Flow

The Skinny Station, in this case a Cisco IP Phone, initiates a connection to the Cisco CallManager, and then Cisco CallManager performs digit analysis before opening a control session with the destination Skinny Station.

**Note**

The Skinny Station messages are written in English so that they can be understood by end-users. Therefore, these messages are not explained in this section. However, these call flow Skinny Station messages are explained in more detail in later sections when traces are being examined.

Cisco CallManager Initialization Process

In this section the initialization process of Cisco CallManager will be explained with the help of traces that are captured from CCM1 (identified by the IP address 172.16.70.228). As described previously, SDI traces are a very effective troubleshooting tool because they detail every packet sent between endpoints.

This section describes the events that occur when Cisco CallManager is initialized. Understanding how to read traces will help you to properly troubleshoot the various Cisco CallManager processes, and the effect of those processes on services such as conferencing and call forwarding.

The following messages from the Cisco CallManager SDI trace utility show the initialization process on one of the Cisco CallManagers, in this case, CCM1.

- The first message indicates Cisco CallManager started its initialization process.
- The second message indicates Cisco CallManager read the default database values (for this case it is the primary or publisher database).
- The third message indicates Cisco CallManager listened to the various messages on TCP port 8002.
- The fourth message shows that after listening to these messages, Cisco CallManager added a second Cisco CallManager to its list: CCM2 (172.16.70.229).
- The fifth message indicates that Cisco CallManager has started and is running Cisco CallManager version 3.1(1).

```
16:02:47.765 CCM|CMPProcMon - CallManagerState Changed - Initialization
Started.
16:02:47.796 CCM|NodeId: 0, EventId: 107 EventClass: 3 EventInfo:
Cisco CM Database Defaults Read
16:02:49.937 CCM| SDL Info - NodeId: [1], Listen IP/Hostname:
[172.16.70.228], Listen Port: [8002]
16:02:49.984 CCM|dBProcs - Adding SdlLink to NodeId: [2], IP/Hostname:
[172.16.70.229]
16:02:51.031 CCM|NodeId: 1, EventId: 1 EventClass: 3 EventInfo:
Cisco CallManager Version=<3.1(1)> started
```

Self-Starting Processes

Once Cisco CallManager is up and running, it starts several other processes within itself. Some of these processes are shown below, including MulticastPoint Manager, UnicastBridge Manager, digit analysis, and route list. The messages described during these processes can be very useful when troubleshooting a problem related to the features in Cisco CallManager.

For example, assume that the route lists are not functioning and are unusable. To troubleshoot this problem, you would monitor these traces to determine whether the Cisco CallManager has started RoutePlanManager and if it is trying to load the RouteLists. In the sample configuration below, RouteListName="ipwan" and RouteGroupName="ipwan" are loading and starting.

```
16:02:51.031 CCM|MulicastPointManager - Started
16:02:51.031 CCM|UnicastBridgeManager - Started
16:02:51.031 CCM|MediaTerminationPointManager - Started
16:02:51.125 CCM|MediaCoordinator(1) - started
16:02:51.125 CCM|NodeId: 1, EventId: 1543 EventClass: 2 EventInfo:
Database manager started
16:02:51.234 CCM|NodeId: 1, EventId: 1542 EventClass: 2 EventInfo:
Link manager started
16:02:51.390 CCM|NodeId: 1, EventId: 1541 EventClass: 2 EventInfo:
Digit analysis started
16:02:51.406 CCM|RoutePlanManager - Started, loading RouteLists
16:02:51.562 CCM|RoutePlanManager - finished loading RouteLists
16:02:51.671 CCM|RoutePlanManager - finished loading RouteGroups
16:02:51.671 CCM|RoutePlanManager - Displaying Resulting RoutePlan
16:02:51.671 CCM|RoutePlanServer - RouteList Info, by RouteList and
RouteGroup Selection Order
16:02:51.671 CCM|RouteList - RouteListName='ipwan'
16:02:51.671 CCM|RouteList - RouteGroupName='ipwan'
16:02:51.671 CCM|RoutePlanServer - RouteGroup Info, by RouteGroup and
Device Selection Order
16:02:51.671 CCM|RouteGroup - RouteGroupName='ipwan'
```

The following trace shows the RouteGroup adding the device 172.16.70.245, which is CCM3 located in Cluster 1 and is considered an H.323 device. In this case, the RouteGroup is created to route calls to CCM3 in Cluster 1 with Cisco IOS Gatekeeper permission. If there is a problem routing the call to a Cisco IP Phone located in Cluster 1, then the following messages would help you find the cause of the problem.

```
16:02:51.671 CCM|RouteGroup - DeviceName='172.16.70.245'
16:02:51.671 CCM|RouteGroup -AllPorts
```

Part of the initialization process shows that Cisco CallManager is adding "Dns" (Directory Numbers). By reviewing these messages, you can determine whether the Cisco CallManager has read the directory number from the database.

```
16:02:51.671 CCM|NodeId: 1, EventId: 1540 EventClass: 2 EventInfo:
Call control started
16:02:51.843 CCM|ProcessDb - Dn = 2XXX, Line = 0,
Display = , RouteThisPattern, NetworkLocation = OffNet,
DigitDiscardingInstruction = 1, WhereClause =
16:02:51.859 CCM|Digit analysis: Add local pattern 2XXX , PID: 1,80,1
16:02:51.859 CCM|ForwardManager - Started
16:02:51.984 CCM|CallParkManager - Started
16:02:52.046 CCM|ConferenceManager - Started
```

In the following traces the Device Manager in Cisco CallManager is statically initializing two devices. The device with IP address 172.17.70.226 is a Gatekeeper and the device with IP address 172.17.70.245 is another Cisco CallManager in a different cluster. That Cisco CallManager is registered as an H.323 Gateway with this Cisco CallManager.

```
16:02:52.250 CCM|DeviceManager: Statically Initializing Device;
DeviceName=172.16.70.226
16:02:52.250 CCM|DeviceManager: Statically Initializing Device;
DeviceName=172.16.70.245
```

Cisco CallManager Registration Process

Another important part of the SDI trace is the registration process. When a device is powered up, it gets information via DHCP, connects to the TFTP server for its .cnf file, and then connects to the Cisco CallManager specified in the .cnf file. The device could be an MGCP Gateway, a Skinny Gateway, or a Cisco IP Phone. Therefore, it is important to be able to discover whether or not devices have successfully registered on the Cisco AVVID network.

In the following trace, Cisco CallManager has received new connections for registration. The registering devices are MTP_nsa-cm1 (MTP services on CCM1), and CFB_nsa-cm1 (Conference Bridge service on CCM1). These are software services running on Cisco CallManager but are treated internally as different external services and are therefore assigned a TCPHandle, socket number, and port number as well as a device name.

```

16:02:52.750 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fbaa00, Socket=0x594, IPAddr=172.16.70.228, Port=3279,
StationD=[0,0,0]
16:02:52.750 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fe05e8, Socket=0x59c, IPAddr=172.16.70.228, Port=3280,
StationD=[0,0,0]
16:02:52.781 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=MTP_nsa-cml, TCPHandle=0x4fbaa00, Socket=0x594,
IPAddr=172.16.70.228, Port=3279, StationD=[1,45,2]
16:02:52.781 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=CFB_nsa-cml, TCPHandle=0x4fe05e8, Socket=0x59c,
IPAddr=172.16.70.228, Port=3280, StationD=[1,96,2]

```

In the following trace, Skinny Station messages are sent between a Cisco IP Phone and Cisco CallManager. The Cisco IP Phone (172.16.70.231) is registering with Cisco CallManager. [Table A-1](#) provides the descriptions of Skinny Station messages. As soon as Cisco CallManager receives the registration request from a Cisco IP Phone, it assigns a TCPHandle number to this device. This number remains the same until the device or Cisco CallManager is restarted. Therefore, you can follow all the events related to a particular device by searching for or keeping track of the device's TCPHandle number, which appears in hexadecimal notation. Also, notice that Cisco CallManager provides the load ID to the Cisco IP Phone. Based on this load ID, the Cisco IP Phone runs the executable file (acquired from the TFTP server) that corresponds to the device.

```

16:02:57.000 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fbbc30, Socket=0x5a4, IPAddr=172.16.70.231, Port=52095,
StationD=[0,0,0]
16:02:57.046 CCM|NodeId: 1, EventId: 1703 EventClass: 2 EventInfo:
Station Alarm, TCP Handle: 4fbbc30, Text: Name=SEP003094C26105
Load=AJ.30 Parm=Status/IPAddr LastTime=A P1: 2304(900) P2:
-414838612(e74610ac)
16:02:57.046 CCM|StationInit - ***** InboundStim - AlarmMessageID
tcpHandle=0x4fbbc30 Message="Name=SEP003094C26105 Load=AJ.30
Parm=Status/IPAddr LastTime=A" Parm1=2304(900) Parm2=-414838612
(e74610ac)
16:02:57.093 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=SEP003094C26105, TCPHandle=0x4fbbc30, Socket=0x5a4,
IPAddr=172.16.70.231, Port=52095, StationD=[1,85,1]
16:02:57.093 CCM|StationInit - InboundStim - IpPortMessageID:
32715(0x7fcb) tcpHandle=0x4fbbc30

```

Cisco CallManager KeepAlive Process

Both the station, device, or service and the Cisco CallManager use the following messages to maintain a knowledge of the communications channel between them. The messages are used to begin the KeepAlive sequence that ensures that the communications link between the Cisco CallManager and the station remains active. The following messages can originate from either the Cisco CallManager or the station.

```
16:03:02.328 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=MTP_nsa-cm2,
TCPHandle=0x4fa7dc0, Socket=0x568, IPAddr=172.16.70.229, Port=1556,
StationD=[1,45,1]
16:03:02.328 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=CFB_nsa-cm2,
TCPHandle=0x4bf8a70, Socket=0x57c, IPAddr=172.16.70.229, Port=1557,
StationD=[1,96,1]
16:03:06.640 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=SEP0010EB001720,
TCPHandle=0x4fbb150, Socket=0x600, IPAddr=172.16.70.230, Port=49211,
StationD=[1,85,2]
16:03:06.703 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=SEP003094C26105,
TCPHandle=0x4fbbc30, Socket=0x5a4, IPAddr=172.16.70.231, Port=52095,
StationD=[1,85,1]
```

The messages in the following trace depict the KeepAlive sequence which indicates that the communications link between the Cisco CallManager and the station is active. Again, these messages can originate from either the Cisco CallManager or the station.

```
16:03:02.328 CCM|MediaTerminationPointControl -
stationOutputKeepAliveAck tcpHandle=4fa7dc0
16:03:02.328 CCM|UnicastBridgeControl - stationOutputKeepAliveAck
tcpHandle=4bf8a70
16:03:06.703 CCM|StationInit - InboundStim - IpPortMessageID:
32715(0x7fcb) tcpHandle=0x4fbbc30
16:03:06.703 CCM|StationD - stationOutputKeepAliveAck
tcpHandle=0x4fbbc30
```

Cisco CallManager Intra-Cluster Call Flow Traces

The following SDI traces explore the intra-cluster call flow in detail. The Cisco IP Phones in the call flow can be identified by the directory number (dn), tcpHandle, and IP address. A Cisco IP Phone (dn: 1001, tcpHandle: 0x4fbbc30, IP address: 172.16.70.231) located in Cluster 2 is calling another Cisco IP Phone in the same Cluster (dn=1000, tcpHandle= 0x4fbb150, IP address= 172.16.70.230). Remember that you can follow a device through the trace by looking at the TCP handle value, time stamp, or name of the device. The TCP handle value for the device remains the same until the device is rebooted or goes offline.

The following traces show that the Cisco IP Phone (1001) has gone off-hook. The trace below shows the unique messages, TCP handle, and the called number, which are displayed on the Cisco IP Phone. There is no calling number at this point, because the user has not tried to dial any digits. The information below is in the form of Skinny Station messages between the Cisco IP Phones and the Cisco CallManager.

```
16:05:41.625 CCM|StationInit - InboundStim - OffHookMessageID
tcpHandle=0x4fbbc30
16:05:41.625 CCM|StationD - stationOutputDisplayText
tcpHandle=0x4fbbc30, Display= 1001
```

The next trace shows Skinny Station messages going from Cisco CallManager to a Cisco IP Phone. The first message is to turn on the lamp on the calling party's Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputSetLamp stim: 9=Line
instance=1 lampMode=LampOn tcpHandle=0x4fbbc30
```

The stationOutputCallState message is used by Cisco CallManager to notify the station of certain call related information.

