About this Guide

Overview vii
Who Should Use This Guide vii
Document Objectives vii
Document Organization viii
Document Conventions viii
Related Documentation ix

Obtaining Documentation ix
World Wide Web ix
Documentation CD-ROM x
Ordering Documentation x
Documentation Feedback x

Obtaining Technical Assistance x
Cisco.com xi
Technical Assistance Center xi
Cisco TAC Web Site xi
Cisco TAC Escalation Center xii

Product Overview

What is SIP? 1-1
Components of SIP 1-2
SIP Clients 1-3
SIP Servers 1-3

What is the Cisco SIP Proxy Server (CSPS)? 1-4
CSPS Components 1-4
CSPS Features 1-5
Server Modes 1-6
Protocol Support 1-6
Spiralled and Looped Request Detection 1-7
Address Translation, Routing, and IP Resolution 1-7
Authentication, Authorization 1-11
Accounting 1-12
Server Farm Support 1-12
Registrar for Multiple Domain 1-13
IP Security 1-16
Access and Error Logging 1-16

Requirements 1-16
Solaris Platform Requirements 1-16
Linux Platform Requirements 1-17

Related Documentation 1-17
Reference Links 1-17

CHAPTER 2
Licensing System
License Keys 2-1
License Key File 2-1
Information in License Key 2-2
License Key Characteristics 2-2
Activating the Licensing System 2-2
Adding Licenses 2-4
Editing Licenses 2-5
Deleting and Reloading Licenses 2-6
Searching and Filtering Licenses 2-6

CHAPTER 3
Provisioning System GUI
Activating the Provisioning System 3-1
Configuring a Farm 3-3
Configuring Advanced Features for a Farm 3-5
Farming 3-6
Virtual Proxy Host 3-9
RAS 3-9
RPM S 3-11
Debug and Logs 3-11
Authentication 3-13
Call Forward 3-15
Number Expansion 3-16
ENUM 3-17
Server Directives 3-18
SIP Server Core 3-20
MySQL 3-23
GKTM P 3-24
Accounting 3-24
Access Control 3-26
Subscribers and Static Registries 3-27
Static Routes 3-30
Seeding 3-32
  Multi-member Farm 3-34
GUI Password 3-34
Searching and Filtering Data 3-34

CHAPTER 4
Configuring the Cisco SIP Proxy Server (CSPS)
Defining the CSPS Configuration File 4-1
  Configuring the Server Global Directives 4-2
  Configuring the Host-Specific Directives 4-3
  Configuring the Cisco SIP Module Core Directives 4-5
  Configuring Cisco SIP Module Standard Directives 4-9
    Configuring MySQL Database Subscriber Table Interface 4-10
    Configuring the GKTMP Interface 4-11
    Configuring Accounting Services 4-11
    Configuring SIP Access Control 4-12
    Configuring Authentication and Authorization 4-14
    Configuring Pre-Authentication Query 4-15
    Configuring Call Forwarding 4-16
    Configuring Number Expansion 4-17
    Configuring E.164 to Request-URI Address Translation 4-18
    Configuring Registry Services 4-18
    Configuring Next Hop Routing 4-21
    Configuring H.323 RAS module 4-24
  Configuring the SIP Proxy Server in a Farm 4-25
    Configuring the Master Member for the Farm 4-26
    Configuring the Slave Member for the Farm 4-27
  Configuring IPSec 4-27
    Configuring IPSec for the Cisco SIP Proxy on Solaris Platforms 4-29

CHAPTER 5
Maintaining the Cisco SIP Proxy Server (CSPS)
Starting and Stopping the CSPS 5-1
  CSPS with All Components Installed (including Provisioning) 5-2
    Starting the CSPS 5-2
    Stopping the CSPS 5-3
    Restarting the CSPS 5-4
    Gracefully Restarting the CSPS 5-5
  CSPS Installed without Provisioning 5-7
    Starting the CSPS 5-7
System Administration Tools

Administration Tool for Routing and Registry Databases 6-1
   Tool Activation for Routing and Registry Databases 6-2
      Examples 6-2
   Tool Menus for Routing and Registry Databases 6-3
      Menu Operation for Routing and Registry Databases 6-4
   Tool Functionalities and Operations for Routing and Registry Databases 6-4
      Selecting a database 6-5
      Adding or Deleting Data 6-5
      Searching the Database 6-7
      Displaying Data in the Database 6-7
      Importing Configuration File to a Database 6-8
      Exporting Database Entries to a Configuration File 6-8
      Displaying Memory Information 6-9
      Sample Error Messages 6-9

Administration Tool for MySQL Databases 6-9
   Tool Activation 6-10
   Tool Menus 6-10
      Menu Operation 6-11
   Tutorial 6-11
      Adding a Subscriber with a Password 6-11
      Listing Subscriber Information 6-12
      Adding a Subscriber with Call Forwarding Features 6-13
      Listing Subscribers 6-17
Renaming an Existing Subscriber Record 6-18
Removing a Subscriber from a MySQL Database 6-20

Tool Functionalities and Operations 6-20
Adding Subscribers to the Databases 6-21
Removing Subscribers from the Databases 6-21
Modifying Subscribers in a Database 6-22
Renaming an Existing Subscriber Record in a Database 6-22
Displaying Subscribers of a Database 6-22
Exiting the Tool 6-23

Sample Error Messages 6-23

CHAPTER 7

CIAGent

CIAGent Overview 7-1
MIBs and Subagents 7-2
Installation 7-2
  Customized Installation 7-3
  Manual installation 7-4
Stopping and Restarting CIAGent Manually 7-5
Uninstalling CIAGent 7-5
Modifying System Information 7-6
Creating a User for the System Administrator 7-6
Configuring SNMP v2c and Trap Target Addresses 7-6
Customizing Configurations 7-7
  Configuring Critical Application Subagent (critagt) 7-8
  Configuring the Script Subagent (smagt) 7-10
  Configuring the Log File Subagent (logagt) 7-11
  Configuring the Event MIB Subagent 7-11
Configuring Trap Sink for CIAGent Traps 7-14
Checking Memory Size 7-14
Checking Disk Space Information 7-14
Checking Link Up/down Status 7-16
Checking Installed and Active Components in the System 7-17
Using the CSPS with CIAGent 7-17
  Starting, Stopping, and Gracefully Restarting the CSPS 7-17
    Starting the CSPS 7-17
    Stopping CSPS 7-17
    Checking Status 7-18
    Restarting the CSPS Gracefully 7-18
Cisco SIP Proxy Server (CSPS) Compliance Information

SIP Compliance with RFC 2543 A-1
SIP Functions A-1
SIP Methods A-1
SIP Responses A-2
1xxResponse—Information Responses A-3
2xxResponse—Successful Responses A-3
3xxResponse—Redirection Responses A-4
4xxResponse—Request Failure Responses A-4
5xxResponse—Server Failure Responses A-6
6xxResponse—Global Responses A-6
SIP Header Fields A-6
Transport Layer Protocols A-8
SIP Security A-8
Authentication A-8
SIP DNS Records Usage A-8

Cisco SIP Proxy Server (CSPS) Call Flows

SIP Messages and Methods B-1
Requests B-1
Responses B-2
The Registration Process B-2
The Invitation Process B-2

Call Flow Scenarios for Successful Calls B-2
SIP Gateway-to-SIP Gateway—Call via SIP Redirect Server B-3
SIP Gateway-to-SIP Gateway—Call via SIP Proxy Server B-6
SIP IP Phone-to-SIP IP Phone Call Forward Unconditionally B-14
SIP IP Phone-to-SIP IP Phone Call Forward on Busy B-19
SIP IP Phone-to-SIP IP Phone Call Forward No Answer B-23
SIP IP Phone-to-SIP IP Phone Call Forward Unavailable B-28

Call Flow Scenarios for Failed Calls B-33
SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User is Busy B-33
SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User Does Not Answer B-35
SIP Gateway-to-SIP Gateway via SIP Redirect Server—Client, Server, or Global Error B-38
SIP Gateway-to-SIP Gateway via SIP Proxy Server—Called User is Busy B-39
SIP Gateway-to-SIP Gateway via SIP Proxy Server—Client or Server Error B-41
SIP Gateway-to-SIP Gateway via SIP Proxy Server—Global Error B-43
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled B-46
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route enabled B-58
SIP Phone to SIP/H.323 Gateway—Call via SIP Redirect Server B-71
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled (Call failed with a 503 Service Unavailable response) B-82

**Appendix C**

Accounting Services Record Attributes

- Standard Attributes C-1
- Vendor-specific Attributes C-2
- Request-specific Attributes C-3
  - Start Record C-3
  - Stop Record C-3

**Glossary**

**Index**
About this Guide

This preface describes the objectives, audience, organization, and conventions of the Cisco SIP Proxy Server Administrator Guide.

This preface contains the following information:
- Overview, pagevii
- Who Should Use This Guide, pagevii
- Document Objectives, pagevii
- Document Organization, pageviii
- Document Conventions, pageviii
- Related Documentation, pageix
- Obtaining Documentation, pageix

Overview

The Cisco Session Initiation Protocol (SIP) Proxy Server Administrator Guide provides information about how to configure a CiscoSIP Proxy Server (CSPS). This administrator guide also includes reference information such as CSPS call flows and compliance information.

Who Should Use This Guide

Network engineers, system administrators, or telecommunication engineers should use this guide to learn the steps required to properly configure a CSPS on the network.

The tasks described are considered to be administration-level tasks and require a working knowledge of Unix, including configuring user shells. Configuring the CSPS also requires an understanding of IP networking and telephony concepts.

Document Objectives

The Cisco SIP Proxy Server Administrator Guide provides necessary information to set up the CSPS to be operational in a Voice-over-IP (VoIP) network.
It is not the intent of this administrator guide to provide information on how to implement a SIP VoIP network. For information on implementing a SIP VoIP network, refer to the documents listed in the “Related Documentation” section on pageix.

**Document Organization**

This administrator guide is divided into the following chapters and appendixes:

- **Chapter 1, “Product Overview”** provides an overview of SIP, its components, and how SIP works; it describes and lists the features and prerequisites of the CSPS.
- **Chapter 2, “Licensing System”** provides information on the License Keys and Licensing GUI functionalities.
- **Chapter 3, “Provisioning System GUI”** describes how to use the Provisioning GUI to configure the CSPS.
- **Chapter 4, “Configuring the Cisco SIP Proxy Server (CSPS)”** describes how to manually edit the text based configuration files to configure the CSPS, assuming the Provisioning System GUI is not used. It also describes how to configure multiple servers as a farm, and IPSec on the servers.
- **Chapter 5, “Maintaining the Cisco SIP Proxy Server (CSPS)”** provides information on starting and stopping the CSPS, working with log files, and backing up and restoring configuration data.
- **Chapter 6, “System Administration Tools”** provides an overview and operation instructions of the system administration tools for routing and registry databases, and MySQL databases of the CSPS.
- **Chapter 7, “CIAgent”** provides information on CIAgent installation and its SNMP functionalities.
- **Appendix A, “Cisco SIP Proxy Server (CSPS) Compliance Information”** describes how the CSPS complies with the IETF definition of SIP as described in RFC 2543, and gives an overview of SIP concepts and services.
- **Appendix B, “Cisco SIP Proxy Server (CSPS) Call Flows”** illustrates how SIP messages are exchanged during various call scenarios.
- **Appendix C, “Accounting Services Record Attributes”** describes the standard and request-specific attributes that are logged in the Call Detail Records generated by the Accounting Services module of the CSPS.
- The glossary provides definitions of the acronyms and terms used in this document.

**Document Conventions**

Table 1 describes the conventions that might be used in this document.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Commands and keywords.</td>
</tr>
<tr>
<td><strong>italic</strong></td>
<td>Command input that is supplied by you.</td>
</tr>
<tr>
<td>[   ]</td>
<td>Keywords or arguments that appear within square brackets are optional.</td>
</tr>
<tr>
<td>{ x</td>
<td>x</td>
</tr>
</tbody>
</table>
About this Guide

The following sections explain how to obtain documentation from Cisco Systems.

Related Documentation

The following is a list of related Cisco SIP VoIP publications.

- Session Initiation Protocol Gateway Call Flows
- Session Initiation for VoIP on Cisco Access Platforms
- Cisco SIP IP Phone 7960 Administrator Guide
- Getting Started Cisco 7960 IP Phone
- Enhancements for the Session Initiation Protocol for VoIP on Cisco Access Platforms
- Cisco IOS Multiservice Applications Command Reference
- Cisco IOS Multiservice Applications Configuration Guide

The following is a list of Cisco VoIP publications that provide information about implementing a VoIP network:

- Service Provider Features for Voice over IP (introduced in Cisco IOS Release 12.0(3)T)
- Cisco IOS IP and IP Routing Configuration Guide
- Cisco IOS Release 12.1 Multiservice Applications Configuration Guide
- Voice over IP for the Cisco 2600 and Cisco 3600 Series Routers
- Voice over IP for the Cisco AS5300 Documents

Obtaining Documentation

The following sections explain how to obtain documentation from Cisco Systems.

World Wide Web

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

http://www.cisco.com

Translated documentation is available at the following URL:

Documentation CD-ROM

Cisco documentation and additional literature are available in a Cisco Documentation CD-ROM package, which is shipped with your product. The CD-ROM package is available as a single unit or through 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 408526-7208 or, elsewhere in North America, by calling 800 553-NETS (6387).

Documentation Feedback

If you are reading Cisco product documentation on Cisco.com, you can submit technical comments electronically. Click Leave Feedback at the bottom of the Cisco Documentation home page. After you complete the form, print it out and fax it to Cisco at 408527-0730.

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:

Cisco Systems
Attn: Document Resource Connection
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 by using the Cisco Technical Assistance Center (TAC) Web Site. Cisco.com registered users have complete access to the technical support resources on the Cisco TAC Web Site.
Cisco.com

Cisco.com is the foundation of a suite of interactive, networked services that provides immediate, open access to Cisco information, networking solutions, services, programs, and resources at any time, from anywhere in the world.

Cisco.com is a highly integrated Internet application and a powerful, easy-to-use tool that provides a broad range of features and services to help you to

- Streamline business processes and improve productivity
- Resolve technical issues with online support
- Download and test software packages
- Order Cisco learning materials and merchandise
- Register for online skill assessment, training, and certification programs

You can self-register on Cisco.com to obtain customized information and service. To access Cisco.com, go to the following URL:

http://www.cisco.com

Technical Assistance Center

The Cisco TAC is available to all customers who need technical assistance with a Cisco product, technology, or solution. Two types of support are available through the Cisco TAC: the Cisco TAC Web Site and the Cisco TAC Escalation Center.

Inquiries to Cisco TAC are categorized according to the urgency of the issue:

- Priority level 4 (P4)—You need information or assistance concerning Cisco product capabilities, product installation, or basic product configuration.
- Priority level 3 (P3)—Your network performance is degraded. Network functionality is noticeably impaired, but most business operations continue.
- Priority level 2 (P2)—Your production network is severely degraded, affecting significant aspects of business operations. No workaround is available.
- Priority level 1 (P1)—Your production network is down, and a critical impact to business operations will occur if service is not restored quickly. No workaround is available.

Which Cisco TAC resource you choose is based on the priority of the problem and the conditions of service contracts, when applicable.

Cisco TAC Web Site

The Cisco TAC Web Site allows you to resolve P3 and P4 issues yourself, saving both cost and time. The site provides around-the-clock access to online tools, knowledge bases, and software. To access the Cisco TAC Web Site, go to the following URL:

http://www.cisco.com/tac

All customers, partners, and resellers who have a valid Cisco services contract have complete access to the technical support resources on the Cisco TAC Web Site. The Cisco TAC Web Site requires a Cisco.com login ID and password. If you have a valid service contract but do not have a login ID or password, go to the following URL to register:

http://www.cisco.com/register/
If you cannot resolve your technical issues by using the Cisco TAC Web Site, and you are a Cisco.com registered user, you can open a case online by using the TAC Case Open tool at the following URL:
http://www.cisco.com/tac/caseopen

If you have Internet access, it is recommended that you open P3 and P4 cases through the Cisco TAC Web Site.

**Cisco TAC Escalation Center**

The Cisco TAC Escalation Center addresses issues that are classified as priority level 1 or priority level2; these classifications are assigned when severe network degradation significantly impacts business operations. When you contact the TAC Escalation Center with a P1 or P2 problem, a Cisco TAC engineer will automatically open a case.

To obtain a directory of toll-free Cisco TAC telephone numbers for your country, go to the following URL:

Before calling, please check with your network operations center to determine the level of Cisco support services to which your company is entitled; for example, SMARTnet, SMARTnet Onsite, or Network Supported Accounts (NSA). In addition, please have available your service agreement number and your product serial number.
Product Overview

This chapter provides an overview of the Cisco SIP Proxy Server (CSPS), its features, and prerequisites. This chapter contains the following sections:

- What is SIP?, page 1-1
- What is the Cisco SIP Proxy Server (CSPS)?, page 1-4
- CSPS Components, page 1-4
- CSPS Features, page 1-5
- Requirements, page 1-16
- Reference Links, page 1-17

Note

For installation updates of Release 2.0, refer to the README file.

What is SIP?

Session Initiation Protocol (SIP) is the Internet Engineering Task Force’s (IETF’s) standard for multimedia conferencing over IP. SIP is an ASCII-based, application-layer control protocol (defined in RFC2543) that can be used to establish, maintain, and terminate calls between two or more end points.

Like other VoIP protocols, SIP is designed to address the functions of signaling and session management within a packet telephony network. **Signaling** allows call information to be carried across network boundaries. **Session management** provides the ability to control the attributes of an end-to-end call.

SIP provides the following capabilities:

- Determine the location of the target end point—Supports address resolution, name mapping, and call redirection.
- Determine the media capabilities of the target end point—By using Session Description Protocol (SDP), SIP determines the highest level of common services between the end points. Conferences are established using only the media capabilities that can be supported by all end points.
- Determine the availability of the target end point—If a call cannot be completed because the target end point is unavailable, SIP determines whether the called party is already on the phone or did not answer in the allotted number of rings. It then returns a message indicating why the target end point is unavailable.
What is SIP?

• Establish a session between the originating and target end point—If the call can be completed, SIP establishes a session between the end points. SIP also supports mid-call changes, such as the addition of another end point to the conference or the changing of a media characteristic or codec.

• Handle the transfer and termination of calls—SIP supports the transfer of calls from one end point to another. During a call transfer, SIP simply establishes a session between the transferee and a new end point (specified by the transferring party) and terminates the session between the transferee and the transferring party. At the end of a call, SIP terminates the sessions between all parties.

Conferences can consist of two or more users and can be established using multicast or multiple unicast sessions.

Note

The term conference means an established session (or call) between two or more end points. In this document, the terms conference and call are used interchangeably.

Components of SIP

SIP is a peer-to-peer protocol. The peers in a session are called User Agents (UAs). A user agent can function in one of the following roles:

• User agent client (UAC)—A client application that initiates the SIP request.

• User agent server (UAS)—A server application that contacts the user when a SIP request is received and that returns a response on behalf of the user.

Typically, a SIP end point is capable of functioning as both a UAC and a UAS, but functions only as one or the other per transaction. Whether the endpoint functions as a UAC or a UAS depends on the UA that initiated the request.

From an architecture standpoint, the physical components of a SIP network can be grouped into two categories: clients and servers. Figure 1-1 illustrates the architecture of a SIP network.

Note

In addition, the SIP servers can interact with other application services, such as Lightweight Directory Access Protocol (LDAP) servers, location servers, a database application, or an extensible markup language (XML) application. These application services provide back-end services such as directory, authentication, and billing services.
SIP Clients

SIP clients include the following:
- Phones—Can act as either a UAS or UAC. Softphones (PCs that have phone capabilities installed) and Cisco SIP IP phones can initiate SIP requests and respond to requests.
- Gateways—Provide call control. Gateways provide many functionalities. The most common one is a translation function between SIP conferencing endpoints and other terminal types. This function includes translation between transmission formats and between communications procedures. In addition, the gateway also translates between audio and video codecs and performs call setup and clearing on both the LAN side and the switched-circuit network side.

SIP Servers

SIP servers include the following:
- Proxy server—The proxy server is an intermediate device that receives SIP requests from a client and then forwards the requests on the client’s behalf. Basically, proxy servers receive SIP messages and forward them to the next SIP server in the network. Proxy servers can provide functions such as authentication, authorization, network access control, routing, reliable request retransmission, and security.
- Redirect server—Provides the client with information about the next hop or hops that a message should take, then the client contacts the next hop server or UAS directly.
- Registrar server—Processes requests from UACs for registration of their current location. Registrar servers are often co-located with a redirect or proxy server.
What is the Cisco SIP Proxy Server (CSPS)?

The CSPS provides the primary capabilities required for call session management in a VoIP network and processes SIP requests and responses as described in RFC 2543. Powered by Apache, the CSPS can be configured to operate as a transaction stateful or stateless server. The CSPS can also be configured to provide additional server modes and features. For example, the CSPS can be configured with the following capabilities:

- Function as a redirect or registrar server
- Translate E.164 numbers to URL via location server protocols such as Telephone Number Mapping (ENUM)
- Perform gateway and Domain Name System (DNS) routing

Note

The CSPS includes software developed by the Apache Software Foundation (http://www.apache.org/).

CSPS Components

This section describes the main components in the CSPS and how they interact. See also Chapter 3, “Provisioning System GUI”.

- SIP Proxy Server (sipd)—handles all call processing and SIP messages. The provisioning client for sipd (spa) is installed if the CSPS GUI-based provisioning system is being used. SNMP is not automatically installed when sipd is installed. It must be manually installed as needed. See Cisco SIP Proxy Server Installation Guide.
- Provisioning Server (pserver)—main server used by the CSPS GUI-based provisioning system. A license manager (lm) component is automatically installed when the pserver is installed.
- MySQL—This database can be used to store and access provisioning system and subscriber feature data. If the provisioning system is installed, subscriber features (call forwarding and local authentication, not RADIUS) are automatically included since mysql already exists.
- Provisioning GUI client—provisioning client of the CSPS GUI-based provisioning system. It can be installed independent of the pserver. It requires installation of the correct version of JRE -1.3.1 (Java Runtime Environment 1.3.1).

The GUI retrieves and displays the current information in MySQL via the pserver. This information may be modified by the end user, prompting the GUI to send the updates to the pserver. If the information changed is licensing related, the license manager which resides with the pserver, does additional processing on this information. The pserver then updates the MySQL database with this information. Meanwhile, each spa (can be more than one on a multi-member farm) has registered for specific changes to the MySQL database. If a spa is notified of a change, it requests the pserver for the changed data and updates sipd.conf and/or routing and registry data. sipd can then use the new registry and routing entries immediately, and it uses the new sipd.conf if gracefully restarted.

Note

Changes to the MySQL database that spa does not register for are subscriber information such as call forwarding destinations and authentication passwords. sipd receives these information directly from MySQL.
CSPS Features

The CSPS offers the following features:

- Ability to function as a transaction stateful or stateless proxy server, stateful or stateless redirect server, and registrar server
- Call forwarding
- Choice of GUI based provisioning system or CLI tools for the following:
  - server configuration
  - embedded routing and registry databases
  - embedded subscriber database
- Ability to fork requests and distinguish spiralled requests from looped requests
- Address translation
  - Registry database (static and dynamic registry entries for contact points)
  - Gatekeeper Transaction Message Protocol (GKTMP) interface with Cisco NAM for 1-800, 1-900, and LNP lookups as well as least-cost routing
  - E.164 to URL address translation (via location server protocols such as ENUM and GKTMP)
  - Number expansion
- Next-hop routing
  - Static E.164 routes (dial plans)
  - Static domain routes
  - DNS SRV and DNS A record lookup and IP resolution
  - RAS module that allows communications between a CSPS and a H.323 directory gatekeeper
- Authentication and authorization via HTTP (Hypertext Transfer Protocol) Digest or HTTP Basic with a backend MySQL database, or via HTTP Digest, HTTP Basic or CHAP with a backend Radius server.
- Accounting via Radius
- Server farm support for sharing database information (registry and routing databases)
- Support for SIP over the following transport protocols:
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)
- Interoperability with Cisco SIP gateways, SIP IP phones, and unified messaging
- IP security (IPSec) for SIP signaling messages
- Access and error logging
- Support for rport and received parameters aiding in NAT traversal
- Domain-Specific registration, authentication, accounting, and subscriber database
- Pre-authorization query to a resource management system (RPMS)
- SNMP interface via CIAgent with basic platform MIBs for server status and start/stop/restart
Server Modes

The CSPS can be configured to function as one of the following types of servers:

- **Transaction stateful or stateless proxy server**
  
  When configured to be a stateful proxy server, the proxy server creates a transaction control block (TCB) and remembers incoming and outgoing requests, providing reliable retransmission of proxied requests and returning the best final response or responses back upstream. One transaction encompasses the received request, the request or requests (if forked) forwarded downstream, responses received from downstream hosts, and the best response returned upstream.

  When configured to be a stateless proxy server, the CSPS forgets all information once a request or response has been processed. As a stateless proxy server, the CSPS will just forward requests and responses.

  The CSPS also performs functions such as authentication, authorization, and access control.

- **Transaction stateful or stateless redirect server**
  
  As a redirect server, the CSPS accepts SIP requests, maps the address in the Request-URI to zero or more new addresses, and returns these addresses as Contacts in a SIP 3xx response to the UAC.

  When configured as a stateless redirect server, the CSPS does not create a TCB on receiving an INVITE request.

- **Registrar server**
  
  As a registrar server, the CSPS processes requests from UACs for registration of their current location. The CSPS also maintains registration information and can share this information with other registrar servers in its server farm. As a registrar server, the CSPS provides location services to the proxy server.

  **Note**
  
  In the CSPS, registrar servers must also be configured to function as either a proxy or redirect server. Standalone registrar servers are not supported.

  For more information on configuring server modes, see Chapter 3, “Provisioning System GUI” and Chapter 4, “Configuring the Cisco SIP Proxy Server (CSPS)”.

Protocol Support

The CSPS supports the following network and transport protocols:

- **Internet Protocol (IP)**
  
  IP is a network layer protocol that sends datagram packets between nodes on the Internet. IP also provides features for addressing, type-of-service (ToS) specification, fragmentation and reassembly, and security. The CSPS supports IP as it is defined in RFC 791 for SIP signaling.

- **User Datagram Protocol (UDP)**
  
  UDP is a protocol that exchanges data packets without acknowledgments or guaranteed delivery. SIP can use UDP as the underlying transport protocol. With UDP, retransmissions are used to ensure reliability. The CSPS supports UDP as it is defined in RFC 768 for SIP signaling.

- **Transport Control Protocol (TCP)**
  
  TCP is a protocol that exchanges data packets with guaranteed delivery. The CSPS supports TCP as it is defined in RFC 793 for SIP signaling.

- **Domain Name System (DNS)**
DNS is used in the Internet for translating names of network nodes into addresses. SIP uses DNS to resolve the host names of end points to IP addresses. The resolution may include DNS SRV and DNS A records.

- Session Description Protocol (SDP)

SDP is an ASCII-based protocol that describes multimedia sessions and their related scheduling information. The Cisco SIP IP phone uses SDP for session description.

### Spiralled and Looped Request Detection

A spiralled request is a request that revisits the CSPS with a new request-uri and is considered a new transaction by the proxy. A looped request is a request that contains the proxy's via header, and the request-uri is the same as it was when it first visited the proxy.

At the proxy, this capability has the following effects:

- Allows the same call to be logged more than once
- May cause invocation of different features
- If Record-Route field in the INVITE message is enabled at the proxy, Record-Route procedure will be performed more than once.

**Note**

The Call-ID, From, To and CSeq.seqnum fields of a spiralled request are the same as that of the request which visited the proxy the first time.

### Address Translation, Routing, and IP Resolution

The following steps are performed by the CSPS to deliver messages from endpoint to endpoint:

1. **Translation**—Process of translating an incoming Request-URI into an outgoing Request-URI.
2. **Next hop routing**—Process of obtaining a set of Fully Qualified Domain Names (FQDNs) or IP addresses with port numbers for each of the SIP entities found in the translation step. Among others, this step involves the following features:
   a. SRV lookup for static route—Process of a SRV lookup on the Static_Route_NextHop field if the Static_Route_NextHopPort field in a static route is not specified or is zero.
   b. RAS LRQ message transmission to gatekeeper—Process of sending the LRQ message to the H.323 gatekeeper to obtain the next hop gateway transport address.
3. **IP resolution**—Process of converting each next hop found in the next hop route lookup step into an IP address.

**Note**

If a Route header is present in the SIP request message, the CSPS translation and next hop routing functions are bypassed. Record-Route must be enabled on the server for subsequent requests to contain a route header.

### Translation

During the translation step, the Request-URI of an incoming request is processed and a list of Contacts returned, each providing a URL to use in the outgoing request.
If Number Expansion is enabled, the global set of expansion rules is applied to the user portion of the relevant URLs for which the host portion is the CSPS. For REGISTER messages, this applies to the To, From, Contact, and optionally the Authorization headers. For INVITE messages, this applies to the Request URI, From, and optionally the Proxy-Authorization headers.

**Note** The headers are not rewritten. The expanded versions are used internally for authentication, accounting, translation, and routing purposes.

The CSPS translation modules, in the order that they are called, are as follows:

- Call Forward Unconditional (mod_sip_call_forward)
- Registry (mod_sip_registry)
- ENUM (mod_sip_enum)
- GKTMP (mod_sip_gktmp)

The first module to return one or more Contacts completes the translation step, and the remaining modules are not called. For example, if the Registry module returns a Contact, then neither the ENUM or GKTMP module are called. If none of the translation modules returns a Contact, the core proxy module (mod_sip) returns a Contact based on the incoming Request-URI and that Request-URI is used in the next hop routing step.

**Next Hop Routing**

This step in SIP request processing is to determine the next hop route for each Contact. Next hop routing takes each translated Request-URI (Contact) and locates a set of next hop SIP entities capable of routing to the new Request-URI. This step involves two advanced features as follows.

**SRV Lookup for Static Route**—If the Static_Route_NextHopPort field of a static route is not specified or is zero, the CSPS performs an SRV lookup on the Static_Route_NextHop field.

**H.323 RAS module**—The RAS module allows communications between a SIP proxy server and a H.323 gatekeeper. The CSPS can send ASN.1 encoded RAS LRQ message to a H.323 gatekeeper which responds with a RAS LCF message. Figure 1-2 illustrates how the proxy server sends a LRQ H.225 RAS message to a gatekeeper which responds with a LCF message that can contain a gateway transport address.
The RAS module supports the following:

**H.323 RIP (Request in Progress) message**—This message contains a time-delay value. Upon receipt of this message, the CSPS resets its timer and waits for the final response from the gatekeeper.

**Directory gatekeeper**—This gatekeeper forwards requests to the other gatekeepers and returns the highest priority and lowest cost gateway information to the requesting endpoint. Value of the TTL (time-to-live) field in the LRQ message must be greater than zero. The default is 6 hops.

**Sequential LRQ**—The CSPS can send LRQ requests sequentially based on the priority specified for the gatekeeper clusters in sipd.conf file. Configurable parameters include timeout value for each individual request, total LRQ request window, and whether to wait for the best LCF within the LRQ time window or to return the first valid LCF response to the proxy server.

**Blast LRQ**—The CSPS can send LRQ requests to all gatekeeper clusters before listening to the responses from the gatekeepers. It can also be configured to either take the first valid LCF response or wait for the best response within the LRQ time window.

**Multi-alternate endpoint**—If the selected LCF from a gatekeeper contains alternate endpoints, the CSPS can parse these endpoints and store them in the route table. The routes in the alternate endpoints have lower priority than the route for the primary endpoint.

**Redundant gatekeeper**—This can be configured by maintaining a list of prioritized gatekeeper clusters in the CSPS configuration. Once this is configured, the following trial process occurs. The proxy server tries one of the gatekeepers randomly in the cluster which has the highest priority. If this trial expires the timeout value specified in the RASTimeoutInterval directive, the proxy server tries the next gatekeeper in the same cluster in a round-robin manner. If LRJ message is received by the proxy server, it tries a gatekeeper in the next highest priority cluster and so forth. **Figure1-3** illustrates the configuration.
Tech prefix—The CSPS prepends the technology prefix (one, two or three digits + ‘#’) to the expanded dialed number to the LRQ request. It can also be included in the Request-URI of the INVITE message forwarded to the SIP/H.323 gateway. Multiple technology prefixes can be configured in the sipd.conf file based on the expanded dialed number.

See directive RASTechPrefix in “Configuring E.164 to Request-URI Address Translation” section on page 4-18 and Chapter 3, “Provisioning System GUI”.

Pavo extension—The CSPS can include Pavo extensions (CallIdentifier, RedirectIEInfo, CallingOctet3a) in the LRQ from the CC-Diversion header of the incoming INVITE message.

To locate the next hop SIP entities, the following actions occur:

- If the host portion of the new Request-URI is the address (FQDN or IP address) of the server itself, the next hop routing is performed, using the user portion of the Request-URI.

  E.164 routing is an implementation of this method.

- If the Static_Route_NextHopPort field of a static route is not specified or has the value of 0, the CSPS tries to do SRV lookup on the Static_Route_NextHop field.

  If this lookup is successful, the algorithm outlined in RFC 2782 is used to select one destination. If this lookup fails, alternate destinations will be tried and the proxy will do simple DNS ‘A’ lookup on the Static_Route_NextHop field of the static route. This field should contain a name for the ‘A’ lookup.

- If no route is found in the E.164 routing, RAS LRQ message is sent by the CSPS to the H.323 gatekeeper cluster which is identified to have the highest priority gateway.

  According to configurations and availability of the gatekeepers being contacted, one of the following messages is returned before the time limit in the configured timeout window expires.

  - LCF (Location Confirm)
  - LRJ (Location Reject)
  - RIP (Response in Progress)

  The RIP message is activated when the gatekeeper cluster is connected to a remote gatekeeper (by UDP). This is related to the number of hops to be attempted. See directive RASTimeToLive in “Configuring E.164 to Request-URI Address Translation” section on page 4-18 and Chapter 3, “Provisioning System GUI”.

Figure 1-3 Gatekeeper Cluster Priority Configuration

![Gatekeeper Cluster Priority Configuration Diagram]
Chapter 1      Product Overview

CSPS Features

• If the *host portion* of the new Request-URI is NOT the address (FQDN or IP address) of the server itself, domain routing is performed using the *host portion* of the Request-URI.

  Domain routing and next hop routing via DNS SRV are implementations of this method.

IP Resolution

This step in processing SIP requests is to convert each next hop found via next hop routing into an IP address. Standard IP resolution (via gethostbyname) is performed via DNS, NIS, NIS+, or host file, depending on the IP resolution subsystem configured on the system where CSPS is located.