```
16:05:41.625 CCM|StationD - stationOutputCallState tcpHandle=0x4fbbc30
```

The stationOutputDisplayPromptStatus message is used by Cisco CallManager to cause a call-related prompt message to be displayed on the Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbbc30
```

The stationOutputSelectSoftKey message is used by Cisco CallManager to cause the Skinny Station to select a specific set of soft keys.

```
16:05:41.625 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
```

The next message is used by Cisco CallManager to instruct the Skinny Station as to the correct line context for the display.

```
16:05:41.625 CCM|StationD - stationOutputActivateCallPlane
tcpHandle=0x4fbbc30
```

In the following message, the digit analysis process is ready to identify incoming digits and check them for potential routing matches in the database. The entry, cn=1001, represents the calling party number where dd="" represents the dialed digit, which would show the called part number. Note that StationInit messages are sent by the phone, StationD messages are sent by Cisco CallManager, and digit analysis is performed by Cisco CallManager.

```
16:05:41.625 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="")
16:05:41.625 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
```

The following debug message shows that the Cisco CallManager is providing inside dial tone to the calling party Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputStartTone: 33=InsideDialTone
tcpHandle=0x4fbbc30
```

Once Cisco CallManager detects an incoming message and recognizes that the keypad button 1 has been pressed on the Cisco IP Phone, it immediately stops the output tone.

```
16:05:42.890 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 1 tcpHandle=0x4fbbc30
16:05:42.890 CCM|StationD - stationOutputStopTone tcpHandle=0x4fbbc30
16:05:42.890 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
16:05:42.890 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="1")
16:05:42.890 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.203 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
16:05:43.203 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="10")
16:05:43.203 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.406 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
```

```

16:05:43.406 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="100")
16:05:43.406 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.562 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
16:05:43.562 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="1000")

```

Once the Cisco CallManager has received enough digits to match, it provides the digit analysis results in a table format. Any extra digits pressed on the phone after this point will be ignored by Cisco CallManager, since a match has already been found.

```

16:05:43.562 CCM|Digit analysis: analysis results
16:05:43.562 CCM| |PretransformCallingPartyNumber=1001
|CallingPartyNumber=1001
|DialingPattern=1000
|DialingRoutePatternRegularExpression=(1000)
|PotentialMatches=PotentialMatchesExist
|DialingSdlProcessId=(1,38,2)
|PretransformDigitString=1000
|PretransformPositionalMatchList=1000
|CollectedDigits=1000
|PositionalMatchList=1000
|RouteBlockFlag=RouteThisPattern

```

The next trace shows that Cisco CallManager is sending out this information to a called party phone (the phone is identified by the tcpHandle number).

```

16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=1000,
CalledParty=1000, tcpHandle=0x4fbb150

```

The next trace indicates that Cisco CallManager is ordering the lamp to blink for incoming call indication on the called party's Cisco IP Phone.

```

16:05:43.578 CCM|StationD - stationOutputSetLamp stim: 9=Line
instance=1 lampMode=LampBlink tcpHandle=0x4fbb150

```

In the following traces, Cisco CallManager is providing ringer, display notification, and other call-related information to the called party's Cisco IP Phone. Again, you can see that all messages are directed to the same Cisco IP Phone because the same tcpHandle is used throughout the traces.

```

16:05:43.578 CCM|StationD - stationOutputSetRinger: 2=InsideRing
tcpHandle=0x4fbb150

```

```

16:05:43.578 CCM|StationD - stationOutputDisplayNotify
tcpHandle=0x4fbb150
16:05:43.578 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbb150
16:05:43.578 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbb150

```

Notice that Cisco CallManager is also providing similar information to the calling party's Cisco IP Phone. Again, the tcpHandle is used to differentiate between Cisco IP Phones.

```

16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=1000, tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=1000,
CalledParty=1000, tcpHandle=0x4fbbc30

```

In the next trace, Cisco CallManager provides an alerting or ringing tone to the calling party's Cisco IP Phone, notifying that the connection has been established.

```

16:05:43.578 CCM|StationD - stationOutputStartTone: 36=AlertingTone
tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputCallState tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbbc30

```

At this point, the called party's Cisco IP Phone goes off-hook. Therefore, Cisco CallManager stops generating the ringer tone to calling party.

```

16:05:45.140 CCM|StationD - stationOutputStopTone tcpHandle=0x4fbbc30

```

In the following messages, Cisco CallManager causes the Skinny Station to begin receiving a Unicast RTP stream. To do so, Cisco CallManager provides the IP address of the called party as well as codec information, and packet size in msec (milliseconds). PacketSize is an integer containing the sampling time in milliseconds used to create the RTP packets.


Note

Normally this value is set to 30msec. In this case it is set to 20msec.

```

16:05:45.140 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x4fbbc30 myIP: e74610ac (172.16.70.231)

```

```
16:05:45.140 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

Similarly, Cisco CallManager provides information to the called party (1000).

```
16:05:45.140 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x4fbb150 myIP: e64610ac (172.16.70.230)
16:05:45.140 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

Cisco CallManager has received the acknowledgment message from called party for establishing the open channel for RTP stream, as well as the IP address of the called party. This message is to inform the Cisco CallManager of two pieces of information about the Skinny Station. First, it contains the status of the open action. Second, it contains the receive port address and number for transmission to the remote end. The IP address of the transmitter (calling part) of the RTP stream is `ipAddr`, and `PortNumber` is the IP port number of the RTP stream transmitter (calling party).

```
16:05:45.265 CCM|StationInit - InboundStim -
StationOpenReceiveChannelAckID tcpHandle=0x4fbb150, Status=0,
IpAddr=0xe64610ac, Port=17054, PartyID=2
```

The following messages are used by Cisco CallManager to order the station to begin transmitting the audio stream to the indicated remote Cisco IP Phone's IP address and port number.

```
16:05:45.265 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x4fbbc30 myIP: e74610ac (172.16.70.231)
16:05:45.265 CCM|StationD - RemoteIpAddr: e64610ac (172.16.70.230)
RemoteRtpPortNumber: 17054 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

In the following traces, the previously explained messages are sent to the called party. These messages are followed by the messages indicating the RTP media stream has been started between the called and calling party.

```
16:05:45.312 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x4fbb150 myIP: e64610ac (172.16.70.230)
16:05:45.328 CCM|StationD - RemoteIpAddr: e74610ac (172.16.70.231)
RemoteRtpPortNumber: 18448 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
16:05:46.203 CCM|StationInit - InboundStim - OnHookMessageID
tcpHandle=0x4fbbc30
```

The calling party's Cisco IP Phone finally goes on-hook, which terminates all the control messages between the Skinny Station and Cisco CallManager as well as the RTP stream between Skinny Stations.

```
16:05:46.203 CCM|StationInit - InboundStim - OnHookMessageID  
tcpHandle=0x4fbbc30
```



Troubleshooting Cisco IP Phone-to-Cisco IOS Gateway Calls

In the case study described in [Appendix A, “Skinny Station Registration Process Descriptions,”](#) the call flow for an intra-cluster call was discussed in detail. The case study in this appendix examines a Cisco IP Phone calling through a Cisco IOS Gateway to a phone connected through a local PBX or on the Public Switched Telephone Network (PSTN). Conceptually, when the call reaches the Cisco IOS Gateway, the gateway will forward the call either to a phone connected to an FXS port, or to the PBX. If the call is forwarded to the PBX, it could terminate to a phone connected to a local PBX or the PBX will forward it over the PSTN and the call will terminate somewhere on the PSTN.

This appendix contains the following topics:

- [Call Flow Traces, page B-2](#)
- [Debug Messages and Show Commands on the Cisco IOS Gatekeeper, page B-5](#)
- [Debug Messages and Show Commands on the Cisco IOS Gateway, page B-6](#)
- [Cisco IOS Gateway with T1/PRI Interface, page B-11](#)
- [Cisco IOS Gateway with T1/CAS Interface, page B-12](#)

Call Flow Traces

This section discusses call flow through examples from the Cisco CallManager trace file CCM000000000. The traces in this case study focus only on the call flow itself, as the more detailed trace information has already been explained in [Appendix A, “Troubleshooting Intra-Cluster Phone Calls”](#) (for example, initialization, registration, and KeepAlive mechanism).

In this call flow, a Cisco IP Phone (directory number 1001) located in the cluster 2 is calling a phone (directory number 3333) located somewhere on the PSTN. Remember that you can follow a device through the trace by looking at the TCP handle value, time stamp, or name of the device. The TCP handle value for the device remains the same until the device is rebooted or goes offline.

In the following traces, the Cisco IP Phone (1001) has gone off-hook. The trace shows the unique messages, TCP handle, and the calling number, which is displayed on the Cisco IP Phone. There is no called number at this point, because the user has not tried to dial any digits.

```
16:05:46.37515:20:18.390 CCM|StationInit - InboundStim -  
OffHookMessageID tcpHandle=0x5138d98
```

```
15:20:18.390 CCM|StationD - stationOutputDisplayText  
tcpHandle=0x5138d98, Display=1001
```

In the following traces, the user is dialing the DN 3333, one digit at a time. The number 3333 is the destination number of the phone, which is located somewhere on the PSTN network. The digit analysis process of the Cisco CallManager is currently active and is analyzing the digits to discover where the call needs to get routed. A more detailed explanation of the digit analysis is provided in [Appendix A, “Troubleshooting Intra-Cluster Phone Calls.”](#)

```
15:20:18.390 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",  
dd="")  
15:20:19.703 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",  
dd="3")  
15:20:20.078 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",  
dd="33")  
15:20:20.718 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",  
dd="333")  
15:20:21.421 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",  
dd="3333")  
15:20:21.421 CCM|Digit analysis: analysis results
```

In the following traces, the digit analysis has been completed, calling and called party has been matched, and the information has been parsed.

```
|CallingPartyNumber=1001
|DialingPattern=3333
|DialingRoutePatternRegularExpression=(3333)
|PretransformDigitString=3333
|PretransformPositionalMatchList=3333
|CollectedDigits=3333
|PositionalMatchList=3333
```

In the following traces, the number 0 indicates the originating location, and the number 1 indicates the destination location. The bandwidth of the originating location is determined by BW = -1. The value -1 implies that the bandwidth is infinite. The bandwidth is infinite because the call was originated from a Cisco IP Phone located in a LAN environment. The bandwidth of the destination location is determined by BW = 64. The call destination is to a phone located in a PSTN, and the codec type is used is G.711 (64Kbps).

```
15:20:21.421 CCM|Locations:Orig=0 BW=-1 Dest=1 BW=64 (-1 implies
infinite bw available)
```

The following traces show the calling and called party information. In this example, the calling party name and number is the same because the administrator has not configured a display name, such as John Smith.

```
15:20:21.421 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=3333, tcpHandle=0x5138d98
```

The following trace shows that the H.323 code has been initialized and is sending an H.225 setup message. You can also see the traditional HDLC SAPI messages, the IP address of the called side in hexadecimal, and the port numbers.

```
15:20:21.421 CCM|Out Message -- H225SetupMsg -- Protocol= H225Protocol
15:20:21.421 CCM|MMan_Id= 1. (iep= 0 ds1= 0 sapi= 0 ces= 0
IpAddr=e24610ac IpPort=47110)
```

The following trace shows the calling and called party information as well as the H.225 alerting message. Also shown is the mapping of a Cisco IP Phone's hexadecimal value to the IP address. The IP address of the Cisco IP Phone (1001) is 172.16.70.231.

```
15:20:21.437 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=3333, tcpHandle=0x5138d98
15:20:21.453 CCM|In Message -- H225AlertMsg -- Protocol= H225Protocol
```

```
15:20:21.953 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x5138d98 myIP: e74610ac (172.16.70.231)
```

The following trace shows the compression type used for this call (G.711 μ -law).

```
15:20:21.953 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

Once the H.225 alert message has been sent, H.323 initializes: H.245. The following trace shows the calling and called party information, and the H.245 messages. Notice the TCP handle value is the same as before, indicating that this is the continuation of the same call.

```
15:20:22.062 CCM|H245Interface(3) paths established ip = e74610ac,
port = 23752
15:20:22.062 CCM|H245Interface(3) OLC outgoing confirm ip = e24610ac,
port = 16758
15:20:22.062 CCM|MediaManager - wait_AuConnectInfo - received
response, forwarding
```

The following trace shows the H.225 connection message, as well as other information that was explained earlier. When the H.225 connection message is received, the call has been connected.

```
15:20:22.968 CCM|In Message -- H225ConnectMsg -- Protocol=
H225Protocol
15:20:22.968 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=3333, tcpHandle=0x5138d98
15:20:22.062 CCM|MediaCoordinator - wait_AuConnectInfoInd
15:20:22.062 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x5138d98 myIP: e74610ac (172.16.70.231)
15:20:22.062 CCM|StationD - RemoteIpAddr: e24610ac (172.16.70.226)
RemoteRtpPortNumber: 16758 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
15:20:22.062 CCM|Locations:Orig=0 BW=-1Dest=1 BW=6(-1 implies infinite
bw available)
```

The following message shows that an on-hook message from the Cisco IP Phone (1001) is being received. As soon as an on-hook message is received, the H.225 and Skinny Station device disconnection messages are sent and the entire H.225 message is seen. This final message indicates that the call has been terminated.

```
15:20:27.296 CCM|StationInit - InboundStim - OnHookMessageID
tcpHandle=0x5138d98
15:20:27.296 CCM|ConnectionManager -wait_AuDisconnectRequest
(16777247,16777248): STOP SESSION
```

```
15:20:27.296 CCM|MediaManager - wait_AuDisconnectRequest - StopSession
sending disconnect to (64,5) and remove connection from list
15:20:27.296 CCM| Device SEP003094C26105 , UnRegisters with SDL Link
to monitor NodeID= 1
15:20:27.296 CCM|StationD - stationOutputCloseReceiveChannel
tcpHandle=0x5138d98 myIP: e74610ac (172.16.70.231)
15:20:27.296 CCM|StationD - stationOutputStopMediaTransmission
tcpHandle=0x5138d98 myIP: e74610ac (172.16.70.231)
15:20:28.328 CCM|In Message -- H225ReleaseCompleteMsg -- Protocol=
H225Protocol
```

Debug Messages and Show Commands on the Cisco IOS Gatekeeper

In the [Call Flow Traces, page B-2](#), the Cisco CallManager SDI trace was discussed in detail. In the topology for this case study, the debug ras command has been turned on in the Cisco IOS Gatekeeper.

The following debug messages show that the Cisco IOS Gatekeeper is receiving the admission request (ARQ) for the Cisco CallManager (172.16.70.228), followed by other successful Remote Access Server (RAS) messages. Finally, the Cisco IOS Gatekeeper sends an admission confirmed (ACF) message to the Cisco CallManager.

```
*Mar 12 04:03:57.181: RASLibRASRecvData ARQ (seq# 3365) rcvd from
[172.16.70.228883] on sock [0x60AF038C]
*Mar 12 04:03:57.181: RASLibRAS_WK_TInit ipsock [0x60A7A68C] setup
successful
*Mar 12 04:03:57.181: RASlibras_sendto msg length 16 from
172.16.70.2251719 to 172.16.70.228883
*Mar 12 04:03:57.181: RASLibRASSendACF ACF (seq# 3365) sent to
172.16.70.228
```

The following debug messages show that the call is in progress.

```
*Mar 12 04:03:57.181: RASLibRASRecvData successfully rcvd message of
length 55 from 172.16.70.228883
```

The following debug messages shows that the Cisco IOS Gatekeeper has received a disengaged request (DRQ) from the Cisco CallManager (172.16.70.228), and the Cisco IOS Gatekeeper has sent a disengage confirmed (DCF) to the Cisco CallManager.

```
*Mar 12 04:03:57.181: RASLibRASRecvData DRQ (seq# 3366) rcvd from
[172.16.70.228883] on sock [0x60AF038C]
*Mar 12 04:03:57.181: RASLibras_sendto msg length 3 from
172.16.70.2251719 to 172.16.70.228883
*Mar 12 04:03:57.181: RASLibRASsendDCF DCF (seq# 3366) sent to
172.16.70.228
*Mar 12 04:03:57.181: RASLibRASRecvData successfully rcvd message of
length 124 from 172.16.70.228883
```

The command `show gatekeeper endpoints` on the Cisco IOS Gatekeeper shows that all four Cisco CallManagers are registered with the Cisco IOS Gatekeeper. Remember that in the topology for this case study, there are four Cisco CallManagers, two in each cluster. This Cisco IOS Gatekeeper has two zones and each zone has two Cisco CallManagers.

R2514-1#show gatekeeper endpoints

```

                                GATEKEEPER ENDPOINT REGISTRATION
                                =====
CallSignalAddr  Port  RASignalAddr  Port  Zone Name          Type
-----
172.16.70.228   2     172.16.70.228   1493  gka.cisco.com
VOIP-GW
H323-ID: ac1046e4->ac1046f5
172.16.70.229   2     172.16.70.229   3923  gka.cisco.com
VOIP-GW
H323-ID: ac1046e5->ac1046f5
172.16.70.245   1     172.16.70.245   1041  gkb.cisco.com
VOIP-GW
H323-ID: ac1046f5->ac1046e4
172.16.70.243   1     172.16.70.243   2043  gkb.cisco.com
VOIP-GW
H323-ID: ac1046f5->ac1046e4
Total number of active registrations = 4
```

Debug Messages and Show Commands on the Cisco IOS Gateway

In [Debug Messages and Show Commands on the Cisco IOS Gatekeeper, page B-5](#), the Cisco IOS Gatekeeper show commands and debug outputs were discussed in detail. This section focuses on the debug output and show commands on the Cisco IOS Gateway. In the topology for this case study, calls are going

through the Cisco IOS Gateways. The Cisco IOS Gateway interfaces to the PSTN or PBX with either T1/CAS or T1/PRI interfaces. Debug output of commands such as debug voip ccapi inout, debug h225 events and debug h225 asn1, are shown below.

In the following debug output, the Cisco IOS Gateway is accepting the TCP connection request from Cisco CallManager (172.16.70.228) on port 2328 for H.225.

```
*Mar 12 04:03:57.169: H225Lib::h225TAccept: TCP connection accepted
from 172.16.70.228:2328 on socket [1]
*Mar 12 04:03:57.169: H225Lib::h225TAccept: Q.931 Call State is
initialized to be [Null].
*Mar 12 04:03:57.177: Hex representation of the received
TPFKT03000065080000100
```

The following debug output shows that the H.225 data is coming from the Cisco CallManager on this TCP session. Important to notice in this debug output is the protocolIdentifier, which indicates the H.323 version being used. The following debug shows that H.323 version 2 is being used. The called and calling party numbers are also shown.

```
- Source Address H323-ID
- Destination Address e164
*Mar 12 04:03:57.177: H225Lib::h225RecvData: Q.931 SETUP
received from socket [1]value H323-UserInformation ::=
*Mar 12 04:03:57.181: {
*Mar 12 04:03:57.181:   h323-uu-pdu
*Mar 12 04:03:57.181:   {
*Mar 12 04:03:57.181:     h323-message-body setup :
*Mar 12 04:03:57.181:     {
*Mar 12 04:03:57.181:       protocolIdentifier { 0 0 8 2250 0 2 },
*Mar 12 04:03:57.181:       sourceAddress
*Mar 12 04:03:57.181:       {
*Mar 12 04:03:57.181:         h323-ID : "1001"
*Mar 12 04:03:57.181:       },
*Mar 12 04:03:57.185:       destinationAddress
*Mar 12 04:03:57.185:       {
*Mar 12 04:03:57.185:         e164 : "3333"
*Mar 12 04:03:57.185:       },
*Mar 12 04:03:57.189:     H225Lib::h225RecvData: State changed to
[Call Present].
```

The following is debug output for Call Control Application Programming Interface (CCAPi). Call Control APi is indicating an incoming call. Called and calling party information can also be seen in the following output. CCAPi

matches the dial peer 0, which is the default dial peer. It is matching dial peer 0 because the CCAPi could not find any other dial peer for the calling number, so it is using the default dial peer.

```
*Mar 12 04:03:57.189: cc_api_call_setup_ind (vdbPtr=0x616C9F54,
callInfo={called=3333, calling=1001, fdest=1 peer_tag=0},
callID=0x616C4838)
*Mar 12 04:03:57.193: cc_process_call_setup_ind (event=0x617A2B18)
handed call to app "SESSION"
*Mar 12 04:03:57.193: sess_appl: ev(19=CC_EV_CALL_SETUP_IND), cid(17),
disp(0)
*Mar 12 04:03:57.193: ccCallSetContext (callID=0x11,
context=0x61782BBC)
Mar 12 04:03:57.193: ssaCallSetupInd finalDest cllng(1001),
cldd(3333)
*Mar 12 04:03:57.193: ssaSetupPeer cid(17) peer list: tag(1)
*Mar 12 04:03:57.193: ssaSetupPeer cid(17), destPat(3333), matched(4),
prefix(), peer(6179E63C)
*Mar 12 04:03:57.193: ccCallSetupRequest (peer=0x6179E63C, dest=,
params=0x61782BD0 mode=0, *callID=0x617A87C0)
*Mar 12 04:03:57.193: callingNumber=1001, calledNumber=3333,
redirectNumber=
*Mar 12 04:03:57.193: accountNumber=,finalDestFlag=1,
guid=0098.89c8.9233.511d.0300.cddd.ac10.46e6
```

The CCAPi matches the dial-peer 1 with the destination pattern, which is the called number 3333. Keep in mind that peer_tag means dial peer. Notice the calling and called party number in the request packet.

```
*Mar 12 04:03:57.193: peer_tag=1
*Mar 12 04:03:57.197: ccIFCallSetupRequest: (vdbPtr=0x617BE064, dest=,
callParams={called=3333, calling=1001, fdest=1, voice_peer_tag=1},
mode=0x0)
```

The following debug output shows the H.225 alerting messages are returning to the Cisco CallManager.

```
*Mar 12 04:03:57.197: ccCallSetContext (callID=0x12,
context=0x61466B30)
*Mar 12 04:03:57.197: ccCallProceeding (callID=0x11, prog_ind=0x0)
*Mar 12 04:03:57.197: cc_api_call_proceeding(vdbPtr=0x617BE064,
callID=0x12, prog_ind=0x0)
*Mar 12 04:03:57.197: cc_api_call_alert(vdbPtr=0x617BE064,
callID=0x12, prog_ind=0x8, sig_ind=0x1)
*Mar 12 04:03:57.201: sess_appl: ev(17=CC_EV_CALL_PROCEEDING),
cid(18), disp(0)
```

```

*Mar 12 04:03:57.201: ssa:
cid(18)st(1)oldst(0)cfid(-1)csize(0)in(0)fDest(0)-cid2(17)st2(1)oldst2
(0)
*Mar 12 04:03:57.201: ssaIgnore cid(18), st(1),oldst(1), ev(17)
*Mar 12 04:03:57.201: sess_appl: ev(7=CC_EV_CALL_ALERT), cid(18),
disp(0)
*Mar 12 04:03:57.201: ssa:
cid(18)st(1)oldst(1)cfid(-1)csize(0)in(0)fDest(0)-cid2(17)st2(1)oldst2
(0)
*Mar 12 04:03:57.201: ssaFlushPeerTagQueue cid(17) peer list: (empty)
*Mar 12 04:03:57.201: ccCallAlert (callID=0x11, prog_ind=0x8,
sig_ind=0x1)
*Mar 12 04:03:57.201: ccConferenceCreate (confID=0x617A8808,
callID1=0x11, callID2=0x12, tag=0x0)
*Mar 12 04:03:57.201: cc_api_bridge_done (confID=0x7,
srcIF=0x616C9F54, srcCallID=0x11, dstCallID=0x12, disposition=0,
tag=0x0)value H323-UserInformation
*Mar 12 04:03:57.201: {
*Mar 12 04:03:57.201:   h323-uu-pdu
*Mar 12 04:03:57.201:   {
*Mar 12 04:03:57.201:     h323-message-body alerting :
*Mar 12 04:03:57.201:     {
*Mar 12 04:03:57.201:       protocolIdentifier { 0 0 8 2250 0 2 },
*Mar 12 04:03:57.205:       destinationInfo
*Mar 12 04:03:57.205:       {
*Mar 12 04:03:57.205:         mc FALSE,
*Mar 12 04:03:57.205:         undefinedNode FALSE
*Mar 12 04:03:57.205:       },

```

Notice in this packet that Cisco IOS is also sending the H.245 address and port number to Cisco CallManager. Sometimes the Cisco IOS Gateway will send the unreachable address, which could cause either no audio or one-way audio.