For more information on configuring directives pertaining to address translation and routing, see “Configuring E.164 to Request-URI Address Translation” section on page 4-18 and Chapter 3, “Provisioning System GUI”.

Authentication, Authorization

The Cisco SIP Server can be configured to provide authentication and authorization. The authentication can occur at a Radius Server or at the Proxy. Three types of authentication are supported by the Proxy in conjunction with the appropriate Radius Server.

• CHAP-Password Authentication
• HTTP Digest Authentication
• HTTP Basic Authentication

Only HTTP Digest Authentication and HTTP Basic Authentication are supported by the Proxy (without a Radius Server and via a MySQL database).

HTTP Digest Authentication and HTTP Basic Authentication are supported by the Proxy.

Radius is an IETF protocol based on UDP. It functions by exchanging a set of attribute/value pairs between the client and server. For example, a CSPS acting as a Radius client exchanges attribute/value pairs with a Radius Server to provide authentication.

For Radius supported authentication, the UAC password is stored at the Radius Server. For Proxy supported authentication, the UAC password is stored in a subscriber table in a MySQL database.

Authentication

The default authentication scheme is the HTTP Digest Authentication performed at the CSPS. HTTP Digest Authentication and HTTP Basic Authentication are performed as described in RFC 2617.

When Digest Authentication and Basic Authentication are used at the Proxy, the Username as found in the Authorization Header or the Proxy-Authorization Header is the key to query the MySQL database.

If the authentication takes place at the Radius Server, the Username is passed as one of the attribute/value pairs from the CSPS to the Radius Server where it can be used to key the User Search, before performing authentication. Additionally, CSPS can be configured to add any desired SIP headers as VSAs in the authentication request to the Radius Server.

The UserName may be expanded by the CSPS before performing the MySQL lookup, or before it is passed to the Radius Server. This enables phone numbers to be expanded to fully expanded E.164 numbers before processing.

If the virtual proxy host feature is turned on, the Username@domain (Username found in the Authorization/proxy-Authorization header and the domain name found in the From header) is the key used to query the MySQL database. This the same with authentication performed on a radius server and
the RadiusUserNameAttrAddDomain directive turned on. Username@domain is passed as one of the attribute/value pairs from the CSPS to the Radius Server where it can be used as the key for the user search feature.

Note
Native Apache based virtual host refers to providing the allusion of more than one server on one system, as differentiated by their hostname. For example, companies sharing a web server can have their own domains (www.company1.com and www.company2.com) and access to the web server. Native Apache based virtual hosts are not supported in CSPS.

The CSPS provides access control lists and user authentication which can be combined to provide more complex access control. Based on Apache access control mechanism, some SIP devices such as SIP gateways, do not authenticate themselves. They may be specified in the access control list, and instruct the CSPS to accept INVITEs from them.

REGISTERs and INVITEs from other devices may be challenged by the CSPS for user authentication. This is configurable.

Authorization
An authenticated user is authorized. Currently, there is no authorization of specific user capabilities for the service provider voice applications.

Accounting
The CSPS uses basic start-stop records with a combination of standard attributes and Cisco Vendor Specific Attributes (VSAs). The server can be configured to provide accounting services via a Radius server. When accounting is enabled, Call Detail Records (CDR) generated during calls are logged on a Radius accounting server as described in RFC 2159.

Accounting uses standard RADIUS attributes where possible. Additionally, CSPS can be configured to add any desired SIP headers as VSAs in the accounting requests. When the accounting services are enabled and the interface to the Radius server is configured, the CSPS creates and sends the Accounting records to the Radius server described as follows.

- The start record is written when a 200 OK is received by the CSPS for an INVITE.
- The stop record is written when BYE is received by the CSPS.

Note
The CSPS receives the BYE only if Record-Route is enabled. See Configuring E.164 to Request-URI Address Translation, page 4-18 and Chapter 3, “Provisioning System GUI”.

Server Farm Support
A server farm is a group of CSPSs that share database information, provide redundancy and high availability. When the members of a farm are defined, all updates to a member database are replicated across every member of the farm. During replication, any member of the farm that is out of synchronization with the other members, will be resynchronized.

By default, a farm is created for the routing and registry databases. To configure a farm, the names of the members of the farm and the ports at which to lock and update the farm members is required.

Database information is shared by mirroring all updates to databases for all members of the farm and by keeping the databases synchronized.
Registrar for Multiple Domain

The CSPS can act as a registrar for multiple domains. These domains can be a single domain (ProxyDomain) for fully expanded E.164 numbers, and zero or more additional domains within which private name spaces can exist.

For example, the E.164 domain can be cisco.com. The additional private domains can be a.com and b.com. The CSPS accepts REGISTER messages for the following domains:

<*>@a.com
<*>@b.com
<*>@cisco.com
<e.164-number>@cisco.com
<e.164-number>@a.com
<e.164-number>@b.com

If an INVITE is received for <e.164-number>@*.com, it is treated as an INVITE for <e.164-number>@cisco.com. All registrations for <e.164-number>.*.com are represented by a single entry in the registration database.

If an INVITE is received for user@a.com, or user@b.com, or user@cisco.com, it is treated as an INVITE for user@a.com, or user@b.com, or user@cisco.com, respectively. Registrations for user@a.com, user@b.com, and user@cisco.com result in three separate entries in the registry database.

The creation and administration of multiple domains of overlapping name spaces is a coordinated effort among the administrator of the CSPS and the administrators of the domains the CSPS supports.

For example, CompanyA and CompanyB are in area code 408. CompanyA may use 4-digit extensions 2000-7999 and CompanyB may also use 4-digit extensions, 2000-7999. There are no DIDs associated with these extensions. For users at CompanyA and CompanyB who need DIDs, CompanyA gets 1-408-555-[8000-8999] and CompanyB gets 1-408-666-[8000-8999].

Registration occurs as follows.

- 2xxx at CompanyA registers as 2xxx@companyA.com
- 2xxx at CompanyB registers as 2xxx@companyB.com
- 8xxx at CompanyA registers as +14085558xxx@companyA.com
- 8xxx at CompanyB registers as +14086668xxx@companyB.com

Note  xxx can be replaced by any number. For example, 2xxx can be 2000, 2001, 2999, etc.

Call processing occurs as follows.

2001 calls 2000

- 2001 at CompanyA calls 2000 at CompanyA as 2000@companyA.com [1]
- 2001 at CompanyA calls 2000 at CompanyB as 2000@companyB.com [2]
- 2001 at CompanyB calls 2000 at CompanyB as 2000@companyB.com [1]
- 2001 at CompanyB calls 2000 at CompanyA as 2000@companyA.com [2]
8001 calls 2000
- 8001 at CompanyA calls 2000 at CompanyA as 2000@companyA.com [1]
- 8001 at CompanyA calls 2000 at CompanyB as 2000@companyB.com [2]
- 8001 at CompanyB calls 2000 at CompanyB as 2000@companyB.com [1]
- 8001 at CompanyB calls 2000 at CompanyA as 2000@companyA.com [2]

2001 calls 8000
- 2001 at CompanyA calls 8000 at CompanyA as +14085558000@companyA.com[3]
- 2001 at CompanyA calls 8000 at CompanyB as +14086668000@companyA.com[4]
- 2001 at CompanyB calls 8000 at CompanyB as +14086668000@companyB.com[3]
- 2001 at CompanyB calls 8000 at CompanyA as +14085558000@companyB.com[4]

8001 calls 8000
- 8001 at CompanyA calls 8000 at CompanyA as +14085558000@companyA.com[3]
- 8001 at CompanyA calls 8000 at CompanyB as +14086668000@companyA.com[4]
- 8001 at CompanyB calls 8000 at CompanyB as +14086668000@companyB.com[3]
- 8001 at CompanyB calls 8000 at CompanyA as +14085558000@companyB.com [4]

2000 and 8000 call 18183635839
- 2000 at CompanyA calls 18183635839 as +18183635839@companyA.com [5]
- 8000 at CompanyA calls 18183635839 as +18183635839@companyA.com [5]
- 2000 at CompanyB calls 18183635839 as +18183635839@companyB.com [5]
- 8000 at CompanyB calls 18183635839 as +18183635839@companyB.com [5]

John Doe, who is not with CompanyA or CompanyB, but is an authorized user, places calls as follows.
- Calls 2000 at CompanyA as 2000@companyA.com [2]
- Calls 2000 at CompanyB as 2000@companyB.com [2]
- Calls 8000 at CompanyA as +14085558000@<proxy>.com [6]
- Calls 8000 at CompanyB as +14086668000@<proxy>.com [6]
Note

- [1] The phones at both companies expand 2xxx to 2xxx@<proxy>
- [2] This is done via full URL dialing
- [3] The phones at companyA expand 8xxx to +14085558xxx@<proxy> the phones at companyB expand 8xxx to +14086668xxx@<proxy>
- [4] The phones at both companies expand [2-9]xxxxxx to +1408xxxxxxx@<proxy>. All authenticated users, regardless of domain, have access to the same routes for forwarding calls for +1408xxxxxxx
- [5] The phones at both companies expand 1xxxxxxxxx to +1xxxxxxxxx@<proxy>. All authenticated users, regardless of domain, have access to the same routes for forwarding calls for +1xxxxxxxxx
- [6] CSPS supports a default domain for e.164 numbers. These are designated by the plus (+) sign. Refer to Figure 1-4 for detail.

Figure 1-4  Multiple Domains Registration
IP Security

IP Security (IPSec) provides security for transmission of sensitive information over unprotected networks such as the Internet. IPSec acts at the network layer, protecting and authenticating IP packets between participating IPSec devices (peers), such as the CSPS and a UAC, SIP gateway, or another CSPS. With IPSec, data can be transmitted across a public network without fear of observation, modification, or spoofing.

All IPSec combinations have been successfully tested and proven to secure traffic to and from the CSPS on the following platforms:

- Solaris 2.8 Operating Environment (via manually-keyed key management method)
- Redhat 7.1 with Linux FreeS/WAN Release 1.5 (via manually keyed and Internet Key Exchange (IKE) key management methods)

Access and Error Logging

The CSPS uses the standard Apache logging functionality in addition to a SIP-specific logging functionality. The standard Apache logging functionality is defined for the CSPS as a whole, while the SIP logging functionality can be configured on a per-module basis.

By default, internal CSPS errors are logged to /usr/local/sip/logs/error_log (on Linux) and /opt/sip/logs/error_log (on Solaris). Access records are logged to /usr/local/sip/logs/access_log (on Linux) and /opt/sip/logs/access_log (on Solaris). For information on interpreting the information in the log files, and customizing the type and amount of information they include, see Chapter 3, “Provisioning System GUI,” Chapter 4, “Configuring the Cisco SIP Proxy Server (CSPS),” and Chapter 5, “Maintaining the Cisco SIP Proxy Server (CSPS).”

Log rotation and size are per standard apache. For information on Apache and Apache logs, see URL www.apache.org.

Requirements

The CSPS can be run on a Solaris or Linux platform. The following sections list the minimum hardware and software requirements for each platform. Before installing and configuring the CSPS, one of the following sets of requirements must be met.

For a list of Cisco supported platforms, see the Cisco SIP Proxy Server (CSPS) Release Note.

Solaris Platform Requirements

For Solaris, the following is required:

- Workstation—Sparc or UltraSparc server class system with a minimum of 256 MB of RAM and 1GB of disk space.
- Solaris 2.8
- IPSec encryption requires the installation of supplemental encryption software. Additional details and free downloads of the software are available at: http://wwws.sun.com/software/solaris/encryption/
Linux Platform Requirements

For Linux, the following is required:

- PC—Intel Pentium III processor operating with a minimum of 128 MB of RAM and 1 GB of disk space.
- Redhat 7.1
- For IPSec, Linux FreeS/WAN with Redhat Linux (See URL www.freeswan.org for the latest implementation of Linux FreeS/WAN IPSec and its targeted Redhat Linux version).

Reference Links

The following is a list of URLs on companies and additional information on technologies referenced within the CSPS software and documentation:

- http://www.apache.org
- http://www.mysql.com
- http://www.sun.com
- http://www.linux.org
- http://www.redhat.com
- http://www.freeswan.org
- http://wwws.sun.com
Licensing System

This chapter provides information on the following:

- License Keys, page 2-1
- Activating the Licensing System, page 2-2

License Keys

This section describes the license key file, information in the license key and characteristics of the license key.

License Key File

This license key file (license.conf) is installed in the /usr/local/sip/conf directory on a Linux system, and in the /opt/sip/conf directory on a Solaris system. It is a text file that the CSPS requires at startup. The CSPS validates the License Key in this file before it can run. License validation debug messages are logged in the error_log file unconditionally.

If the csps_setup script is used for installation, it prompts the user for the license key and automatically updates the license.conf file. If this script is not used for installation, the license key may be entered either by using the license GUI (see following sections), or by manually entering the license key into the default license.conf file. The license.conf file format is as follows.

```
LicenseKey RSRCJDEkIAYo3j16dsNDE6MDolsdUwJDU0MjYKnZI=
LicenseDebug Off
```

The file contains the following parameters:

- LicenseDebug flag—Enables low level debug functionalities. Default is Off.
- LicenseKey—Contains the license key provided to the customer

The license key file must be installed before starting the CSPS. The license key in the above example is not a valid license key.
Information in License Key

The license key specifies the following:

- Whether this is a Eval (evaluation) license or a Permanent license—A Eval license has an expiration date. A Permanent license has no expiration date.
- The major release supported by the license—Use the same license key with different minor releases of the same major release. Each major release requires a different license key.
- Expiration date of the license—This is applicable to Eval license only. The expiration date is calculated from the date and time when the license key is generated.

License Key Characteristics

Whenever the CSPS is started or restarted, the validity of the license is verified. If the license fails the validity check, the proxy server does not start.

With an Evaluation license, you cannot start or restart the CSPS after the expiration date. You can gracefully restart it. A warning is issued when the time to expiration is less than or equal to seven days. The time to expiration is logged only when the CSPS is started, restarted or gracefully restarted.

**Note**

A Base license can turn on any of the infrastructure functionalities in sipd.conf. However, these functionalities are disabled. With an Infrastructure license, you can optionally turn off any of the infrastructure functionalities in sipd.conf.

Activating the Licensing System

This section describes how to use the Licensing System GUI.

**Step 1**

For Linux or Solaris systems, if the csps_setup script is used to install the system, the script automatically creates a sysadmin_csps_license script in the /usr/local/sip/bin or /opt/sip/bin directory respectively. In the respective directory, run the script (/sysadmin_csps_license) to activate the licensing GUI.

For Windows 2000, the licensing.jar on the Linux or Solaris system (in /usr/local/sip/bin or /opt/sip/bin directory respectively) must be copied to your Windows system. If these files are in a folder named CSPS in the C drive directory (Program Files), use the following command from a dos/cmd/ command line window to activate the GUI:

```
java -jar c:/"Program Files"/CSIPS/licensing.jar
```
You can also double click on a text file icon with file extension .bat (batch file). The batch file contains the following line:

```
java -jar c:"Program Files"/CSPS/licensing.jar
```

Enter the Password in the Login Dialog Box as shown in Figure2-1.

**Figure2-1 Licensing Login Dialog Box**

![Licensing Login Dialog Box](image)

**Note** The default password is cspsuser. To change the password, use the GUI password option in the GUI.

**Step 2** Click **OK** to access the Licensing window or click **more>>** to access the following dialog box, Figure2-2, to enter the host name and port number.

**Note** If the Licensing GUI is run from a system which is not the pserver, enter the hostname and port number of that system. These entries are automatically saved and reappears at each login.

**Figure2-2 Licensing Host and Port Dialog Box**

![Licensing Host and Port Dialog Box](image)

A display similar to Figure2-3 appears.
Adding Licenses

Step 1  To add a license, click Add in the Licensing window. A display similar to Figure2-4 appears.

Step 2  Enter license information and settings.

Step 3  If you click Browse to enter License Key, A display similar to Figure2-5 appears.
Step 4 Select or type the wanted file name. Click **Open**.

**Note**
The license file must be a text file that contains the license key only. License key can also be entered in the License Key entry box.

Step 5 Click **Submit** or **Cancel**.

---

**Editing Licenses**

**Step 1** To edit a license, click **Edit** or double click the wanted license in the licensing window. A display similar to **Figure2-6** appears.

**Note** The current IP and License Key appear in the IP and License Key fields.

**Figure2-6** **Edit License Window**
Step 2 Enter license information and settings.
Step 3 If you click Browse to edit License Key, select or type the wanted file name in the Open File window. Click Open.
Step 4 Click Submit or Cancel.

Deleting and Reloading Licenses

Step 1 To delete a license, select the license and click Delete.
Step 2 To reload all the licenses, click Reload All.

Note Reload redisplays the data. If it was filtered, the redisplayed data are unfiltered. See “Searching and Filtering Licenses” section on page 2-6.

Searching and Filtering Licenses

Step 1 To search or filter data, use one of the following methods.
  • Click Operations on top of the Licensing window. Click Search.
  • Right click the Licensing window. Click Search.
A display similar to Figure 2-7 appears.

Figure 2-7 Search License Window

Step 2 Select search query and operators by clicking the down arrows respectively. Displays similar to Figure 2-8 and Figure 2-9 appear.
**Step 3** Enter a search string in the Search string box. Click **OK**.
This chapter provides information on using the Provisioning System GUI (Graphical User Interface) as follows.

- Activating the Provisioning System, page 3-1
- Configuring a Farm, page 3-3
- Configuring Advanced Features for a Farm, page 3-5
- Subscribers and Static Registries, page 3-27
- Static Routes, page 3-30
- Seeding, page 3-32
- GUI Password, page 3-34
- Searching and Filtering Data, page 3-34

It is highly recommended that the csps_setup installation script be used when you install the CSPS with a provisioning system. This script prompts you for the license key during installation, and automatically creates a license.conf file. If the csps_setup script is not used during installation, the license key must be entered by using the license GUI. See Chapter 2, “Licensing System”.

**Note**

The sysadmin_csps_regroute tool should not be used to add, modify, or delete data if the GUI based provisioning system is used. The tool can be used for viewing data, such as dynamic registrations that the GUI cannot provide, or to import files containing static registration or routing entries.

See also Chapter 1, “Product Overview”, “CSPS Components” section on page 1-4 for detail on CSPS components, and Chapter 5, “Maintaining the Cisco SIP Proxy Server (CSPS)”, “Starting and Stopping the CSPS” section on page 5-1.

### Activating the Provisioning System

This section describes how to activate the Provisioning System by using the GUI.

**Step 1**

For Linux or Solaris systems, if the csps_setup script is used to install the provisioning system, the script automatically creates a sysadmin_csps_provision script in the /usr/local/sip/bin or /opt/sip/bin directory respectively. To activate the GUI, run the script, ./sysadmin_csps_provision, from the respective directory.
For Windows 2000, the mascarpone.jar on your Linux or Solaris system (in /usr/local/sip/bin or /opt/sip/bin directory respectively) must be copied to your Windows system. If this file is in a folder named CSPS in your C drive directory, Program Files, use the following command from a dos/cmd/command line window to activate the GUI.

```java
java -jar c:/Program Files/CSPS/mascarpone.jar
```

You can also double click on a text file icon with file extension, .bat (batch file). The batch file contains the following line:

```java
java -jar /mascarpone.jar
```

A Login Dialog Box appears (Figure3-1).

![Figure3-1 Provisioning Login Dialog Box](image)

**Step 2** Enter the Password in the Login Dialog Box.

**Note** The default password is cspsuser. You can use the GUI password option in the GUI to change the password.

**Step 3** Click **OK** to access the provisioning GUI or click **more>>** to access the dialog box (Figure3-2) to enter the host name and port number.

**Note** If the Provisioning GUI is run from a system which is not the pserver, enter the hostname and port number of that system. These entries are automatically saved and reappears at each login.

![Figure3-2 Provisioning Host and Port Dialog Box](image)

The main provisioning menu appears (Figure3-3).
Configuring a Farm

See also “Defining the CSPS Configuration File” section on page 4-1.

To configure a farm, use the following steps:

**Step 1**  
Click Farm/Proxies in the option side-bar in the main menu. The following screen appears (Figure 3-4).

**Figure 3-4 Farm Configuration Window**

![Farm Configuration Window](image)
Step 2  Enter the farm label, server root and proxy domain (Figure 3-4).

Note  This information is automatically filled in if the csps_setup script was used for installing this system.

Step 3  Click Add Proxy to enter a host name and a port number.

Step 4  To view or edit more information on a farm member, click Show Additional Fields. The following data appears (Figure 3-5).

**Figure 3-5 Additional Farm Data**

![Edit existing Farm](image)

To hide the additional data, click Hide Additional Fields.

If you want to remove unwanted proxy data, click Delete Proxy.

Step 5  Click Submit to submit the data to the provisioning server. The following message appears (Figure 3-6).

**Figure 3-6 Farm Configuration Submission Message**

![Message](image)

Alternatively, you can click Cancel to close the window and the data are not submitted to the provisioning server.

Note  Farm label is an unique identifier. For Linux, the server root is /usr/local/sip as shown in Figure 3-4. For Solaris, it is /opt/sip. When a proxy is added or deleted through the GUI, the addition or deletion is virtual, as considered by the provisioning server only. That is, to physically add or delete the software or a process to or from the system, you have to install/uninstall the software or start/stop the process.
Configuring Advanced Features for a Farm

This section describes configurations for the following advanced features for a farm.

- Farming, page 3-6
- Virtual Proxy Host, page 3-9
- RAS, page 3-9
- RPMS, page 3-11
- Debug and Logs, page 3-11
- Authentication, page 3-13
- Call Forward, page 3-15
- Number Expansion, page 3-16
- ENUM, page 3-17
- Server Directives, page 3-18
- SIP Server Core, page 3-20
- MySQL, page 3-23
- GKTMP, page 3-24
- Accounting, page 3-24
- Access Control, page 3-26

For more detail on configuring these advanced features, see Chapter 4, “Configuring the Cisco SIP Proxy Server (CSPS)”.

To configure advanced features for a farm, select Advanced in the Farm Configuration window. The following screen appears (Figure3-7).
Figure 3-7 Farm Configuration Advanced Features Window

After a configuration is set up, use one of the following command buttons:

- Click **Submit** to submit configurations. Farm Configuration Submission Message appears.
- Click **Cancel** to cancel the configuration.
- Click **<<advanced** to return to the Farm Configuration window.

**Note**

In the configuration windows, an asterisk, *, next to an entry box indicates that the entry is required.

**Farming**

To configure routing and registry, use the following steps:

**Step 1** Click **Farming** in the Farm Configuration Advanced Feature window. The following screen appears (Figure 3-8).
Step 2  Select or enter the appropriate settings in the Routing and Registry sections.

Step 3  Scroll down the window to display the Farm Member section shown in Figure3-9.
Step 4

Add or delete farm members.

To add a farm member, enter the host and port in the Farming Configuration Window. Registry and Route information are synchronized between farm members that show On in the In Registry Farm and In Route Farm columns.

Note

When a member is not part of a synchronized farm (Off is displayed in the In Registry Farm and In Route Farm columns), it still receives route/registry updates as long as its host is in the Farm Member table. Database synchronization limits the number of registries/routes to around 7000. Removing database synchronization allows up to 50,000 registries/routes to be supported. In this case, none of the members want to be in a synchronized farm, so database synchronization can be eliminated, but the provisioning system should update each farm member in the table with routes/registries.

Only one hostname or IP is needed in a synchronized farm, since the other farm members automatically get the updates. However, in an unsynchronized farm, all farm member hostnames or IP addresses should be entered and separated by commas, so all members are deleted/seeded with the same route/registry data from the provisioning system. See Seeding, page 3-32.
Virtual Proxy Host

To configure virtual proxy host, use the following steps:

Step 1  Click **Virtual Proxy Host** in the Farm Configuration Advanced Feature window. The following screen appears (**Figure3-10**).

Figure3-10  Virtual Proxy Host Configuration Window

![Virtual Proxy Host Configuration Window](image)

**Step 2**  Select **On** for Use Virtual Proxy Host.

**Step 3**  Add or delete proxy hosts as needed.

RAS

To configure RAS, use the following steps:

**Step 1**  Click **RAS** in the Farm Configuration Advanced Feature window. The following screen appears (**Figure3-11**).

![RAS Configuration Window](image)
Step 2  Select or enter the settings for RAS.

Step 3  Add or delete tech prefix in the Tech Prefix section.

Step 4  Scroll down to display the Gatekeeper Clusters section shown in Figure3-12.

Figure3-12  RAS Configuration Window (cont’d)

Step 5  Add or delete gatekeeper clusters.
RPMS

To configure RPM, use the following steps:

**Step 1**  
Click **RPMS** in the Farm Configuration Advanced Feature window. The following screen appears (Figure3-13).

**Figure3-13  RPMS Configuration Window**

![RPMS Configuration Window](Image)

**Step 2**  
Select or enter the settings for RPMS.

**Step 3**  
Add or delete items in the Server IP Port Secret and Previous Hops sections respectively.

Debug and Logs

To configure debug flags, log level and format, use the following steps:

**Step 1**  
Click **Debug and Logs** in the Farm Configuration Advanced Feature window. The following screen appears (Figure3-14).
Step 2  In the Debug Flags section, click in the checkboxes to select debug flags.
Step 3  In the Error Log entry box, enter the file path for error log.
Step 4  In the Log Level dropdown list, select the log level.
Step 5  In the Custom Log section, add or delete custom log file.
Step 6  Scroll down to display the Log Format section shown in Figure3-15.
Figure 3-15 Debug and Logs Configuration Window (cont’d)

Step 7 In the Log Format section, add or delete log formats.

Step 8 Select or enter settings for SIP Stats Log, SIP Stats Interval, Shared Memory Stats Log, and Shared Memory Stats Interval.

Note TransferLog is not provisionable from the GUI, since the CustomLog can be set up to have the same functionality as TransferLog.

Authentication

To configure authentication, use the following steps:

Step 1 Click Authentication in the Farm Configuration Advanced Feature window. The following screen appears (Figure 3-16).
Step 2  Select or enter the settings for authentication.

Step 3  Enter IP address, Port number and password in the Primary Radius Server and the Secondary Radius Server sections shown in Figure3-17.
Call Forward

To configure Call Forwarding, use the following steps:

**Step 1** Click **Call Forward** in the Farm Configuration Advanced Feature window. The following screen appears (Figure3-18).
Step 2 Select or enter the settings for call forwarding.

**Number Expansion**

To configure Number Expansion, use the following steps:

**Step 1** Click **Number Expansion** in the Farm Configuration Advanced Feature window. The following screen appears *(Figure3-19).*
Figure 3-19 Number Expansion Configuration Window

Step 2 Select On for Number Expansion.

Step 3 Add or delete number plans in the Number Plan section.

ENUM

To configure ENUM, use the following steps:

Step 1 Click ENUM in the Farm Configuration Advanced Feature window. The following screen appears (Figure 3-20).
Step 2  Select On or Off for ENUM, and enter private domain and global domain as needed.

Server Directives

To configure the server directives, use the following steps:

Step 1  Click Server Directives in the Farm Configuration Advanced Feature window. The following screen appears (Figure3-21).
Step 2  Enter server directives and pool size information.

Step 3  Scroll down to display the Listen section shown in Figure3-22.
Figure 3-22  Server Directives Configuration Window

Step 4  Add or delete data in the Listen section and enter User, Group and Server Name.

Step 5  Select On or Off for Hostname Lookups.

SIP Server Core

To configure the SIP Server Core, use the following steps:

Step 1  Click SIP Server Core in the Farm Configuration Advanced Feature window. The following screen appears (Figure 3-23).
Step 2  Select or enter settings for the SIP server core, and scroll down to the SIP over TCP section shown in Figure 3-24 and Figure 3-25.
Step 3  Select or enter settings in the SIP over TCP section.
Note
The Persistent Connection File field is not editable. This file must reside in the conf directory and must be named persistent_tcp.conf.

MySQL

To configure MySQL database, use the following steps:

**Step 1**
Click MySQL in the Farm Configuration Advanced Feature window. The following screen appears (Figure 3-26).

**Figure 3-26 MySQL Configuration Window**

![MySQL Configuration Window](image)

**Step 2**
Select or enter database settings.

Note
The MySQL database name, table name and field names are not provisionable from the GUI, since the GUI expects specific names.
To configure GKTMP, use the following steps:

**Step 1** Click GKTMP in the Farm Configuration Advanced Feature window. The following screen appears (Figure3-27).

![GKTMP Configuration Window](image)

**Step 2** Select **On** for GKTMP Connection.

**Step 3** Enter information for the Primary and Secondary Servers.

---

**Accounting**

To configure accounting, use the following steps:

**Step 1** Click **Accounting** in the Farm Configuration Advanced Feature window. The following screen appears, Figure3-28.
Figure3-28  Accounting Configuration Window

Step 2  Select On for Accounting, then select Radius for Record format.
Step 3  Select time format.
Step 4  Enter information for the Primary and Secondary Radius Servers.
Step 5  Scroll down to display the SIP Headers section shown in Figure3-29.
Figure3-29 Accounting Configuration Window (cont’d)

**Step 6** Add or delete SIP headers.

**Access Control**

To configure access control, use the following steps:

**Step 1** Click **Access Control** in the Farm Configuration Advanced Feature window. The following screen appears, **Figure3-30**.
Chapter 3  Provisioning System GUI

Configuring a Farm

Figure 3-30  Access Control Configuration Window

Step 2  Select On for Access Control.

Step 3  Select the settings for Access Order and Satisfy.

Step 4  Add or delete Deny and Allow settings.

Subscribers and Static Registries

Note  If the same subscriber is statically configured and dynamic registered, there may be a mismatch between what the provisioning system considers as registered as opposed to the registry information in shared memory that call processing uses. It is not recommended to have users statically and dynamically registered at the same time.

To configure a subscriber or a static registry, use the following steps:

Step 1  Click Subscriber/Static Registry in the option side-bar in the main menu. The following window appears (Figure 3-31).
Note: To hide or resize a column, click Column in the window. You can resize a column by placing the cursor on the vertical line dividing the column headers, then drag the cursor to resize the column. Order of the columns can be rearranged by placing the cursor on the header and drag it to a desired position or order in the window.

Step 2: To add a subscriber and/or registry, click Add. The following screen appears (Figure 3-32).
**Chapter 3  Provisioning System GUI**

**Configuring a Farm**

**Figure 3-32  Add New Subscribers Window**

Note  The First, Middle and Last name fields are useful for functionalities such as sorting, searching and filtering in the provisioning system. They are not used by the CSPS.

**Step 3** Enter subscriber information and URLs.

Note  To configure a registry only, enter the User and Domain in the Subscriber window as a minimum requirement.

**Step 4** Click **Submit** or **Cancel**.

**Step 5** To edit a subscriber, select the subscriber and click **Edit** or double-click the subscriber.

Note  To sort the listings, click the column header that you want to sort on. An arrow appears in the column header which is the sort key.

**Step 6** To delete a subscriber, select the subscriber and click **Delete**.

**Step 7** To reload all the subscribers, click **Reload All**.

Note  **Reload** redisplays data. If the data was filtered, the redisplayed data are unfiltered. See Searching and Filtering Data, page 3-34.

**Step 8** Click **Static Registry** to add or delete static registries. The following screen appears (Figure 3-33).
Figure 3-33  Static Registry Window

Step 9  Add, delete or rearrange static registries.

Step 10  Click Submit or Cancel.

**Note**  The associated subscriber's User and Domain data must be filled in before the Static Registry contacts can be submitted.

**Note**  If a static registry with an expiration time is entered from the provisioning system, it automatically expires appropriately from CSPS. The provisioning data still contains it until it is manually removed.

### Static Routes

To configure static routes, use the following steps:

Step 1  Click Static Routes in the option side-bar in the main menu. The following window appears (Figure 3-34).
Configuring a Farm

Figure 3-34  Static Routes Configuration Window

Note
To hide or resize a column, click Column in the window. You can resize a column by placing the cursor on the vertical line dividing the column headers, then drag the cursor to resize the column. Orders of the columns can be rearranged by placing the cursor on the header and drag it to a desired position or order in the window.

Step 2  Click Add to add new routes. The following screen appears (Figure 3-35).

Figure 3-35  Add New Static Route Window

Step 3  Enter data and settings for the static route.

Step 4  Click Submit or Cancel.
To edit a route, select the route and click **Edit** or double-click the route.

**Note**
To sort the listings, click the column header that you want to sort on. An arrow appears in the column header which is the sort key.

To delete a route, select the route and click **Delete**.

To reload all the routes, click **Reload All**.

**Note**
- The majority of help text are displayed when the cursor is pointed to a column header.
- If a module is turned off, all associated directives are greyed out. Changes can only be done when the module is turned on.

**Seeding**

The provisioning system GUI provides a seeding functionality to remove all routes and permanent registries from shared memory, then re-seed it with the data stored in the MySQL database of the provisioning system. There is an option to seed without deleting any routes and registries first.

The seeding functionality assumes that the MySQL database of the provisioning server contains the master list of routes and registries. Any routes and permanent registries in shared memory are deleted if one of the delete options is selected, and all routes and registries in the MySQL database of the provisioning server is pushed into shared memory. When registries are deleted, dynamic and non-permanent registries are not removed from the database.

To revert to the running information that was in shared memory, it is recommended to use the export functionality of the `sysadmin_csps_regroute` tool to save the routes and registries that are in shared memory before executing the seeding functionality of the provisioning system GUI. This operation may take several minutes, depending on the number of routes and registries. It is recommended to do this during low traffic hours.

The seeding functionality is useful for the following purposes:
- When the administrator changes data via the GUI, but CSPS is down and the changes are not propagated to the CSPS shared memory. If there is only added data, the seed option can be used. If the data is modified or deleted, the delete and seed option should be used. This takes more time which depends on the size of the database.
- To make sure that the CSPS shared memory is synchronized with the data in the MySQL database of the provisioning system.

To configure seeding, use the following steps:

**Step 1**
Click **Seeding** in the option side-bar in the main menu. The following window appears (Figure 3-36).
Figure 3-36 Seeding Configuration Window

Note To hide or resize a column, click Column in the window. You can resize a column by placing the cursor on the vertical line dividing the column headers, then drag the cursor to resize the column. Orders of the columns can be rearranged by placing the cursor on the header and drag it to a desired position or order in the window.

Step 2 In a single member farm configuration, click the only member displayed, then click Edit. The following window appears (Figure 3-37).

Figure 3-37 Edit Seeding Entry Window

Step 3 Click the checkboxes to delete and/or seed the Routing and Registry databases.

Step 4 Click Submit or Cancel.

Step 5 All checkboxes are unchecked once Submit is selected and the seeding operation is completed, or Cancel is selected. To refresh screen display or find out if the seeding operation is completed, click Reload All in the Seeding Configuration window.
Multi-member Farm

In a multi-member farm, there are two basic configurations as follows.

- The member’s route and/or registry information are farmed together
  
  In the Seeding Configuration window, click only one member, then click Edit. The other farm members automatically receive these changes.

- The member’s route and/or registry information are not farmed together
  
  In the Seeding Configuration window, click all members (click on top member, press Shift, click on the bottom member), then click Edit. The Host information is blank since there are multiple hosts that will be seeded or deleted.

Click the appropriate checkboxes in the Edit Seeding Entry window, then click Submit or Cancel.

Note

If you click Delete and Seed and Seed only, a delete and a single seed occur. The system do not seed twice.

GUI Password

To change the password, use the following steps:

Step 1  Click GUI Password in the option side-bar in the main menu. The following window appears (Figure3-38).

![GUI Password Window](image)

Step 2  Enter the new password.

Step 3  Click Submit or Cancel.

Searching and Filtering Data

This section describes how to search and filter data.

Step 1  To search or filter data, use one of the following methods.
• Click **Operations** on top of the subscriber/registry table or static route table in the Subscribers and Static Registries window, and Static Route window respectively. Click **Search**.

• Right click the subscriber/registry or static route table in the Subscribers and Static Registries window, and Static Route window respectively. Click **Search**.

**Figure 3-39 Subscriber/Registry Table**

A display similar to the following screen appears (Figure 3-40).

**Figure 3-40 Search Window**

Step 2 Select search query and operators by clicking the down arrows respectively. The following screens appear (Figure 3-41, Figure 3-42).

**Figure 3-41 Search query Selection—Last Name**
Step 3 Enter a search string in the Search string box. Click OK. The following screen appears (Figure 3-43).

Note Filter status message (lower left corner) in search result window displays the search/filter query used. To hide or resize a column, click Column on top of the window. You can also resize a column by placing the cursor on the vertical line dividing the column headers, then drag the cursor to resize a column.
This chapter explains how to configure the CSPS. It contains the following information:

- Defining the CSPS Configuration File, page 4-1
- Configuring the SIP Proxy Server in a Farm, page 4-26
- Configuring IPSec, page 4-28

### Defining the CSPS Configuration File

**Note**

The preferred method of configuring the CSPS is to use the provisioning system GUI. See Chapter 3, “Provisioning System GUI”. However, all CSPS 1.x versions require configuration with text-based files. For backward compatibility, manual editing of all configuration files is supported. If the GUI is used, manual editing of the configuration files is not recommended. Manual changes to any configuration file written by the GUI will be overwritten when the GUI is used again.

The CSPS main configuration file is sipd.conf. A default sipd.conf configuration file is copied into /usr/local/sip/conf/ when installation is on a Linux platform and copied into /opt/sip/conf when installation is on a Solaris platform. In most cases, the default configuration is valid for starting the CSPS and for placing some test registrations and calls through it. The defaults should be customized as needed for the given environment.

Before beginning any of the configuration tasks in this chapter, change to the directory in which the sipd.conf file is located and open the file by using a text editor such as vi.

Similar to the Apache Server, the CSPS directives are grouped into major categories. The major categories of CSPS directives are:

- Server global directives—Define the overall operation of the CSPS. See the “Configuring the Server Global Directives” section on page 4-2.

- Host-specific directives—Define the basic configuration of the main CSPS which responds to requests that are not handled by a virtual host.

  Virtual host refers to maintaining more than one server on one machine, as differentiated by their hostname. For example, companies sharing a web server can have their own domains (www.company1.com and www.company2.com) and access to the web server. See the “Configuring the Host-Specific Directives” section on page 4-3.
Defining the CSPS Configuration File

- Core SIP server directives—Define the primary SIP functionality of the CSPS. If a core SIP server directive is not specified, the server uses the default. See the “Configuring the Cisco SIP Module Core Directives” section on page 4-5.

- SIP server module directives—Define the CSPS interfaces and additional functionality on a per module basis. See the “Configuring Cisco SIP Module Standard Directives” section on page 4-9.

Note
Alternatively and preferably, use the provisioning system GUI to configure the CSPS.

Configuring the Server Global Directives

The server global directives are generic server directives that define the overall operation of the server. These directives exclude those that configure protocol specific (HTTP or SIP) details. For example the global directive section of the sipd.conf file specifies the directory in which the Cisco SIP software resides and how child processes of the CSPS functions.

Note
The directives that configure the CSPS global environment are standard Apache directives. If the default for an Apache directive has been changed for the CSPS usage, the new default is documented. For more detail on the Apache directives, see www.apache.org.

To configure the server global directives, define values as necessary for the following directives:

- **ServerRoot**—Directory in which the CSPS executables, configuration, error, and log files reside (`bin/`, `conf/` and `logs/`). On Linux, the default directory for these subdirectories is `/usr/local/sip`. On Solaris, the default directory is `/opt/sip`. Do not add a forward slash (/) to the end of the directory path.

- **LockFile**—Path to the lockfile used when the CSPS is compiled with either `USE_FCNTL_SERIALIZED_ACCEPT` or `USE_FLOCK_SERIALIZED_ACCEPT`. This directive should normally contain its default value. The main reason for changing it is if the logs directory is NFS mounted, since the lockfile must be stored on a local disk. The PID of the main server process is automatically appended to the filename. The default is `logs/accept.lock`.

Note
If the server is to be on a NFS or networked mounted filesystem, see LockFile documentation at url: http://www.apache.org/docs/mod/core.html#lockfile.

- **PidFile**—Path and file to which the CSPS records its process ID when it is started. If the filename does not begin with a forward slash (/), it is assumed to be relative to the ServerRoot. The default is `logs/sipd.pid`.

- **ScoreBoardFile**—Memory-mapped file in which internal server process information are stored. The ScoreBoardFile is automatically created if your architecture requires it. If this file is automatically created, ensure that no two servers share the same file. The default is `logs/apache_runtime_status`.

- **prefork MPM module**—How the CSPS child processes will operate. When this is configured, child processes are monitored. When necessary, additional child processes are spawned to process incoming SIP requests and responses. When the monitor determines that too few requests and responses are taking place, it tears down some of the idle child processes.
Chapter 4 Configuring the Cisco SIP Proxy Server (CSPS)

Defining the CSPS Configuration File

The maximum and minimum values for the following prefork MPM module directives are dependent on your available platform resources. Modify as required. The prefork module directives are ignored if the server is run in single process mode for debugging purpose as follows: /sipd -DONE_PROCESS.

To configure the prefork module, specify values for the following directives:

- **StartServers**—Number of child processes to create when the CSPS starts. The default is 5.
- **MinSpareServers**—Minimum number of idle child processes (not handling requests). The default is 5.
- **MaxSpareServers**—Maximum number of idle child processes (not handling requests). Idle child processes that exceed this number are torn down. Do not set this parameter to a large number. The default is 10.
- **MaxClients**—Maximum number of simultaneous requests that can be supported; not more than the number of child processes to be created. The default is 20.
- **MaxRequestsPerChild**—Maximum number of requests that a child process can process. This directive is for recovering from memory leaks. If this number is exceeded, the child process will be killed and replaced by a new child process. The default is 0. This value indicates the child process will never be torn down.

Each child process handles udp traffic, ipc traffic and timeouts. Timeouts happen every 50 milliseconds (even if there is no SIP traffic), and the per child MaxRequestPerChild counter is updated. To replace the child processes on the order of hours or days, set this value to 100,000 or 1,000,000.

- **Listen**—Binds the server to specific IP addresses and specifies whether the server should listen to more than one IP address or port. By default, it responds to requests on all IP interfaces, but only on the port specified in the Port directive. The entries can include IP:port or a port, but not IP only. See the following example:

  3000
  12.23.56.78:5060

Configuring the Host-Specific Directives

The host-specific directives define the basic configuration of the CSPS. The basic configuration consists of values used by the main server which responds to requests that are not handled by a virtual host. The host-specific directives define the server access, error logs, and with what frequency the logs rotate.

The directives that define the basic configuration of the CSPS environment are standard Apache directives. If the default for an Apache directive has been changed for the CSPS usage, the new default is documented. For more detail on the Apache directives see www.apache.org.

To configure the host-specific directives, define values for the following (located in the sipd.conf file):

- **Port**—Port on which the CSPS listens. The default is SIP port 5060. If this directive is set to a value less than 1023, the CSPS (sipd daemon) initially must be run as root. This is still true if sipd is to run as a different user or group.
Defining the CSPS Configuration File

- **User** — Name or number of the user to run the sipd process as when sipd is started by the root user. The default is csps.

- **Group** — Name or number of the group to run the sipd process as when sipd is started by the root user. The default is csps.

- **ServerName** — A host name which is used by clients in Request-URIs that is different from the one the server normally recognizes as its own. For example, use sip-proxy.company.com instead of the host’s real name. It is useful for building a server farm and publishing a single hostname for the farm. This directive is optional.

  *Note* The ServerName, if defined, must be a valid DNS name for the host. If the host does not have a resolvable name, use its IP address.

- **HostnameLookups** — Whether client DNS host names or IP addresses are logged. Valid values are On (log host names) or Off (log IP addresses). The default is Off.

- **ErrorLog** — Location of the error log file to which the CSPS logs debug and error messages are stored. The default is `logs/error_log`.

  To automatically rotate error/debug log without having to tear down the CSPS (sipd daemon), use the following examples for Linux and Solaris respectively.

  ErrorLog “/usr/local/sip/bin/rotatelogs /usr/local/sip/log/error_log 86400”

  ErrorLog “/opt/sip/bin/rotatelogs opt/sip/log/error_log 86400”

  Rotation time default is 86400 seconds (24 hours).

- **LogLevel** — Verbosity of messages recorded in the error logs. Valid values (in order of decreasing significance) are:
  - `emerg` — Emergencies, and system is unusable
  - `alert` — Action must be taken immediately
  - `crit` — Critical conditions
  - `error` — Error conditions
  - `warn` — Warning conditions
  - `notice` — Normal but significant condition
  - `info` — Informational
  - `debug` — Debugging messages

  The default is warn.

- **LogFormat** — Format nicknames of logfile. These are used with the CustomLog directive.

  LogFormat "%h %l %u %t "%r" %>s %b "%{Referer}i" "%{User-Agent}i"" combined

  LogFormat "%h %l %u %t "%r" %>s %b" common

  LogFormat "%{Referer}i -> %U" referer

  LogFormat "%{User-agent}i" agent

- **CustomLog** — Location and format of the access log file. The default is `logs/access_log common`.

  For an agent and referer logfiles, use the following directives by removing the comment markers.

  CustomLog logs/referer_log referer

  CustomLog logs/agent_log agent
For a single logfile with access, agent and referer information (combined logfile format), use the following directive by removing the comment markers.

- CustomLog logs/access_log combined

- TransferLog—With what frequency (in seconds) to rotate CSPS logs without having to tear down the CSPS (sipd daemon). To specify a value for this directive, specify the full path of the log file to be rotated and the rotation time.

You can specify a value similar to /user/local/sip/bin/rotatelogs
/user/local/sip/logs/request_log 86400 in this directive to have access records such as a REGISTER request logged to both the access_log and request_log.0974732040 (number extension is calculated and added base on the current time stamp and the specified rotation frequency). If the CustomLog directive has its comment marker removed, access records are logged to the file specified in the TransferLog directive.

Configuring the Cisco SIP Module Core Directives

The CSPS core module implements an RFC 2543 compliant SIP server that can function as a redirect, registrar, or proxy server. If configured to be a proxy, the CSPS can be configured to function as a transaction stateful or stateless server.

To define the CSPS core configuration, define values for the following directives in the CSPS Core module (mod_sip).

- ProxyDomain—DNS domain of the CSPS. The DNS domain suffix must be entered in a standard Fully Qualified Domain (FQDN) format, mydomain.com or company.mydomain.com. There is no default for this directive.

- StatefulServer—Whether the CSPS will be a transaction stateful or transaction stateless server.

When configured to function as a stateful server, the CSPS creates a TCB in which it maintains the transaction state upon receipt of a SIP request.

As a stateful proxy server, from the time a SIP request is received until the final response is sent, the transaction is considered as one. Stateful proxy servers do not originate any SIP requests except for the SIP CANCEL request or an ACK for a non-200 OK final response.

When configured to function as a stateless proxy server, the CSPS forwards every request downstream and every response upstream.

As a stateful redirect server, the CSPS looks in its registry database upon receiving a SIP request and returns a 302 response upstream. As a stateless redirect server, the CSPS returns a final response upon receiving any request and does not forward any response or request.

Valid values are On and Off. The default is On.

- SipResolveLocalContactsInRedirectMode—Whether to return next hop routing information when the CSPS is configured to function as a redirect server. A redirect server typically returns the contact location it knows. However, if this directive is set to On, next hop routing occurs and the contact information is possibly updated before the SIP 3xx response is returned. Valid values are On and Off. The default is Off.

- UseCallerPreferences—Whether to use user-defined or administrator-defined preferences when handling requests. Request handling preferences are controlled by a server administrator but can be overridden by a UAC. Preferences include decisions such as whether to proxy or redirect a request, whether to fork a request (sequential or parallel), whether to recursively search, and to which URI to proxy or redirect a request. Valid values are On (use user-defined preferences) or Off (ignore user-defined preferences). The default is On.
• **Recursive**—Whether the CSPS recursively tries addresses returned in a SIP 3xx redirection response or return the lowest-numbered (best) response. Valid values are On (the server recursively tries addresses on the contact list returned) or Off (the server uses the lowest-numbered response). The default is On.

• **ServerType**—Whether the CSPS functions as a proxy or redirect server. As a proxy server, the CSPS processes and routes SIP requests. As a redirect server, the CSPS provides contact information via SIP redirect responses (3xx). Possible values are Proxy and Redirect. The default is Proxy.

• **MaxForks**—Maximum number of branches that can be forked when the CSPS is configured to function as a stateful server. The range is 1 to 6. The default is 5.

• **NumericUsernameInterpretation**—Lookup order for numeric user information in the Request-URI header field when the “;user=IP/PHONE” parameter is missing.

  The following are valid values:
  - IP_164—Process the Request-URI entries as URLs first and then as E.164 numbers.
  - E164_IP—Process the Request-URI entries as E.164 numbers first and then as URLs.
  - IP—Process the Request-URI entries as URLs only.
  - E164—Process the Request-URI entries as E.164 numbers.

  The default is E164_IP.

• **NumericUsernameCharacterSet**—Set of characters used to determine whether the user information portion of the Request-URI is in a category of users that will be applied to the “NumericUsernameInterpretation” processing step. This set does not apply to any user information parameters. The default is +0123456789.-() (global phone number combinations). For more information on this directive, see the sipd.conf file.

• **OrigUserNameSource**—Origin of the User Name attribute in the accounting request message. Possible values are From and Auth. If the value is From, the user part of the URL in the From SIP header is used to perform authentication and populate standard radius accounting attribute #1 (User Name). If the value is Auth, the user provided for authentication in the authorization or proxy-authorization header is used for authentication and billing. If no proxy-authorization header is present, the user will be taken from the From header in the billing records. The default is Auth.

• **NumExpandAuthUserName**—Apply Number Expansion rules to the UserName received in Authorization or Proxy-Authorization header [On | Off]. Default is On.

• **SrvForFqdnOnly**—Whether to perform DNS Server (SRV) lookups only for hosts that are FQDNs. If the host portion of the Request-URI header field does not contain an IP address, but contains a period, the CSPS determines the host to be an FQDN. Valid values are On (perform DNS SRV lookups only on FQDN hosts) or Off (perform DNS SRV lookups for any host that does not contain a target port). The default is Off.

• **SipT1InMs**—Amount of time (in milliseconds) after which a request is first retransmitted, assuming no response is received. The default is 500.

• **SipT2InMs**—Amount of time (in milliseconds) after which the backoff interval for non-INVITE requests does not increase exponentially. The default is 4000.

• **SipT3InMs**—Default amount of time (in milliseconds) the CSPS waits after receiving a provisional response when processing an INVITE request. This value is used whenever there is no Expires parameter in the INVITE. Otherwise, the value in the Expires parameter is used if it does not exceed SipMaxT3InMs. The default value is 60000.
• **SipMaxT3InMs**—Maximum amount of time (in milliseconds) the CSPS waits after receiving a provisional response when processing an INVITE request. The default value is 180000. If a client includes an Expires value greater than SipMaxT3InMs in an INVITE, the value of SipMaxT3InMs is used instead.

• **SipT4InMs**—Amount of time (in milliseconds) that the TCB is maintained after a final response to a SIP INVITE is proxied. The default is 32000.

• **MaxInviteRetxCount**—Maximum number of times that a SIP INVITE request can be retransmitted by the CSPS. The range is 0 to 6. The default is 6.

• **MaxNonInviteRetxCount**—Maximum number of times that a SIP request other than an INVITE request can be retransmitted. The range is 0 to 10. The default is 10.

• **SharedMemorySize**—Shared memory size to be allocated for transaction control block (TCB). The valid range is 32MB to 512MB. The default is 32 MB.

  **Note**  
  The recommended size is 128 MB.

• **RegistryCleanupRate**—Interval (in milliseconds) at which expired or deleted entries are removed from the registry. The default is 180000 milliseconds.

• **AddRecordRoute**—Whether the CSPS will add the Record-Route header field to an initial SIP INVITE message. The Record-Route header field contains a globally reachable Request-URI that identifies that proxy server. When a proxy server adds the Request_URI to the Record-Route field in SIP messages, the proxy server is kept in the path of subsequent requests for the same call leg. Valid values are On (add the Record-Route field) and Off (do not add the Record-Route field). The default is Off. The ServerType directive must be set to Proxy for this directive to be applied.

• **SipRouteHdrTransportType**—Transport type for routes specified in Route headers of SIP requests handled by the CSPS.

  **Note**  
  If the route contains an explicit transport parameter, this directive is ignored and the transport identified in the route header is used. This can be TCP or UDP. Default is UDP.

• **DiffServValue**—The value (in hex) to mark the Type of Service (TOS) byte of the IP header field of the transmitted SIP packets. This field is in 8 bits (1 byte). Default value is 0x00. DiffServ values and their meaning are specified in RFC2474, RFC2475, RFC2597, RFC2598. The following are the DiffServ values:
  - Expedited Forwarding (EF) queue (RFC2598) value: 0xB8
  - Assured Forwarding (AF) queue (RFC2597) values:

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Drop</td>
<td>0x28</td>
<td>0x48</td>
<td>0x68</td>
</tr>
<tr>
<td>Medium Drop</td>
<td>0x30</td>
<td>0x50</td>
<td>0x70</td>
</tr>
<tr>
<td>High Drop</td>
<td>0x38</td>
<td>0x58</td>
<td>0x78</td>
</tr>
</tbody>
</table>

Some networks may alternatively recognize the Type-Of-Service (RFC1349, RFC1812) bitmasks as follows.
  - Minimize delay: 0x10
Defining the CSPS Configuration File

- Maximize throughput: 0x08
- Maximize reliability: 0x04
- Minimize cost: 0x02

**Note**  
This directive only marks the IP packets to a specified value. Marked packets receive special treatment only if the network’s IP routers/switches are configured to do so.

- **Sip_Token_Port**—Port that will be used by the synchronization server on the CSPS. This port must be the same for all servers in a farm. The default is 22794.
- **Sip_Services_Port**—Port on the synchronization server. The default is 52931.
- **RadiusUserNameAttrAddDomain**—Appends a domain to the user name in the radius User Name attribute (user@domain format). The domain in the From header is used. If the domain is not ProxyDomain (default) or one of the domains in Virtual_Proxy_Domain directive, it will not be appended. Default is On.
- **RadiusRetransmissionInterval**—Time (in milliseconds) between retransmissions to radius server.
- **RadiusRetransmissionCount**—Number of times to retransmit before deciding that the radius server is unreachable.
- **RadiusRetransmissionAfterFailure**—The number of times to retransmit the current request if all the attempts to send the previous radius request end in failure. The default is 0.
- **RadiusRetryTime**—Time (in seconds) before retrying the primary server, if primary radius server is marked out-of-service. Default is 300 seconds (5 minutes).
- **UseIpAddrInPathHeaders**—Controls which IP Address is used in Path headers (Via and Record-Route) when the UseIpAddrInPathHeaders is On. If it is Off, the ServerName is used (if defined). Otherwise, the first value returned from the gethostbyname directive is used. The primary purpose is to control which address is used on multi-homed servers. The default is Off.
- **IPAddrInPathHeaders**—Which IP address will be used in the Via and Record-Route header fields when the UserIpAddrInPathHeaders field is set to On. If an address is not defined in this directive, the first value returned via gethostbyname is used.
- **IgnoreProxyRequire**—If this is On, the server behaves as if any ProxyRequire headers are not present in the request. For example, INVITE contains ProxyRequire:extension-foo, but the proxy has no formal logic to understand extension-foo. RFC 2543 requires that a 420 response to be returned, but with IgnoreProxyRequire configured, the INVITE is processed as if that particular header does not present.
- **SIPStatsLog**—Whether the CSPS will print statistics to the stats_log file. The default is On.
- **SIPStatsInterval**—Interval (in seconds) at which statistics are logged. The default is 3600.
- **SharedMemoryStatsLog**—This flag controls whether debugging for shared memory is to be turned On or Off.
- **SharedMemoryStatsInterval**—This specifies the time interval the log is written to the sharedmem_stats_log file in the logs directory. The default is 5 minutes.
- **SipTcpMaxTCPConnections**—Controls the number of SIP TCP connections that can be open at any time. Default is 128.
Defining the CSPS Configuration File

**Note**
The value of this directive is ignored in favor of using limits enforced by the operating system. For both Linux and Solaris, this limit is 1024 (= FD_SETSIZE) by default. It is not recommended to increase it because it may degrade performance.

- **SipTcpMaxConnectTimeout**—Controls the maximum time for which the server waits to connect to the client. It is specified in units of milliseconds. Default is 1 second (1000ms). The maximum is 10 seconds and the minimum is 150 milliseconds.

- **SipTcpReuseConnection**—Determines whether the TCP connection should be reused for subsequent transactions to the same entity. All SIP entities using TCP for transport to one another should preferably have the same value for this directive. The reason is, if one proxy has this flag On, it continuously tries and reuses the same connection, which the other hop where this flag is turned Off, is being torn down. This results in performance degradation and potential call failures. For best performance, turn this flag On when all elements support it, so less time is spent in opening and closing connections. Default is Off.

**Note**
To interwork with Cisco IOS GWs, set SipTcpReuseConnection to Off. If other entities in the network reuse connections, define persistent connections to those entities in the conf/persistent_tcp.conf file.

- **DebugFlag StateMachine**—Whether to enable the logging of LogLevel information on the operation of the per sipd child SIP state machine to ServerRoot/logs/error_log. Valid values are On (enable logging) or Off (disable logging). The default is Off.

- **DebugFlag Radius**—Whether to enable the logging of LogLevel information for Radius messages to ServerRoot/logs/error_log. Valid values are On (enable logging) or Off (disable logging). The default is Off.

- **DebugFlag Parser**—Whether to enable the logging of LogLevel information on the operation of the per sipd child SIP parser to ServerRoot/logs/error_log. Valid values are On (enable logging) or Off (disable logging). The default is Off.

- **DebugFlag SipTcp**—Whether to enable the logging of LogLevel information on TCP transport of SIP messages by TCP services to ServerRoot/logs/error_log. Valid values are On (enable logging) or Off (disable logging). The default is Off.

**Configuring Cisco SIP Module Standard Directives**

The CSPS contains eight additional modules that can be used to configure a variety of interfaces and additional features. The following sections describe the interfaces and services that can be configured on the CSPS by using modules, including those for call forwarding features on a per user basis.

- Configuring MySQL Database Subscriber Table Interface, page 4-10
- Configuring Number Expansion, page 4-17
- Configuring SIP Access Control, page 4-12
- Configuring Authentication and Authorization, page 4-14
- Configuring Accounting Services, page 4-11
- Configuring Call Forwarding, page 4-16
- Configuring Registry Services, page 4-18
Defining the CSPS Configuration File

- Configuring E.164 to Request-URI Address Translation, page 4-18
- Configuring Next Hop Routing, page 4-21
- Configuring Next Hop Routing, page 4-21
- Configuring H.323 RAS module, page 4-24

Configuring MySQL Database Subscriber Table Interface

The CSPS MySQL module (mod_sip_db_mysql) provides the ability to configure an interface to a MySQL database subscriber table to maintain subscriber records for user authentication, authorization, accounting, and per user call forwarding features.

If a MySQL database subscriber table exists in the network, use directives in the MySQL module to map the field names used by the CSPS to those used in the MySQL database subscriber table. If a MySQL subscriber table does not already exist in the network, create one by using the install_mysql_db script (See Cisco SIP Proxy Server Installation Guide) or cut and paste the subscriber.sql into an existing MySQL database, then start the CSPS with the MySQL interface enabled.

Note

If the GUI-based provisioning system is used, the MySQL database tables are created as part of installing the provisioning system, and the field names used in the tables cannot be modified. See Cisco SIP Proxy Server Installation Guide for details in installing the CSPS with the GUI-based Provisioning System.

Note

For terminating features such as the Call Forwarding features, the "user" portion of the Request-URI is the key to query the MySQL database. For originating features such as Authentication, the UserName from the Authorization, Proxy-Authorization Header, or From header is the key for the MySQL query. In either case, the key may be expanded to a fully expanded E164 number before the MySQL query, depending on the relevant configuration directives.

To configure the interface to the MySQL database subscriber table and map field names used by the CSPS to an existing MySQL subscriber table, specify values for the following directives in the mod_sip_db_mysql module:

- **DB_MySQL**—Enable or disable the interface to the MySQL database. Enabling the interface will establish a TCP connection with the database. Valid values are On (enable the interface) or Off (disable the interface). The default is Off.
- **DB_MySQL_HostName**—Host name or IP address of the system on which the MySQL database resides.
- **DB_MySQL_DB**—Name of the database in which the subscriber table is stored and maintained.
- **DB_MySQL_Username**—Login username to the database account.
- **DB_MySQL_Password**—Login password to the database account.
- **DB_MySQL_SubscriberTable**—Name of the table in which the subscriber entries will be stored.
- **DB_MySQL_Connect_Timeout**—The timeout value (in seconds) when attempt to connect to the MySQL database server. When it expires, the CSPS marks the connection as bad to prevent more child processes from blocking on the connect attempt. The CSPS resets the connection flag as soon as the MySQL server returns online. Default is 3 seconds.
Note

This value should be adjusted according to the traffic load of the server. If the timeout value is too large, more child processes can be blocked.

- **DB_MySQL_XXX_Field**—Name equivalent in an existing MySQL database subscriber table. Use these directives as necessary to map the field names being used by the CSPS to the equivalent entry in an existing MySQL subscriber table.

- **DebugFlag DBMySQL**—Whether to enable the printing of all mod-sip-db-mysql debug messages to logs/error_log. Valid values are On (print messages) or Off (do not print messages). The default is Off.

For information on working with MySQL databases, see www.mysql.com.

### Configuring the GKTMP Interface

The CSPS GKTMP interface module (mod_sip_gktmp) is a translation module that enables the CSPS to translate SIP PDUs to the GKTMP protocol for LNP lookups, 1-800 and 1-900 number translations, and end-point resolutions. When the GKTMP interface is configured, as the CSPS processes are started, the processes initiate a TCP connection with a Network Application Manager (NAM) server via the GKTMP interface.

To configure the GKTMP interface, specify values for the following directives located in the mod_sip_gktmp module:

- **GktmpConnection**—Whether the GKTMP interface is enabled or disabled. Possible values are On (interface is enabled) or Off (interface is disabled). The default is Off.

- **MasterServerHostname**—Hostname of the primary NAM server.

- **MasterServerIpAddress**—IP address of the primary NAM server.

- **MasterServerPort**—Destination port number of the primary NAM server and LNP lookup services.

- **SecondaryServerHostname**—Hostname of the secondary NAM server.

- **SecondaryServerIpAddress**—IP address of the secondary NAM server.

- **SecondaryServerPort**—Destination port number of the secondary NAM server.

- **GktmpTransportType**—Transport type for routes specified in GKTMP responses received by the CSPS. It can be TCP or UDP. Default is UDP.

- **Debug Flag**—Whether to enable the printing of mod_sip_gktmp module debug messages to logs/error_log. Valid values are GKTMP On (print messages) or GKTMP Off (do not print messages). The default is Gktmp Off.

- **DebugFlag Gktmp API**—Whether to enable the printing of mod_sip_gktmp API debug messages to logs/error_log. Valid values are On (print messages) or Off (do not print messages). The default is Off.

### Configuring Accounting Services

The CSPS can be configured to generate and forward transaction or call information to a Radius accounting server. This information is forwarded by the CSPS in the form of a Radius Accounting-Request message that contains standard billing information such as the user name, IP address of the proxy server that set up the call, the message status type, type of port, session time, the ID of the end point that is called and the ID of the end point that calls.
When the accounting services are enabled and the interface to the Radius server is configured, the CSPS creates and sends the Accounting records to the Radius server described as follows.

- The start record is written when a 200 OK is received by the CSPS for an INVITE.
- The stop record is written only when BYE is received by the CSPS.

Accounting services can be configured with the following directives:

- **Accounting**—Whether or not the SIP server will log accounting information on a Radius account server. Possible values are On (accounting is enabled) and Off (accounting is disabled). The default is Off.
- **AccountingRecordFormat**—Record format used for accounting. Currently, Radius is the only valid option.
- **AccountingTimeFormat**—Whether timestamps is in local or GMT time.
- **PrimaryRadiusAcctIp**—IP address or host name of the primary Radius server to be used for accounting.
- **PrimaryRadiusAcctPort**—Destination port number of the primary Radius server to be used for accounting.
- **PrimaryRadiusAcctSecret**—Secret text string shared between the CSPS and the primary Radius account server.
- **SecondaryRadiusAcctIp**—IP address or host name of the secondary Radius server to be used for accounting.
- **SecondaryRadiusAcctPort**—Destination port number of the secondary Radius server to be used for accounting.
- **SecondaryRadiusAcctSecret**—Secret text string shared between the CSPS and the secondary Radius account server.
- **AcctIncludeSIPHeader**—SIP header to be sent in VSA #1 (AVPair) within radius accounting messages. They are sent as they are received by the proxy, that is, with complete header line (from the 200 OK for the start request, and from the BYE for the stop request). A maximum of 50 headers can be configured in sipd.conf. For Radius, this directive is included as the value of Cisco AVPair 1 and attribute name, sip-hdr.
- **DebugFlag**—Turns the Authentication module debug message On or Off. Default is Off.

## Configuring SIP Access Control

The SIP Access Control directives determine access to the CSPS.

- **Allow**—Determines which hosts can access an area of the server. Access can be controlled by hostname, IP Address, IP Address range, or by other characteristics of the client request captured in environment variables.

  The first argument to this directive is always the *from* hostname. The subsequent arguments can take two different forms (*all* and *host*). If **Allow from all** is specified, all hosts are allowed access. This is subject to the configuration of the Deny and Order directives (see below). To allow only particular hosts or groups of hosts to access the server, the host can be specified in any of the following formats.

  - A partial domain-name
    
    ```
    Allow from company.com
    ```

    In this example, hosts whose names match or end in this string are allowed access.
- A full IP address
  Allow from 10.1.2.3

  In this example, an IP address of a host is allowed access.

- A partial IP address
  Allow from 10.1

  This example shows the first 1 to 3 bytes of an IP address for subnet restriction.

- A network/netmask pair
  Allow from 10.1.0.0/255.255.0.0

  This example shows a network a.b.c.d, and a netmask w.x.y.z. for finer subnet restriction.

- A network/nnn CIDR specification
  Allow from 10.1.0.0/16

  This example is the same as the previous example except the netmask consists of nnn high-order 1 bits.

- **Deny**—Allows access to the server with restrictions that base on hostname, IP address, or environment variables. The arguments for the Deny directive are identical to the arguments for the Allow directive.

- **Satisfy**—Determines access policy for both types of access control (Allow and Deny) and authentication checks.

  The parameter can be either *all* or *any*. If *all* is specified, the sending host should be *allowed* and authenticated. If *any* is specified, the sending host can be granted access if it passes either access control allow or authentication check. In either case, authentication module must be turned on.

  **Note** If authentication module is turned off, the authentication check is considered successful.

- **Order**—Controls the default access state and the order in which Allow and Deny directives are evaluated. Valid orders are:

  - Deny,Allow—The Deny directives are evaluated before the Allow directives. Access is allowed by default. Any client which does not match a Deny directive or does match an Allow directive will be allowed access to the server.

  - Allow,Deny—The Allow directives are evaluated before the Deny directives. Access is denied by default. Any client which does not match an Allow directive or does match a Deny directive will be denied access to the server.

  **Note** Keywords can only be separated by a comma, and whitespace is NOT allowed between them. In all cases, every Allow and Deny statement is evaluated.

In the following example, all hosts in the company.com domain are allowed access, and all other hosts are denied access.

Order Deny,Allow
Deny from all
Allow from company.com
In the following example, all hosts in the company.com domain are allowed access, except for the hosts which are in the foo.company.com subdomain, who are denied access. All hosts not in the company.com domain are denied access because the default state is to deny access to the server.

Order Allow,Deny
Allow from company.com
Deny from foo.company.com

If the Order in the last example is changed to Deny,Allow, all hosts will be allowed access. Regardless of the ordering of the directives in the configuration file, the Allow from company.com will be evaluated last and will override the Deny from foo.company.com. All hosts not in the company.com domain will also be allowed access because the default state will change to Allow.

Configuring Authentication and Authorization

The CSPS Authentication module, mod_sip_authen, ensure that users are authenticated before a transaction is serviced by the CSPS. Authentication between an endpoint and the CSPS can be provided via HTTP Digest Authentication (described in RFC2617), HTTP Basic Authentication (described in RFC 2617) or via CHAP-Password. The HTTP Digest Authentication and HTTP Basic Authentication can be performed at the CSPS or at an appropriate Radius Server. The CHAP-Password Authentication is always performed by a Radius Server.

Authentication of a user is based on the user name which is extracted from the Authorization header, the Proxy-Authorization header, or the From header as defined by the OrigUserNameSource directive. This is regardless of whether the authentication takes place at the CSPS or remotely at the Radius Server.

The UserName can be expanded to a fully expanded E164 number before the user is located in the MySQL database, or before passing the UserName to the Radius Server. If the user name is taken from the From header, the expansion is dependent on the user type and the NumericUserNameInterpretation directive. If the user name is taken from the Authorization or Proxy-Authorization header, the expansion is dependent on the NumExpandAuthUserName directive. Additionally, the domain of the user, as determined by the host portion of the From, Authorization or Proxy-Authorization header, may be included in the MySQL query and/or Radius request based on the value of the RadiusUserNameAttrAddDomain directive.