```

*Mar 12 04:03:57.205:   h245Address ipAddress :
*Mar 12 04:03:57.205:   {
*Mar 12 04:03:57.205:     ip 'AC1046E2'H,
*Mar 12 04:03:57.205:     port 011008
*Mar 12 04:03:57.205:   },
*Mar 12 04:03:57.213: Hex representation of the ALERTING TPKT to
send.0300003D0100
*Mar 12 04:03:57.213:
*Mar 12 04:03:57.213:   H225Lib::h225AlertRequest: Q.931 ALERTING
sent from socket [1]. Call state changed to [Call Received].
*Mar 12 04:03:57.213: cc_api_bridge_done (confID=0x7,
srcIF=0x617BE064, srcCallID=0x12, dstCallID=0x11, disposition=0,
tag=0x0)

```

The following debug output shows that the H.245 session is coming up. You can see the capability indication for codec negotiation, as well as how many bytes will be present in each voice packet.

```
*Mar 12 04:03:57.217: cc_api_caps_ind (dstVdbPtr=0x616C9F54,
dstCallId=0x11, srcCallId=0x12, caps={codec=0xEBFB, fax_rate=0x7F,
vad=0x3, modem=0x617C5720 codec_bytes=0, signal_type=3})
*Mar 12 04:03:57.217: sess_appl: ev(23=CC_EV_CONF_CREATE_DONE),
cid(17), disp(0)
*Mar 12 04:03:57.217: ssa:
cid(17)st(3)oldst(0)cfid(7)csz(0)in(1)fDest(1)-cid2(18)st2(3)oldst2(
1)
*Mar 12 04:03:57.653: cc_api_caps_ind (dstVdbPtr=0x617BE064,
dstCallId=0x12, srcCallId=0x11, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x01, codec_bytes=160, signal_type=0})
```

The following debug output shows that both parties negotiated correctly and agreed on G.711 codec with 160 bytes of data.

```
*Mar 12 04:03:57.653: cc_api_caps_ack (dstVdbPtr=0x617BE064,
dstCallId=0x12, srcCallId=0x11, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x01, codec_bytes=160, signal_type=0})
*Mar 12 04:03:57.653: cc_api_caps_ind (dstVdbPtr=0x617BE064,
dstCallId=0x12, srcCallId=0x11, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x0, codec_bytes=160, signal_type=0})
*Mar 12 04:03:57.653: cc_api_caps_ack (dstVdbPtr=0x617BE064,
dstCallId=0x12, srcCallId=0x11, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x01, codec_bytes=160, signal_type=0})
*Mar 12 04:03:57.657: cc_api_caps_ack (dstVdbPtr=0x616C9F54,
dstCallId=0x11, srcCallId=0x12, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x01, codec_bytes=160, signal_type=0})
*Mar 12 04:03:57.657: cc_api_caps_ack (dstVdbPtr=0x616C9F54,
dstCallId=0x11, srcCallId=0x12, caps={codec=0x01, fax_rate=0x2,
vad=0x2, modem=0x01, codec_bytes=160, signal_type=0})
```

The H.323 connect and disconnect messages follow.

```
*Mar 12 04:03:59.373: cc_api_call_connected(vdbPtr=0x617BE064,
callID=0x12)
*Mar 12 04:03:59.373: sess_appl: ev(8=CC_EV_CALL_CONNECTED), cid(18),
disp(0)
*Mar 12 04:03:59.373: ssa:
cid(18)st(4)oldst(1)cfid(7)csz(0)in(0)fDest(0)-cid2(17)st2(4)oldst2(
3)
*Mar 12 04:03:59.373: ccCallConnect (callID=0x11)
*Mar 12 04:03:59.373: {
*Mar 12 04:03:59.373:   h323-uu-pdu
*Mar 12 04:03:59.373:   {
*Mar 12 04:03:59.373:     h323-message-body connect :
```

```
*Mar 12 04:03:59.373:          {
*Mar 12 04:03:59.373:          protocolIdentifier { 0 0 8 2250 0 2 },
*Mar 12 04:03:59.373:          h245Address ipAddress :
*Mar 12 04:03:59.373:          {
*Mar 12 04:03:59.377:              ip 'AC1046E2'H,
*Mar 12 04:03:59.377:              port 011008
*Mar 12 04:03:59.377:          },
*Mar 12 04:03:59.389: Hex representation of the CONNECT TPKT to
send.03000052080
*Mar 12 04:03:59.393: H225Lib::h225SetupResponse: Q.931 CONNECT sent
from socket [1]
*Mar 12 04:03:59.393: H225Lib::h225SetupResponse: Q.931 Call State
changed to [Active].
*Mar 12 04:04:08.769: cc_api_call_disconnected(vdbPtr=0x617BE064,
callID=0x12, cause=0x10)
*Mar 12 04:04:08.769: sess_appl: ev(12=CC_EV_CALL_DISCONNECTED),
cid(18), disp(0)
```

Cisco IOS Gateway with T1/PRI Interface

As explained earlier, there are two types of calls going through the Cisco IOS Gateways: the Cisco IOS Gateway interfaces to the PSTN or PBX with either T1/CAS or T1/PRI interfaces. The following are the debug outputs when the Cisco IOS Gateways use T1/PRI interface.

The debug isdn q931 command on the Cisco IOS Gateway has been turned on, enabling Q.931, a Layer Three signaling protocol for D-channel in the ISDN environment. Each time a call is placed out of the T1/PRI interface, a setup packet must be sent. The setup packet always has (protocol descriptor) pd = 8 and it generates a random hexadecimal value for the callref. The callref is used to track the call. For example, if two calls are placed, the callref value can determine the call for which the RX (received) message is intended. Bearer capability 0x8890 means a 64kb/s data call. If it were a 0x8890218F, then it would be a 56kb/s data call and 0x8090A3 if it is a voice call. In the debug output below, the bearer capability is 0x8090A3, which is for voice. Called and calling party numbers are also shown.

The callref uses a different value for the first digit (to differentiate between TX and RX) and the second value is the same (SETUP had a 0 for the last digit and CONNECT_ACK also has a 0). The router is completely dependent upon the PSTN or PBX to assign a Bearer channel (B-channel). If the PSTN or PBX does not assign a channel to the router, the call will not be routed. In our case, a

CONNECT message is received from the switch with the same reference number as was received for ALERTING (0x800B). Finally, you can see the exchange of the DISCONNECT message followed by RELEASE and RELEASE_COMP messages as the call is being disconnected. RELEASE_COMP messages are followed by a cause ID for the call rejection. The cause ID is a hexadecimal value. The meaning of the cause can be found by decoding the hexadecimal value and following up with your provider.

```
*Mar 1 225209.694 ISDN Se115 TX -> SETUP pd = 8 callref = 0x000B
*Mar 1 225209.694 Bearer Capability i = 0x8090A3
*Mar 1 225209.694 Channel ID i = 0xA98381
*Mar 1 225209.694 Calling Party Number i = 0x2183, '1001'
*Mar 1 225209.694 Called Party Number i = 0x80, '3333'
*Mar 1 225209.982 ISDN Se115 RX <- ALERTING pd = 8 callref =
0x800B
*Mar 1 225209.982 Channel ID i = 0xA98381
*Mar 1 225210.674 ISDN Se115 RX <- CONNECT pd = 8 callref = 0x800B
*Mar 1 225210.678 ISDN Se115 TX -> CONNECT_ACK pd = 8 callref =
0x000B
*Mar 1 225215.058 ISDN Se115 RX <- DISCONNECT pd = 8 callref =
0x800B
*Mar 1 225215.058 Cause i = 0x8090 - Normal call clearing
225217 %ISDN-6
DISCONNECT Int S10 disconnected from unknown , call lasted 4 sec
*Mar 1 225215.058 ISDN Se115 TX -> RELEASE pd = 8 callref = 0x000B
*Mar 1 225215.082 ISDN Se115 RX <- RELEASE_COMP pd = 8 callref =
0x800B
*Mar 1 225215.082 Cause i = 0x829F - Normal, unspecified or Special
intercept, call blocked group restriction
```

Cisco IOS Gateway with T1/CAS Interface

There are two types of calls going through the Cisco IOS Gateways: the Cisco IOS Gateway interface to the PSTN or PBX with either T1/CAS or T1/PRI interfaces. The following are the debug outputs when the Cisco IOS Gateways has T1/CAS interface. The debug cas on the Cisco IOS Gateway has been turned on.

The following debug message shows that the Cisco IOS Gateway is sending an off-hook signal to the switch.

```
Apr 5 17:58:21.727: from NEAT(0): (0/15): Tx LOOP_CLOSURE (ABCD=1111)
```

The following debug message indicates the switch is sending wink after receiving the loop closure signal from the Cisco IOS Gateway.

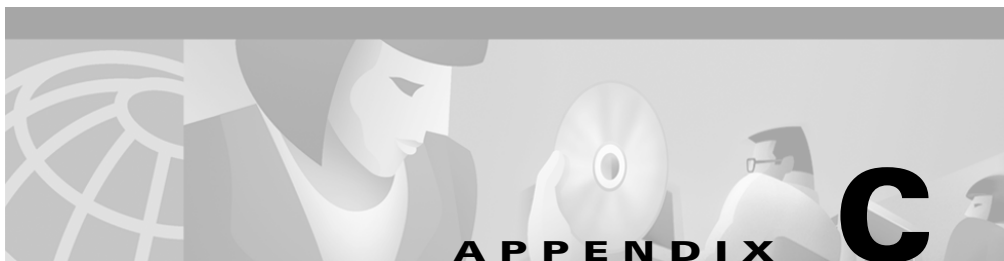
```
Apr  5 17:58:21.859: from NEAT(0): (0/15): Rx LOOP_CLOSURE (ABCD=1111)
Apr  5 17:58:22.083: from NEAT(0): (0/15): Rx LOOP_OPEN (ABCD=0000)
```

The following debug message indicates the Cisco IOS Gateway is going off-hook.

```
Apr  5 17:58:23.499: from NEAT(0): (0/15): Rx LOOP_CLOSURE (ABCD=1111)
```

The following is the output of the show call active voice brief on the Cisco IOS Gateway when the call is in progress. The called and calling party number and other useful information are also shown.

```
R5300-5#show call active voice brief
<ID>: <start>hs.<index> +<connect> pid:<peer_id> <dir> <addr> <state>
tx:<packets>/<bytes> rx:<packets>/<bytes> <state>
  IP <ip>:<udp> rtt:<time>ms pl:<play>/<gap>ms
lost:<lost>/<early>/<late> delay:<last>/<min>/<max>ms <codec>
  FR <protocol> [int dlci cid] vad:<y/n> dtmf:<y/n> seq:<y/n>
sig:<on/off> <codec> (payload size)
  Tele <int>: tx:<tot>/<v>/<fax>ms <codec> noise:<l> acom:<l>
i/o:<l>/<l> dBm
511D : 156043737hs.1 +645 pid:0 Answer 1001 active
  tx:1752/280320 rx:988/158080
  IP172.16.70.228:18888 rtt:0ms pl:15750/80ms lost:0/0/0
delay:25/25/65ms g711ulaw
511D : 156043738hs.1 +644 pid:1 Originate 3333 active
  tx:988/136972 rx:1759/302548
  Tele 1/0/0 (30): tx:39090/35195/0ms g711ulaw noise:-43 acom:0
i/o:-36/-42 dBm
```

Troubleshooting Inter-Cluster Calls

The case study in this appendix examines a Cisco IP Phone calling another Cisco IP Phone located in a different cluster. This type of call is also known as an inter-cluster Cisco IP Phone call.

This appendix contains the following topics:

- [Sample Topology, page C-1](#)
- [Inter-Cluster H.323 Communication, page C-1](#)
- [Call Flow Traces, page C-2](#)
- [Failed Call Flow, page C-4](#)

Sample Topology

The following is the sample topology used in this case study. There are two clusters, each having two Cisco CallManagers. There are also Cisco IOS Gateways and a Cisco IOS Gatekeeper in place.

Inter-Cluster H.323 Communication

The Cisco IP Phone in Cluster 1 is making a call to the Cisco IP Phone in Cluster 2. Inter-cluster Cisco CallManager communication takes place using the H.323 Version 2 protocol. There is also a Cisco IOS Gatekeeper for admission control.

The Cisco IP Phone can talk to the Cisco CallManager via Skinny Station protocol, and the Cisco CallManager can talk with the Cisco IOS Gatekeeper using the H.323 Remote Access Server (RAS) protocol. The admission request message (ARQ) is sent to the Cisco IOS Gatekeeper, which sends the admission confirmed message (ACF) after making sure the inter-cluster call can be made using H.323 version 2 protocol. Once this happens, the audio path is made using the RTP protocol between Cisco IP Phones in different clusters.

Call Flow Traces

This section discusses the call flow using SDI trace examples captured in the CCM000000000 file. The traces discussed in this case study focus only on the call flow itself.

In this call flow, a Cisco IP Phone (2002) located in Cluster 2 is calling a Cisco IP Phone (1001) located in Cluster 1. Remember that you can follow a device through the trace by looking at the TCP handle value, time stamp, or name of the device. The TCP handle value for the device remains the same until the device is rebooted or goes offline.

In the following traces, the Cisco IP Phone (2002) has gone off-hook. The trace shows the unique messages, TCP handle, and the calling number, which is displayed on the Cisco IP Phone. The called number (1001), H.225 connect, and H.245 confirm messages can be seen in the following debug output. The codec type is G.711 μ -law.

```
16:06:13.921 CCM|StationInit - InboundStim - OffHookMessageID
tcpHandle=0x1c64310
16:06:13.953 CCM|Out Message -- H225ConnectMsg -- Protocol=
H225Protocol
16:06:13.953 CCM|Ie - H225UserUserIe IEData= 7E 00 37 05 02 C0 06
16:06:13.953 CCM|StationD - stationOutputCallInfo CallingPartyName=,
CallingParty=2002, CalledPartyName=1001, CalledParty=1001,
tcpHandle=0x1c64310
16:06:14.015 CCM|H245Interface(2) OLC indication chan number = 2
16:06:14.015 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x1c64310 myIP: e74610ac (172.16.70.231)
16:06:14.015 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
16:06:14.062 CCM|StationInit - InboundStim -
StationOpenReceiveChannelAckID tcpHandle=0x1c64310, Status=0,
IpAddr=0xe74610ac, Port=20444, PartyID=2
```

```

16:06:14.062 CCM|H245Interface(2) paths established ip = e74610ac,
port = 20444
16:06:14.187 CCM|H245Interface(2) OLC outgoing confirm ip = fc4610ac,
port = 29626

```

The calling and called party number, which is associated with an IP address and a hexadecimal value, can be seen in the following traces.

```

16:06:14.187 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x1c64310 myIP: e74610ac (172.16.70.231)
16:06:14.187 CCM|StationD - RemoteIpAddr: fc4610ac (172.16.70.252)

```

The following traces show the packet sizes and the MAC address of the Cisco IP Phone (2002). These traces are followed by the disconnect, then on-hook messages.

```

RemoteRtpPortNumber: 29626 msecPacketSize: 20
compressionType:(4)Media_Payload_G711Ulaw64k
16:06:16.515 CCM| Device SEP003094C26105 , UnRegisters with SDL Link
to monitor NodeID= 1
16:06:16.515 CCM|StationD - stationOutputCloseReceiveChannel
tcpHandle=0x1c64310 myIP: e74610ac (172.16.70.231)
16:06:16.515 CCM|StationD - stationOutputStopMediaTransmission
tcpHandle=0x1c64310 myIP: e74610ac (172.16.70.231)
16:06:16.531 CCM|In Message -- H225ReleaseCompleteMsg -- Protocol=
H225Protocol
16:06:16.531 CCM|Ie - Q931CauseIe -- IEData= 08 02 80 90
16:06:16.531 CCM|Ie - H225UserUserIe -- IEData= 7E 00 1D 05 05 80 06
16:06:16.531 CCM|Locations:Orig=1 BW=64Dest=0 BW=-1 (-1 implies
infinite bw available)
16:06:16.531 CCM|MediaManager - wait_AuDisconnectRequest - StopSession
sending disconnect to (64,2) and remove connection from list
16:06:16.531 CCM|MediaManager - wait_AuDisconnectReply - received all
disconnect replies, forwarding a reply for party1(16777219) and
party2(16777220)
16:06:16.531 CCM|MediaCoordinator - wait_AuDisconnectReply - removing
MediaManager(2) from connection list
16:06:16.734 CCM|StationInit - InboundStim - OnHookMessageID
tcpHandle=0x1c64310

```

Failed Call Flow

The following section describes an unsuccessful inter-cluster call flow, as seen in the SDI trace. In the traces below, the Cisco IP Phone (1001) has gone off-hook. A TCP handle is assigned to the Cisco IP Phone.

```
16:05:33.468 CCM|StationInit - InboundStim - OffHookMessageID
tcpHandle=0x4fbbc30
16:05:33.468 CCM|StationD - stationOutputDisplayText
tcpHandle=0x4fbbc30, Display= 1001
16:05:33.484 CCM|StationD - stationOutputSetLamp stim: 9=Line
instance=1 lampMode=LampOn tcpHandle=0x4fbbc30
```

In the following traces, the user is dialing the called number (2000) of the Cisco IP Phone, and the process of digit analysis is trying to match the number.

```
16:05:33.484 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="")
16:05:33.484 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:35.921 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="2")
16:05:35.921 CCM|Digit
analysis:potentialMatches=ExclusivelyOffnetPotentialMatchesExist
16:05:36.437 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="20")
16:05:36.437 CCM|Digit
analysis:potentialMatches=ExclusivelyOffnetPotentialMatchesExist
16:05:36.656 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="200")
16:05:36.656 CCM|Digit
analysis:potentialMatches=ExclusivelyOffnetPotentialMatchesExist
16:05:36.812 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="2000")
```

The digit analysis has now been completed and the results are shown in the following traces. It is important to note that the PotentialMatches=NoPotentialMatchesExist reference below indicates that the Cisco CallManager is unable to match this directory number. Finally a reorder tone is sent to the calling party (1001), which is followed by an on-hook message.

```
16:05:36.812 CCM|Digit analysis: analysis results
16:05:36.812 CCM||PretransformCallingPartyNumber=1001
|CallingPartyNumber=1001
|DialingPattern=2XXX
|DialingRoutePatternRegularExpression=(2XXX)
|PotentialMatches=NoPotentialMatchesExist
```

```
|CollectedDigits=2000
16:05:36.828 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=2000, tcpHandle=0x4fbbc30
16:05:36.828 CCM|StationD - stationOutputStartTone: 37=ReorderTone
tcpHandle=0x4fbbc30
16:05:37.953 CCM|StationInit - InboundStim - OnHookMessageID
tcpHandle=0x4fbbc30
```

Failed Call Flow



Call Detail and Call Management Records

This appendix provides detailed information about Call Detail Records (CDR) and Call Management Records (CMR).

CDR records are written to a database for use in post-processing activities. These activities include many functions but this appendix will focus primarily on billing and network analysis.

The database is a Microsoft SQL Server 7.0 database. Access to the database can be made via Open DataBase Connectivity (ODBC).

Access is provided to all tables in the database in a read-only fashion, and to the CDR and CMR tables in a read or write fashion.

To use CDR record data, you may want to read other tables in the database in an effort to obtain information about the type of device the CDR describes. This correlation between devices in the device table and the IP address listed in the CDR record is not straightforward and is listed as a known issue later in this appendix.

This appendix contains the following topics:

- [Writing Records, page D-2](#)
- [Reading Records, page D-2](#)
- [Removing Records, page D-3](#)
- [Table Schema, page D-4](#)
- [Issues, page D-4](#)
- [Field Data Conversions, page D-4](#)

- [Call Records Logged By Call Type](#), page D-20
- [Call Management Records Logged By Call Type](#), page D-22
- [Codec Types \(Compression / Payload types\)](#), page D-23
- [Cause Codes](#), page D-24
- [Alarms](#), page D-28

Writing Records

Cisco CallManager writes CDR records to the SQL database as calls are made in a manner consistent with the configuration of each individual Cisco CallManager. This configuration is made using the Service Parameters screen in Cisco CallManager Administration.

All records are written to the primary database for a cluster. If the primary database is not available, then they will be written to any of the other backup databases. Once the primary database becomes available, then writing new records will continue on the primary database and the locally written records will be moved to the primary.

Reading Records

The easiest way to read data from the SQL database may be to use ODBC. A good connection string would look like:

```
DRIVER={SQL Server};SERVER=machineX;DATABASE=CCM0300
```

**Note**

Be sure to use the correct database name. If you have installed a Cisco CallManager Release 3.1(1) version over an existing installation, then the database might be migrated if called for by the new installation. In this case, the old database will still exist, and the new database will also exist. The names will differ by adding one to the number of the name. For instance, the original name is CCM0300. After a migration, the newer database name will be CCM0311. You should use the highest number database.

Refer to the Cisco CallManager Release 3.1 documentation available online at the following location:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm

The primary database (machine and name) currently in use by the cluster can be found by clicking on Details in Cisco CallManager Administration (click Help to reach the Welcome screen where Details is located). The registry on machines hosting a database can also be checked. Look at the registry key: \\HKEY_LOCAL_MACHINE\Software\Cisco Systems Inc.\DBL for the item called DBConnection0. This string item contains a connection string similar to that shown above with the machine name and database name of the primary database.

Access is controlled by use of SQL Users. The following table specifies the UserID and password that should be used when accessing the Cisco CallManager database.

Tables	SQL UserID	Password	Capability
CallDetailRecord	CiscoCCMCDR	dipsy	Read/Write
CallDetailRecordDiagnostic			
(Other)	CiscoCCMReader	cowboys	Read Only

Removing Records

Because Cisco CallManager relies on third-party applications to post-process the CDR data, you should remove the CDR data when all applications are through with the data.

If CDR records accumulate to a configured maximum (10,000,000 CDR records), then the oldest CDR records will be removed along with related CMR records once a day.

When removing CDR data after analysis, be sure to remove all related CMR records also.

Table Schema

Detailed information about the format and use of each field in the CDR is provided in [Table D-1](#).

The main tables you should use are the CallDetailRecord table, which holds CDR records, and the CallDetailRecordDiagnostic table, which holds CMR records. The CallDetailRecord table is related to the CallDetailRecordDiagnostic table via the two GlobalCallID columns, GlobalCallID_callManagerId and GlobalCallID_callId. There may be more than one CMR per CDR.

The CallDetailRecord table holds information about the endpoints of the call and other call control/routing aspects of the call. The CallDetailRecordDiagnostic table holds information about the quality of the streamed audio of the call.

Issues

Cisco CallManager Release 3.1(1) may have issues with the CDR data. Contact Cisco TAC or refer to the Release Notes for Cisco CallManager 3.1(1) at the following location:

http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/rel_note/index.htm

Field Data Conversions

There are some fields that require conversion from decimal format to another format for displays. This appendix defines their values, and how to convert them or where to get information on how to convert them.

Time Values

All time values are represented as unsigned 32 bit integers. This unsigned integer value is displayed from the database as a signed integer.

This field is a `time_t` value that is obtained from the Windows NT (2000) system routines. The value is a coordinated universal time (UTC) value, and represents the number of seconds since Midnight (00:00:00) Jan. 1, 1970.

Deciphering the Time Stamp

Using Microsoft Excel, you can write a formula to make converting this time stamp a little easier. If the value is in cell A1, you can make another cell:

```
=A1/86400+DATE(1970,1,1)
```

There are 86400 seconds in a day. Then format the resulting cell as a date/time field in Excel.

IP Addresses

All IP addresses are stored in the system as unsigned integers. The database displays them as signed integers. To convert the signed decimal value to an IP address, first convert the value to a hexadecimal number (taking into consideration that it is really an unsigned number). The 32bit hexadecimal value represents four bytes. The four bytes are in reverse order (Intel standard). To get the IP address, reverse the order of the bytes, and convert each byte to a decimal number. The resulting four bytes represent the 4-byte fields of the IP address in dotted notation.

**Note**

The database will display a negative number when the low byte of the IP address has the most significant bit set.

Converting IP Addresses

For example: IP Address 192.168.18.188 would be displayed as follows

Database Display = -1139627840.

This converts to a hexadecimal value of 0xBC12A8C0.

Reverse the hexadecimal bytes = C0A812BC

C0 A8 12 BC

Bytes Converted from hexadecimal to Decimal = 192 168 18 188 which would be displayed as 192.168.18.188.

Example2: IP Address 192.168.18.59

Database Display = 991078592

This converts to a hexadecimal value of 0x3B12A8C0

Reverse Byte order = C0A8123B

C0 A8 12 3B

Bytes Converted from hexadecimal to Decimal = 192 168 18 59 which would be displayed as 192.168.18.59.

CDR Field Definition

[Table D-1](#) provides field definitions for CDRs.

Table D-1 CDR Field Definitions

Field	Definition
cdrRecordType	Type of this record unsigned integer Specifies the type of this specific record. It could be a Start call record(0), End call record(1), or a CMR record(2).
globalCallIdentifier	Global Call Identifier The Global Call Identifier consists of two fields that are both unsigned integers. The values must be treated as unsigned integers. The two fields are: Unsigned integer GlobalCallID_CallID This is the call identifier that is assigned to the entire call. All records associated with a standard call will have the same global call identifier.
origLegCallIdentifier	Origination leg call identifier unsigned integer This is a unique identifier that is used to track the origination leg of a call. It is unique within a cluster.

Table D-1 CDR Field Definitions (continued)

Field	Definition
dateTimeOrigination	Date/time of call origination unsigned integer This represents the time that the device originating the call went off hook, or the time that an outside call was first recognized by the system (it received the Setup message). The value is a coordinated universal time (UTC) value, and represents the number of seconds since Midnight (00:00:00) Jan. 1, 1970.
origNodeId	Originator's node ID unsigned integer This field represents the node within the Cisco CallManager cluster where the call originator was registered at the time of this call.
origSpan	Originator's span or port unsigned integer This field contains the originator's port or span number if the call originated through a gateway. If not, this field contains zero (0).
callingPartyNumber	Calling party number up to 25 characters This is the directory number of the device from which the call originated.
origIpPort	Calling party's IP port unsigned integer This field contains the IP Port of the device from which the call originated.

Table D-1 CDR Field Definitions (continued)

Field	Definition
origIpAddr	<p>Calling party's IP address unsigned integer</p> <p>This field contains the IP address of the device from which the call originated.</p>
originalCallingPartyNumberPartition	<p>Calling party's partition up to 50 characters</p> <p>This field contains the Partition associated with the calling party.</p>
origCause_Location	<p>SDN location value unsigned integer</p> <p>This field contains the location value from the Cause information Element.</p>
origCause_Value	<p>Calling party cause Of call termination unsigned integer</p> <p>This cause represents why the call to the originating device was terminated. In the case of transfers, forwards, and so on, the cause of call termination may be different for the originating device and the termination device. Thus, there are two cause fields associated with each call. Usually they will be the same.</p>
origMediaTransportAddress_IP	<p>The IP address for the originator's media connection unsigned integer</p> <p>This is the destination IP Address to which the Media Stream from the originator was connected.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
origMediaTransportAddress_Port	<p>The port for the originator's media connection</p> <p>unsigned integer</p> <p>This is the destination port to which the Media Stream from the originator was connected.</p>
origMediaCap_payloadCapability	<p>The codec type used by the originator</p> <p>unsigned integer</p> <p>This field contains the Codec type (compression or payload type) that the originator used on the sending side during this call. It may be different than the codec type used on its receiving side.</p>
origMediaCap_maxFramesPerPacket	<p>The number of milliseconds of data per packet</p> <p>unsigned integer</p> <p>This field contains the number of milliseconds of data per packet sent to the destination, by the originator of this call. The actual data size depends on the codec type being used to generate the data.</p>
origMediaCap_g723BitRate	<p>The bit rate to be used by G.723</p> <p>unsigned integer</p> <p>Defines the bit rate to be used by G.723. There are two bit rate values. They are: 1 =5.3K bit rate, and 2 = 6.3K bit rate.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
lastRedirectDn	<p>Directory number of the party that last redirected this call</p> <p>up to 25 characters</p> <p>This is the directory number of the last device that redirected this call. This field applies only to calls that were redirected, such as conference calls, call forwarded calls, and so on.</p>
lastRedirectDnPartition	<p>Partition of the phone that last redirected this call</p> <p>up to 50 characters</p> <p>This is the Partition of the last device that redirected this call. This field applies only to calls that were redirected such as conference calls, call forwarded calls, and so on.</p>
destLegIdentifier	<p>The call identifier for the destination leg of the call</p> <p>unsigned integer</p> <p>This is a unique identifier that is used to track the destination leg of this call. It is unique within a cluster.</p>
destNodeId	<p>The node identifier for the node where the destination of the call was registered</p> <p>unsigned integer</p> <p>The node within the Cisco CallManager cluster where the destination device was registered at the time of this call.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
dest Span	<p>The destination span or port unsigned integer</p> <p>This field contains the destination port or span number if the call was terminated through a gateway. If not, this field contains a (0) zero.</p>
destIpAddr	<p>The IP address to which the call was delivered unsigned integer</p> <p>This field contains the IP address of the signaling connection on the device that terminated the call.</p>
destIpPort	<p>The IP port to which the call was delivered unsigned integer</p> <p>This field contains the IP port of the signaling connection on the device that terminated the call.</p>
originalCalledPartyNumber	<p>The destination received from the call originator up to 25 characters</p> <p>This field contains the Directory Number to which the call was originally extended based on the digits dialed by the originator of the call. If the call completes normally (meaning it was not forwarded), this Directory Number should always be the same as the finalCalledPartyNumber. If the call was forwarded, this field contains the original destination of the call before it was forwarded.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
originalCalledPartyNumberPartition	<p>Called party's partition up to 50 characters</p> <p>This field contains the partition associated with the called party.</p>
finalCalledPartyNumber	<p>The destination to which the call was delivered up to 25 characters</p> <p>This field contains the Directory Number to which the call was actually extended. If the call completes normally (meaning it was not forwarded), this Directory Number should always be the same as the originalCalledPartyNumber. If the call was forwarded, this field contains the Directory Number of the final destination of the call after all forwards were completed.</p>
finalCalledPartyNumberPartition	<p>The partition associated with the final destination of the call. up to 50 characters</p> <p>This field contains the partition associated with the destination to which the call was actually extended. In a normal call, this field should be the same as originalCalledPartyNumberPartition. If the call was forwarded, this field contains the partition of the final destination of the call after all forwards were completed.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
destCause_location	Called party cause location unsigned integer This is the ISDN Location value from the Cause Information Element.
destCause_value	Called party cause of call termination unsigned integer This cause describes why the device was terminated. In the case of transfers, forwards, and so on, the cause of call termination may be different for the recipient of the call and the originator of the call. Thus, there are two cause fields associated with each call. Usually they will be the same. When an attempt is made to extend a call to a busy device that is forwarded, the cause code will reflect Busy even though the call was connected to a forward destination.
destMediaTransportAddress_IP	The IP address for the destination outgoing media connection unsigned integer This is the origination IP Address from which the Media Stream from the destination was connected.
origMediaTransportAddress_Port	The port for the destination outgoing media connection unsigned integer This is the originator's port from which the Media Stream from the destination was connected.

Table D-1 CDR Field Definitions (continued)

Field	Definition
destMediaCap_payloadCapability	<p>The codec type used by the destination on the sending side.</p> <p>unsigned integer</p> <p>This field contains the Codec type (compression or payload type) that the destination used on its sending side during this call. It may be different than the codec type used on its receiving side.</p>
destMediaCap_maxFramesPerPacket	<p>The number of milliseconds of data per packet</p> <p>unsigned integer</p> <p>This field contains the number of milliseconds of data per packet sent to the originator, by the destination of this call. The actual data size depends on the codec type being used to generate the data.</p>
destMediaCap_g723BitRate	<p>The bit rate to be used by G.723</p> <p>unsigned integer</p> <p>Defines the bit rate to be used by G.723. There are two bit rate values. They are: 1 = 5.3K bit rate, and 2 = 6.3K bit rate.</p>

Table D-1 CDR Field Definitions (continued)

Field	Definition
dateTimeConnect	<p>Date/time of connect</p> <p>unsigned integer</p> <p>This is the date and time that the call was connected between the originating and terminating devices. This is the date and time that the call was connected between the originating and terminating devices. The value is a coordinated universal time (UTC) value, and represents the number of seconds since Midnight (00:00:00) Jan. 1, 1970.</p>
dateTimeDisconnect	<p>Date/time of disconnect</p> <p>unsigned integer</p> <p>This is the time that the call was disconnected between the originating and terminating devices, or when the call was torn down even if it was never connected. The value is a coordinated universal time (UTC) value, and represents the number of seconds since Midnight (00:00:00) Jan. 1, 1970.</p>
duration	<p>Call duration</p> <p>This is the number of seconds that the call was connected. It is the difference between the date/time of connect and the date/time of disconnect.</p>

Table D-2 provides field definitions for CMRs (diagnostic CDRs).

Table D-2 CMR Field Definitions

Field	Definition
cdrRecordType	Type of this record unsigned integer Specifies the type of this specific record. It will be set to CMR record.
globalCallIdentifier	Global Call Identifier for this call The Global Call Identifier consists of two fields which are both unsigned integers. The values must be treated as unsigned integers. The two fields are: Unsigned integer GlobalCallID_CallID Unsigned integer GlobalCallID_CallManagerID This is the call identifier that is assigned to the entire call. All records associated with a standard call will have the same global call identifier.
nodeID	The Cisco CallManager node identifier The node within the Cisco CallManager cluster where this record was generated.
callIdentifier	Call Identifier unsigned integer This is a call leg identifier that identifies to which call leg this record pertains.
directoryNum	Directory number used on this call This is the directory number of the device from which these diagnostics were collected.

Table D-2 CMR Field Definitions (continued)

Field	Definition
directoryNumPartition	The partition associated with the directory number This is the partition of the directory number in this record.
dateTimeStamp	Date/time of call termination This represents the approximate time that the device went on hook. The time is put into the record when the phone responds to a request for diagnostic information. This is a time_t value.
numberPacketsSent	Number of packets sent The total number of RTP data packets transmitted by the device since starting transmission on this connection. The value is zero if the connection was set in receive only mode.
numberOctetsSent	Number of octets (bytes) of data sent to the other party The total number of payload octets (that is, not including header or padding) transmitted in RTP data packets by the device since starting transmission on this connection. The value is zero if the connection was set in receive only mode.