Expansion is based on the number expansion rules which are defined in the number expansion module.

To enable authentication and authorization, specify values for the following directives in the mod_sip_authen module:

- **Authentication**—Whether the proxy server will require users be authenticated before servicing their transactions. Valid values are On (user must be authenticated) or Off (user does not have to be authenticated). The default is Off.

  **Note** User authentication will not occur in the following case because access control is already satisfied: Access control is being used; the hostname or IP address of the sender is covered by a corresponding Allow directive; the Satisfy directive is set to Any instead of All.

- **AuthRealm**—Realm used in authentication response headers. The default is CISCO.

- **AuthServer**—The server on which the User Authentication takes place (Radius Server or CSPS). Possible values are Radius or Proxy. Default is Proxy.

- **AuthScheme**—Type of authentication method to be used when users are required to obtain authentication before receiving service from the CSPS. Possible values are HTTP_Digest, HTTP_Basic or HTTP_CHAP. The default is HTTP_Digest.
• **AuthDigestQop**—The value of the QOP parameter to be included in a Digest Challenge to the user. This value is used in Authentication Response headers. Possible values are None, Auth or Auth-Int. None indicates no QOP parameter is included in the challenge, Auth indicates QOP="auth", Auth-Int indicates QOP="auth-int". Default is None.

• **AuthDigestAlgorithm**—The value of the algorithm value to be included in a Digest Challenge to the user. This value is used in Authentication Response headers. Possible values are MD5 or MD5-sess. MD5 indicates ALG="md5" and MD5-sess indicates ALG="md5-sess". Default is MD5.

• **AuthConsumeProxyAuthHdr**—Specifies whether the proxy should consume or leave the Proxy-Authorization header before forwarding a request downstream. Values are On (consume) and Off (leave). The default is On as recommended in RFC 2543. This causes the proxy to consume the header. When the header is needed by a downstream device to identify the originator of the request, the directive should be set to Off.

• **AuthAllow3rdPartyRegistration**—Checks unauthorized redirection of calls by a third-party registration. If the value of this directive is set to Off, the user name in the To header is matched with the user name in the From or Authorization header. If the user names in these two headers do not match, registration is rejected. This is independent of the authentication scheme which can be Basic, Digest or CHAP. The default is Off.

• **AuthAllow3rdPartyInvite**—Allows or disallows 3rd party INVITEs for all forms of authentication (Digest/Basic/CHAP). When this is turned on, user in the From header can be different from user used for authentication. When this is turned off, the user in the From header in the INVITE message must match user used for authentication. Default is On.

• **RadiusAuthSkew**—Amount of time (in seconds) that a challenge is valid. The default is 30.

• **PrimaryRadiusAuthIp**—IP address or host name of the primary Radius server to be used for user authentication.

• **PrimaryRadiusAuthPort**—Destination port number of the primary Radius server to be used for user authentication.

• **PrimaryRadiusAuthSecret**—Secret text string shared between the CSPS and the primary RADIUS authentication server.

• **SecondaryRadiusAuthIp**—IP address or host name of the secondary Radius server to be used for user authentication.

• **SecondaryRadiusAuthPort**—Destination port number of the secondary Radius server to be used for user authentication.

• **SecondaryRadiusAuthSecret**—Secret text string shared between the CSPS and the secondary Radius authentication server.

• **AcctIncludeSIPHeader**—SIP header to be sent in VSA #1 (AVPair) within radius accounting messages. They are sent as they are received by the proxy, that is, with complete header line (from the 200 OK for the start request, and from the BYE for the stop request). A maximum of 50 headers can be configured in sipd.conf. For Radius, this directive is included as the value of Cisco AVPair 1 and attribute name, sip-hdr.

### Configuring Pre-Authentication Query

Another authentication module, mod_sip_rpms.c, sends pre-authorization query for new INVITE to a RPMS server. If the module is turned on in sipd.conf, and if the INVITE's previous hop matches any in the PreAuthPreviousHopList, a pre-auth query Radius message is sent on behalf of the INVITE to a RPMS server in the RPMS_ServerIpPortSecret list. If the CSPS receives an Accept message from the RPMS server, the new INVITE is processed as normal. If the CSPS receives a Reject message, a 408
(temporarily unavailable) is returned to the caller. If a time lapse occurs before receiving any response from the RPMS server, the next entry in the RPMS_ServerIpPortSecret list is tried. If all RPMS servers failed to respond, the INVITE is processed as normal, as if the module is Off. If any RPMS server respond, that server is marked as good for the next call to do pre-auth query with.

Since pre-auth query messages are Radius messages, the new rpms module uses the Radius module to build and send pre-auth query Radius messages. If you want to see debug messages, turn on the following debug flags:

- DebugFlag StateMachine
- DebugFlag Radius
- DebugFlag RPMS

The various Radius timers work the same way for the pre-auth query message as they control how Radius module behaves in general.

- RadiusRetransmissionInterval—2000
- RadiusRetransmissionCount—2
- RadiusRetransmissionAfterFailure—0
- RadiusRetryTime—300

To enable pre-authorization query, specify values for the following directives in the mod_sip_rpms.c module:

- PreAuthorization—Whether the proxy server do pre-authorization for new INVITE. Possible values are On and Off. Default is Off.
- PreAuthRequestType—Whether the proxy server supports query. Possible value is Query.
- RPMS_ServerIpPortSecret—Specifies RPMS server's IP address, port number, and secret (password). You can specify up to ten RPMS servers in this format. The list of RPMS servers are queried in descending order until the proxy server gets either an AccessAccept or AccessReject response, or till all the servers fail to respond.

In the case where the Proxy received a response (Accept or Reject) from a RPMS server, that server is used first for subsequent queries if the time from the radius directive, RadiusRetryTime, has not expired since the time the very first RPMS server from this list failed to respond. In the case where all RPMS servers fail to respond to a previous query, or the RadiusRetryTime expires, the next query is sent to the very first server again. Whether a query is starting with the first RPMS server in the list, or with a middle RPMS server (because it responded to a previous query), query is sent in descending order and stops at the last server in this list.

- PreAuthPreviousHop—Specifies an IP address, hostname, domain, or keyword ALL, etc. for Hops. The maximum is 100 entries. The format of an entry is the same as that in the access list. If a new INVITE's Previous Hop IP matches any entries from this list, the proxy server does pre-authorization by sending a pre-auth query to a RPMS server. A first match, using the same matching rules as those of the access list, results in the proxy server doing a pre-auth query if the PreAuthentication flag is On.
- DebugFlag RPMS—Turns the RPMS module debug messages On or Off. Possible values are On and Off.

## Configuring Call Forwarding

The CSPS Call Forwarding module (mod_sip_call_forward) enables or disables call forwarding functionality on the CSPS.
Call forwarding support requires a MySQL database.

To configure call forwarding support on the CSPS, specify values for the following directives located in the mod_sip_call_forward module:

- **CallForwardUnconditional**—Whether to forward calls unconditionally. Possible values are On (forward calls unconditionally, based on contents of subscriber record) or Off (do not forward any calls unconditionally).

- **CallForwardNoAnswer**—Whether to forward calls when a call is not answered. Possible values are On (forward calls when a call is not answered within CallForwardNoAnswerTimer time, based on contents of subscriber record) or Off (do not forward any calls when a call is not answered).

- **CallForwardBusy**—Whether to forward calls when a SIP 486 Busy Here response is received. Possible values are On (forward busy calls, based on contents of subscriber record) or Off (do not forward any busy calls).

- **CallForwardUnavailable**—Whether to forward calls when a UAC is unavailable. When this is enabled, calls for users that exist in the subscriber database but do not have an active registration, or that have a valid registration but do not respond within CallForwardUnavailableTimer time, will be forwarded to the call forward unavailable location. Possible values are On (forward calls, based on contents of subscriber record, when called device is unavailable) or Off (do not forward any calls, when called device is unavailable).

- **CallForwardNoAnswerTimer**—Time (in milliseconds) after which to forward a call when a call is unanswered. The default is 24000. The setting for this directive is valid only when the CallForwardNoAnswer directive is enabled.

- **CallForwardUnavailableTimer**—Time (in milliseconds) after which to forward a call when a UAC is unavailable. The default is 24000. The setting for this directive is only valid when the CallForwardUnavailable directive is enabled.

- **AddDiversionHeader**—Whether the CC-Diversion header will be included in the SIP messages. Inclusion of the CC-Diversion header enables conveying call-redirection information during a call setup phase. Possible values are On (include the CC-Diversion header) and Off (exclude the Diversion header). The default is On if call forwarding is enabled.

- **DiversionHeaderName**—Name used for Diversion headers generated by this proxy. Possible values are Diversion, CC-Diversion, CC-Redirect. Default is CC-Diversion.

### Configuring Number Expansion

The CSPS supports global number expansion plans. To use number expansion plans, you must enable the function and then define the global number expansion plan.

#### Enabling or Disabling Number Expansion

To enable number expansion, specify values for the following directives in the Number Expansion module (mod_sip_numexpand) module:

- **Cisco_Numexpand**—Whether to use number expansion on the CSPS. Valid values are On (use number expansion) or Off (do not use number expansion). The default is On.

- **DebugFlagNumexpand**—Whether to enable the printing of all number expansion-related debug messages. Valid values are On (print messages) or Off (do not print messages). The default is Off.
Configuring the Numbering Plan

After enabling number expansion, define the global number plan. A period (.) may be used as a wildcard to represent any digit. To define the global number plan, use the following steps:

**Step 1**
Start the number plan by assigning it a unique identifier by specifying the following:

```
<NumberPlan Global>
```

where ID is the unique identifier of the number plan (for example, global).

**Step 2**
Specify the number plan using one or more NumExp directives:

```
NumExp <unexpanded-pattern> <expanded-pattern>
</NumberPlan>
```

The following is an example of a global number expansion plan:

```
<NumberPlan Global>
  NumExp 2.... +1919392....
  NumExp 7.... +1408527....
  NumExp 8... 5555...
</NumberPlan>
```

Configuring E.164 to Request-URI Address Translation

The ENUM module (mod_sip_enum) is a translation module that enables the CSPS to translate an E.164 number (or any number in a number plan) into a list of Request-URIs via DNS lookup.

To enable ENUM translation, define values for the following directives located in the mod_sip_enum module:

- **Cisco_Enum**—Whether E.164 to Request-URI translation is enabled. Possible values are On (translate) or Off (do not translate). The default is Off.
- **Cisco_Enum_Domain**—Private search domain for a private ENUM number plan. If a Request-URI user begins with the plus (+) character, this directive is not used because the plus character indicates that the number is part of a global ENUM number plan, which is e164.arpa.
- **Cisco_Enum_Global_Domain**—Domain to use when the Request-URI user begins with a plus (+) character (indicating a global domain) or to use when a value is not specified for the Cisco_Enum_Domain directive. The default is e164.arpa.
- **DebugFlag Enum**—Whether to enable the printing of mod_sip_enum API debug messages to logs/error_log. Valid values are On (print messages) or Off (do not print messages). The default is Off.

Configuring Registry Services

The CSPS Registry module (mod_sip_registry) enables the CSPS to process requests from UACs that have been configured to register their location. When registry services have been configured, the CSPS can do the following:

- Add a new registration
- Delete an existing registration
- Update an existing registration
- Delete all registrations of a user
Enabling for Disabling Registration Services

To enable the registry services and establish a registration database, define values for the following directives in the mod_sip_registry module:

- **Cisco_Registry** — Whether registry services are enabled or disabled on the CSPS. Possible values are On (function as a registrar server) or Off (do not function as a registrar server).
- **Cisco_Registry_Shared_Memory_Address** — Memory location of the registration table. The default value is 0x30000000.
- **Cisco_Registry_Rendezvous_Name** — Rendezvous name of the database containing registration information. The default is registry_db.
- **Cisco_Registry_Rendezvous_Name** — Location of the registration database. The default is <ServerRoot>/logs.
- **Cisco_Registry_Remote_Update_Port** — Port number of the registration database server for all members of a farm of servers. The value for this directive must be the same for all members of the farm. The default is 22913.
- **Cisco_Registry_Farm_Members** — Names of the CSPS included as members of the registry farm. This list of names must be defined on all of the CSPSs identified as part of a farm. If it is not defined or omitted, the registry farm contains only the local host. The local host should not be included in its own list. It is included implicitly. The entries to this directive are used by the sysadmin_csps_regroute tool in relation to the registry farm.

See “Configuring the SIP Proxy Server in a Farm” section on page 4-26 for details on configuring a CSPS farm. Also see Cisco_Routing_Farm_Members directive.

- **Cisco_Registry_Max_DB_Age_on_Boot** — Maximum age (in seconds) of the database backing store file from starting of the system. If the age of the database backing store file exceeds this age, the file will be deleted. The default is 86400 seconds (24 hours). This value must be greater than the registry ageout value.

- **DebugFlag Registry** — Whether to enable the printing of mod_sip_registry module debug messages to logs/error_log. Valid values are On (print messages) or Off (do not print messages). The default is Off.

- **Cisco_Registry_Use_Virtual_Proxy_Host** — Whether Virtual Proxy Host services are enabled or disabled on the CSPS. Possible values are On (function as a Virtual Proxy Host) or Off (do not function as a Virtual Proxy Host).

- **Virtual_Proxy_Domain** — Unique DNS domain which proxy handles as the domain of this VirtualProxyHost. Virtual_Proxy_Domain must be different than ProxyDomain. See “Configuring the Cisco SIP Module Core Directives” section on page 4-5 for detail on ProxyDomain field. This directive is required. Example of format is as follows.

  - somedomain.org, or foo.com

- **Virtual_Proxy_Server_Name** — Unique name which proxy handles as ServerName of this VirtualProxyHost. Virtual_Proxy_Server_Name must be different from the ServerName of the proxy. This directive is optional. Example of format is as follows.

  - usa.somedomain.org, or usa.foo.com
Defining the CSPS Configuration File

- **Virtual_Proxy_Server_IP**—Unique IP address which proxy handles as IP address of this VirtualProxyHost. Virtual_Proxy_Server_IP must be different from the actual IP address of Proxy. This directive is optional. Example of format is as follows.
  206.23.2.2, or 192.168.2.2.

Configuring Virtual Proxy Hosts

After enabling virtual proxy host, define one or more virtual proxy host entries. The CSPS supports up to 10 virtual proxy hosts. To configure a virtual proxy host, use the following steps:

**Step 1**
Assign a unique identifier to the virtual proxy host as follows.

```xml
<VirtualProxyHost ID>
```

ID is a unique identifier for this VirtualProxyHost configuration stanza.

**Step 2**
Define values for the following directives:
- `Virtual_Proxy_Domain`
- `Virtual_Proxy_Server_Name`
- `Virtual_Proxy_Server_IP`

**Step 3**
At the end of the entry, specify the following:

```xml
</VirtualProxyHost>
```

Example of a Virtual Proxy Host entry

```xml
<VirtualProxyHost 1.1>
Virtual_Proxy_Domain foo.bar
Virtual_Proxy_Server_Name usa.foo.bar
Virtual_Proxy_Server_IP 61.12.1.1
</VirtualProxyHost>
```

Configuring Static Registry Entries

After enabling registry services, a static registry entry may be defined for each default contact point for each user.

**Note**
Static registrations are not required in general since most user agents register all contacts dynamically.

Prior to CSPS version 1.2, the static registry entries were defined in the main server configuration file, sipd.conf. Starting from version 1.2, an administrator should place these entries in a separate file. If version 1.2 or higher is installed as an upgrade from version 1.0 or 1.1, the administrator should move static entries from old sipd.conf to this new file. The CSPS provides a default template file, sip_registry.conf-dist, which has definitions for each stanza entry and examples. The administration tool, sysadmin_csps_regroute, can import the configuration file. See “Configuring the SIP Proxy Server in a Farm” section on page 4-26.

To configure a registry contact entry, use the following steps:

**Step 1**
Start the registry entry by assigning it a unique identifier by specifying the following:

```xml
<StaticRegistry ID>
```

ID is the unique identifier of the registry entry.
Step 2 Define values for the following directives:

- **Static_Registry_User_Type**—Type of destination pattern for the entry. Valid values are IP and PHONE. The default is IP.
- **Static_Registry_User**—Registration key.
- **Static_Registry_Contact**—Default contact point for the user indicated in the **Static_Registry_User** directive. Valid value is `user@FQDN` or `user@ipaddress`.
- **Static_Registry_Contact_User_Type**—Type of contact point for the user defined in the user portion of the value in the **Static_Registry_Conact** directive. Possible values are IP or PHONE. If a value is not specified, the lookup order specified in the **NumericUsernameInterpretation** directive will be used to determine the type of the contact point.
- **Static_Registry_ContactPort**—Port on the contact point to be used. The default is 5060.
- **Static_Registry_TransportProtocol**—Transport protocol to use when contacting this contact. The valid values are UDP and TCP. The default is UDP.
- **Static_Registry_ConactAge**—Length of validity of this registry contact. Possible values are Permanent (default), or the UNIX system time (secs since 1970). This directive is optional.
- **Static_Registry_Delete_or/Add**—Whether to add or delete this registry entry. Possible values are Add or Delete. The default is Add.

Step 3 End the entry by specifying the following:

```xml
</StaticRegistry>
```

The following is an example of a registry entry for an IP phone:

```xml
<StaticRegistry 10.1>
Static_Registry_User_Type IP
Static_Registry_User jdoe
Static_Registry_Contact 001515551212@mycompany.com
Static_Registry_Contact_User_Type PHONE
Static_Registry_ContactPort 5060
Static_Registry_TransportProtocol UDP
Static_Registry_ContactAge Permanent
Static_Registry_Delete_or_Add ADD
</StaticRegistry>
```

### Configuring Next Hop Routing

The CSPS Routing module (mod_sip_routing) provides the ability to perform next hop route lookups for final Request-URIs via static route entries. The static routes are configured by parsing directives such as the destination pattern, transport protocol, and target address into a table entry that can be retrieved by the CSPS when necessary.

#### Enabling Next Hop Routing

To enable next hop routing and establish a routing database, define values for the following directives in the mod_sip_routing module:

- **Cisco_Routing**—Whether next hop routing is enabled or disabled on the CSPS. Possible values are On (next hop routing is enabled) or Off (next hop routing is disabled). The default is On.
- **Cisco_Routing_Shared_Memory_Address**—Memory location of the routing table. The default value is 0x35000000.
• **Cisco_Routing_Rendezvous_Name** — Rendezvous name of the database containing routing information. The default is routing_db.

• **Cisco_Routing_Rendezvous_Directory** — Location of the routing database. The default is `<ServerRoot>/logs`.

• **Cisco_Routing_Remote_Update_Port** — Port number of the routing database server for all members of a farm of servers. The value for this directive must be the same for all members of the farm. The default is 22913.

• **Cisco_Routing_Use_Domain_Routing** — Whether to use domain next hop routing. Domain next hop routing uses the host portion of the Request-URI as the key in obtaining the next hop or hops for a request. Valid values are On (use domain routing) or Off (do not use domain routing). The default is Off.

• **Cisco_Routing_Max_DB_Age_on_Boot** — Maximum age (in seconds) of the database backing store file when the system is started. If the age of the database backing store file exceeds this age, the file will be deleted. The default is 86400 seconds (24 hours). This value must be greater than the registry ageout value.

  This field must contain an appropriate value if an external routing process exists to keep the database current. Otherwise, the externally populated routes may be deleted on system start up.

• **Cisco_Routing_Farm_Members** — Names of the CSPSs to include as members of the routing farm. This list of names must be defined on all of the CSPSs identified as part of a farm. If it is not defined or omitted, the routing farm contains only the local host. The local host should not be included in its own list. It is included implicitly. The entries to this directive are used by the sysadmin_csps_regroute tool in relation to the routing farm.

  See “Configuring the SIP Proxy Server in a Farm” section on page 4-26 for details on configuring a CSPS farm. Also see Cisco_Registry_Farm_Members directive.

• **Cisco_Routing_Wildcard_Expand_Length** — When the destination pattern type is phone (see Static_Route_Type in the following section), this directive specifies the maximum length for a destination pattern wildcard entry. If a wildcard entry is longer than the maximum length specified in this directive, the wildcard character (*) is removed from the entry, and the destination pattern entered by the user is not expanded to a variable length. Then the portion of the destination pattern entry (without the wildcard) is added or deleted in the routing database as specified in the Static_Route_Delete_or_Add directive for this destination pattern entry. The default length is 25 digits. This length is also used when this directive is omitted in the system.

• **DebugFlag Routing** — Whether to enable the printing of all mod-sip-routing module debug messages to `logs/error_log`. Valid values are On (print messages) or Off (do not print messages). The default is Off.

### Configuring Static Route Entries

After enabling next hop routing, define a route for each single next hop.

Prior to CSPS version 1.2, the static registry entries were defined in the main server configuration file, sipd.conf. Starting from version 1.2, an administrator should always place these entries in a separate file. If version 1.2 or later is installed as an upgrade from version 1.1 or 1.0, the administrator should move static entries from old sipd.conf to this new file. The CSPS provides a default template file, sip_routing.conf-dist, which has definitions for each stanza entry and examples. The administration tool, sysadmin_csps_regroute, can import the configuration file. See “Configuring the SIP Proxy Server in a Farm” section on page 4-26.
When defining a static route destination pattern, use the period (.) as a wildcard to represent any digits and the asterisk (*) as a wildcard to represent one or more digits. The period and the asterisk may be used only at the end of the destination pattern.

To configure a static route entry, use the following steps:

Step 1  Assign the route entry a unique identifier by specifying the following:
<StaticRoute ID>

ID is the unique identifier of the route entry.

Step 2  Define values for the following directives:

- **Static_Route_DestinationPattern**—Destination pattern for the route. The value in this field depends on the route type defined in the Static_Route_Type directive.
  
  For E.164 numbers, this is an International Direct Dial (IDD) that has trailing digits represented by a period (.) as the wildcard digit. The length of the IDD and the number of wildcard digits are specific to the dial plan they represent. For example, each wildcard IDD destination pattern that has a different length will be a unique dial plan.

  An example of an E.164 destination pattern is +1555666...., which dictates to match all number in the United States at area code 555 with a 666 exchange. If the plus sign (+) does not begin the pattern, it will not be internationally unique.

- **Static_Route_Type**—Type of destination pattern. Valid values are IP or PHONE. The default is IP.

  When a wildcard (*) is used at the end of a destination pattern string of which the type is phone, the string is expanded with periods (.) to a length specified in Cisco_Routing_Wildcard_Expand_Length directive (see previous section). For example, the wildcard destination pattern is 408* and the length specified in Cisco_Routing_Wildcard_Expand_Length directive is 10, the expanded patterns are 408, 408., 408.., 408..., till the length of the last expanded pattern is 408....... which is the length of 10 digits.

- **Static_Route_NextHop**—Next hop for the destination pattern defined in the Static_Route_DestinationPattern directive. Valid values are the FQDN or IP address of the next hop.

- **Static_Route_NextHopPort**—Port on the next hop to be used. The default is 5060. A value of zero (0) specifies that a DNS-SRV lookup should be performed on the associated Static_Route_NextHop.

- **Static_Route_TransportProtocol**—Transport protocol to use for this route. The valid values are UDP and TCP. The default is UDP.

- **Static_Route_Priority**—Priority of this static route. This value is a 16-bit integer. The route with the highest priority will be used over a route with a lower priority. Priority 1 (default) is the highest. Multiple routes with the same priority can be selected based on their weight. See Static_Route_Weight directive.

- **Static_Route_Weight**—Weight of this static route. The higher the weight, the higher the probability of selection over routes with the same priority. This value is a 16-bit integer. The default is 1. See Static_Route_Priority directive.

- **Static_Route_TechPrefix_Action**—If the destination number pattern starts with a tech prefix, for example 003#, this directive informs the CSPS to take one of the following actions: STRIP or INCLUDE the tech prefix in the user portion of the request-uri in the forwarded INVITE message.
Chapter 4: Configuring the Cisco SIP Proxy Server (CSPS)

Defining the CSPS Configuration File

Note

The CSPS only recognizes tech-prefix format that starts with digits and ends with a number sign (#) character. The # sign is not a legal character for SIP URLs. CSPS has special handling for them. This also applies to Cisco SIP gateways, but not to other servers to which CSPS may forward messages containing the # sign. The action for all routes of the same destination number pattern must be the same. Default action is INCLUDE.

- Static_Route_Delete_or_Add—Whether to add or delete this static route entry. Valid values are Add or Delete. The default is Add.

Step 3

End the entry by specifying the following:

```xml
</StaticRoute>
```

The following is an example of a static route definition:

```xml
<StaticRoute 1>
  Static_Route_Destination Pattern 001555666....
  Static_Route_Type PHONE
  Static_Route_NextHop sip_gw1.mycompany.com
  Static_Route_NextHopPort 5060
  Static_Route_TransportProtocol UDP
  Static_Route_TechPrefix_Action INCLUDE
  Static_Route_Delete_or_Add Add
</StaticRoute>
```

Configuring H.323 RAS module

The RAS module allows communications between a SIP proxy server and a H.323 gatekeeper. The CSPS can send ASN.1 encoded RAS LRQ message to a provisioned H.323 gatekeeper and receive LCF, LRJ or RIP messages returned by the gatekeeper.

To configure H.323 RAS support on the CSPS, specify values for the following directives located in the mod_sip_ras module:

- **RASModule**—Turn the RAS module on/off.
- **RASAcceptLCF**—Choice of gatekeeper. The values are as follows.
  - First—take the first valid LCF message from any gatekeepers.
  - Best—wait for the best LCF message within the LRQ time window. This LCF is from the gateway that has the lowest cost and highest priority.
- **RASTimeoutInterval**—Time (in milliseconds) that the proxy waits for a single response from a gatekeeper. If the specified timeout expires, the proxy tries another gatekeeper within the same cluster. Default is 300 milliseconds. This is specifically used in sequential mode.
- **RASTransportType**—Specifies the transport type used in the Route entry learned from a Gatekeeper via RAS. The transport type can be TCP. Default is UDP.
- **RASLRQMethod**—The methods used to send LRQ messages, as follows.
  - Sequential—LRQ is sent sequentially and will wait for the response in time specified in RASTimeoutInterval directive.
  - Blast—LRQs are sent in a parallel manner before detecting any response from the gatekeepers.
• **RASLRQWindow**—Maximum time (in milliseconds) that the proxy waits for all responses returned from gatekeepers contacted. Default is 3000 milliseconds. This value can be overridden by the RIP message.

• **RASTimeToLive**—The TTL(TimeToLive) value in the RAS LRQ non-standard message body. Default is 6 (in hops).

• **RASAllowTranslation**—Set canMapAlias field in LRQ message to True or False. The values are as follows.
  - On—Sets canMapAlias field to True. This allows gatekeeper to replace the dialed phone number (destinationInfo field).
  - Off—Sets canMapAlias field to False. This does not allow the gatekeeper to change the original dialed phone number. If the gatekeeper replaces addressing information from the LRQ and canMapAlias field is False, the gatekeeper rejects the LRQ.

• **RASGateKeeperCluster**—Gatekeepers can be grouped into clusters. This directive sets priorities for gatekeeper clusters so the CSPS can query the clusters in priority order. The gatekeeper child directive in this directive contains two parameters (IP address and port number of the target gatekeeper). Valid priority value range for each cluster is from 1 to 65535.

  Sample cluster configurations are as follows.

  - Cluster 1
    
    &lt;RASGatekeeperCluster 1&gt;
    RASGatekeeper gatekeeper1.company.com 1719
    RASGatekeeper gatekeeper2.company.com 1719
    &lt;/RASGatekeeperCluster&gt;

  - Cluster 2
    
    &lt;RASGatekeeperCluster 2&gt;
    RASGatekeeper gatekeeper3.company.com 1719
    RASGatekeeper gatekeeper4.company.com 1719
    &lt;/RASGatekeeperCluster&gt;

• **RASGateKeeper**—Specify an individual gatekeeper in a gatekeeper cluster. Maximum number of gatekeepers in a cluster is 5. The values are as follows.

  - IP address or FQDN
  - Port number

• **RASDefaultTechPrefixAction**—Default action taken by the CSPS when a tech prefix exists and no specific local rules applies. The values are as follows.

  - Strip—Remove the tech prefix in the outgoing INVITE message or 302.
  - Include—Include the tech prefix in the outgoing INVITE message or 302.

• **RAS TechPrefix**—The technology prefix used when the dialed number matches the specified number pattern. This directive include three parameters as shown in the following examples.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Dialed-number pattern</th>
<th>Tech-prefix string</th>
<th>Include/Strip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
<td>1919321...</td>
<td>001#</td>
<td>INCLUDE</td>
</tr>
<tr>
<td>Example 2</td>
<td>1919456...</td>
<td>002#</td>
<td>STRIP</td>
</tr>
</tbody>
</table>
Sample directive
RASTechPrefix 1919456... 002# STRIP

- **DebugFlag**—Turn the RasAPI and RAS module debug messages on or off. This allows display of how the CSPS handles incoming and outgoing messages.
  - RasAPI—RAS message encoding/decoding and sending/receiving.
  - RAS—Module specific tasks such as module configuration, socket creation, message conversion and routes insertion.

## Configuring the SIP Proxy Server in a Farm

Before setting up a SIP Proxy Server Farm, do the following and observe the notes below.

### Step 1
Make sure the IP network among the farm members is working.

### Step 2
Enter the host names of all servers (excluding the local server) as the value for `Cisco_Registry_Farm_Members` directive in `sipd.conf`.

Example of a farm member entry for a farm consisting of host1.cisco.com, host2.cisco.com, and host3.cisco.com is as follows.

- **In sipd.conf on host1.cisco.com:**
  ```
  Cisco_Registry_Farm_Members *host2.cisco.com, host3.cisco.com*
  ```
- **In sipd.conf on host2.cisco.com:**
  ```
  Cisco_Registry_Farm_Members *host1.cisco.com, host3.cisco.com*
  ```
- **In sipd.conf on host3.cisco.com:**
  ```
  Cisco_Registry_Farm_Members *host1.cisco.com, host2.cisco.com*
  ```

### Step 3
Enter the host names as the value for `Cisco_Routing_Farm_Members` directive in `sipd.conf`.

Example of a farm member entry for a farm consisting of host1.cisco.com, host2.cisco.com, and host3.cisco.com is as follows.

- **In sipd.conf on host1.cisco.com:**
  ```
  Cisco_Routing_Farm_Members *host2.cisco.com, host3.cisco.com*
  ```
- **In sipd.conf on host2.cisco.com:**
  ```
  Cisco_Routing_Farm_Members *host1.cisco.com, host3.cisco.com*
  ```
- **In sipd.conf on host3.cisco.com:**
  ```
  Cisco_Routing_Farm_Members *host1.cisco.com, host2.cisco.com*
  ```

The registry and routing farms can contain different sets of members, but the configuration should be consistent across all farm members. For example, the registry can contain all three members (host1, host2, host3), and the routing farm can contain only the local member. In this case, `Cisco_Registry_Farm_Members` is configured on all members as shown in the previous example, and the comment marker for `Cisco_Routing_Farm_Members` is removed on all servers.
Chapter 4      Configuring the Cisco SIP Proxy Server (CSPS)

Configuring the CSPS in a Farm

Note

It is recommended to designate one farm member as the master and the rest as slaves. The sysadmin_csp_regroute tool can be executed on the master member to load any initial registry or routing seed files for the farm, and to perform any registry or routing updates for the farm. The sysadmin_csp_regroute tool should not be used on slave members to seed or update registry or routing information. The master member synchronizes the slave members initially. They synchronize with each other whenever an update occurs.

When farming the CSPS, the system clock must be synchronized among all farm members. It is recommended to use the Network Time Protocol (NTP) to achieve the synchronization.

NTP is used to synchronize the time of a computer client or server with another server or reference time source, such as a radio, satellite receiver or modem. It provides accuracy within a millisecond on LANs and up to a few tens of milliseconds on WANs relative to Coordinated Universal Time (UTC) via a Global Positioning Service (GPS) receiver, as an example.

Typical NTP configurations utilize multiple redundant servers and diverse network paths in order to achieve high accuracy and reliability. Some configurations include cryptographic authentication to prevent accidental or malicious protocol attacks and some provide automatic server discovery using IP multicast. See URL http://www.eecis.udel.edu/~ntp/ for more detail on NTP.

Configuring the Master Member for the Farm

To configure the master member for a farm, do the following steps:

---

**Step 1**

Prepare two seed data files for registry and routing separately for the CSPS.

For additional information, see template files sip_registry.conf-dist and sip_routing.conf-dist.

Note

For upgrades from CSPS 1.0 or CSPS 1.1, copy static entries from sipd.conf to two separate files for registry and routing.

Use sysadmin_csp_regroute to import the files, or use the provisioning system GUI to seed the registry and routing entries. See Chapter 3, “Provisioning System GUI”.

Note

Only use the seed data to start the master member for the first time or when databases are not valid. Seed data is not required between normal shutdown and start. A database is invalid when it has corrupted data or when it has expired. If the databases get corrupted, stop the CSPS, remove the database files (registry_db and routing_db in the opt/sip/logs or usr/local/sip/logs directory for Solaris and Linux respectively), then seed the database after the CSPS starts. If the database has expired, the farm member automatically clears its old database and gets the most current database from another farm member.

**Step 2**

Do periodic backup of the database files. Use sysadmin_csp_regroute to export the database contents of both registry and routing databases. These database files can be used as seed data files if the databases are corrupted. See Chapter 5, “Maintaining the Cisco SIP Proxy Server (CSPS)” , Backing Up and Restoring CSPS, page 5-12.
Configuring the Slave Member for the Farm

Do not place any seed entries in the default template files or sipd.conf, and do not use sysadmin_csps_regroute on a slave member to update a slave database to avoid conflicts.

**Note**

Use sysadmin_csps_regroute to read on a slave member.

Configuring IPSec

**Note**

The following procedures should only be performed by system administrators.

IPSec is a framework of open standards developed by the Internet Engineering Task Force (IETF). It provides security for transmission of sensitive information over unprotected networks such as the Internet. IPSec resides in the network layer, protecting and authenticating IP packets between participating IPSec devices (peers), such as Cisco SIP Proxies.

IPSec provides the following network security services which are optional. In general, local security policy dictates the use of one or more of these services.

- **Data Confidentiality**—The IPSec sender can encrypt packets before transmitting them across a network.
- **Data Integrity**—The IPSec receiver can authenticate packets sent by the IPSec sender to ensure that the data has not been altered during transmission.
- **Data Origin Authentication**—The IPSec receiver can authenticate the source of the IPSec packets sent. This service is dependent upon the data integrity service.
- **Anti-Replay**—The IPSec receiver can detect and reject replayed packets.