numberPacketsReceived	The number of data packets received during this call The total number of RTP data packets received by the device since starting reception on this connection. The count includes packets received from different sources if this is a multicast call. The value is zero if the connection was set in send only mode.

Table D-2 CMR Field Definitions (continued)

Field	Definition
numberOctetsReceived	<p>The number of octets (bytes) of data received during this call</p> <p>The total number of payload octets (that is, not including header or padding) received in RTP data packets by the device since starting reception on this connection. The count includes packets received from different sources, if this is a multicast call. The value is zero if the connection was set in send only mode.</p>
numberPacketsLost	<p>Lost RTP packets during this connection</p> <p>The total number of RTP data packets that have been lost since the beginning of reception. This number is defined as the number of packets expected less the number of packets actually received, where the number of packets received includes any that are late or duplicates. Thus, packets that arrive late are not counted as lost, and the loss may be negative if there are duplicates. The number of packets expected is defined to be the extended last sequence number received, as defined next, less the initial sequence number received. The value is zero if the connection was set in send only mode. (For details, refer to RFC 1889)</p>

Table D-2 CMR Field Definitions (continued)

Field	Definition
jitter	The interarrival jitter during this connection An estimate of the statistical variance of the RTP data packet interarrival time, measured in milliseconds and expressed as an unsigned integer. The interarrival jitter J is defined to be the mean deviation (smoothed absolute value) of the difference D in packet spacing at the receiver compared to the sender for a pair of packets. Detailed computation algorithms are found in RFC 1889. The value is zero if the connection was set in send only mode.
latency	The latency experienced during this connection The value is an estimate of the network latency, expressed in milliseconds. This is the average value of the difference between the NTP timestamp indicated by the senders of the RTCP messages and the NTP timestamp of the receivers, measured when these messages are received. The average is obtained by summing all the estimates, then dividing by the number of RTCP messages that have been received. (For details refer to RFC 1889)

Call Records Logged By Call Type

Each normal call between two parties logs one CDR End Call record. Each End Call record contains all fields identified above, but some fields may not be used. If a field is not used, it will be blank if it is an ASCII string field, or 0 if it is a numeric field. When supplementary services are involved in a call, more End Call records may be written.

In addition to the CDR End Call record, there may be up to one CMR record per endpoint involved in a call. In a normal call between two parties each using a Cisco IP Phone, there will be two CMR records written: one for the originator, and one for the destination of the call.

This section describes the records written for different call types in the system.

Normal Calls (Cisco IP Phone-to-Cisco IP Phone)

Normal calls log three records per call. They are: EndCall plus two diagnostic records, one for each endpoint. In the EndCall record, all fields may contain valid information. The duration will always be non-zero unless the CdrLogCallsWithZeroDurationFlag flag is enabled (set to true). The originalCalledPartyNumber field will contain the same directory number as the finalCalledPartyNumber field.

Abandoned Calls

The logging of calls with zero duration is optional. Normally these records will not be logged. If logging calls with zero duration is enabled, the following things should be noted.

- If the call was abandoned (such as when a phone is taken off hook, and placed back on hook), various fields will not contain data. In this case, the originalCalledPartyNumber, finalCalledPartyNumber, the partitions associated with them, destIpAddr, and the dateTimeConnect fields will be blank. All calls that were not connected will have a duration of zero seconds. When a call is abandoned, the cause code is 0.
- If the user dialed a directory number and then abandoned the call before it was connected, the First Dest and Final Dest fields and their associated partitions will contain the directory number and partition to which the call would have been extended. The Dest Ip field will be blank, and the duration will be zero.

Forwarded or Redirected Calls

The call records for forwarded calls will be the same as those for normal calls except for the originalCalledPartyNumber field, and the originalCalledPartyNumberPartition fields. These fields will contain the directory number and partition for the destination that was originally dialed by the originator of the call. If the call was forwarded, the finalCalledPartyNumber and finalCalledPartyNumberPartition fields will be different, and will contain the

directory number and partition of the final destination of the call. Also, when a call is forwarded, the lastRedirectDn and lastRedirectDnPartition fields will contain the directory number and partition of the last phone that forwarded or redirected this call.

Calls With Busy or Bad Destinations

These calls will be logged as a normal call with all relevant fields containing data. The Called Party Cause field will contain a cause code indicating why the call was not connected, and the Called Party IP and Date/Time Connect fields will be blank. If the originator abandoned the call, the cause will be NO_ERROR (0). The Duration will always be zero seconds. These calls will not be logged unless CdrLogCallsWithZeroDurationFlag is enabled.

Call Management Records Logged By Call Type

Each normal call between two Cisco IP Phones logs exactly two CMR records. Each call CMR record contains all fields identified above. When supplementary services are involved in a call, more than one record may be written. This section describes when diagnostic records are written for different call types in the system.

Normal Calls

Normal calls log exactly two CMR records per call, one for each phone involved in the call. Currently only Cisco IP Phones and MGCP gateways are capable of responding to the diagnostic information request. All fields will contain valid information.

Abandoned Calls

If the call was abandoned (such as when a phone is taken off-hook and placed back on hook), all fields related to streaming data will be blank (zero). This is because no streaming connection was established, and therefore no data was transferred. All records with blank fields will not be logged if the `CdrLogCallsWithZeroDurationFlag` is disabled.

Forwarded Calls

The call records for forwarded calls will be the same as those for normal calls.

Calls With Busy or Bad Destinations

In the normal case, only records that represent calls that were actually connected will be logged. In order to log calls with bad destinations, you must enable `CdrLogCallsWithZeroDurationFlag`. If it is enabled, then all calls will be logged including the case where the user goes off-hook and then on-hook again.

If the calls are logged, they will be logged as normal calls with all relevant fields containing data. There will only be one record per call since the calls were never connected to a destination. The record will be for the originator of the call.

Codec Types (Compression / Payload types)

[Table D-3](#) provides values and descriptions for codec types.

Table D-3 *Codec Description*

Codec	Description
1	NonStandard
2	G711A-law 64k
3	G711A-law 56k
4	G711 μ -law 64k

Table D-3 *Codec Description (continued)*

Codec	Description
5	G711 μ -law 56k
6	G722 64k
7	G722 56k
8	G722 48k
9	G7231
10	G728
11	G729
12	G729AnnexA
13	Is11172AudioCap
14	Is1381AudioCap
15	G729AnnexB
32	Data 64k
33	Data 56k
80	GSM
81	ActiveVoice
82	G726_32K
83	G726_24K
84	G726_16K

Cause Codes

[Table D-4](#) provides a list of cause codes that may appear in the Cause fields.

Table D-4 *Cause Code Descriptions*

Cause Code	Description
0	No error
1	unallocated (unassigned) number

Table D-4 Cause Code Descriptions (continued)

Cause Code	Description
2	No route to specified transit network (national use)
3	No route to destination
4	Send special information tone
5	Misdialed trunk prefix (national use)
6	Channel unacceptable
7	Call awarded and being delivered in an established channel
8	Preemption
9	Preemption—circuit reserved for reuse
16	Normal call clearing
17	User busy
18	No user responding
19	No answer from user (user alerted)
20	Subscriber absent
21	Call rejected
22	Number changed
26	Non-selected user clearing
27	Destination out of order
28	Invalid number format (address incomplete)
29	Facility rejected
30	Response to STATUS ENQUIRY
31	Normal, unspecified
34	No circuit/channel available
38	Network out of order
39	Permanent frame mode connection out of service

Table D-4 Cause Code Descriptions (continued)

Cause Code	Description
40	Permanent frame mode connection operational
41	Temporary failure
42	Switching equipment congestion
43	Access information discarded
44	Requested circuit/channel not available
46	Precedence call blocked
47	Resource unavailable, unspecified
49	Quality of Service not available
50	Requested facility not subscribed
53	Service operation violated
54	Incoming calls barred
55	Incoming calls barred within CUG (Closed User Group)
57	Bearer capability not authorized
58	Bearer capability not presently available
62	Inconsistency in designated outgoing access information and subscriber class
63	Service or option not available, unspecified
65	Bearer capability not implemented
66	Channel type not implemented
69	Requested facility not implemented
70	Only restricted digital information bearer capability is available (national use)

Table D-4 Cause Code Descriptions (continued)

Cause Code	Description
79	Service or option not implemented, unspecified
81	Invalid call reference value
82	Identified channel does not exist
83	A suspended call exists, but this call identity does not
84	Call identity in use
85	No call suspended
86	Call having the requested call identity has been cleared
87	user not member of CUG (Closed User Group)
88	Incompatible destination
90	Destination number missing and DC not subscribed
91	Invalid transit network selection (national use)
95	Invalid message, unspecified
96	Mandatory information element is missing
97	Message type non-existent or not implemented
98	Message is not compatible with the call state, or the message type is non-existent or not implemented
99	An information element or parameter does not exist or is not implemented
100	Invalid information element contents
101	The message is not compatible with the call state

Table D-4 Cause Code Descriptions (continued)

Cause Code	Description
102	The call was terminated when a timer expired and a recovery routine was executed to recover from the error
103	Parameter non-existent or not implemented - passed on (national use)
110	Message with unrecognized parameter discarded
111	Protocol error, unspecified
126	Call split. This is a Cisco-specific code. It is used when a call is terminated during a transfer operation because it was split off and terminated (was not part of the final transferred call). This can help determine which calls were terminated as part of a transfer operation.
127	Interworking, unspecified

Alarms

An alarm is issued when CDR or Diagnostic data is enabled, and the system is unable to write the data into the database.

Unable to write CDR data. (Alarm # 1711 - Major Alarm)

The system attempted to open the database, and was unsuccessful. Probable causes include:

- Cisco CallManager does not have sufficient privileges to open the file for writing to the database. Make sure Cisco CallManager has privileges that will permit write operations.
- The path is not set up, or the database server is down.



A

- A-law** ITU-T companding standard used in the conversion between analog and digital signals in PCM systems. A-law is used primarily in European telephone networks and is similar to the North American μ -law standard.
- ACF** Admission Confirm.
- ANI** Answer Number Indication. The calling number (number of the calling party).
- ARQ** Admission Request.

B

- B-Channel** Bearer channel. In ISDN, a full-duplex, 64-kbps channel used to send user data.

C

- Calling Search Space** The Calling Search Space defines what directory numbers and route patterns a given device can call. It is a grouping of partitions to look through when making a call. For example, assume there are several Partitions in a Calling Search Space named Executive. If a Cisco IP Phone number is in the Executive Calling Search Space, then when initiating a call, it looks for the example NYInternationalCall, NYLongDistance, NYLocalCall, and NY911 Partitions available to search through. A Cisco IP Phone number that has a Guest Calling Search Space, for example, might only be allowed to search through NYLocalCall and NY911 Partitions, so that if the user tries to dial an international number, it will not find a match and the call cannot be routed.

CCAPI	Call Control API. Used by Cisco IOS to handle VoIP call processing.
CCO	Cisco Connection Online (http://www.cisco.com). Provides the latest information on Cisco products, technical support information, and technical documentation.
CDR	Call Detail Record. Information about call origination, destination, and duration, used to create billing records.
Cisco IOS	Cisco system software that provides common functionality, scalability, and security for all products under the CiscoFusion architecture. Cisco IOS allows centralized, integrated, and automated installation and management of internetworks, while ensuring support for a wide variety of protocols, media, services, and platforms.
Cluster	Cisco CallManager cluster. A logical grouping of several Cisco CallManager servers.
CMR	Call Management Records, also known as Diagnostic CDRs. Records that contain the count of bytes sent, packets sent, jitter, latency, dropped packets, and so on.
codec	Coder-Decoder. A DSP software algorithm used to compress/decompress speech or audio signals.
.cnf	Configuration file used by devices.

D

D-Channel	Data channel. Full-duplex, 16-kbps (BRI) or 64-kbps (PRI) ISDN channel. Used for signaling and control.
DCF	Disengage Confirm.
DHCP	Dynamic Host Configuration Protocol. Provides a mechanism for allocating IP addresses dynamically so that addresses can be reused when hosts no longer need them.

DN	Directory Number. This is the phone number of an end device. It can be a number assigned to a Cisco IP Phone, a Cisco IP SoftPhone, fax machine, or analog phone attached to a gateway. Examples include 1000 and 24231.
DNIS	Dialed Number Identification Service.
DNS	Domain Name System. System used in the Internet for translating names of network nodes into addresses.
DRQ	Disengage Request.
DTMF	Dual tone multifrequency. Use of two simultaneous voice-band tones for dialing (such as touch tone).

F

Flow	Stream of data traveling between two endpoints across a network (for example, from one LAN station to another). Multiple flows can be transmitted on a single circuit.
Full duplex	Capability for simultaneous data transmission from both a sending station and a receiving station.

G

G.711	Describes the 64-kbps PCM voice coding technique. In G.711, encoded voice is already in the correct format for digital voice delivery in the PSTN or through PBXs. Described in the ITU-T standard in its G-series recommendations.
G.729	Describes CELP compression where voice is coded into 8-kbps streams. There are two variations of this standard (G.729 and G.729 Annex A) that differ mainly in computational complexity; both provide speech quality similar to 32-kbps ADPCM. Described in the ITU-T standard in its G-series recommendations.
Gateway	A gateway allows H.323 terminals to communicate with non-H.323 terminals by converting protocols. A gateway is the point at which a circuit-switched call is encoded and repackaged into IP packets.

H

- H.225** An ITU standard that governs H.225 session establishment and packetization. H.225 actually describes several different protocols: RAS, use of Q.931, and use of RTP.
- H.245** An ITU standard that governs H.245 endpoint control.
- H.323** Extension of ITU-T standard H.320 that enables videoconferencing over LANs and other packet-switched networks, as well as video over the Internet.
- Half Duplex** Capability for data transmission in only one direction at a time between a sending station and a receiving station. BSC is an example of a half-duplex protocol.
- Hookflash** Short on-hook period usually generated by a telephone-like device during a call to indicate that the telephone is attempting to perform a dial-tone recall from a PBX. Hookflash is often used to perform call transfer.

I

- ICCP** Intra-Cluster Control Protocol.
- ISDN** Integrated Services Digital Network. Communication protocol, offered by telephone companies, that permits telephone networks to carry data, voice, and other source traffic.

J

- Jitter** The variation in the arrival times of voice packets.

K

- kbps** Kilobits per second.
- Keepalive Message** A notice sent by one network device to another that the virtual circuit (VC) between the two is still active.

L

- Line Side** A circuit from a subscriber's telephone to an end office (EO) or central office (CO).
- Logical Connection** A call that makes a physical connection to a network, establishes communication with another device on the network, and remains connected until the user initiates a disconnect request.
- Loop Back** A method of performing transmission tests on a circuit that does not require the assistance of personnel at the far end.

M

- MAC Address** Media Access Control. A standardized data link layer address that is required for every port or device that connects to a LAN. Other devices in the network use these addresses to locate specific ports in the network and to create and update routing tables and data structures. Also called a physical address.
- MGCP** Media Gateway Control Protocol. A protocol for Cisco CallManager to control VoIP gateways (MGCP endpoints).
- MTP** Media Termination Point. A virtual device that allows transfer, forward, conference, and hold features on any G.711 μ -law call between an IP phone and any H.323 gateway, gatekeeper, or client. A call using MTP will automatically convert A-law to μ -law (and vice versa), if required.

N

- NANP** North American Numbering Plan.
- Network Address** Network layer address referring to a logical, rather than a physical, network device. Also called a protocol address.

P

- Partition** A Partition is a logical grouping of Directory Numbers and Route Patterns with similar reachability characteristics. For simplicity, these are usually named for their characteristic, such as NYLongDistance or NY911. When a DN or Route Pattern is placed into a certain partition, this creates a rule for who can call that device or Route List.
- PBX** Private Branch Exchange. Digital or analog telephone switchboard located on the subscriber premises and used to connect private and public telephone networks.
- PRI** Primary Rate Interface. Primary rate access consists of a single 64-Kbps D channel plus 23 (T1) or 30 (E1) B channels for voice or data.
- PSTN** Public Switched Telephone Network. General term referring to the variety of telephone networks and services in place worldwide.

Q

- Q.931** An ITU standard that describes ISDN signaling. The H.225.0 standard uses a variant of Q.931 to establish and disconnect H.323 sessions.

R

- RAS** Registration, Admission, and Status protocol. Protocol used in the H.323 protocol suite for discovering and interacting with a gatekeeper.

Route Filter	A route filter can be used not only to restrict dialing, but also to identify a subset of a wildcard pattern (when using the @ wildcard in the North American Dialing Plan). For example, it could be used to block the dialing of 900 area codes. It can also be used in conjunction with Partitions and Calling Search Spaces to set up complex rules. For example, assume you have three user groups established, Executive, Staff, and Guest. A Route Filter can allow the Executive user group to dial international numbers; while the Staff user group can only dial local numbers or long distance calls; and the Guest user group can only dial local numbers, 911, and 800 numbers.
Route Group	A Route Group is a list of one or more gateways or ports on gateways that are seen as equal access. It is analogous to a trunk group in traditional PBX terminology. For instance, you may have two PRI circuits to the same carrier that can be used arbitrarily. A gateway (or a particular port on a gateway) can only be added to one Route Group.
Route List	Formerly called Route Point, the Route List allows Cisco CallManager to hunt through a list of Route Groups in a configured order of preference. Multiple Route Lists can point to the same Route Groups.
Route Pattern	A specific number or, more commonly, a range of dialed numbers that will be used to route calls to a device (such as a Cisco Access DT-24+ Gateway or a voice-capable router) or indirectly via a Route List. For example, 1XXX signifies 1000 through 1999. The X in 1XXX signifies a single digit, a wildcard. There are other wildcards (such as @, ., !). A Route Pattern does not have to be unique within a partition as long as the Route Filter is different.
RRJ	Registration Reject.
RTP	Real-Time Transport Protocol. One of the IPv6 protocols. RTP is designed to provide end-to-end network transport functions for applications transmitting real-time data, such as audio, video, or simulation data, over Multicast or Unicast network services. RTP provides services such as payload type identification, sequence numbering, time stamping, and delivery monitoring to real-time applications.

S

Silence Suppression (Voice Activation Detection) Silence Suppression allows a Cisco IP Phone to detect the absence of audio and does not transmit packets over the network. The sound quality may be slightly degraded but the connection may also use less bandwidth. Silence Suppression is disabled by default.

SNMP Simple Network Management Protocol. Network management protocol used almost exclusively in TCP/IP networks. SNMP provides a means to monitor and control network devices, and to manage configurations, statistics collection, performance, and security.

SQL Structured Query Language. International standard language for defining and accessing relational databases.

T

T1/CAS T1 is a digital WAN carrier facility, transmitting DS-1-formatted data at 1.544 Mbps through the telephone-switching network, using AMI or B8ZS coding. CAS is a Channel Associated Signaling interface.

T1/PRI T1 is a digital WAN carrier facility, transmitting DS-1-formatted data at 1.544 Mbps through the telephone-switching network, using AMI or B8ZS coding. PRI is Primary Rate Interface. Primary rate access consists of a single 64-Kbps D channel plus 23 (T1) or 30 (E1) B channels for voice or data.

TCP Transmission Control Protocol. Connection-oriented transport layer protocol that provides reliable full-duplex data transmission. TCP is part of the TCP/IP protocol stack.

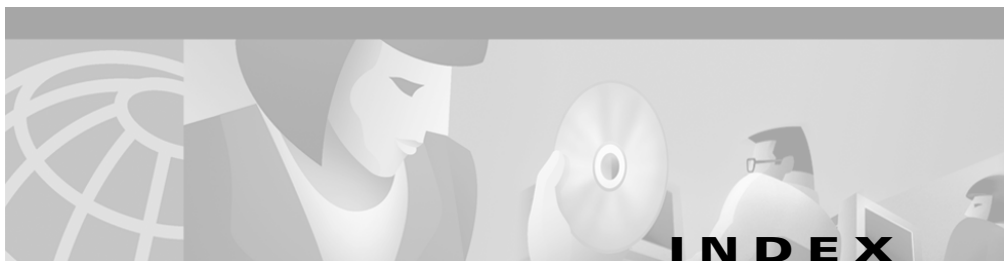
- TFTP** Trivial File Transfer Protocol. Simplified version of FTP that allows files to be transferred from one computer to another over a network.
- Translation Pattern** Used to translate called (DNIS) and calling (ANI) numbers before routing the call. For example, a call may come in to a set of numbers 919 392-3XXX that need to be translated to a set of Cisco IP Phones that are in the range of 2XXX. Cisco CallManager has a Translation Pattern set up for 919 392-3XXX. This pattern translates the leading 919 392-3 simply to 2 while leaving the remaining digits intact. Then the call is routed to the appropriate Cisco IP Phone. Translation Patterns are used only for true translations and should not be used for simple digit stripping and prefixing.

U

- μ-law** Companding technique commonly used in North America. μ-law is standardized as a 64-kbps codec in ITU-T G.711
- UDP** User Datagram Protocol. Connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols. UDP is defined in RFC 768.

V

- Voice Activation Detection (Silence Suppression)** Voice Activation Detection allows a Cisco IP Phone to detect the absence of audio and does not transmit packets over the network. The sound quality may be slightly degraded but the connection may also use less bandwidth. VAD/Silence Suppression is disabled by default.
- VoIP** Voice over IP.
- VLAN** Virtual LAN. Group of devices on one or more LANs that are configured (using management software) so that they can communicate as if they were attached to the same wire, when in fact they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, they are extremely flexible.



C

case study

Cisco IP Phone-to-Cisco IOS Gateway calls [B-1](#)

inter-cluster Cisco IP Phone-to-Cisco IP Phone calls [C-1](#)

intra-cluster phone calls [A-1](#)

CDR

description [D-1](#)

CDRs

disabling [2-9](#)

enabling [2-9](#)

using [2-9](#)

checking

RTP header compression [3-8](#)

Cisco CallManager Administration

system information [2-1](#)

using [2-1](#)

Cisco CallManager problems

diagnosing [3-1](#)

Cisco IP Phone

diagnosing audio problems [3-4](#)

testing speakerphone and volume levels [3-6](#)

Cisco IP Phones

case study [C-1](#)

diagnosing crackling sounds [3-5](#)

CMR

description [D-1](#)

CMRs

description [2-9](#)

using [2-10](#)

D

diagnosing

audio problems from Cisco IP Phone [3-4](#)

Cisco CallManager feature issues [3-11](#)

Cisco CallManager problems [3-1](#)

codec and region mismatches [3-11](#)

conference bridge problems [3-12](#)

crackling sounds [3-5](#)

dial plan problems [3-18](#)

dropped calls [3-10](#)

echo [3-5](#)

gatekeeper problems [3-31](#)

gateway registration problems [3-24](#)

gateway reorder tone problems [3-23](#)

location and bandwidth problems [3-11](#)

lost or distorted audio problems [3-2](#)

MTP resource problems [3-16](#)

one-way or no audio [3-7](#)
 partition problems [3-20](#)
 phone resets [3-9](#)
 security problems [3-23](#)
 slow server response [3-23](#)
 transcoding problems [3-14](#)
 voice quality issues [3-2](#)

H

http

[//www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm](http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/index.htm) [D-3](#)

[//www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/rel_note/index.htm](http://www.cisco.com/univercd/cc/td/doc/product/voice/c_callmg/3_1/rel_note/index.htm) [D-4](#)

L

loads

checking [3-5](#)

logs

echo log [3-6](#)

M

Microsoft Event Viewer

opening [2-2, 2-3](#)

using [2-2, 2-3](#)

Microsoft Performance

opening [2-2](#)

using [2-2](#)

Microsoft performance

collecting system statistics [2-1](#)

using [2-1](#)

monitoring

tools and utilities [2-1](#)

MTP

checking [3-7](#)

S

SDI Trace

using [2-4](#)

SDL Trace

enabling [2-6](#)

using [2-5](#)

Sniffer Trace

applications [2-9](#)

description [2-8](#)

T

testing

Cisco CallManager configuration [3-7](#)

gateways [3-6](#)

speakerphone and volume [3-6](#)

topology

checklist [1-2](#)

- understanding [1-1](#)
- traces
 - configuring [2-5](#)
 - Sniffer [2-8](#)
- troubleshooting
 - admission rejects [3-31](#)
 - ARJs [3-31](#)
 - Cisco IP Phone-to-Cisco IOS Gateway calls [B-1](#)
 - intra-cluster phone calls [A-1](#)
 - overview [1-1](#)
 - registration rejects [3-32](#)
 - RRJs [3-32](#)
 - tools and utilities [2-1](#)