**Note**

The term data authentication is generally referred to data integrity and data origin authentication. In this document, it also includes anti-replay services, unless otherwise specified.

IPSec for the CSPS is supported on the following network configurations:

- Solaris system to Solaris system via manual keying
- Linux system to Linux system via the following:
  - manual keying
  - IKE via configuration files
  - IKE via mod_ipsec_auto.c
- Solaris system to Linux system via manual keying
Configuring IPSec for the Cisco SIP Proxy on Solaris Platforms

IPSec on Solaris is a suite of security protocols that secure communication channels and ensure that only authorized parties can communicate on those channels.

The CSPS implementation of Solaris IPSec requires Authentication and Data Encryption. By default, Authentication is installed with the Solaris Operating Environment 2.8. Data Encryption is available on a Solaris supplemental CD or may be downloaded without charge from http://wwws.sun.com/software/solaris/encryption/.

To verify Data Encryption has been installed, determine the existence of /kernal/strmod/encrdes and the /kernel/strmod/encr3des files.

To configure IPSec for the CSPS on a Solaris platform, you must complete the following tasks:

• Configure the system security policy
• Install the Authentication and Data Encryption security keys

Additional details are available at: http://wwws.sun.com/software/solaris/encryption/
Maintaining the Cisco SIP Proxy Server (CSPS)

This chapter contains the following information:

- Starting and Stopping the CSPS, page 5-1
- Working with Logs, page 5-10
- Backing Up and Restoring CSPS, page 5-12

See Chapter 1, “Product Overview”, “CSPS Components” section on page 1-4 for detail on CSPS components.

Starting and Stopping the CSPS

This section describes how to start, stop, and restart CSPS by using the `sip` script. If SNMP is used, CSPS can also be started, stopped, restarted, and gracefully restarted via SNMP using CIAgent. See Chapter 7, “CIAgent”.

The `sip` script, newly offered with CSPS 2.0, provides the functionality in the `sipdctl` script in previous releases. The `sip` script is generated from running `csps_setup` during installation of CSPS. If CSPS is installed without using `csps_setup`, a default `sip` script is generated. See Cisco SIP Proxy Server (CSPS) Installation Guide for installation detail.

The default `sip` script assumes CSPS is installed without the provisioning system. When `csps_setup` is used, a new `sip` script is created, taking into account that all the components are installed. The location of the script is `<ServerRoot>/bin/sip`. Scripts to start and stop each individual component exist in the same directory. It is suggested not to use the component scripts individually. The `sip` script should be used to invoke all the component scripts in the appropriate order and with the necessary precondition checking.

The following sections describe the procedures for starting and stopping CSPS by using the `sip` script. The actions and results of using this script vary depending on which components are installed on the system. Two configurations are illustrated. The first configuration has all components exist on the system (default configuration for CSPS 2.0, set up by using option 1 of `csps_setup` during installation). The second configuration only has CSPS server exist on the system, and the provisioning system is not used (standard configuration supported for CSPS 1.x and also CSPS 2.0).

**Note**

- Before starting the CSPS, be sure to properly install CSPS and all desired components. See Cisco SIP Proxy Server (CSPS) Installation Guide for detail.
If errors occur when the CSPS starts, stops, restarts, or gracefully restarts, the corresponding error messages are displayed in /var/log/messages (Linux) or on the screen (Solaris). Additional details can be found by viewing the log files for each process. See “Working with Logs” section on page 5-10.

**CSPS with All Components Installed (including Provisioning)**

**Starting the CSPS**

To start the CSPS, use the following steps.

**Step 1** Execute the **sip** script with the **start** argument, as follows.

```
[ /usr/local/sip/bin ] ./sip start
```

The following display appears.

```
Starting pserver:                                         [ OK ]
Starting license manager:                                  [ OK ]
Starting spa:                                              [ OK ]
Starting sipd:                                             [ OK ]
```

**Note** For Linux, the default location for the script is /usr/local/sip/bin. For Solaris, the location is /opt/sip/bin.

If the server initializes without error, the following appears in the file /var/log/messages (for Linux) or on the screen (for Solaris).

```
pserverctl: /usr/local/sip/bin/pserverctl start: pserver started
sip: Starting pserver:  succeeded
lmctl: /usr/local/sip/bin/lmctl start: licenseMgr started
sip: Starting license manager:  succeeded
spactl: /usr/local/sip/bin/spactl start: spa started
spactl: /usr/local/sip/bin/spactl start: Waiting for sipd.conf from spa..
spactl: .
spactl: /usr/local/sip/bin/spactl start: sipd.conf written
sip: Starting spa:  succeeded
sipdctl: Version of CSPS        : 2.0.x.x - Experimental Version
sipdctl: Version in Config file : 2.0.x.x - Experimental Version
sipdctl: Software release version of CSPS validated successfully with your license
sipdctl: License validated successfully
sipdctl: This is Permanent license, with Infrastructure functionality
sipdctl: /usr/local/sip/bin/sipdctl start: sipd started
sip: Starting sipd:  succeeded
```

**Step 2** To verify the CSPS is running properly, issue the following command to view all csps processes:

```
[ /usr/local/sip/bin ] ps -ef | grep "/sip/bin"
```

An output similar to the following appears.

```
csps  4040  1 0 17:38 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
csps  4054  1 0 17:38 ?  00:00:00 /usr/local/sip/bin/licenseMgr /u
csps  4057  4040 0 17:38 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
csps  4058  4057 0 17:38 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
```
Starting and Stopping the CSPS

There should be 4 pserver processes, 3 licenseMgr processes, and 3 spa processes. By default, there should be 7 sipd processes. In this example, the first sipd process, with the parent process id as 1, is the parent sipd. The other sipd processes are the TCP I/O process and 5 child processes. Sip_Services is an additional process required to maintain synchronization among local and remote farm members.

**Stopping the CSPS**

To stop all CSPS processes, use the following steps.

**Step 1**

Issue the following command:

```
[/usr/local/sip/bin] ./sip stop
```

The following display appears.

```
Stopping sipd: [ OK ]
Stopping spa: [ OK ]
Stopping license manager: [ OK ]
Stopping pserver: [ OK ]
```

Output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris).

```
sipdctl: Waiting for process to stop.
sipdctl: .
sipdctl: /usr/local/sip/bin/sipdctl stop: sipd stopped
sipdctl: Waiting for process to stop.
sipdctl: .
sipdctl: /usr/local/sip/bin/sipdctl stop: Sip_Services stopped
sip: Stopping sipd: succeeded
spactl: Waiting for process to stop.
spactl: .
spactl: /usr/local/sip/bin/spactl stop: spa stopped
sip: Stopping spa: succeeded
lmctl: Waiting for process to stop.
lmctl: .
```

```
lmctl: /usr/local/sip/bin/lmctl stop: licenseMgr stopped
sip: Stopping license manager: succeeded
pserverctl: Waiting for process to stop.
pserverctl: .
pserverctl: /usr/local/sip/bin/pserverctl stop: pserver stopped
sip: Stopping pserver: succeeded
```
Starting and Stopping the CSPS

Step 2  At this point, all CSPS processes are stopped and the server is no longer able to process calls. This may be verified by issuing the following command:

```
[/usr/local/sip/bin] ps -ef | grep "/sip/bin"
```

csp5 16421 15876 0 09:47 pts/0 00:00:00 grep sip/bin

If pserver, licenseMgr, spa, sipd processes or Sip_Services processes still exist, it is advisable to stop them manually by using the Unix command, `kill`.

Restarting the CSPS

To restart (stop and then start) all CSPS processes, use the following steps.

Step 1  Issue the following command:

```
[/usr/local/sip/bin] ./sip restart
```

The following display appears.

```
Stopping sipd: [ OK ]
Stopping spa: [ OK ]
Stopping license manager: [ OK ]
Stopping pserver: [ OK ]
Starting pserver: [ OK ]
Starting license manager: [ OK ]
Starting spa: [ OK ]
Starting sipd: [ OK ]
```

As shown above, this command stops all server processes, then starts them. An output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris):

```
sipdctl: Waiting for process to stop.
sipdctl: /usr/local/sip/bin/sipdctl stop: sipd stopped
sipdctl: Waiting for process to stop.
sipdctl: /usr/local/sip/bin/sipdctl stop: Sip_Services stopped
sip: Stopping sipd: succeeded
spactl: Waiting for process to stop.
spactl: /usr/local/sip/bin/spactl stop: spa stopped
sip: Stopping spa: succeeded
lmctl: Waiting for process to stop.
lmctl: /usr/local/sip/bin/lmctl stop: licenseMgr stopped
sip: Stopping license manager: succeeded
pserverctl: Waiting for process to stop.
pserverctl: /usr/local/sip/bin/pserverctl stop: pserver stopped
sip: Stopping pserver: succeeded
pserverctl: /usr/local/sip/bin/pserverctl start: pserver started
sip: Starting pserver: succeeded
lmctl: /usr/local/sip/bin/lmctl start: licenseMgr started
spactl: /usr/local/sip/bin/spactl start: spa started
spactl: /usr/local/sip/bin/spactl start: Waiting for sipd.conf from spa..
spactl: /usr/local/sip/bin/spactl start: sipd.conf written
sip: Starting spa: succeeded
sipdctl: Version of CSPS : 2.0.x.x - Experimental Version
```
Chapter 5      Maintaining the Cisco SIP Proxy Server (CSPS)

Starting and Stopping the CSPS

sipdctl: Version in Config file : 2.0.x.x - Experimental Version
sipdctl: Software release version of CSPS validated successfully with your license
sipdctl: License validated successfully
sipdctl: /usr/local/sip/bin/sipdctl start: sipd started
sip: Starting sipd: succeeded

Step 2

To verify that CSPS is running properly, issue the following command to view all csps processes:

```
[/<usr/local/sip/bin]> ps -ef | grep "/sip/bin"
```

An output similar to the following appears.

```
[/<usr/local/sip/bin>]
```

```
csps 4216  1  0 17:58 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
  csps 4225  1  0 17:58 ?  00:00:00 /usr/local/sip/bin/licenseMgr /u
  csps 4232 4225  0 17:58 ?  00:00:00 /usr/local/sip/bin/licenseMgr /u
  csps 4233 4232  0 17:58 ?  00:00:00 /usr/local/sip/bin/licenseMgr /u
  csps 4235 4216  0 17:58 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
  csps 4236 4235  0 17:58 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
  csps 4240 4235  0 17:58 ?  00:00:00 /usr/local/sip/bin/pserver -c /u
  csps 4241  1  0 17:58 ?  00:00:00 /usr/local/sip/bin/spa /usr/loc
  csps 4245  1  0 17:58 ?  00:00:00 /usr/local/sip/bin/Sip_Services
  csps 4252 4241  0 17:58 ?  00:00:00 /usr/local/sip/bin/spa /usr/loc
  csps 4253 4252  0 17:58 ?  00:00:00 /usr/local/sip/bin/spa /usr/loc
  csps 4264  1  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4266 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4268 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4269 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4270 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4271 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  csps 4276 4264  0 17:58 pts/1 00:00:00 /usr/local/sip/bin/sipd
  root 4279 1387  0 18:00 pts/1 00:00:00 grep /bin/sipd
```

All the process ids are different from those that were used before issuing the restart. During the time between the stop and start, the server cannot process any calls.

Gracefully Restarting the CSPS

Graceful restart provides a mechanism to prompt spa to write a new sipd.conf file, then the parent sipd reread the configuration file (sipd.conf), tear down the child sipd processes as they become idle, and spawn new child processes with the new configuration.

Graceful restart is usually used when a configuration change is made and the change is to be put in service. To gracefully restart the CSPS, use the following steps.

Step 1

Issue the following command:

```
[/<usr/local/bin>]/sip graceful
```

The following display appears.

```
Gracefully restarting pserver: [ OK ]
Gracefully restarting license manager: [ OK ]
Gracefully restarting spa: [ OK ]
Gracefully restarting sipd: [ OK ]
```
An output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris):

```
pserverctl: /usr/local/sip/bin/pserverctl graceful: pserver (pid 3749 3769 3770 3775) already running
sip: Gracefully restarting pserver: succeeded
lmctl: /usr/local/sip/bin/lmctl graceful: licenseMgr (pid 3764 3766 3767) already running
sip: Gracefully restarting license manager: succeeded
spactl: Waiting for process to stop.
spactl: /usr/local/sip/bin/spactl stop: spa stopped
spactl: Wait 3 seconds before restarting the application...
spactl: /usr/local/sip/bin/spactl start: Waiting for sipd.conf from spa..
spactl: .
spactl: /usr/local/sip/bin/spactl start: sipd.conf written
sip: Gracefully restarting spa: succeeded
spactl: /usr/local/sip/bin/spactl start: sipd.conf written
spactl: /usr/local/sip/bin/spactl start: sipd.conf written
sip: Gracefully restarting sipd: succeeded
sipdctl: /usr/local/sip/bin/sipdctl graceful: sipd gracefully restarted
sip: Gracefully restarting sipd: succeeded
```

Step 2 Issue the following command to view all csps processes:

```
[/usr/local/sip/bin] ps -ef | grep "sip/bin"
```

The following process listing shows the original pserver and licenseMgr processes as they are not affected by a graceful restart. The spa processes are restarted to force the writing of a new sipd.conf file. The parent sipd process, the original Sip_Services, and the TCP I/O sipd process remain the same as from the previous start of the server. All other sipd child processes have been restarted and have new process ids.

```
[/usr/local/sip/bin] ps -ef | grep "sip/bin"
```

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>GROUP</th>
<th>COMMAND</th>
<th>STATE</th>
<th>Command Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>4216</td>
<td>csps</td>
<td>root</td>
<td>pserver</td>
<td>0:17</td>
<td>/usr/local/sip/bin/pserver -c /u</td>
</tr>
<tr>
<td>4225</td>
<td>csps</td>
<td>root</td>
<td>licenceMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4232</td>
<td>csps</td>
<td>root</td>
<td>licenseMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4233</td>
<td>csps</td>
<td>root</td>
<td>licenseMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4235</td>
<td>csps</td>
<td>root</td>
<td>licenseMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4236</td>
<td>csps</td>
<td>root</td>
<td>licenseMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4240</td>
<td>csps</td>
<td>root</td>
<td>licenseMgr</td>
<td>0:17</td>
<td>/usr/local/sip/bin/licenseMgr /u</td>
</tr>
<tr>
<td>4264</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4266</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4334</td>
<td>csps</td>
<td>root</td>
<td>spa</td>
<td>0:17</td>
<td>/usr/local/sip/bin/spa /usr/loca</td>
</tr>
<tr>
<td>4337</td>
<td>csps</td>
<td>root</td>
<td>Sip_Services</td>
<td>0:17</td>
<td>/usr/local/sip/bin/Sip_Services</td>
</tr>
<tr>
<td>4344</td>
<td>csps</td>
<td>root</td>
<td>Sip_Services</td>
<td>0:17</td>
<td>/usr/local/sip/bin/Sip_Services</td>
</tr>
<tr>
<td>4345</td>
<td>csps</td>
<td>root</td>
<td>Sip_Services</td>
<td>0:17</td>
<td>/usr/local/sip/bin/Sip_Services</td>
</tr>
<tr>
<td>4357</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4358</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4359</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4360</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4361</td>
<td>csps</td>
<td>root</td>
<td>sipd</td>
<td>0:17</td>
<td>/usr/local/sip/bin/sipd</td>
</tr>
<tr>
<td>4370</td>
<td>root</td>
<td>root</td>
<td>grep</td>
<td>0:13</td>
<td>grep /sip/bin</td>
</tr>
</tbody>
</table>

Note When there is a server configuration change, perform a graceful restart to activate the new configuration without dropping any existing or pending calls. When the **sip graceful** command is used, the sipd daemon (parent process) remains alive and it kills the child processes as soon as they are not processing calls. Call processing is not interrupted as a result.

If the TCP I/O process of CSPS becomes unresponsive, the parent sipd performs its own graceful restart (up to five times) to activate the TCP I/O process. If the graceful restart fails to activate the TCP I/O process, you can perform graceful restart manually after one minute.
CSPS Installed without Provisioning

Starting the CSPS

To start the CSPS, use the following steps.

---

**Step 1**

Execute the `sip` script with the `start` argument, as follows:

```
[/usr/local/sip/bin] ./sip start
Starting sipd:
```

```
[ OK ]
```

**Note**

For Linux, the default location for the script is `/usr/local/sip/bin`. For Solaris, the location is `/opt/sip/bin`.

If the server initializes without error, the following appears in the file `/var/log/messages` (for Linux) or on the screen (for Solaris).

```
sipdctl: Version of CSPS        : 2.0.x.x - Experimental Version
sipdctl: Version in Config file : 2.0.x.x - Experimental Version
sipdctl: Software release version of CSPS validated successfully with your license
sipdctl: License validated successfully
sipdctl: This is Permanent license, with Infrastructure functionality
sipdctl: /usr/local/sip/bin/sipdctl start: sipd started
sip: Starting sipd:  succeeded
```

**Step 2**

To verify the CSPS is running properly, issue the following command to view all `csp` processes:

```
[/usr/local/sip/bin] ps -ef | grep "/sip/bin"
```

An output similar to the following appears.

```
csp 16394 1 4 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16396 1 1 09:40 pts/0 00:00:00 /usr/local/sip/bin/Sip_Services
```

```
csp 16397 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16399 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16400 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16401 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16402 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16408 16394 0 09:40 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

```
csp 16410 15876 0 09:54 pts/0 00:00:00 grep sip/bin
```

By default, there should be 7 sipd processes. In this example, the first sipd process, with the parent process id as 1, is the parent sipd. The other sipd processes are the TCP I/O process and 5 child processes. `Sip_Services` is an additional process required to maintain synchronization among local and remote farm members.

---

Stopping the CSPS

To stop all CSPS processes, use the following steps.

---

**Step 1**

Issue the following command:

```
[/usr/local/sip/bin] ./sip stop
Stopping sipd:
```

```
[ OK ]
```
Output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris):

```
sipdctl: Waiting for process to stop.
sipdctl: /usr/local/sip/bin/sipdctl stop: sipd stopped
sip: Stopping sipd: succeeded
```

### Step 2
At this point, all CSPS processes are stopped and the server is no longer able to process calls. This may be verified by issuing the following command:

```bash
[ /usr/local/sip/bin ] ps -ef | grep "/sip/bin"
csps 16421 15876 0 09:47 pts/0 00:00:00 grep sip /bin
```

If sipd processes or Sip_Services processes still exist, it is advisable to stop them manually by using the Unix command, `kill`.

---

### Restarting the CSPS

To restart (stop and then start) all CSPS processes, use the following steps.

### Step 1
Issue the following command:

```bash
[ /usr/local/sip/bin ] ./sip restart
```

An output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris):

```
sipdctl: Waiting for process to stop.
sipdctl: /usr/local/sip/bin/sipdctl stop: sipd stopped
sip: Stopping sipd: succeeded
sipdctl: Version of CSPS : 2.0.x.x - Experimental Version
sipdctl: Version in Config file : 2.0.x.x - Experimental Version
sipdctl: Software release version of CSPS validated successfully with your license
sipdctl: License validated successfully
sipdctl: This is Permanent license, with Infrastructure functionality
sipdctl: /usr/local/sip/bin/sipdctl start: sipd started
sip: Starting sipd: succeeded
```

### Step 2
To verify that CSPS is running properly, issue the following command to view all csps processes:

```bash
[ /usr/local/sip/bin ] ps -ef | grep "/sip/bin"
```

An output similar to the following appears.

```
csps 16454 1 0 09:53 pts/0 00:00:00 /usr/local/sip/bin/sipd
```

---

---

---
All the process ids are different from those that were used before issuing the restart. During the time between the stop and start, the server cannot process any calls.

---

Gracefully Restarting the CSPS

Graceful restart provides a mechanism to prompt the parent sipd process to reread the configuration file (sipd.conf), tear down the child sipd processes as they become idle, and spawn new child processes with the new configuration.

Graceful restart is usually used when a configuration change is made and the change is to be put in service. To gracefully restart the CSPS, use the following steps.

**Step 1**

Issue the following command:

```
[/usr/local/bin] ./sip graceful
```

An output similar to the following appears in /var/log/messages (Linux) or on the screen (Solaris):

```
sipdctl: /usr/local/sip/bin/sipdctl graceful: sipd gracefully restarted
sip: Gracefully restarting sipd: succeeded
```

**Step 2**

Issue the following command to view all csps processes:

```
[/usr/local/sip/bin] ps -ef | grep "sip/bin"
```

The following process listing shows the original parent sipd process, the original Sip_Services, and TCP I/O sipd process. They are from the previous start of the server. All other sipd child processes have been restarted and have new process ids.

```
psps 16454 1 0 09:53 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 16456 1 0 09:53 pts/0 00:00:00 /usr/local/sip/bin/Sip_Services
psps 16457 16454 0 09:53 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 15876 101 01:10 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 15874 16454 1 01:10 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 16502 16454 0 10:01 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 16503 16454 0 10:01 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 16504 16454 0 10:01 pts/0 00:00:00 /usr/local/sip/bin/sipd
psps 16511 15876 0 10:01 pts/0 00:00:00 grep sip/bin
```

When there is a server configuration change, perform a graceful restart to activate the new configuration without dropping any existing or pending calls. When the `sip graceful` command is used, the sipd daemon (parent process) remains alive and it kills the child processes as soon as they are not processing calls. Call processing is not interrupted as a result.

---

If the TCP I/O process of CSPS ever becomes unresponsive, the parent sipd performs its own graceful restart (up to five times) to activate the TCP I/O process. If the graceful restart fails to activate the TCP I/O process, the user can perform graceful restart manually after one minute.
Working with Logs

During operation, each component of CSPS writes to one or more log files. The log files written by each component are as follows.

- SIP Proxy Server (sipd): access_log, error_log and stats_log
- Provisioning client for sipd (spa): spa_log
- Provisioning Server (pserver): pserver_log
- License manager (lm): licenseMgr_log
- Installation script (csps_setup): csps_setup_log

The default location for all log files is `<ServerRoot>/logs`.

Note: For Linux, `<ServerRoot>` is `/usr/local/sip/`. For Solaris, `<ServerRoot>` is `/opt/sip/`

Viewing and Customizing Logs

All log files are text files which can be viewed with any text editor. In most cases, the content and level of detail of the log files is controlled by configuration directives. The following sections describe these log files.

SIP Proxy Server (sipd): access_log, error_log and stats_log

sipd reuses Apache existing logging facilities with enhancements to selectively enable and disable a particular module or functionality. Access logging, error logging, and statistics logging are configured and controlled by the DebugFlag configuration file directives. These can be modified by using the provisioning system GUI, or by manually editing sipd.conf if not using the GUI. By default, all internal errors are logged to `<ServerRoot>/logs/error_log`, access records are logged to `<ServerRoot>/logs/access_log`, and periodic statistics are written to `<ServerRoot>/logs/stats_log`.

The logs in error_log can contain the following formats, depending on the values of the DebugFlags.

Format 1
This format is printed unconditionally. Logs in this format are usually informational and contain some important error messages.

[Fri Apr 20 21:44:51 2001] [notice] A new Apache child process (27413) has started.

Format 2
This format is printed when a component DebugFlag is turned on. For instance, if the StateMachine DebugFlag directive is turned on, a call trace similar to the following example is logged to the error_log file.

[Fri Apr 13 22:29:37 2001] sip_protocol.c(4322) Received 291 bytes UDP packets from 10.80.36.85:50117
REGISTER sip:64.102.93.77 SIP/2.0
Via:SIP/2.0/UDP 10.80.36.85:5060
From:sip:IPphone-2@64.102.93.77
To:sip:IPphone-2@10.80.36.85:5060
Call-ID:c3943000-ee2f9c88-23f9821e-382e3031@10.80.36.85
CSeq:101 REGISTER
Contact:<sip:IPphone-2@10.80.36.85:5060>
Provisioning client for sipd (spa): spa_log

spa logs to spa_log based on the value of the configuration directive DebugLevel in spa.conf. spa.conf is located in <ServerRoot>/conf/prov. To change the level of detail provided in the log file, manually edit the spa.conf file and set the DebugLevel to the desired level, in the order of decreasing verbosity: debug, info, notice, warn, error, crit, alert, emerg.

Provisioning Server (pserver): pserver_log

pserver logs to pserver_log based on the value of the configuration directive DebugLevel in ps.conf. ps.conf is located in <ServerRoot>/conf/prov. To change the level of detail provided in the log file, manually edit the ps.conf file and set the DebugLevel to the desired level, in the order of decreasing verbosity: debug, info, notice, warn, error, crit, alert, emerg.

License manager (lm): licenseMgr_log

lm logs to licenseMgr_log based on the value of the configuration directive DebugLevel in lm.conf. lm.conf is located in <ServerRoot>/conf/prov. To change the level of detail provided in the log file, manually edit the lm.conf file and set the DebugLevel to the desired level, in the order of decreasing verbosity: debug, info, notice, warn, error, crit, alert, emerg.

Installation script (csps_setup): csps_setup_log

csps_setup logs to csps_setup_log. There is no configuration directive for it. It should be consulted to verify the correctness of an installation. In the event of a failures during installation or at first startup, it may provide helpful clues to the problems.
Rotating Logs

Note
This section applies to log files written by sipd only.

The size of the error_log can grow significantly if many DebugFlags are turned on. To better maintain the log file and preserve the server information, utilize the log rotation facility included in the CSPS.

- To turn on error_log rotation, enter the following command in the Custom Logs section of the provisioning system GUI. If provisioning is not used, remove the comment marker in the following line in the sipd.conf file. This instructs the CSPS to rotate the logs/error_log file every 86400 seconds (24 hours).

  ErrorLog "|ServerRoot/bin/rotatelogs ServerRoot/logs/error_log 86400"

- To turn on error_log rotation, enter the following command in the Custom Logs section of the provisioning system GUI. If provisioning is not used, remove the comment marker in the following line in the sipd.conf file. This instructs the CSPS to rotate the logs/access_log file every 86400 seconds (24 hours).

  TransferLog "|ServerRoot/bin/rotatelogs ServerRoot/logs/access_log 86400"

Note
The comment markers of "CustomLog logs/access_log common" and "ErrorLog logs/error_log" should be removed if the comment markers of the above two rotatelogs lines are removed.

Backing Up and Restoring CSPS

In order to be able to recover quickly from catastrophic failures which render one or more servers unavailable, it is advisable to back up all important CSPS data on a regular basis.

The following sections describe the backup procedures, and how to use the backed up data to restore CSPS on a new system.

Backing Up Data

The following steps should be performed on a regular basis.

Note
It is strongly recommended to store the backed up data on a separate system or eternal media for safeguard reasons.

Step 1
If MySQL is run for the provisioning system, or subscriber features, or both, save all the data to a flat file with the following commands.

  mysqlldump -u guest -p --databases sip > <outside_directory/file>
Enter password: <default password is "nobody">

Note
If MySQL is not on the same system as CSPS, run the above commands on the system where MySQL is running.
Step 2 If registries exist, export them to a file by using the sysadmin_csps_regroute tool and make the following selections:

- Select registry (default) or routing database
- Use registry database
- Return to Main menu
- Export current database entries to a configuration <outside_directory/file>

Step 3 If static routes exist, export them to a file by using the sysadmin_csps_regroute tool and make the following selections:

- Select registry (default) or routing database
- Use routing database
- Return to Main menu
- Export current database entries to a configuration <outside_directory/file>

Step 4 Copy the license.conf, persistent_tcp.conf, and sipd.conf files:

a. For Solaris
   - cp /opt/sip/conf/license.conf <outside_directory/file>
   - cp /opt/sip/conf/persistent_tcp.conf <outside_directory/file>
   - cp /opt/sip/conf/sipd.conf <outside_directory/file>

b. For Linux
   - cp /usr/local/sip/conf/license.conf <outside_directory/file>
   - cp /usr/local/sip/conf/persistent_tcp.conf <outside_directory/file>
   - cp /usr/local/sip/conf/sipd.conf <outside_directory/file>

Note If provisioning is used, it is not necessary to back up sipd.conf. It will be regenerated from the information stored in MySQL.

Restoring Backed Up Data

Before restoring CSPS on a system, it is recommended to uninstall CSPS so the system is in a known state, if CSPS is originally installed on this system. Then install CSPS. See Cisco SIP Proxy Server (CSPS) Installation Guide for details on uninstalling and installing CSPS.

Note It is recommended to use csps_setup script for installation. When the license key prompt appears, reference the saved license.conf file in case the license key from the initial installation is not readily available.

Step 1 If MySQL is run for the provisioning system, for subscriber features, or for both, restore the database from the saved flat file with the following commands:

a. Delete existing sip database (if any)
   - mysql -u guest -p
   - Enter password: <default password is "nobody”>
   - at mysql prompt type:
     - drop database sip;
     - quit;

b. Restore previously saved sip database
   - mysql -u guest -p < <mysql_backup_file>
   - Enter password: <default password is "nobody”>
c. Delete any old provisioning server connection data (required only if using provisioning)

```bash
mysql -u guest -p
Enter password: <default password is "nobody">
at mysql> prompt type:
    use sip;
    delete from DBSubscriberTable;
    quit;
```

**Step 2** Import saved registries, if any, to shared memory from the backup file by using the sysadmin_csps_regroute tool and make the following selections:

- `<S>` Select registry (default) or routing database
- `<Y>` use registry database
- `<M>` return to Main menu
- `<I>` Import a configuration <file> with route/registry entries <registry_backup_file>

**Note** If this is a member of a multiple member farm, import to shared memory on the first member only. If an active member already exists with a current registry database, this step may be skipped, since the farm members will synchronize the database automatically.

**Step 3** Import saved static routes to shared memory from the backup file by using the sysadmin_csps_regroute tool and make the following selections:

- `<S>` Select registry (default) or routing database
- `<Z>` use routing database
- `<M>` return to Main menu
- `<I>` Import a configuration <file> with route/registry entries <route_backup_file>

**Note** If this is a member of a multiple member farm, import to shared memory on the first member only. If an active member already exists with a current route database, this step may be skipped, since the farm members will synchronize the database automatically.

**Step 4** Restore the license.conf, persistent_tcp.conf, and sipd.conf files as follows.

**For Solaris**

```bash
cp <license.conf_backup file> /opt/sip/conf/license.conf
cp <persistent_tcp.conf_backup_file> /opt/sip/conf/persistent_tcp.conf
cp <sipd.conf_backup_file> /opt/sip/conf/sipd.conf
```

**For Linux**

```bash
cp <license.conf_backup file> /usr/local/sip/conf/license.conf
cp <persistent_tcp.conf_backup_file> /usr/local/sip/conf/persistent_tcp.conf
cp <sipd.conf_backup_file> /usr/local/sip/conf/sipd.conf
```

**Note** If csps_setup was used to install CSPS, it is not necessary to restore license.conf as it will be generated from the information entered while csps_setup is being run. If provisioning is used, it is not necessary to restore sipd.conf, since it will be regenerated from the information stored in MySQL at the next start, restart, or graceful restart.
Step 5  Gracefully restart CSPS to use the new sipd.conf

a. For Solaris

/opt/sip/bin/sip graceful

b. For Linux

/usr/local/sip/bin/sip graceful
System Administration Tools

This chapter provides an overview and operation instructions of the following system administration tools for the CSPS.

- Administration Tool for Routing and Registry Databases, page 6-1
- Administration Tool for MySQL Databases, page 6-9

The following are covered in this chapter.

- Tool Menus
- Tool Activations
- Tutorial for MySQL Database
- Tool Functionalities and Operations
- Sample Error messages

Administration Tool for Routing and Registry Databases

This tool allows a user to manipulate data in the routing and registry databases without interrupting the proxy server operation.

The current version of the administration tool, sysadmin_csps_regroute, allows an administrator to run multiple copies of the tool. It is strongly recommended that only one copy of the tool is used to update (add, delete, modify) the databases.

For example, if you use copy A to add a registry contact for a user X, and mistakenly use copy B to import an old configuration file which has a different contact for the same user, user X will have both contacts.

Note

It is acceptable to use multiple copies of the tool to read data. However, the tool does not enforce a read/write mode, therefore, it is recommended to use one copy of the tool.
Tool Activation for Routing and Registry Databases

Before activating the tool, set the UNIX environment variable LD_LIBRARY_PATH to point to where libSip_Services_Cli.so resides, usually, in <ServerRoot>/libexec. For example, setenv LD_LIBRARY_PATH $([LD_LIBRARY_PATH]:/opt/sip/libexec.

At the system prompt, type sysadmin_csps_regroute, press Enter.

The following options are available for this command.

- **[-i] configuration file name**—Imports a configuration file into the database.
- **[-p] absolute path and config file name**—Allows a user to specify an absolute path and a file name for a sipd configuration file. If this is not specified, the libexec path in the environment variable, LD_LIBRARY_PATH, is searched. If this is found, libexec is removed, conf/ and a default file (sipd.conf) are appended to this path.
- **[-x] configuration file name**—Allows a user to specify a file name. The regroute tool opens this file for appending the entries to and exporting the entries from the database specified by the -m option (see below). The file format is the standard stanza format of a sipd configuration file for static route and registry entries.
- **[-m] database mode for export/list all 1, 2, 3**—Specifies which database to use. It must be used in conjunction with -x and/or -l option. A value of 1 specifies route database; 2 specifies registry database; 3 specifies both databases. If -m is omitted, both databases are used.
- **[-P] absolute path and config file name**—Exports the routing database and/or the registry database to the MySQL database of the provisioning system. The registry and/or routing database can be specified by using the -m option. Generally, this option is used when converting a legacy CSPS system without provisioning to a CSPS 2.0 system with provisioning.

The following example command exports the route database to the provisioning MySQL database (-P -m 1), and uses the sipd.conf file in /opt/sip/conf.

```
sysadmin_csps_regroute -P -m 1 -p /opt/sip/conf/sipd.conf
```

- **[-l]**—list all entries in the database specified by the -m option.

This tool automatically ends after -i, -x, or -l option is used. Option -m only works with option -x or -l.

Examples

**Example 1**

```
sysadmin_csps_regroute -i xyz.conf
```

**Example 2**

```
sysadmin_csps_regroute -x xyz.conf -l
```

**Example 3**

```
sysadmin_csps_regroute -i abc.conf -x xyz.conf -m 2
```

**Example 4**

```
sysadmin_csps_regroute -p /opt/sip/xyz.conf
```
Tool Menus for Routing and Registry Databases

This administration tool contains multi-level menus as follows.

- **Main Menu**—This appears when the tool is activated. See Figure 6-1.

Figure 6-1  Main Menu for Routing and Registry Databases

![Main Menu for Routing and Registry Databases](image)

- **Database Interface Menu**—This appears when the user selects option D from the Main menu. In this menu shown in Figure 6-2, a user can select to add and delete data in a database, or query the database.

Figure 6-2  Database Interface Menu for Routing and Registry Databases

![Database Interface Menu for Routing and Registry Databases](image)

**Note**

For options in this menu, a wildcard character, *, can be used in place of the default selections where applicable. For example, when you select to add data to the registry database, you are prompted to enter a value for User Type. The default value is Phone. Instead of entering Phone, you can type *.

- **Configuration Menu**—This menu appears when the user selects option S in the Main menu. In this menu shown in Figure 6-3, the user can select to access either the routing database or the registry database.

Figure 6-3  Configuration Menu for Routing and Registry Databases

![Configuration Menu for Routing and Registry Databases](image)
Menu Operation for Routing and Registry Databases

To select an option from a menu, type the character that precedes the option, then press Enter. You can use upper or lower case. For example, type Z and press Enter to select the routing database in the Configuration menu.

Some options require specific type of entries. Required entries are specified in a pair of <> bracket that are shown with the options on the screen. For example, in Figure6-4 as follows, option E requires a registry user ID entry.

Tool Functionalities and Operations for Routing and Registry Databases

This section describes operations of the following functionalities.

- Selecting a database, page 6-5
- Adding or Deleting Data, page 6-5
- Searching the Database, page 6-7
- Displaying Data in the Database, page 6-7
- Importing Configuration File to a Database, page 6-8
- Exporting Database Entries to a Configuration File, page 6-8
- Displaying Memory Information, page 6-9
- Sample Error Messages, page 6-9

After using each functionality, you can select one of the following options:
Chapter 6  System Administration Tools

Administration Tool for Routing and Registry Databases

Selecting a database

This section describes how to select between a routing and a registry database.

Step 1

In the main menu, select option S by typing S and press Enter. A display similar to Figure 6-5 appears.

![Figure 6-5  Configuration Menu for Registry Database]

Step 2

Select option Z to use the routing database, or select option Y to use the registry database.

The text line under the Configuration menu title on the screen indicates which database is selected.

Adding or Deleting Data

This section describes how to add or delete data from the database.

Adding Data

Step 1

In the Main menu, select option D. A display similar to Figure 6-6 appears.

![Figure 6-6  Database Interface Menu for Registry Database]
Step 2  Select option A to add an entry. A display similar to Figure 6-7 appears.

Figure 6-7  Add an Entry Menu for Routing Database

Step 3  Select option E and enter the required data as prompted. The added data appears above option E on the screen.

Deleting Data

This section describes how to delete data from the database.

Step 1  In the Main menu, select option D. A display similar to Figure 6-9 appears.

Figure 6-8  Database Interface Menu for Registry Database

Step 2  Select option D to delete an entry. A display similar to Figure 6-9 appears.
Step 3 Select option E and enter the required data as prompted.

Searching the Database

This functionality can be used to query the database.

Step 1 In the Main menu, select option D to access the Database Interface Menu.
Step 2 Select option S to search the database. A display similar to Figure6-10 appears.

Displaying Data in the Database

This section describes how to list all data in the database.

Step 1 In the Main menu, select option D to access the Database Interface Menu.
Step 2 Select option L. Data display similar to Figure6-11 appears.
This section describes how to import content of a configuration file into a database. This updates both the routing and registry databases with the configuration file content as long as matching entries are found.

**Step 1**  
In the Main menu, select option I by typing i and the configuration file name as shown in the following example.

```
i tlk
```

When the import is done, a message similar to Figure6-12 appears.

**Exporting Database Entries to a Configuration File**

This section describes how to export content of a database to a configuration file. This functionality appends the database entries to a specified file. These entries are exported from the database (routing or registry) as specified by the -m option.

**Step 1**  
In the Main menu, select option X by typing x and the configuration file name as shown in the following example.

```
x tlk
```

When the export is done, a message similar to Figure6-13 appears.
Chapter 6  System Administration Tools

Administration Tool for MySQL Databases

Displaying Memory Information

Step 1  In the Main menu, Select option S to access the Configuration Menu.

Step 2  Select option D. The “Using Shared Memory” message appears.

Sample Error Messages

An error message is enclosed between two lines of asterisks. See sample error message in Figure 6-14.

Figure 6-14 Sample Error Message for Routing and Registry Databases

Administration Tool for MySQL Databases

This section describes the MySQL Database Administration tool. The tool can be used to modify a MySQL server on a local or remote system.

Note

If the administration tool is running remotely, the remote MySQL server is not required to be a Linux or Solaris system. This tool has been successfully tested on Redhat 7.1 and Solaris 2.8 platforms.

This tool can be used for the following tasks:

• Add and remove subscriber records to and from the database.

• Modify the following:
  – A subscriber’s password
  – Call Forwarding on No Answer destination URL
  – Call Forwarding on Busy destination URL
  – Call Forwarding on Unavailable destination URL
  – Call Forwarding Unconditional destination URL
Tool Activation

This section describes how to activate the Administration tool for MySQL databases.

**Step 1**

At the system prompt, activate the tool from the bin directory of the CSPS. For Linux, use the following command:

```
% /usr/local/sip/bin/sysadmin_mysql_user
```

For Solaris, use the following command:

```
% /opt/sip/bin/sysadmin_mysql_user
```

A display similar to Figure 6-15 appears.

**Figure 6-15** Prompt for MySQL Server Hostname

```
We're now going to prompt you for the hostname/IP address of the MySQL server where the subscriber database is installed.
We will also prompt you for the username and password used to login to that MySQL server.

Enter MySQL Server hostname (or IP address): [localhost]
```

**Step 2**

Enter hostname, user name and password accordingly. A display similar to Figure 6-16 appears.

**Figure 6-16** Prompt for MySQL Database Name and Subscriber Table Name

```
You're now going to prompt you for the MySQL database name and the name of the subscriber table in that database.
Unless you know otherwise, you should accept the defaults.

Enter MySQL database name: [sip]

Enter subscriber table name: [subscriber]
```

**Step 3**

Enter database name and table name accordingly. The Main menu appears.

Tool Menus

This tool contains the following menus.

- Main menu—This appears when the tool is activated. See Figure 6-17.
Subscribers attributes submenu—This appears when a selection is made to add a subscriber. See Figure 6-18.

Figure 6-18 Subscriber Attribute Submenu

Menu Operation

To select an option from a menu, type the character that precedes the option, then press Enter. Upper or lower case is allowed. For example, type A and press Enter to add a subscriber to the MySQL databases.

Note

After data is entered in a subscriber record by using the Subscribers attribute submenu, data in a field of the record can be removed by pressing Enter.

Tutorial

The following are demonstrated in this section.

- Adding a Subscriber with a Password, page 6-11
- Adding a Subscriber with Call Forwarding Features, page 6-13
- Listing Subscribers, page 6-17
- Renaming an Existing Subscriber Record, page 6-18
- Removing a Subscriber from a MySQL Database, page 6-20

Adding a Subscriber with a Password

The following steps add a subscriber record for subscriber Alice.
Step 1  In the Main menu, select A.

Step 2  Enter the subscriber’s username and domain name as prompted. Figure6-19 shows these steps.

**Figure6-19  Prompt for Adding Subscriber**

```
Entering main_menu
<table>
<thead>
<tr>
<th>Main menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. (A)dd subscriber</td>
</tr>
<tr>
<td>R. (R)move subscriber</td>
</tr>
<tr>
<td>G. (G)oodify subscriber</td>
</tr>
<tr>
<td>S. (S)how subscriber</td>
</tr>
<tr>
<td>L. (L)ist all subscribers</td>
</tr>
<tr>
<td>X. (X)it</td>
</tr>
</tbody>
</table>

Enter a command:
A
Enter subscriber id (eg. “username”, “+19195551234”): alice
Enter domain name for alice (eg. “cisco.com”):
cisco.com
```

Step 3  In the Subscriber attribute submenu, select P.

Step 4  Enter a password for Alice. Figure6-20 shows these steps.

**Figure6-20  Prompt for Password**

```
<table>
<thead>
<tr>
<th>Subscriber attribute submenu</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. (C)reate/change to a different subscriber record.</td>
</tr>
<tr>
<td>L. User (I)D:</td>
</tr>
<tr>
<td>B. (B)domain:</td>
</tr>
<tr>
<td>P. (P)assword:</td>
</tr>
<tr>
<td>N. Call Forward (W)on Answer:</td>
</tr>
<tr>
<td>U. Call Forward (U)navailable:</td>
</tr>
<tr>
<td>C. Call Forward (C)onditional:</td>
</tr>
<tr>
<td>X. (X)it</td>
</tr>
</tbody>
</table>

Enter a command:
```

Step 5  To exit from the menu, select X.

**Listing Subscriber Information**

The following steps display subscriber information of subscriber Alice that was entered.

Step 1  In the Main menu, select option L.

Step 2  Enter Y to show details for each subscriber record to be displayed, or Enter N to see a summary listing of the subscriber names. Figure6-21, Figure6-22, and Figure6-23 show the actions and results of these steps.
Chapter 6  System Administration Tools

Administration Tool for MySQL Databases

Figure 6-21  Prompt for Listing All Subscribers

```
Entering main menu

A. (O)ld subscriber
B. (R)eview subscriber
M. (O)pen subscriber
S. (C)hange subscriber
L. (C)hange all subscribers
.X(x)it

Enter a command:
```

Figure 6-22  Detailed Single Subscriber Listing

```
Detailed subscriber listing:
user_id:      5001
domain_name: cisco.com
password:    gsc
dest_url.cnf: sip:5002@cisco.com
dest_url.ctbi: 
dest_url.ends: 
dest_url.cfnum: 
dest_url.cnfmsg:
datetime_created: 2002-03-21 12:06:29
datetime_modified: 2002-03-21 12:06:49
```

Figure 6-23  Summary Information on Subscribers

```
user_id  domain_name
5001    cisco.com
Bob     cisco.com
5003    cisco.com
Carol   cisco.com
Alice   cisco.com
5551234 cisco.com
```

Adding a Subscriber with Call Forwarding Features

The following steps add subscriber information to a subscriber Bob. Call forwarding features (No Answer, Busy and Unavailable) will also be added.

**Step 1**  In the Main menu, select option A.
**Step 2**  Enter subscriber ID, Bob, and domain name, Cisco.com, as prompted. Figure 6-24 shows these steps.

Figure 6-24  Prompts for Adding a Subscriber

```
Enter a command:
a
Enter subscriber id (eg. "username", "+19195551234"): bob
Enter domain name for bob (eg. "cisco.com"): cisco.com
```
The Subscriber attribute submenu appears.

Adding Call Forward on No Answer URL

The following sections show how to set options call forward on (user) no answer, call forward on busy, and call forward on (device) unavailable to Bob’s voicemail server.

**Step 1**  
In the Subscriber attribute submenu, select option N for Call Forward on (N)o Answer for user.

**Step 2**  
Enter Call Forward on No Answer URL as prompted. *Figure6-25* shows these steps.

*Figure6-25 Prompt for Call Forward No Answer*

Adding Call Forward on Busy URL

**Step 1**  
In the Subscriber attribute submenu, select option B for Call Forward on (B)usy.

**Step 2**  
Enter Call Forward on Busy URL as prompted. *Figure6-26* shows these steps.

*Figure6-26 Prompt for Call Forward on Busy URL*

Adding Call Forward on Unavailable URL

**Step 1**  
In the Subscriber attribute submenu, select option V for Call Forward Una(V)ailable.

**Step 2**  
Enter Call Forward on Unavailable URL as prompted. *Figure6-27* and *Figure6-28* shows the actions and result of these steps.
Adding a New Subscriber with Call Forwarding (Unconditional)

In this section shows how to add a new subscriber, Carol, who wants to forward all calls from her old phone number to her new SIP user agent. Her subscriber ID is 5551234. This can be configured by setting the Call Forward Unconditional field of Carol’s 5551234 subscriber record to the URL of her SIP user agent.

**Step 1**
In the Subscriber attributes menu, select option C for Create/change to a different subscriber record.

**Step 2**
Enter subscriber ID of Carol (5551234) as prompted.

**Step 3**
Enter Y to the prompt “Add this user_id?” (y/n). Figure 6-29 and Figure 6-30 show the actions and result of these steps.
Step 4  Select option U for Call Forward (U)nconditional.

Step 5  Enter Call Forward Unconditional URL as prompted.

Figure6-31, Figure6-32 and Figure6-33 show the actions and result of these steps. Also, a syntax error detection causes a prompt for the user to enter the destination URL again.

Figure6-31  Prompts for Call Forward Unconditional URL and Error Message

Figure6-32  Call Forward Unconditional URL Re-Entry
**Figure 6-33 User ID Display for Call Forward Unconditional URL Re-Entry**

```
+----------------------------------------+
| Subscriber attribute submenu           |
| C. (C)reate/change to a different subscriber record. |
|   I. User (ID): 5551234                |
|   D. (D)omain: cisco.com               |
|   P. (P)assword:                       |
|   N. Call Forward <N>o Answer:        |
|   B. Call Forward <B>usy:             |
|   U. Call Forward <U>nonAvailable:    |
|   X. F(x)It                            |
+----------------------------------------+
```

**Step 6** Select X to return to the Main menu.

---

**Listing Subscribers**

This section describes how to list subscribers in summary and detail formats.

**Step 1** In the Main menu, select option L.

**Step 2** Enter Y for the prompt “Show details? [n]”. Default is N for no detail.

*Figure 6-34* shows the result of these steps. A detail listings of all subscribers appears.
Chapter 6  System Administration Tools

Administration Tool for MySQL Databases

Figure 6-34  Detailed Multiple Subscriber Listings

If you enter N to the prompt “Show details? [n]”, only user IDs of all the subscribers appears as in Figure 6-35.

Figure 6-35  Summary Listings on Multiple Subscribers

Renaming an Existing Subscriber Record

In this section, a subscriber record is renamed from Alice to Ali by changing the subscriber ID.

Step 1  In the Main menu, select option M.
Step 2  Enter subscriber ID, Alice or alice, as prompted. Figure 6-36 shows these steps and result.
Step 3  In the Subscriber attribute submenu, select option I.

Step 4  Enter new subscriber ID for this record, ali, as prompted. Figure 6-37 and Figure 6-38 show these steps and result.

Figure 6-37  Prompt for New Subscriber ID

Figure 6-38  New Subscriber ID and Password Display
Removing a Subscriber from a MySQL Database

In this section, subscriber Bob is removed from the subscriber database.

**Step 1**
In the Main menu, select option R.

**Step 2**
Enter subscriber ID and domain name of Bob as prompted. Figure 6-39 shows these steps.

![Figure 6-39 Prompts for Removing a Subscriber](image)

If you select to list all subscriber in detail format, Figure 6-40 appears without the subscriber record of Bob.

![Figure 6-40 Detail Multiple Subscriber Listings after Removal of a Subscriber](image)

Tool Functionalities and Operations

This section provides a summary on operations of the following functionalities in addition to the tutorial.

- Adding Subscribers to the Databases, page 6-21
- Removing Subscribers from the Databases, page 6-21
- Modifying Subscribers in a Database, page 6-22
- Renaming an Existing Subscriber Record in a Database, page 6-22
- Displaying Subscribers of a Database, page 6-22
- Exiting the Tool, page 6-23
Adding Subscribers to the Databases

This functionality adds subscribers to the databases.

**Step 1** In the Main menu, select option A.

**Step 2** Enter subscriber ID domain name at the next prompt and press Enter key. Figure 6-41 appears.

![Subscriber Attribute Submenu for Adding Subscribers](image)

**Step 3** From the Subscriber attribute submenu, select option P to enter a password. The password appears next to the password option in the menu on the screen.

**Step 4** Select option N, B, V or U to enable the appropriate call forwarding options.

**Step 5** Enter the URL for the selected call forwarding option as prompted, and press Enter. The URL for each selected option appears next to the option in the menu.

**Step 6** To leave the menu, select option X.

Removing Subscribers from the Databases

This functionality removes subscribers from the databases.

**Step 1** In the Main menu, select option R.

**Step 2** Enter subscriber ID at the next prompt and press Enter. Message on operation completion appears.
Modifying Subscribers in a Database

This functionality changes subscriber information.

Step 1
In the Main menu, select option M.

Step 2
Enter subscriber ID at the next prompt and press Enter. Subscriber attribute submenu with attributes of
the selected subscriber appears.

Step 3
Modify the attributes by selecting the appropriate options in the menu accordingly.

Step 4
To leave the menu, select option X.

Renaming an Existing Subscriber Record in a Database

This functionality renames an existing subscriber record with a new subscriber ID.

Step 1
In the Main menu, select option A.

Step 2
Enter an existing subscriber ID as prompted.

Step 3
In the Subscriber attribute submenu, select option I.

Step 4
Enter new subscriber ID for this record as prompted.

Displaying Subscribers of a Database

This functionality displays information on one or all subscribers.

Displaying a Single Subscriber

Step 1
In the Main menu, select option S.

Step 2
Enter subscriber ID at the next prompt and press Enter.

Step 3
Subscriber attribute submenu with attributes of the selected subscriber appears.

Step 4
To leave the menu, select option X.

Displaying All Subscribers

Step 1
In the Main menu, select option L.

Step 2
Enter Y or N to the question “Show detail? [n]”
If you enter Y, all information, including attributes, related to the subscriber appears.
If you enter N, only the subscriber IDs appears.
Exiting the Tool

In the Main menu, Select option X.

Sample Error Messages

A sample of error messages are shown in Figure6-42, Figure6-43, Figure6-44, Figure6-45 and Figure6-46.

Error Message 1

Figure6-42  Sample MySQL Error Message 1

```
  Attempting to connect to <local> MySQL server...
  mysql -u u-guest --password=--null < /tmp/mysql_commands_12628
  ERROR 2002: Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (111)
```

Problem
The MySQL database is not installed.

Solution
Install the database before running the tool.

Error Message 2

Figure6-43  Sample MySQL Error Message 2

```
  Attempting to connect to <local> MySQL server...
  mysql -u u-guest --password=--null < /tmp/mysql_commands_12656
  ERROR 1045: Access denied for user: 'u-guest@localhost' (Using password: YES)
  Operation failed.
```

Problem
• You have entered an invalid MySQL username and MySQL password, or
• The username and password you have specified have insufficient permission to access the specified MySQL server database.

Solution
Enter correct username and password.

Error Message 3

Figure6-44  Sample MySQL Error Message 3

```
  ERROR 1146 at line 2: Table 'sip_subscribe1' doesn't exist
  Operation failed.
```

Problem
The table 'sip_subscribe1' does not exist.

Solution
Create the table or verify the table exists before running the tool.
Problem
The database name you entered for the database does not exist.

Solution
Enter a valid name or re-install the database.

Error Message 4

Figure 6-45  Sample MySQL Error Message 4

Problem
Invalid syntax specified for the subscriber ID.

Solution
Enter a valid subscriber ID.
Error Message 5

Figure 6-46 Sample MySQL Error Message 5

```plaintext
ERROR: Invalid dest_url.cfna syntax
The syntax of the dest_url.cfna field which you have entered is not recognized.
Please enter a valid value for this field (or press Enter to clear this field).
Examples:
sip:Bob@call-phone-gateway.company.com:user-ip
sip:call@proxy.company.com:user-phone
tel:+19195551234
```

**Problem**
Invalid syntax specified for a Call Forwarding destination URL.

**Solution**
Enter a valid URL.
CIAgent Overview

CIAgent is a SNMP agent that has the following functionalities:

- Start, stop and gracefully restart the CSPS
- Monitor the proxy server log file sizes (error_log and access_log)
- Monitor CPU usage
- Check memory size, disk space
- Check Link up/down status on the system running the proxy server

CIAgent has a master agent (snmpdm) that communicates with a few subagents (critagt, smagt, logagt, etc.) which service their respective MIBs. Each SNMP request that the master agent receives is passed to the appropriate subagent which retrieves or sets a particular MIB object value. Responses from subagents are passed back through the master agent to the requester.
Setting a MIB object value can also trigger a command to run, start, stop and gracefully restart CSPS. In the subagent managed MIBs, traps are defined so when a certain event occurs, a trap is sent by the subagent through the master agent to the management station or a trap sink. SNMP traps can be used to notify the CSPS administrator when the CSPS comes up or goes down, when the size of a log file exceeds a specified limit, or when the CPU load rises above or falls below specified thresholds.

CIAgent runs as a set of processes as root on the same system that runs CSPS. In a farm setup of CSPS, each farm member should have its own CIAgent co-existing on the same system. This CIAgent controls and monitors a particular farm member. A SNMP management station can send requests to all the CIAgents on the farm members, and each CIAgent responds to the management station independently.

### MIBs and Subagents

The MIB files are located in the CIAgent installation directory under the mibs directory. They are text files with the extension, .my, and is a good source to learn what a MIB/subagent can provide.

The following are CIAgent MIBs being used and the subagents that service them:

- CRITAPP-MIB (critagt)—start and stop CSPS
- LOG-MIB (logagt)—monitor CSPS error_log and access_log sizes

The following are standard MIBs and subagents in the CIAgent which are also used:

- DISMAN-SCRIPT-MIB (smagt)—gracefully restart CSPS
- DISMAN-EVENT-MIB (eventagt)—monitor CPU load
- HOST-RESOURCES-MIB (hostagt)—check memory size, disk space
- RFC1213-MIB (mib2agt)—check link up/down status
- SYSAPPL-MIB (sappagt)—check what applications are installed and running on the system

### Installation

Use either the customized or manual installation methods to install and upgrade CIAgent.

**Note**

In a farm setup of CSPS, each farm member should have its own CIAgent installed on the same system. Each farm member has a one-to-one relationship with its co-located CIAgent in the same system, though a SNMP manager station can control or monitor all farm members through their individual CIAgent.

Before installation, check if a SNMP agent is running on the system by using the `ps -ef|grep snmp` command. If no snmp agent is running, proceed to the last paragraph of this section.

If a SNMP agent is running on port 161 or port 161 is in use and the existing SNMP agent is running on the port specified for snmp in the `/etc/services` file, stop the SNMP agent or run CIAgent on a non-standard port.

To run CIAgent on a different port, set an environment variable, `SR_SNMP_TEST_PORT`, to an available port. If necessary, an environment variable, `SR_TRAP_TEST_PORT`, can be set to specify a non-standard trap port, beside port 162. Otherwise, if the environment variable, `SR_SNMP_TEST_PORT`, is set, the trap port will be of the value `SR_SNMP_TEST_PORT +1`. 
Optionally, you can set an environment variable, SR_HTTP_TEST_PORT, to specify a port besides 280 for snmp web management access. If any of the three environment variables are set for CIAgent, they must be included in the .login, .cshrc or any shell init files that are relevant to maintain consistency.

If a previous version of CIAgent is installed, use its uninstall.CIA script to uninstall it. Then use the following steps for installation.

**Customized Installation**

This section describes the customized installation for CIAgent.

**Note**

The customized script is only recommended for first time installation of CIAgent, and NOT for upgrades because it does not keep the previous configuration. For CIAgent upgrades, use the steps in “Manual installation” section on page 7-4.

**Step 1**
Login as root user.

**Step 2**
Run the installation script, csps_ciagent_install, by using the following command:

```
./csps_ciagent_install
```

On a Linux system, CIAgent is installed in the `/usr/local/ciagent` directory and CSPS is assumed to be installed in the `/usr/local/sip` directory. On a Solaris system, CIAgent is installed in the `/opt/ciagent` directory and CSPS is assumed to be installed in the `/opt/sip` directory.

During installation, the following configuration files are modified to reflect the CSPS installation path and they are copied into the `/etc/srconf/agt` directory. These files provide a basic set of configurations to CIAgent for use with CSPS. You can modify them to suit your environment.

- critagt.cnf
- smagt.cnf
- logagt.cnf
- eventagt.cnf

A script, smPopScript, is activated to populate the script MIB subagent (smagt) with commands to do graceful restart for CSPS. After installation, the CIAgent is automatically started.

**Step 3**
Add the CIAgent installation path and its bin directory to the search path as shown in the following example.

For Linux:

```
setenv PATH $PATH:/usr/local/ciagent:/usr/local/ciagent/bin
```

For Solaris:

```
setenv PATH $PATH:/opt/ciagent:/opt/ciagent/bin
```
Manual installation

To install CIAgent manually, use the CIAgent default installation script, `install`, then use the following steps.

**Step 1** Log in as a **root** user.

**Step 2** Change directory to where the CIAgent installation program is located with the following commands.

For Linux:
```
cd /usr/local/sip/ciagent
```

For Solaris:
```
cd opt/sip/ciagent
```

**Step 3** Run the installation program, `install`, with a parameter indicating your preferred CIAgent installation directory, as shown in the following examples.

For Linux:
```
./install /usr/local
```

For Solaris:
```
./install /opt
```

**Note** It is recommended to install CIAgent in the same parent directory tree as CSPS for easy location. For example, if CSPS is installed on Linux in the `/usr/local/sip` directory, pass `/usr/local` to the script. If CSPS is installed on Solaris in the `/opt/sip` directory, pass `/opt` to the script. If no parameters are provided, CIAgent is installed in the `/usr/local/ciagent` on both Linux and Solaris.

CIAgent is automatically started.

**Step 4** Use the following command to check if CIAgent is running.
```
ps -ef|grep snmpd
```

**Step 5** Add the CIAgent installation path and its bin directory to the bin search path with the following commands.

For Linux:
```
setenv PATH ${PATH}:/usr/local/ciagent:/usr/local/ciagent/bin
```

For Solaris:
```
setenv PATH ${PATH}:/opt/ciagent:/opt/ciagent/bin
```

When CIAgent is loaded, the following processes which run as root are displayed:

- `/snmpdm -tcplocal` (can be three on Linux)
- `/brassagt`
- `/critagt`
- `/mib2agt`
- `/eventagt`
- `/fsagt`
Access CIAgent Dr-Web interface by opening a web browser and enter the following URL:
http://<localhost, IP or hostname of the system running CIAgent>:280

Note 280 is the default standard port for snmp web management access. However, if the environment variable, SR_HTTP_TEST_PORT, was set to a different port value before running the installation, that port must be used to access the Dr-Web page.

For login and password, enter root and webRootPassword respectively.
Information on CIAgent and its configurations can be found in the CIAgent Online Manual on this homepage. You can also click on the list of CIAgent subagents to do configuration through Dr-Web, or refer to the manual configuration instructions in this document.

### Stopping and Restarting CIAgent Manually

You must be a root user to install, start, stop and uninstall CIAgent.

For Linux, to stop and restart CIAgent which is installed in the /usr/local/ciagent directory, use the following commands:

```bash
cd /usr/local/ciagent/
ociagent stop
ociagent start
```

For Solaris, to stop and restart CIAgent which is installed in the /opt/ciagent directory, use the following commands:

```bash
cd /opt/ciagent/
ociagent stop
ociagent start
```

### Uninstalling CIAgent

To uninstall CIAgent on both Linux and Solaris as a root user, issue the following command at the system prompt.
For Linux:

```bash
cd /usr/local/ciagent/
uninstall.CIA
```
Modifying System Information

To modify managed system information in the file `/etc/srconf/agt/snmpd.cnf`, change the default information in the following entries.

- sysLocation—The physical location of this managed system. For example, 2nd rack, 3rd floor
- sysContact—The contact person for this managed system
- sysName—FQDN of this managed system

Creating a User for the System Administrator

Step 1
To create a user for the system administrator to access Dr-Web interface, add a line to the end of the file, `/etc/srconf/agt/snmpd.cnf`, as shown in the following example.

```
httpUserNameEntry  <YOUR-LOGIN-NAME> SystemAdmin - nonVolatile <YOUR-LOGIN-PASSWORD>
```

Remove the comment markers of or remove entirely the other `httpUserNameEntry(s)` to increase security and limit access to your system administrator.

Step 2
Change the file permission to read-only by root afterwards, then stop and restart CIAgent to activate the new configuration.

Note
Access the Dr-Web interface with configured login and password, and configure the subagents for the environment.

Configuring SNMP v2c and Trap Target Addresses

To add or configure SNMP v2c community strings for the CSPS administrators and configure trap target addresses, refer to CIAgent online manual section, *The Emanate Master Agent*. The subsections on SNMP Community/userName Configuration and Trap Configuration provide information on how to make additions and modifications for the system environment. See following examples from the file `/etc/srconf/agt/snmpd.cnf`.

This example adds community string `cspsAdmin` with security level `proxySec`.

```
snmpCommunityEntry  t0000001 cspsAdmin proxySec localSnmpID - - nonVolatile
```

This example adds a new group, `proxyGroup`, with snmpv2c access permissions.

```
vacmAccessEntry  proxyGroup - snmpv2c noAuthNoPriv exact All All All nonVolatile
```

This example associates security level `proxySec` with group, `proxyGroup`.

```
vacmSecurityToGroupEntry  snmpv2c proxySec proxyGroup nonVolatile
```
This example specifies an IP address, 128.107.140.131, to send traps to.

```
snmpTargetAddrEntry  40 snmpUDPDomain 128.107.140.131:0 100 3 Console v2cExampleParams
nonVolatile 255.255.255.255:0
```

This example specifies community string, `cspAdmin`, to be used in v2c traps:

```
snmpTargetParamsEntry v2cExampleParams 1 snmpv2c proxySec noAuthNoPriv nonVolatile
```

## Customizing Configurations

It is recommended to use CIAgent's Dr-Web interface to configure CIAgent's subagents. If the `csps_ciagent_install` script is used to install CIAgent, a preset configuration is included in the installation. This configuration is extensible.

To send SNMP requests to CIAgent, use the Dr-Web interface, command-line CIAgent utility, or any other SNMP product. If the command-line CIAgent utility is used, a default community string, `cspAdmin`, is provided in the `snmpd.cnf` file.

CIAgent supports SNMP v1, v2c and v3, and the examples provided in this document and the readme file use v2c to show how to retrieve and set certain MIB objects for use of the CSPS. If security is a major concern, v3 MIB should be used. Refer to CIAgent's Online Manual on Dr-Web home page for more detail on CIAgent configurations and how to set up v3 user and passwords.

Observe the following settings:

- sipd's Intended Operation Status is initialized to be down. To start CSPS, change the status to up.
- Start and stop commands are also set into the Critical Applications MIB with appropriate path to the CSPS installation directory.
- The graceful restart command is set into the script MIB with appropriate path to the CSPS installation directory.
- The CSPS log files monitored by the CIAgent are pre-set to error_log and accesss_log with maximum size set to 5MB. If either one grows bigger than 5MB, a trap is sent.
- CPU average load over the last minute is monitored by the Event subagent and a rising threshold is set to 75% and a falling threshold is set to be 20%. When the CPU average load over the last minute exceeds the rising threshold or falls below the falling threshold, a trap is sent. To change these pre-set values or modify any CIAgent configurations by using the Dr-Web interface, or access them in their respective cnf files, refer to the CIAgent Online Manual for more detail.

To access the interface, use the following steps.

### Step 1
Enter the following URL at a web browser:

```
http://<localhost, machine IP or hostname running CIAgent>:280
```

### Step 2
Enter login and password as shown in the following examples.

```
login:proxy
password:cspAdmin
```

**Note** You can always remove or add logins in the file, `/etc/srconf/agt/snmpd.cnf`. See CIAgent Online Manual, Chapter 6, *The EMANATE Master Agent*, for the format and directives in `snmpd.cnf`. 
Step 3  At the CIAgent home page, access the CIAgent Online Manual on how to configure the subagents. Click on the listed subagents to see a sample list of CSPS specific configurations. The following list contains sample CIAgent’s subagents and how they are used with CSPS.

- Critical Application Monitor (critagt)—starts and stops CSPS
- Script MIB (smagt, smPopScript)—gracefully restarts the CSPS by setting a MIB object (smLaunchStart.1.67.1.68) to 0 via SNMP set command or CIAgent setany utility program.
- Log File Monitor (logagt)—monitors CSPS error_log and access_log file sizes, and sends traps when specified thresholds are exceeded.
- Host Resources (hostagt)—checks CPU usage, physical memory size and disk space usage on the system.
- Event MIB (eventagt)—monitors CPU usage by setting specified rising and falling thresholds and send traps when the thresholds are exceeded.
- System Applications Monitor (sappagt)—Checks what are installed and running on the system.
- MIB-II (mib2agt)—Checks link up and down status

Note  The Script MIB subagent cannot be configured through this home page. The MIB files are located in CIAgent installation directory under mibs. There are utility programs in the CIAgent bin directory that can interact with MIB objects. They are setany, getone, getmany, trapcv.

Configuring Critical Application Subagent (critagt)

To configure the critagt to start and stop CSPS, use the Dr-Web interface to add an entry for sipd, provide the Start command and Terminate command, and set desired options.

To do this, directly modify the /etc/srconf/agt/critagt.cnf file by adding a critAppProcEntry line for sipd, spa, pserver, and licenseMgr.

The following example is for Linux. To use this example for Solaris, change the path to sip from /usr/local/sip/bin/sip to /opt/sip/bin/sip.

# Entry type: critAppProcEntry
# Entry format: integer - index number (continuous positive integer)
# octetString - process name (real process name)
# octetString - start command (string of characters)
# octetString - stop command (string of characters)
# integer - intended operation status (up(1), down(2))
# integer - restart on exit (true or false)
# integer - restart interval in centisecond
# integer - send trap on exit (true or false)
# integer - send trap on start (true or false)
# integer - find process on startup (true or false)
critAppProcEntry 1 sipd \
   "/usr/local/sip/bin/sip start" \
   "/usr/local/sip/bin/sip stop" 2 false 3000 \
   true true true

critAppProcEntry 2 spa \
   "/usr/local/sip/bin/sip start" \
   "/usr/local/sip/bin/sip stop" 2 false 3000 \
   true true true
Stop and restart the critagt to activate the new configuration by using the following commands.

```
ps -ef|grep critagt
kill -9 <critagt's PID>
cd /usr/local/ciagent/bin (on solaris, cd /opt/ciagent/bin)
./critagt &
```

To start or stop CSPS, use Dr-Web to change the Intended Operation Status on the GUI, or use the following commands.

**Note**
setany is a SNMP utility program from CIAgent. Other SNMP set command can be used. Refer to the MIB files (located in the CIAgent installation directory under mibs) for the object information before setting it.

To start the CSPS, use the following command.

```
setany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin
critAppAdminStatus.1 1
```

To stop the CSPS, use the following command.

```
setany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin
critAppAdminStatus.1 2
```

To get the CSPS running status, use the following command.

```
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin
critAppOperStatus.1
```

To find out about the CSPS start command, use the following command.

```
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin
critAppStartCommand.1
```

To find out about the CSPS stop command, use the following command.

```
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin
critAppTerminateCommand.1
```
Configuring the Script Subagent (smagt)

To configure the smagt to do graceful restart for CSPS, use the following steps.

**Step 1** Modify the script, smPopScript, which is located in the CSPS distribution directory under the ciagent/conf directory. Use a text editor to modify the following variables in the script for the environment.

```
Agent="localhost"
Version="-v2c"
User="cspsAdmin"
AuthPassword=""
PrivPassword=""
```

Change the following line to reflect the CSPS installation path to do graceful restart. For Solaris, change path to sip, so the full path is /opt/sip/bin/sip.

```
setany $Version $Agent $User \ smCodeRowStatus.$SOwner.$SName.1 createAndGo \ smCodeText.$SOwner.$SName.1 -D "exec("/usr/local/sip/bin/sip graceful");" \
```

**Step 2** Copy this file to the CIAgent installation bin directory (/usr/local/ciagent/bin for Linux; /opt/ciagent/bin for Solaris), and change file permission to be executable by root.

**Step 3** Run the script to populate the script MIB table by using the following command.

```
./smPopScript
```

**Step 4** To invoke a graceful restart, use the following command. This assumes the CSPS is already running.

```
setany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin smLaunchStart.1.67.1.68 0
```

To stop and restart CIAgent, run this script to populate the script MIB subagent. To avoid doing this every time, add a call to the script in the ciagent script file (CIAgent installation directory). See the following example in the ciagent script file.

```
./critagt
./mib2agt
./eventagt &
./fsagt &
./hostagt &
./htmlagt
./logagt
./mappagt &
./schdagt
./smagt
sleep 5
./smPopScript
```

**Note** This example assumes the smPopScript file is modified and exists in the CIAgent bin directory. The **sleep 5** command runs smPopScript after sleeping 5 seconds for smagt to fully started. An alternative is to use the customized start/stop script, csps_ciaagent_ctl (in /usr/local/sip/ciagent for Linux and /opt/sip/ciagent for Solaris). You can copy it to your CIAgent installation directory (/usr/local/ciagent for Linux and /opt/ciagent for Solaris) and give it execution permission. The customized script calls the smPopScript script after starting CIAgent.
Configuring the Log File Subagent (logagt)

To configure the logagt to monitor CSPS log file sizes at certain intervals and to send traps when they exceed the thresholds, modify file /etc/srconf/agt/logagt.cnf directly or by using the Dr-Web interface. The following example is for Linux. To use this example for Solaris, change the path for error_log and access_log to /opt/sip/logs/error_log and /opt/sip/logs/access_log respectively.

```
# Entry type: siLogGlobalPollInterval
# Entry format: integer
siLogGlobalPollInterval 60
# Entry type: siLogEntry
# Entry format: integer - index number
# octetString - description of the file to be monitored
# octetString - full path to the file
# octetString - regular expression to match in the file
# integer - leave it as is
# integer - character position to start matching
# integer - character position to stop matching
# integer - number of matches found so far
# octetString - command to run on match
# integer - send trap on match (yes(1), no(2))
# integer - current size of the file in bytes
# integer - maximum file size as threshold
# octetString - command to run when maximum size reached
# integer - send trap on maximum size (yes(1), no(2))
# integer - polling interval in seconds
# integer - leave it as is
# octetString - file owner
# integer - leave it as is
siLogEntry 1 "CSPS error log" \
/usr/local/sip/logs/error_log - 2 0 0 0 - 2 \n316687 5000000 - 1 10 2 csps 1 316687 "tent-Length: 0\n\n"
siLogEntry 2 "CSPS access log" \
/usr/local/sip/logs/access_log - 2 0 0 0 - 2 \n316687 5000000 - 1 10 2 csps 1 316687 "tent-Length: 0\n\n"
```

View your current log file size from Dr-Web, or use the following commands to retrieve it.

```
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin siLogSize.1
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin siLogSize.2
```

**Note**
In the preset configurations, siLogSize.1 refers to log file size for the file at index 1 (error_log). siLogSize.2 refers to access_log.

Configuring the Event MIB Subagent

To configure the Event MIB subagent to monitor CPU usage and send traps when CPU average-usage-over-a-minute passes a specified threshold, use one of the following methods.

- Use the Dr-Web interface to configure a Trigger-Event-Notification set for hrProcessorLoad object in Host Resources MIB for CPU rising and falling thresholds.
- Modify the following sample eventagt.cnf file accordingly. Refer to CIAgent Online Manual on Dr-Web for more information on each directives in the eventagt.cnf configuration file.
In this example, the rising threshold is set to 75%, the falling threshold is set to 20%, and the polling interval is set to 5 seconds.

```plaintext
# Entry type: mteResourceSampleMinimum
# Entry format:
# mteResourceSampleMinimum (integer) mteResourceSampleMinimum 1
# Entry type: mteResourceSampleInstanceMaximum
# Entry format:
# mteResourceSampleInstanceMaximum (unsigned) mteResourceSampleInstanceMaximum 0u
# Entry type: mteTriggerEntry
# Entry format:
# mteOwner (text) mteTriggerName (text) mteTriggerComment (text) mteTriggerTest (bits) mteTriggerSampleType (integer) mteTriggerValueID (ObjectID) mteTriggerValueIDWildcard (integer) mteTriggerTargetTag (text) mteTriggerContextName (text) mteTriggerContextNameWildcard (integer) mteTriggerFrequency (unsigned) mteTriggerObjectsOwner (text) mteTriggerObjects (text) mteTriggerEnabled (integer) mteTriggerEntryStatus (integer) mteTriggerEntry 61 loadTrigger * * 20 1 iso.3.6.1.2.1.25.3.3.1.2.1 2 - - 2 \ 5u - - 1 1
# Entry type: mteTriggerDeltaEntry
# Entry format:
# mteTriggerDeltaDiscontinuityID (ObjectID) mteTriggerDeltaDiscontinuityIDWildcard (integer) mteTriggerDeltaDiscontinuityIDType (integer) mteOwner (text) mteTriggerName (text)
# Entry type: mteTriggerExistenceEntry
# Entry format:
# mteTriggerExistenceTest (bits) mteTriggerExistenceStartup (bits) mteTriggerExistenceObjectsOwner (text) mteTriggerExistenceObjects (text) mteTriggerExistenceEventOwner (text) mteTriggerExistenceEvent (text) mteOwner (text) mteTriggerName (text)
# Entry type: mteTriggerBooleanEntry
# Entry format:
# mteTriggerBooleanComparison (integer) mteTriggerBooleanValue (integer) mteTriggerBooleanStartup (integer) mteTriggerBooleanObjectsOwner (text) mteTriggerBooleanObjects (text) mteTriggerBooleanEventOwner (text) mteTriggerBooleanEvent (text) mteOwner (text) mteTriggerName (text)
# Entry type: mteTriggerThresholdEntry
# Entry format:
# mteTriggerThresholdStartup (integer) mteTriggerThresholdRising (integer) mteTriggerThresholdFalling (integer) mteTriggerThresholdDeltaRising (integer) mteTriggerThresholdDeltaFalling (integer)
```
The hrProcessorLoad object represents the average CPU usage over the last minute in a system unit. This is not the same as the CPU usage output in Unix program "top" which shows CPU usage in the sampling moment. This means the hrProcessorLoad value rises up and drops down slowly, because it is an average value over a minute.

To check current hrProcessorLoad object value, use the following command.

```
getone -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin hrProcessorLoad.1
```
Configuring Trap Sink for CIAgent Traps

To set up a trap sink for CIAgent traps, run the CIAgent utility, traprcv, as root and configure in the snmpd.cnf file where trap sink is located accordingly. Then restart CIAgent. Usually, the loopback address (127.0.0.1) is one of the default trap sink addresses. When a trap triggering event occurs, such as the CSPS going up or down, a trap message should appear in the traprcv program window. The same applies to log files that exceed limits and CPU load that is over or under thresholds.

Checking Memory Size

To check memory size (amount of kilobytes in physical main memory in the host), use the following command.

```
getmany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin hrMemorySize
```

Checking Disk Space Information

To check disk space information, use the following command.

```
getmany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin hrStorageEntry
```

Note: See HrStorageEntry in Host Resources MIB for more detail.

Example storage types is given as:

- `hrStorageType.1 = hrStorageRam`
- `hrStorageType.2 = hrStorageVirtualMemory`
- `hrStorageType.3 = hrStorageFixedDisk`
- `hrStorageType.4 = hrStorageFixedDisk`
- `hrStorageType.5 = hrStorageOther`
- `hrStorageType.6 = hrStorageOther`
- `hrStorageType.7 = hrStorageFixedDisk`
- `hrStorageType.8 = hrStorageRamDisk`
- `hrStorageType.9 = hrStorageRamDisk`
- `hrStorageType.10 = hrStorageFixedDisk`
- `hrStorageType.11 = hrStorageFixedDisk`
- `hrStorageType.12 = hrStorageFixedDisk`
- `hrStorageType.13 = hrStorageFixedDisk`
- `hrStorageType.14 = hrStorageFixedDisk`

Example storage description corresponding to the indices is given as:

- `hrStorageDescr.1 = Physical RAM`
- `hrStorageDescr.2 = Virtual Memory`
Checking Disk Space Information

- hrStorageDescr.3 = /
- hrStorageDescr.4 = /usr
- hrStorageDescr.5 = /proc
- hrStorageDescr.6 = /etc/mnttab
- hrStorageDescr.7 = /var
- hrStorageDescr.8 = /var/run
- hrStorageDescr.9 = /tmp
- hrStorageDescr.10 = /opt
- hrStorageDescr.11 = /auto/vvs
- hrStorageDescr.12 = /users/kevinmc
- hrStorageDescr.13 = /users/liuhong
- hrStorageDescr.14 = /users/ribiere

Example storage allocation unit size (in bytes) corresponding to the indices is given as:
- hrStorageAllocationUnits.1 = 8192
- hrStorageAllocationUnits.2 = 8192
- hrStorageAllocationUnits.3 = 1024
- hrStorageAllocationUnits.4 = 1024
- hrStorageAllocationUnits.5 = 512
- hrStorageAllocationUnits.6 = 512
- hrStorageAllocationUnits.7 = 1024
- hrStorageAllocationUnits.8 = 8192
- hrStorageAllocationUnits.9 = 8192
- hrStorageAllocationUnits.10 = 1024
- hrStorageAllocationUnits.11 = 512
- hrStorageAllocationUnits.12 = 512
- hrStorageAllocationUnits.13 = 512
- hrStorageAllocationUnits.14 = 512

Example storage total size in terms of the allocation units corresponding to the indices is given as:
- hrStorageSize.1 = 65536
- hrStorageSize.2 = 256214
- hrStorageSize.3 = 962571
- hrStorageSize.4 = 4032142
- hrStorageSize.5 = 0
- hrStorageSize.6 = 0
- hrStorageSize.7 = 962571
- hrStorageSize.8 = 291743
- hrStorageSize.9 = 292422
- hrStorageSize.10 = 9340301
Checking Link Up/down Status

To check link up/down status from MIB-2, use the following command.

```
getmany -v2c <localhost, IP address or hostname of system running CIAgent> cspsAdmin ifTable
```

Note: Refer to MIB-2 for more detail.

Example link description:
- ifDescr.1 = lo0
- ifDescr.2 = hme0

Example link up/down status:
- ifOperStatus.1 = up(1)
- ifOperStatus.2 = up(1)

Example link type:
- ifType.1 = softwareLoopback(24)
- ifType.2 = ethernet_csmacd(6)

Example storage size being used in terms of the allocation units corresponding to the indices is given as:
- hrStorageSize.11 = 251658240
- hrStorageSize.12 = 83886080
- hrStorageSize.13 = 83886080
- hrStorageSize.14 = 83886080

Example storage size being used in terms of the allocation units corresponding to the indices is given as:
- hrStorageUsed.1 = 26255
- hrStorageUsed.2 = 0
- hrStorageUsed.3 = 53225
- hrStorageUsed.4 = 1211605
- hrStorageUsed.5 = 0
- hrStorageUsed.6 = 0
- hrStorageUsed.7 = 147081
- hrStorageUsed.8 = 1
- hrStorageUsed.9 = 680
- hrStorageUsed.10 = 41936
- hrStorageUsed.11 = 229118304
- hrStorageUsed.12 = 41664336
- hrStorageUsed.13 = 41664336
- hrStorageUsed.14 = 41664336
Example link’s MTU:
• ifMtu.1 = 8232
• ifMtu.2 = 1500

Checking Installed and Active Components in the System

To check what software is installed and running in the system from SYSAPPL-MIB, it is recommended to use Dr-Web interface by clicking on System Applications Monitor from the CIAgent home page.

The `getmany` command may retrieve a long list of entries. See the following examples.

```
getmany -v2c <IP address of system running CIAgent or localhost> cspsAdmin sysApplInstalled
getmany -v2c <IP address of system running CIAgent or localhost> cspsAdmin sysAppRun
getmany -v2c <IP address of system running CIAgent or localhost> cspsAdmin sysApplInstallPkgProductName
getmany -v2c <IP address of system running CIAgent or localhost> cspsAdmin sysApplInstallPkgDate
getmany -v2c <IP address of system running CIAgent or localhost> cspsAdmin \ sysApplInstallPkgLocation
```

Using the CСПS with CIAgent

Starting, Stopping, and Gracefully Restarting the CСПS

Starting the CСПS

To start the CСПS from CIAgent, use the Dr-Web interface as follows.

**Step 1**
Click on the Critical Application Monitor and set the sipd's **Intended Operation Status** to up.

**Step 2**
Reload the page to check the operational status to verify that the CСПS is running.
Alternatively, from the command line, run the CIAgent utility programs by using the following command:

```
setany -v2c <IP address of system running CIAgent or localhost> cspsAdmin critAppAdminStatus.1 1
```

Stopping CСПS

To stop the CСПS from CIAgent, use the Dr-Web interface as follows.

**Step 1**
Click on the Critical Application Monitor and set the sipd's **Intended Operation Status** to down.
Step 2  Reload the page to check the operational status to verify that the CSPS is not running.
Alternatively, from command line, run the CIAgent utility programs by using the following command:

```
setany -v2c <IP address of system running CIAgent or localhost> cspsAdmin
critAppAdminStatus.1 2
```

Checking Status

To check the CSPS running status, use the following command:

```
getone -v2c <IP address of system running CIAgent or localhost> cspsAdmin
critAppOperStatus.1
```

To find out about the CSPS start command, use the following command:

```
getone -v2c <IP address of system running CIAgent or localhost> cspsAdmin
critAppStartCommand.1
```

To find out about the CSPS stop command, use the following command:

```
getone -v2c <IP address of system running CIAgent or localhost> cspsAdmin
critAppTerminateCommand.1
```

Restarting the CSPS Gracefully

To gracefully restart CSPS, use the following command:

```
setany -v2c <IP address of system running CIAgent or localhost> cspsAdmin
smLaunchStart.1.67.1.68 0
```
Cisco SIP Proxy Server (CSPS) Compliance Information

This appendix describes how the CSPS complies with the IETF definition of SIP (Internet Draft draft-ietf-sip-rfc2543bis-04.txt, based on RFC 2543). It also provides an overview of SIP services.

SIP Compliance with RFC 2543

This section contains compliance information on the following:

- SIP Functions, page A-1
- SIP Methods, page A-1
- SIP Responses, page A-2
- SIP Header Fields, page A-6
- Transport Layer Protocols, page A-8
- SIP Security, page A-8

SIP Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy Server</td>
<td>Yes (Transaction stateful, parallel forking, and recursive)</td>
</tr>
<tr>
<td>Redirect Server</td>
<td>Yes</td>
</tr>
<tr>
<td>Registrar Server</td>
<td>Yes</td>
</tr>
</tbody>
</table>

SIP Methods

Five of the six methods used by the SIP are supported:

<table>
<thead>
<tr>
<th>Method</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVITE</td>
<td>Yes</td>
<td>The CSPS proxies SIP INVITE requests.</td>
</tr>
<tr>
<td>ACK</td>
<td>Yes</td>
<td>The CSPS proxies the SIP ACK method.</td>
</tr>
</tbody>
</table>
### SIP Compliance with RFC 2543

#### SIP Responses

The CSPS supports the following SIP responses:

- **1xx Response**—Information Responses, page A-3
- **2xx Response**—Successful Responses, page A-3
- **3xx Response**—Redirection Responses, page A-4
- **4xx Response**—Request Failure Responses, page A-4
- **5xx Response**—Server Failure Responses, page A-6
- **6xx Response**—Global Responses, page A-6

<table>
<thead>
<tr>
<th>Method</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONS</td>
<td>See comments</td>
<td>The CSPS proxies the OPTIONS and BYE methods.</td>
</tr>
<tr>
<td>BYE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CANCEL</td>
<td>Yes</td>
<td>The CSPS proxies the SIP CANCEL method.¹</td>
</tr>
<tr>
<td>REGISTER</td>
<td>Yes</td>
<td>The CSPS supports both user and device registration.</td>
</tr>
</tbody>
</table>

¹ The CSPS can generate a local ACK for a non-200 OK final response to an INVITE request.

² The CSPS can generate a local CANCEL for a pending branch when it receives a 200 OK or 6xx response from the branch.
### 1xx Response—Information Responses

<table>
<thead>
<tr>
<th>1xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 Trying</td>
<td>Yes</td>
<td>The CSPS generates and proxies this response for an incoming INVITE. Upon receiving this response, the server waits for a 180 Ringing, 183 Session progress, or 200 OK response.</td>
</tr>
<tr>
<td>180 Ringing</td>
<td>Yes</td>
<td>None</td>
</tr>
<tr>
<td>181 Call is being forwarded</td>
<td>See comments</td>
<td>The CSPS proxies these responses.</td>
</tr>
<tr>
<td>182 Queued</td>
<td></td>
<td></td>
</tr>
<tr>
<td>183 Session Progress</td>
<td></td>
<td>The CSPS does not generate this message. Upon receiving this message, the proxy server forwards the message upstream.</td>
</tr>
</tbody>
</table>

### 2xx Response—Successful Responses

<table>
<thead>
<tr>
<th>2xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Yes</td>
<td>The CSPS can generate a 200 OK response to a REGISTER or CANCEL request. The CSPS proxy 200 OK responses to other requests.</td>
</tr>
</tbody>
</table>
3xx Response— Redirection Responses

<table>
<thead>
<tr>
<th>3xx Response</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 Multiple</td>
<td>Yes</td>
<td>When in Redirect mode, the CSPS can only generate the 300 Multiple Choices response. When in Proxy mode, the CSPS can generate or proxy this response.</td>
</tr>
<tr>
<td>Choices</td>
<td></td>
<td></td>
</tr>
<tr>
<td>301 Moved Permanently</td>
<td>Yes</td>
<td>The CSPS can proxy this response.</td>
</tr>
<tr>
<td>302 Moved Temporarily</td>
<td>Yes</td>
<td>When in Redirect mode, the CSPS can only generate the 302 Moved Temporarily response when a matching registration is located. When in Proxy mode, the CSPS can generate or proxy this response.</td>
</tr>
<tr>
<td>305 Use Proxy</td>
<td>Yes</td>
<td>None.</td>
</tr>
<tr>
<td>380 Alternate Service</td>
<td>No</td>
<td>None.</td>
</tr>
</tbody>
</table>

4xx Response— Request Failure Responses

<table>
<thead>
<tr>
<th>4xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>400 Bad Request</td>
<td>Yes</td>
<td>The proxy server can generate and proxy a 400 Bad Request response.</td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>Yes</td>
<td>This response is proxied by the CSPS. If the CSPS is configured as a registrar and authentication is turned on, the proxy server generates this response.</td>
</tr>
<tr>
<td>402 Payment Required</td>
<td>See comments</td>
<td>These responses are generated if the CSPS is in registrar mode and is proxied by the proxy server.</td>
</tr>
<tr>
<td>403 Forbidden</td>
<td></td>
<td></td>
</tr>
<tr>
<td>404 Not Found</td>
<td>Yes</td>
<td>These responses are generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>405 Method Not Allowed</td>
<td>See comments</td>
<td>These responses are proxied in this release.</td>
</tr>
<tr>
<td>406 Not Acceptable</td>
<td>See comments</td>
<td></td>
</tr>
<tr>
<td>407 Proxy Authentication Required</td>
<td>Yes</td>
<td>This response is proxied by the CSPS. If authentication is turned on, the CSPS generates this response.</td>
</tr>
<tr>
<td>408 Request Timeout</td>
<td>See comments</td>
<td>These responses are generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>409 Conflict</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 4xx Response

<table>
<thead>
<tr>
<th>4xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>410 Gone</td>
<td>See comments</td>
<td>This response is only proxied by the CSPS in this release. The 410 Gone response indicates that a resource is no longer available at the server and no forwarding address is known.</td>
</tr>
<tr>
<td>411 Length Required</td>
<td>Yes</td>
<td>This response is proxied in this release. This response indicates that the user refuses to accept the request without a defined content length.</td>
</tr>
<tr>
<td>413 Request Entity Too Large</td>
<td>See comments</td>
<td>This response is only proxied by the CSPS in this release. If a retry after header field is contained in this response, then the user can attempt the call once again in the retry time provided.</td>
</tr>
<tr>
<td>414 Request—URL Too Long</td>
<td>Yes</td>
<td>This response is generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>415 Unsupported Media</td>
<td>Yes</td>
<td>This response is proxied in this release.</td>
</tr>
<tr>
<td>420 Bad Extension</td>
<td>Yes</td>
<td>This response is generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>480 Temporarily Unavailable</td>
<td>Yes</td>
<td>This response is proxied by the CSPS in this release. It is generated if pre-authentication is enabled and fails.</td>
</tr>
<tr>
<td>481 Call Leg/Transaction Does Not Exist</td>
<td>Yes</td>
<td>These responses are generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>482 Loop Detected</td>
<td></td>
<td></td>
</tr>
<tr>
<td>483 Too Many Hops</td>
<td></td>
<td></td>
</tr>
<tr>
<td>484 Address Incomplete</td>
<td>See comments</td>
<td>This response is proxied by the CSPS.</td>
</tr>
<tr>
<td>485 Ambiguous</td>
<td>See comments</td>
<td>This response is only proxied by the CSPS in this release.</td>
</tr>
<tr>
<td>486 Busy Here</td>
<td></td>
<td></td>
</tr>
<tr>
<td>487” ; Request Terminated</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>488” ; Not Acceptable Here</td>
<td>See comments</td>
<td>This response is proxied by the CSPS.</td>
</tr>
</tbody>
</table>
Appendix A Cisco SIP Proxy Server (CSPS) Compliance Information

5xx Response—Server Failure Responses

<table>
<thead>
<tr>
<th>5xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>500 Internal Server Error</td>
<td>Yes</td>
<td>These responses are generated and proxied by the CSPS.</td>
</tr>
<tr>
<td>501 Not Implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>502 Bad Gateway</td>
<td>See comments</td>
<td>This response is only proxied by the CSPS in this release.</td>
</tr>
<tr>
<td>503 Service Unavailable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>504 Gateway Timeout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>505 Version Not Supported</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

6xx Response—Global Responses

<table>
<thead>
<tr>
<th>6xx Response</th>
<th>Supported?</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>600 Busy Everywhere</td>
<td>See comments</td>
<td>These responses are only proxied by the CSPS in this release.</td>
</tr>
<tr>
<td>603 Decline</td>
<td></td>
<td></td>
</tr>
<tr>
<td>604 Does Not Exist Anywhere</td>
<td></td>
<td></td>
</tr>
<tr>
<td>606 Not Acceptable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SIP Header Fields

Note: All SIP header fields that concern the CSPS are correctly handled and parsed except for the Hide and Encryption header fields. Header fields that do not directly affect the CSPS or are unknown to the CSPS are passed unaltered in the SIP request.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Yes</td>
</tr>
<tr>
<td>Accept-Encoding</td>
<td>Yes</td>
</tr>
<tr>
<td>Accept-Language</td>
<td>Yes</td>
</tr>
<tr>
<td>Alert-Info</td>
<td>Yes</td>
</tr>
<tr>
<td>Allow</td>
<td>Yes</td>
</tr>
<tr>
<td>Also</td>
<td>Yes</td>
</tr>
<tr>
<td>Authorization</td>
<td>Yes</td>
</tr>
<tr>
<td>Call-ID</td>
<td>Yes</td>
</tr>
<tr>
<td>Call-Info</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### SIP Compliance with RFC 2543

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contact</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Disposition</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Length</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-Type</td>
<td>Yes</td>
</tr>
<tr>
<td>Cseq</td>
<td>Yes</td>
</tr>
<tr>
<td>Date</td>
<td>Yes</td>
</tr>
<tr>
<td>Encryption(^1)</td>
<td>No</td>
</tr>
<tr>
<td>Error-Info</td>
<td>Yes</td>
</tr>
<tr>
<td>Expires</td>
<td>Yes</td>
</tr>
<tr>
<td>From</td>
<td>Yes</td>
</tr>
<tr>
<td>In-Reply-To</td>
<td>Yes</td>
</tr>
<tr>
<td>Max-Forwards</td>
<td>Yes</td>
</tr>
<tr>
<td>Organization</td>
<td>Yes</td>
</tr>
<tr>
<td>MIME-Version</td>
<td>Yes</td>
</tr>
<tr>
<td>Priority</td>
<td>Yes</td>
</tr>
<tr>
<td>Proxy-Authenticate</td>
<td>Yes</td>
</tr>
<tr>
<td>Proxy-Authorization</td>
<td>Yes</td>
</tr>
<tr>
<td>Proxy-Require</td>
<td>Yes</td>
</tr>
<tr>
<td>Record-Route</td>
<td>Yes</td>
</tr>
<tr>
<td>Require</td>
<td>Yes</td>
</tr>
<tr>
<td>Response-Key</td>
<td>Yes</td>
</tr>
<tr>
<td>Retry-After</td>
<td>Yes</td>
</tr>
<tr>
<td>Route</td>
<td>Yes</td>
</tr>
<tr>
<td>Server</td>
<td>Yes</td>
</tr>
<tr>
<td>Subject</td>
<td>Yes</td>
</tr>
<tr>
<td>Supported</td>
<td>Yes</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Yes</td>
</tr>
<tr>
<td>To</td>
<td>Yes</td>
</tr>
<tr>
<td>Unsupported</td>
<td>Yes</td>
</tr>
<tr>
<td>User-Agent</td>
<td>Yes</td>
</tr>
<tr>
<td>Via</td>
<td>Yes</td>
</tr>
<tr>
<td>Warning</td>
<td>Yes</td>
</tr>
<tr>
<td>WWW-Authenticate</td>
<td>Yes</td>
</tr>
</tbody>
</table>

\(^1\) When a SIP message is received that contains this header field, the message is processed with the field ignored.
Transport Layer Protocols

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicast UDP</td>
<td>Yes</td>
</tr>
<tr>
<td>Multicast UDP</td>
<td>No</td>
</tr>
<tr>
<td>TCP</td>
<td>Yes</td>
</tr>
</tbody>
</table>

SIP Security

Authentication

<table>
<thead>
<tr>
<th>Authentication</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Authentication</td>
<td>Yes</td>
</tr>
<tr>
<td>Digest Authentication</td>
<td>Yes</td>
</tr>
<tr>
<td>Proxy Authentication</td>
<td>Yes</td>
</tr>
<tr>
<td>PGP</td>
<td>No</td>
</tr>
</tbody>
</table>

SIP DNS Records Usage

<table>
<thead>
<tr>
<th>DNS Resource Record Type</th>
<th>Supported?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type A</td>
<td>Yes</td>
</tr>
<tr>
<td>Type SRV</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Cisco SIP Proxy Server (CSPS) Call Flows

This appendix describes the types of SIP methods used by the CSPS and the flow of these messages. This appendix contains the following sections:

- SIP Messages and Methods, page B-1
- Call Flow Scenarios for Successful Calls, page B-2
- Call Flow Scenarios for Failed Calls, page B-33

SIP Messages and Methods

All SIP messages are either requests from a server or client or responses to a request. The messages are formatted according to RFC 822, “Standard for the format of ARPA internet text messages.” For all messages, the general format is:

- A start line
- One or more header fields
- An empty line
- A message body (optional)

Each line must end with a carriage return-line feed (CRLF).

Requests

SIP uses six types (methods) of requests:

- INVITE—Indicates a user or service is being invited to participate in a call session.
- ACK—Confirms that the client has received a final response to an INVITE request.
- BYE—Terminates a call and can be sent by either the caller or the callee.
- CANCEL— Cancels any pending searches but does not terminate a call that has already been accepted.
- OPTIONS—Queries the capabilities of servers.
- REGISTER—Registers the address listed in the To header field with a SIP server.
Responses

The following types of responses are used by SIP and generated by the CSPS:

- SIP 1.xx—Informational Responses
- SIP 2.xx—Successful Responses
- SIP 3.xx—Redirection Responses
- SIP 4.xx—Client Failure Responses
- SIP 5.xx—Server Failure Responses
- SIP 6.xx—Global Failure Responses

The Registration Process

A registration occurs when a client needs to inform a proxy or redirect server of its location. During this process, the client sends a REGISTER request to the proxy or redirect server and includes the address (or addresses) at which it can be reached.

The Invitation Process

An invitation occurs when one SIP endpoint (user A) “invites” another SIP endpoint (user B) to join in a call. During this process, user A sends an INVITE message requesting that user B join a particular conference or establish a two-party conversation. If user B wants to join the call, it sends an affirmative response (SIP 2xx). Otherwise, it sends a failure response (SIP 4xx). Upon receiving the response, user A acknowledges the response with an ACK message. If user A no longer wants to establish this conference, it sends a BYE message instead of an ACK message.

Call Flow Scenarios for Successful Calls

This section describes call flows for the following scenarios, which illustrate successful calls:

- SIP Gateway-to-SIP Gateway—Call via SIP Redirect Server, page B-3
- SIP Gateway-to-SIP Gateway—Call via SIP Proxy Server, page B-6
- SIP IP Phone-to-SIP IP Phone Call Forward Unconditionally, page B-14
- SIP IP Phone-to-SIP IP Phone Call Forward on Busy, page B-19
- SIP IP Phone-to-SIP IP Phone Call Forward No Answer, page B-23
- SIP IP Phone-to-SIP IP Phone Call Forward Unavailable, page B-28

Note

The messages are provided as examples for reference only.
SIP Gateway-to-SIP Gateway—Call via SIP Redirect Server

Figure B-1 illustrates a successful gateway-to-gateway call setup and disconnect via a SIP redirect server. In this scenario, the two end users are identified as User A and User B. User A is located at PBX A. PBX A is connected to SIP gateway 1 via a T1/E1. SIP gateway 1 is using a SIP redirect server. User B is located at PBX B. PBX B is connected to SIP gateway 2 via a T1/E1. User B’s phone number is 555-0002. SIP gateway 1 is connected to SIP gateway 2 over an IP network.

The call flow scenario is as follows:

1. User A calls User B via SIP gateway 1 using a SIP redirect server.
2. User B answers the call.
3. User B hangs up.

Figure B-1  SIP Gateway-to-SIP Gateway—Call via SIP Redirect Server

![SIP Gateway-to-SIP Gateway—Call via SIP Redirect Server diagram](image)
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP redirect server | SIP gateway 1 sends an INVITE request to the SIP redirect server. The INVITE request is an invitation to User B to participate in a call session.  

In the INVITE request:  
- The phone number of User B is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of User B and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to User B appears as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- PBX A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability User A is ready to receive is specified.  
- The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | 300 Multiple Choice—SIP redirect server to SIP gateway 1 | The SIP redirect server sends a 300 Multiple Choice response to SIP gateway 1. The 300 Multiple Choice response indicates that the SIP redirect server accepted the INVITE request, contacted a location server with all or part of User B’s SIP URL, and the location server provided a list of alternative locations where User B might be located. The SIP redirect server returns these possible addresses to SIP gateway 1 in the 300 Multiple Choice response. |
| 4    | ACK—SIP gateway 1 to SIP redirect server | SIP gateway 1 acknowledges the 300 Multiple Choice response with an ACK. |
| 5    | INVITE—SIP gateway 1 to SIP gateway 2 | SIP gateway 1 sends a new INVITE request to SIP gateway 2. The new INVITE request includes the first contact listed in the 300 Multiple Choice response as the new address for User B, a higher transaction number in the CSeq field, and the same Call-ID as the first INVITE request. |
| 6    | Setup—SIP gateway 2 to PBX B | SIP gateway 2 receives the INVITE request from SIP gateway 1 and initiates a Call Setup with User B via PBX B. |
| 7    | Call Proceeding—SIP gateway 1 to PBX A | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request. |
| 8    | 100 Trying—SIP gateway 2 to SIP gateway 1 | SIP gateway 2 sends a 100 Trying response to the INVITE request sent by SIP gateway 1. The 100 Trying response indicates that the INVITE request was received by SIP gateway 2, but that User B is not yet located. |
| 9    | Call Proceeding—PBX B to SIP gateway 2 | PBX B sends a Call Proceeding message to SIP gateway 2 to acknowledge the Call Setup request. |
### Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Alerting—PBX B to SIP gateway 2</td>
<td>PBX B locates User B and sends an Alert message to SIP gateway 2. User B’s phone begins to ring.</td>
</tr>
<tr>
<td>11</td>
<td>180 Ringing—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a 180 Ringing response to SIP gateway 1. The 180 Ringing response indicates that SIP gateway 2 has located, and is trying to alert User B.</td>
</tr>
<tr>
<td>12</td>
<td>Alerting—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends an Alert message to PBX A. User A hears ring back tone.</td>
</tr>
</tbody>
</table>

At this point, a one-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Connect—PBX B to SIP gateway 2</td>
<td>User B answers phone. PBX B sends a Connect message to SIP gateway 2. The Connect message notifies SIP gateway 2 that the connection has been made.</td>
</tr>
</tbody>
</table>
| 14   | 200 OK—SIP gateway 2 to SIP gateway 1 | SIP gateway 2 sends a 200 OK response to SIP gateway 1. The 200 OK response notifies SIP gateway 1 that the connection has been made.  
If User B supports the media capability advertised in the INVITE message sent by SIP gateway 1, it advertises the intersection of its own and User A’s media capability in the 200 OK response. If User B does not support the media capability advertised by User A, it sends back a 400 Bad Request response with a 304 Warning header field. |
| 15   | Connect—SIP gateway 1 to PBX A | SIP gateway 1 sends a Connect message to PBX A. The Connect message notifies PBX A that the connection has been made. |
| 16   | Connect ACK—PBX A to SIP gateway 1 | PBX A acknowledges SIP gateway 1’s Connect message. |
| 17   | ACK—SIP gateway 1 to SIP gateway 2 | SIP gateway 1 sends an ACK to SIP gateway 2. The ACK confirms that the 200 OK response has been received.  
The call is now in progress over a two-way voice path via RTP. |
| 18   | Connect ACK—SIP gateway 2 to PBX B | SIP gateway 2 acknowledges PBX B’s Connect message. |

At this point, a two-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>Disconnect—PBX B to SIP gateway 2</td>
<td>Once User B hangs up, PBX B sends a Disconnect message to SIP gateway 2. The Disconnect message starts the call session termination process.</td>
</tr>
<tr>
<td>20</td>
<td>BYE—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a BYE request to SIP gateway 1. The BYE request indicates that User B wants to release the call. Because it is User B that wants to terminate the call, the Request-URI field is now replaced with PBX A’s SIP URL and the From field contains User B’s SIP URL.</td>
</tr>
<tr>
<td>21</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>22</td>
<td>Release—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 sends a Release message to PBX B.</td>
</tr>
<tr>
<td>23</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>24</td>
<td>200 OK—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a 200 OK response to SIP gateway 2. The 200 OK response notifies SIP gateway 2 that SIP gateway 1 has received the BYE request.</td>
</tr>
</tbody>
</table>
Figure B-2 and Figure B-3 illustrate a successful gateway-to-gateway call setup and disconnect via a proxy server. In these scenarios, the two end users are User A and User B. User A is located at PBX A. PBX A is connected to SIP gateway 1 via a T1/E1. SIP gateway 1 is using a proxy server. SIP gateway 1 is connected to SIP gateway 2 over an IP network. User B is located at PBX B. PBX B is connected to SIP gateway 2 (a SIP gateway) via a T1/E1. User B’s phone number is 555-0002.

In the scenario illustrated by Figure B-2, the record route feature is enabled on the proxy server. In the scenario illustrated by Figure B-3, record route is disabled on the proxy server.

When record route is enabled, the proxy server adds the Record-Route header to the SIP messages to ensure that it is in the path of subsequent SIP requests for the same call leg. The Record-Route field contains a globally reachable Request-URI that identifies the proxy server. When record route is enabled, each proxy server adds its Request-URI to the beginning of the list.

When record route is disabled, SIP messages flow directly through the SIP gateways once a call has been established.

The call flow is as follows:

1. User A calls User B via SIP gateway 1 using a proxy server.
2. User B answers the call.
3. User B hangs up.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Release Complete—PBX B to SIP gateway 2</td>
<td>PBX B sends a Release Complete message to SIP gateway 2.</td>
</tr>
<tr>
<td>26</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the session is terminated.</td>
</tr>
</tbody>
</table>

**SIP Gateway-to-SIP Gateway—Call via SIP Proxy Server**
Figure B-2  SIP Gateway-to-SIP Gateway—Call via SIP Proxy Server with Record Route Enabled
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to proxy server | SIP gateway 1 sends an INVITE request to the SIP proxy server. The INVITE request is an invitation to User B to participate in a call session.  
In the INVITE request:  
- The phone number of User B is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of User B and takes a form similar to an email address (user@host) where user is the telephone number and host is either a domain name or a numeric network address. For example, the Request-URI field in the INVITE request to User B appears as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- PBX A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability User A is ready to receive is specified.  
- The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | Call Proceeding—SIP gateway 1 to PBX A | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request. |
| 4    | INVITE—SIP proxy server to SIP gateway 2 | The SIP proxy server checks whether its own address is contained in the Via field (to prevent loops), directly copies the To, From, Call-ID, and Contact fields from the request it received from SIP gateway 1, changes the Request-URI to indicate the server to which it intends to send the INVITE request, and sends a new INVITE request to SIP gateway 2. |
| 5    | 100 Trying—SIP proxy server to SIP gateway 1 | The SIP proxy server sends a 100 Trying response to SIP gateway 1. |
| 6    | Setup—SIP gateway 2 to PBX B | SIP gateway 2 receives the INVITE request from the SIP proxy server and initiates a Call Setup with User B via PBXB. |
| 7    | 100 Trying—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a 100 Trying response to the SIP proxy server. The SIP proxy server might or might not forward the 100 Trying response to SIP gateway 1. |
| 8    | Call Proceeding—PBX B to SIP gateway 2 | PBX B sends a Call Proceeding message to SIP gateway 2 to acknowledge the Call Setup request. |
| 9    | Alerting—PBX B to SIP gateway 2 | PBX B locates User B and sends an Alert message to SIP gateway 2. User B’s phone begins to ring. |
| 10   | 180 Ringing—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a 180 Ringing response to the SIP proxy server. |
### Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>180 Ringing—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the 180 Ringing response to SIP gateway 1.</td>
</tr>
<tr>
<td>12</td>
<td>Alerting—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends an Alert message to User A via PBX A. The Alert message indicates that SIP gateway 1 has received a 180 Ringing response. User A hears the ringback tone that indicates that User B is being alerted.</td>
</tr>
</tbody>
</table>

At this point, a one-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Connect—PBX B to SIP gateway 2</td>
<td>User B answers the phone. PBX B sends a Connect message to SIP gateway 2. The connect message notifies SIP gateway 2 that the connection has been made.</td>
</tr>
<tr>
<td>14</td>
<td>200 OK—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 sends a 200 OK response (including the Record-Route header received in the INVITE) to the SIP proxy server. The 200 OK response notifies the SIP proxy server that the connection has been made. If User B supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and User A’s media capability in the 200 OK response. If User B does not support the media capability advertised by User A, it sends back a 400 Bad Request response with a 304 Warning header field. The SIP proxy server must forward 200 OK responses upstream.</td>
</tr>
<tr>
<td>15</td>
<td>200 OK—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the 200 OK response that it received from SIP gateway 2 to SIP gateway 1. In the 200 OK response, the SIP proxy server forwards the Record-Route header to ensure that it is in the path of subsequent SIP requests for the same call leg. In the Record-Route field, the SIP proxy server adds its Request-URI.</td>
</tr>
<tr>
<td>16</td>
<td>Connect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Connect message to PBX A. The Connect message notifies PBX A that the connection has been made.</td>
</tr>
<tr>
<td>17</td>
<td>Connect ACK—PBX A to SIP gateway 1</td>
<td>PBX A acknowledges SIP gateway 1’s Connect message.</td>
</tr>
<tr>
<td>18</td>
<td>ACK—SIP gateway 1 to SIP proxy server</td>
<td>SIP gateway 1 sends an ACK to the SIP proxy server. The ACK confirms that SIP gateway 1 has received the 200 OK response from the SIP proxy server.</td>
</tr>
<tr>
<td>19</td>
<td>ACK—SIP proxy server to SIP gateway 2</td>
<td>Depending on the values in the To, From, CSeq, and Call-ID field, the SIP proxy server might process the ACK locally or proxy it. If the fields in the ACK match those in previous requests processed by the SIP proxy server, the server proxies the ACK. If there is no match, the ACK is proxied as if it were an INVITE request. The SIP proxy server forwards SIP gateway 1’s ACK response to SIP gateway 2.</td>
</tr>
<tr>
<td>20</td>
<td>Connect ACK—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 acknowledges PBX B’s Connect message. The call session is now active. The 2-way voice path is established directly between SIP gateway 1 and SIP gateway 2; not via the SIP proxy server.</td>
</tr>
</tbody>
</table>

At this point, a two-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>Disconnect—PBX B to SIP gateway 2</td>
<td>After the call is completed, PBX B sends a Disconnect message to SIP gateway 2. The Disconnect message starts the call session termination process.</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>BYE—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 sends a BYE request to the SIP proxy server. The BYE request indicates that User B wants to release the call. Because it is User B that wants to terminate the call, the Request-URI field is now replaced with PBX A’s SIP URL and the From field contains User B’s SIP URL.</td>
</tr>
<tr>
<td>23</td>
<td>BYE—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the BYE request to SIP gateway 1.</td>
</tr>
<tr>
<td>24</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>25</td>
<td>Release—SIP gateway 2 to PBX B</td>
<td>After the call is completed, SIP gateway 2 sends a Release message to PBX B.</td>
</tr>
<tr>
<td>26</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>27</td>
<td>200 OK—SIP gateway 1 to SIP proxy server</td>
<td>SIP gateway 1 sends a 200 OK response to the SIP proxy server. The 200 OK response notifies SIP gateway 2 that SIP gateway 1 has received the BYE request.</td>
</tr>
<tr>
<td>28</td>
<td>200 OK—SIP proxy server to SIP gateway 2</td>
<td>The SIP proxy server forwards the 200 OK response to SIP gateway 2.</td>
</tr>
<tr>
<td>29</td>
<td>Release Complete—PBX B to SIP gateway 2</td>
<td>PBX B sends a Release Complete message to SIP gateway 2.</td>
</tr>
<tr>
<td>30</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session is terminated.</td>
</tr>
</tbody>
</table>
Figure B-3  SIP Gateway-to-SIP Gateway—Call via a Proxy Server with Record Route Disabled
### Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP proxy server | SIP gateway 1 sends an INVITE request to the SIP proxy server. The INVITE request is an invitation to User B to participate in a call session. In the INVITE request:  
  • The phone number of User B is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of User B and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to User B appears as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
  • PBX A is identified as the call session initiator in the From field.  
  • A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  • The transaction number within a single call leg is identified in the CSeq field.  
  • The media capability User A is ready to receive is specified.  
  • The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | Call Proceeding—SIP gateway 1 to PBX A | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request. |
| 4    | INVITE—SIP proxy server to SIP gateway 2 | The SIP proxy server checks whether its own address is contained in the Via field (to prevent loops), directly copies the To, From, Call-ID, and Contact fields from the request it received from SIP gateway 1, changes the Request-URI to indicate the server to which it intends to send the INVITE request, and sends a new INVITE request to SIP gateway 2. |
| 5    | 100 Trying—SIP proxy server to SIP gateway 1 | The SIP proxy server sends a 100 Trying response to SIP gateway 1. |
| 6    | Setup—SIP gateway 2 to PBX B | SIP gateway 2 receives the INVITE request from the SIP proxy server and initiates a Call Setup with User B via PBXB. |
| 7    | 100 Trying—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a 100 Trying response to the SIP proxy server. The SIP proxy server might or might not forward the 100 Trying response to SIP gateway 1. |
| 8    | Call Proceeding—PBX B to SIP gateway 2 | PBX B sends a Call Proceeding message to SIP gateway 2 to acknowledge the Call Setup request. |
| 9    | Alerting—PBX B to SIP gateway 2 | PBX B locates User B and sends an Alert message to SIP gateway 2. User B’s phone begins to ring. |
| 10   | 180 Ringing—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a 180 Ringing response to the SIP proxy server. |
### Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>180 Ringing—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the 180 Ringing response to SIP gateway 1.</td>
</tr>
<tr>
<td>12</td>
<td>Alerting—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends an Alert message to User A via PBX A. The Alert message indicates that SIP gateway 1 has received a 180 Ringing response. User A hears the ringback tone that indicates that User B is being alerted. At this point, a one-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.</td>
</tr>
<tr>
<td>13</td>
<td>Connect—PBX B to SIP gateway 2</td>
<td>User B answers the phone. PBX B sends a Connect message to SIP gateway 2. The Connect message notifies SIP gateway 2 that the connection has been made.</td>
</tr>
<tr>
<td>14</td>
<td>200 OK—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 sends a 200 OK response to the SIP proxy server. The 200 OK response notifies the SIP proxy server that the connection has been made. If User B supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and User A’s media capability in the 200 OK response. If User B does not support the media capability advertised by User A, it sends back a 400 Bad Request response with a 304 Warning header field. The SIP proxy server must forward 200 OK responses upstream.</td>
</tr>
<tr>
<td>15</td>
<td>200 OK—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the 200 OK response that it received from SIP gateway 2 to SIP gateway 1.</td>
</tr>
<tr>
<td>16</td>
<td>Connect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Connect message to PBX A. The Connect message notifies PBX A that the connection has been made.</td>
</tr>
<tr>
<td>17</td>
<td>Connect ACK—PBX A to SIP gateway 1</td>
<td>PBX A acknowledges SIP gateway 1’s Connect message.</td>
</tr>
<tr>
<td>18</td>
<td>ACK—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends an ACK to SIP gateway 2. The ACK confirms that SIP gateway 1 has received the 200 OK response from the SIP proxy server.</td>
</tr>
<tr>
<td>19</td>
<td>Connect ACK—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 acknowledges PBX B’s Connect message. The call session is now active. The 2-way voice path is established directly between SIP gateway 1 and SIP gateway 2: not via the SIP proxy server. At this point, a two-way voice path is established between SIP gateway 1 and PBX A and between SIP gateway 2 and PBX B. A two-way RTP channel is established between SIP gateway 1 and SIP gateway 2.</td>
</tr>
<tr>
<td>20</td>
<td>Disconnect—PBX B to SIP gateway 2</td>
<td>After the call is completed, PBX B sends a Disconnect message to SIP gateway 2. The Disconnect message starts the call session termination process.</td>
</tr>
<tr>
<td>21</td>
<td>BYE—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a BYE request to SIP gateway 1. The BYE request indicates that User B wants to release the call. Because it is User B that wants to terminate the call, the Request-URI field is now replaced with PBX A’s SIP URL and the From field contains User B’s SIP URL.</td>
</tr>
<tr>
<td>22</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>23</td>
<td>Release—SIP gateway 2 to PBX B</td>
<td>After the call is completed, SIP gateway 2 sends a Release message to PBX B.</td>
</tr>
<tr>
<td>24</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
</tbody>
</table>
## Call Flow Scenarios for Successful Calls

**SIP IP Phone-to-SIP IP Phone Call Forward Unconditionally**

FigureB-4 and FigureB-5 illustrate a successful SIP IP phone-to-SIP IP phone call forward unconditionally via a SIP proxy. In these scenarios, the three end users and endpoints are identified as Alice at SIP IP phone A, Bob at SIP IP phone B, and Carol at SIP IP phone C. Bob’s calls are configured to forward to Carol unconditionally. FigureB-4 illustrates the call as processed by a recursive proxy and FigureB-5 illustrates the call as processed by a non-recursive proxy.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>200 OK—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a 200 OK response to SIP gateway 2. The 200 OK response notifies SIP gateway 2 that SIP gateway 1 has received the BYE request.</td>
</tr>
<tr>
<td>26</td>
<td>Release Complete—PBX B to SIP gateway 2</td>
<td>PBX B sends a Release Complete message to SIP gateway 2.</td>
</tr>
<tr>
<td>27</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session is terminated.</td>
</tr>
</tbody>
</table>
Figure B-4  SIP IP Phone-to-SIP IP Phone Call Forward Unconditionally Call Setup via Recursive Proxy

1. INVITE Bob@company.com
2. INVITE Carol@IPhoneC.company.com
   CC-Diversion: Bob@company.com, reason="unconditional"
3. 180 Ringing
4. 180 Ringing
5. 200 OK
6. 200 OK
7. ACK

2-way RTP channel 1 between SIP IP phones A and C established
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- Alice at SIP IP phone A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability SIP IP phone A is ready to is specified in the SDP.  
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | INVITE—SIP proxy server to SIP IP phone C | The SIP proxy server determines that Bob’s calls have been configured to forward unconditionally to Carol at SIP IP phone C. The SIP proxy server sends an INVITE request to Carol at SIP IP phone C. In the INVITE request, the proxy server changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 3    | 180 Ringing—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 180 Ringing response to the SIP proxy server. |
| 4    | 180 Ringing—SIP proxy server to SIP IP phone A | The SIP proxy server forwards the 180 Ringing response to SIP IP phone A. |
| 5    | 200 OK—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 200 OK response to SIP IP phone A. The 200 OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook).  
If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |
| 6    | ACK—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that User A’s SIP IP phone has received the 200 OK response from User C’s SIP IP phone. |

At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP C phone.
Figure B-5  SIP IP Phone-to-SIP IP Phone Call Forward Unconditionally via Non-recursive Proxy

1. INVITE Bob@company.com
2. 302 Moved Temporarily
   Contact Carol@IPhoneC.company.com
   CC: diversion: Bob@company.com, reason="unconditional"
3. INVITE Carol@IPhoneC.company.com
   CC: diversion: Bob@company.com, reason="unconditional"
4. 180 Ringing
5. 200 OK
6. ACK

2-way RTP channel between SIP IP phones A and C established
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- Alice at SIP IP phone A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability SIP IP phone A is ready to is specified in the SDP.  
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | 302 Moved Temporarily—SIP proxy server to SIP IP phone A | The SIP proxy server determines that Bob’s calls have been configured to forward unconditionally to Carol at SIP IP phone C. The SIP proxy server sends an 302 Moved Temporarily message to SIP IP phone A. In the 302 Moved Temporarily message, Carol at SIP IP phone C is added as the Contact and a CC-Diversion header is added that contains the Request-URI from the initial INVITE and the reason for the diversion. |
| 3    | INVITE—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an INVITE request to Carol at SIP IP phone C. The INVITE request contains a CC-Diversion header that contains the Request-URI from the initial INVITE request and the reason for the diversion. |
| 4    | 180 Ringing—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 180 Ringing response to SIP IP phone A. |
| 5    | 200 OK—SIP IP phone C to SIP IP phone A | SIP IP phone C sends a 200 OK response to SIP IP phone A. The 200 OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook). If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |
| 6    | ACK—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C. |

At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.
SIP IP Phone-to-SIP IP Phone Call Forward on Busy

FigureB-6 and FigureB-7 illustrate a successful SIP IP phone-to-SIP IP phone call forward on busy via a SIP proxy. In these scenarios, the three end users are identified as User A, User B, and User C. User B’s calls are configured to forward to User C when User B’s SIP IP phone sends a 486 Busy Here response. FigureB-6 illustrates the call as processed by a recursive proxy and FigureB-7 illustrates the call as processed by a non-recursive proxy.
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- Alice at SIP IP phone A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability SIP IP phone A is ready to is specified in the SDP.  
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
<p>| 2    | INVITE—SIP proxy server to SIP IP phone B | The proxy server forwards the INVITE request to Bob at SIP IP phone B. |
| 3    | 486 Busy Here—SIP IP phone B to the SIP proxy server | SIP IP phone B sends a 486 Busy response to the SIP proxy server. The 486 Busy Here response is a client error response that indicates that Bob at SIP IP phone B was successfully contacted but Bob was either unwilling or unable to take another call. |
| 4    | INVITE—SIP proxy server to SIP IP phone C | The SIP proxy server sends an INVITE request to Carol at SIP IP phone C to which Bob’s calls have been configured to forward on busy. In the INVITE request, the proxy server changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 5    | 180 Ringing—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 180 Ringing response to the SIP proxy server. |
| 6    | 180 Ringing—SIP proxy server to SIP IP phone A | The SIP proxy server forwards the 180 Ringing response to SIP IP phone A. |
| 7    | 200 OK—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 200 OK response to SIP IP phone A. If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |</p>
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>200 OK—SIP proxy server to SIP IP phone A</td>
<td>The SIP proxy server forwards the 200 OK response to SIP IP phone A. The 200OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook).</td>
</tr>
<tr>
<td>9</td>
<td>ACK—SIP IP phone A to SIP IP phone C</td>
<td>SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C.</td>
</tr>
</tbody>
</table>

At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.

**Figure B-7  SIP IP Phone-to-SIP IP Phone Call Forward on Busy Call Setup via Non-recursive Proxy**
## Call Flow Scenarios for Successful Calls

### Step 1: INVITE—SIP IP phone A to SIP proxy server

Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session.

In the INVITE request:

- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (`user@host` where `user` is the telephone number and `host` is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.
- Alice at SIP IP phone A is identified as the call session initiator in the From field.
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.
- The transaction number within a single call leg is identified in the CSeq field.
- The media capability SIP IP phone A is ready to be specified in the SDP.
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP.

### Step 2: INVITE—SIP proxy server to SIP IP phone B

The SIP proxy server forwards the INVITE request to Bob at SIP IP phone B.

### Step 3: 486 Busy Here—SIP IP phone B to the SIP proxy server

SIP IP phone B sends a 486 Busy response to the SIP proxy server. The 486 Busy Here response is a client error response that indicates that Bob at SIP IP phone B phone was successfully contacted but Bob was either unwilling or unable to take another call.

### Step 4: 302 Moved Temporarily—SIP proxy server to SIP IP phone A

The SIP proxy server sends an 302 Moved Temporarily message to SIP IP phone A. In the 302 Moved Temporarily message, Carol at SIP IP phone C is added as the Contact and a CC-Diversion header is added that contains the Request-URI from the initial INVITE and the reason for the diversion.

### Step 5: INVITE—SIP proxy server to SIP IP phone C

The SIP proxy server sends an INVITE request to Carol at SIP IP phone C to which Bob’s calls have been configured to forward on busy. In the INVITE request, the proxy server changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion.

### Step 6: 180 Ringing—SIP IP phone C to SIP IP phone A

SIP IP phone C sends a 180 Ringing response to SIP IP phone A.
### Call Flow Scenarios for Successful Calls

**SIP IP Phone-to-SIP IP Phone Call Forward No Answer**

FigureB-8 and FigureB-9 illustrate a successful SIP IP phone-to-SIP IP phone call forward when no answer via a SIP proxy. In these scenarios, the three end users are identified as User A, User B, and User C. User B’s calls are configured to forward to User C when an response timeout occurs. FigureB-8 illustrates the call as processed by a recursive proxy and FigureB-9 illustrates the call as processed by a non-recursive proxy.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 7    | 200 OK—SIP IP phone C to SIP IP phone A | SIP IP phone C sends a 200 OK response to SIP IP phone A. The 200OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook).
If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |
| 8    | ACK—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C. |

At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.
Figure B-8  SIP IP Phone-to-SIP IP Phone Call Forward No Answer Call Setup via Recursive Proxy

1. INVITE Bob@company.com
2. INVITE Bob@IPphoneB.company.com
3. 180 Ringing
   Call forward no answer timeout occurs
4. INVITE Carol@IPphoneC.company.com
   CC-Diversion: Bob@company.com;reason="no-answer"
5. 180 Ringing
6. 200 OK
7. 200 OK
8. ACK

2-way RTP channel between SIP IP phones A and C established
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- Alice at SIP IP phone A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability SIP IP phone A is ready to is specified in the SDP.  
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | INVITE—SIP proxy server to SIP IP phone B | The proxy server forwards the INVITE request to Bob at SIP IP phone B. |
| 3    | 180 Ringing—SIP IP phone B to the SIP proxy server | SIP IP phone B sends a 180 Ringing response to the SIP proxy server.  
Call forward no answer timer expires. |
| 4    | INVITE—SIP proxy server phone to SIP IP phone C | The SIP proxy server sends an INVITE request to Carol at SIP IP phone C to which Bob’s calls have been configured to forward when there is no answer. In the INVITE request, SIP IP phone A changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 5    | 180 Ringing—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 180 Ringing response to the SIP proxy server. |
| 6    | 200 OK—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 200 OK response to SIP IP phone A.  
If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |
| 7    | 200 OK—SIP proxy server to SIP IP phone A | The SIP proxy server forwards the 200 OK response to SIP IP phone A. The 200 OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook). |
### Call Flow Scenarios for Successful Calls

**Figure B-9** SIP Phone-to-SIP Phone Call Forward No Answer Setup via Non-recursive Proxy

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>ACK—SIP IP phone A to SIP IP phone C</td>
<td>SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C. At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.</td>
</tr>
<tr>
<td>Step</td>
<td>Action</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
  - Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
  - Alice at SIP IP phone A is identified as the call session initiator in the From field.  
  - A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  - The transaction number within a single call leg is identified in the CSeq field.  
  - The media capability SIP IP phone A is ready to is specified in the SDP.  
  - The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | INVITE—SIP proxy server to SIP IP phone B | The SIP proxy server forwards the INVITE request to Bob at SIP IP phone B. |
| 3    | 180 Ringing—SIP IP phone B to the SIP proxy server | SIP IP phone B sends a 180 Ringing response to the SIP proxy server. Timeout to INVITE request occurs. |
| 4    | 302 Moved Temporarily—SIP proxy server to SIP IP phone A | The SIP proxy server sends an 302 Moved Temporarily message to SIP IP phone A. In the 302 Moved Temporarily message, Carol at SIP IP phone C is added as the Contact and a CC-Diversion header is added that contains the Request-URI from the initial INVITE and the reason for the diversion. |
| 5    | INVITE—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an INVITE request to Carol at SIP IP phone C to which Bob’s calls have been configured to forward when Bob is unavailable. In the INVITE request, the SIP proxy server changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 6    | 180 Ringing—SIP IP phone C to SIP IP phone A | SIP IP phone C sends a 180 Ringing response to SIP IP phone A. |
SIP IP Phone-to-SIP IP Phone Call Forward Unavailable

FigureB-10 and FigureB-11 illustrate a successful SIP IP phone-to-SIP IP phone call forward when the callee is unavailable via a SIP proxy. In these scenarios, the three end users are identified as User A, User B, and User C. UserB’s calls are configured to forward to User C when User B is unavailable. FigureB-10 illustrates the call as processed by a recursive proxy and FigureB-11 illustrates the call as processed by a non-recursive proxy.
Figure B-10 SIP IP Phone-to-SIP IP Phone Call Forward Unavailable Call Setup via Recursive Proxy
## Call Flow Scenarios for Successful Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
- Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
- Alice at SIP IP phone A is identified as the call session initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability SIP IP phone A is ready to is specified in the SDP.  
- The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | 100 Trying—SIP proxy server to SIP IP phone A | The SIP proxy server sends a 100 Trying response to the INVITE request sent by SIP IP phone A. The 100 Trying response indicates that the INVITE request has been received by the SIP proxy server but that Bob at SIP IP phone B has not yet been located and that some unspecified action, such as a database consultation, is taking place. |
| 3 to 5 | INVITE—proxy server to SIP IP phone B | The SIP proxy server forwards the INVITE request to Bob at SIP IP phone B.  
If call forward unavailable timer expires. |
| 6    | INVITE—SIP proxy server to SIP IP phone C | The SIP proxy server sends an INVITE request to Carol at SIP IP phone C to which Bob’s calls have been configured to forward when there is no answer. In the INVITE request, SIP IP phone A changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 7    | 180 Ringing—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 180 Ringing response to the SIP proxy server. |
| 8    | 200 OK—SIP IP phone C to SIP proxy server | SIP IP phone C sends a 200 OK response to SIP IP phone A.  
If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field. |
### Call Flow Scenarios for Successful Calls

#### Figure B-11  SIP IP Phone-to-SIP IP Phone Call Forward Unavailable Call Setup via Non-recursive Proxy

<table>
<thead>
<tr>
<th>Step</th>
<th>Action Description</th>
<th>Detailed Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>200 OK—SIP proxy server to SIP IP phone A</td>
<td>The SIP proxy server forwards the 200 OK response to SIP IP phone A. The 200 OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone went off-hook).</td>
</tr>
<tr>
<td>10</td>
<td>ACK—SIP IP phone A to SIP IP phone B</td>
<td>SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C.</td>
</tr>
</tbody>
</table>

At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE—SIP IP phone A to SIP proxy server | Alice’s SIP IP phone A sends an INVITE request to the proxy server. The INVITE request is an invitation to Bob to participate in a call session. In the INVITE request:  
  • Bob’s phone number is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies Bob’s address and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to Bob might appear as “INVITE sip:555-0002@companyb.com; user=phone.” The “user=phone” parameter distinguishes that the Request-URI address is a telephone number rather than a user name.  
  • Alice at SIP IP phone A is identified as the call session initiator in the From field.  
  • A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  • The transaction number within a single call leg is identified in the CSeq field.  
  • The media capability SIP IP phone A is ready to is specified in the SDP.  
  • The port on which SIP IP phone A is prepared to receive the RTP data is specified in the SDP. |
| 2    | 100 Trying—SIP proxy server to SIP IP phone A | The SIP proxy server sends a 100 Trying response to the INVITE request sent by SIP IP phone A. The 100 Trying response indicates that the INVITE request has been received by the SIP proxy server but that Bob has not yet been located and that some unspecified action, such as a database consultation, is taking place. |
| 3 to 5 | INVITE—proxy server to SIP IP phone B | The SIP proxy server forwards the INVITE request to Bob at SIP IP phone B. Call forward unavailable timer expires. |
| 6    | 302 Moved Temporarily—SIP proxy server to SIP IP phone A | The SIP proxy server sends an 302 Moved Temporarily message to SIP IP phone A. In the 302 Moved Temporarily message, Carol at SIP IP phone C is added as the Contact and a CC-Diversion header is added that contains the Request-URI from the initial INVITE and the reason for the diversion. |
| 7    | INVITE—SIP IP phone A to SIP IP phone C | SIP IP phone A sends an INVITE request to Carol at SIP IP phone C to which Bobs calls have been configured to forward when there is no answer. In the INVITE request, SIP IP phone A changes the Request-URI to divert the request to Carol at SIP IP phone C and adds a CC-Diversion header containing the Request-URI from the initial INVITE request and the reason for the diversion. |
| 8    | 180 Ringing—SIP IP phone C to SIP IP phone A | SIP IP phone C sends a 180 Ringing response to SIP IP phone A. |
### Call Flow Scenarios for Failed Calls

This section describes call flows for the following scenarios, which illustrate successful calls:

- **SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User is Busy**, page B-33
- **SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User Does Not Answer**, page B-35
- **SIP Gateway-to-SIP Gateway via SIP Redirect Server—Client, Server, or Global Error**, page B-38
- **SIP Gateway-to-SIP Gateway via SIP Proxy Server—Called User is Busy**, page B-39
- **SIP Gateway-to-SIP Gateway via SIP Proxy Server—Client or Server Error**, page B-41
- **SIP Gateway-to-SIP Gateway via SIP Proxy Server—Global Error**, page B-43
- **SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled**, page B-46
- **SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route enabled**, page B-58
- **SIP Phone to SIP/H.323 Gateway—Call via SIP Redirect Server**, page B-71
- **SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled (Call failed with a 503 Service Unavailable response)**, page B-82

**Note:**

The messages are provided as examples for reference only.

### SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User is Busy

*Figure B-12* illustrates an unsuccessful call in which User A initiates a call to User B but User B is on the phone and is unable or unwilling to accept another call.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>200 OK—SIP IP phone C to SIP IP phone A</td>
<td>SIP IP phone C sends a 200 OK response to SIP IP phone A. The 200 OK response notifies the SIP IP phone A that Carol has answered the phone (for example, the handset of SIP IP phone C went off-hook). If SIP IP phone C supports the media capability advertised in the INVITE message sent by the SIP proxy server, it advertises the intersection of its own and SIP IP phone A’s media capability in the 200 OK response. If SIP IP phone C does not support the media capability advertised by SIP IP phone A, it sends back a 400 Bad Request response with a “Warning: 304 Codec negotiation failed” header field.</td>
</tr>
<tr>
<td>10</td>
<td>ACK—SIP IP phone A to SIP IP phone C</td>
<td>SIP IP phone A sends an ACK to SIP IP phone C. The ACK confirms that SIP IP phone A has received the 200 OK response from SIP IP phone C. At this point, a two-way RTP channel is established between SIP IP phone A and SIP IP phone C.</td>
</tr>
</tbody>
</table>
Call Flow Scenarios for Failed Calls

Figure B-12  SIP Gateway-to-SIP Gateway Call via a SIP Redirect Server—Called User is Busy

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP redirect server | SIP gateway 1 sends an INVITE request to the SIP redirect server. The INVITE request is an invitation to User B to participate in a call session. In the INVITE request:  
• The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.  
• PBX A is identified as the call session initiator in the From field.  
• A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
• The transaction number within a single call leg is identified in the CSeq field.  
• The media capability User A is ready to receive is specified.  
• The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | 302 Moved Temporarily—SIP redirect server to SIP gateway 1 | SIP redirect server sends a 302 Moved Temporarily message to SIP gateway 1. The message indicates that User B is not available and includes instructions to locate User B. |
| 4    | ACK—SIP gateway 1 to SIP redirect server | SIP gateway 1 acknowledges the 302 Moved Temporarily response with an ACK.                                                                                                                                   |
Call Flow Scenarios for Failed Calls

SIP Gateway-to-SIP Gateway via SIP Redirect Server—Called User Does Not Answer

FIGURE B-13 illustrates an unsuccessful call in which User A initiates a call to User B but User B does not answer.
Call Flow Scenarios for Failed Calls

Figure B-13 SIP Gateway-to-SIP Gateway Call via a SIP Redirect Server—Called User is Does Not Answer

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP redirect server | SIP gateway 1 sends an INVITE request to the SIP redirect server. The INVITE request is an invitation to User B to participate in a call session. In the INVITE request:  
  - The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.  
  - PBX A is identified as the call session initiator in the From field.  
  - A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  - The transaction number within a single call leg is identified in the CSeq field.  
  - The media capability User A is ready to receive is specified.  
  - The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>302 Moved Temporarily—SIP redirect server to SIP gateway 1</td>
<td>SIP redirect server sends a 302 Moved Temporarily message to SIP gateway 1. The message indicates that User B is not available and includes instructions to locate User B.</td>
</tr>
<tr>
<td>4</td>
<td>ACK—SIP gateway 1 to SIP redirect server</td>
<td>SIP gateway 1 acknowledges the 302 Moved Temporarily response with an ACK.</td>
</tr>
<tr>
<td>5</td>
<td>INVITE—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a new INVITE request to User B. The new INVITE request includes a new address for User B, a higher transaction number in the CSeq field, but the same Call-ID as the first INVITE request.</td>
</tr>
<tr>
<td>6</td>
<td>Call Proceeding—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request.</td>
</tr>
<tr>
<td>7</td>
<td>Setup—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 receives the INVITE request from SIP gateway 1 and initiates a Call Setup with User B via PBX B.</td>
</tr>
<tr>
<td>8</td>
<td>100 Trying—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a 100 Trying response to the INVITE request sent by SIP gateway 1. The 100 Trying message indicates that the INVITE request has been received by SIP gateway 2 but that User B has not yet been located and that some unspecified action, such as a database consultation, is taking place.</td>
</tr>
<tr>
<td>9</td>
<td>Call Proceeding—PBX B to SIP gateway 2</td>
<td>PBX B sends a Call Proceeding message to SIP gateway 2 to acknowledge the Call Setup request.</td>
</tr>
<tr>
<td>10</td>
<td>Alerting—PBX B to SIP gateway 2</td>
<td>PBX B sends an Alert message to SIP gateway 2. User B’s phone begins to ring.</td>
</tr>
<tr>
<td>11</td>
<td>180 Ringing—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a 180 Ringing response to SIP gateway 1. The 180 Ringing response indicates that SIP gateway 2 has located, and is trying to alert User B.</td>
</tr>
<tr>
<td>12</td>
<td>Alerting—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends an Alert message to PBX A.</td>
</tr>
<tr>
<td>13</td>
<td>CANCEL (Ring Timeout)—SIP gateway 1 to SIP gateway 2</td>
<td>Because SIP gateway 2 did not return an appropriate response within the time specified by the Expires header in the INVITE request, SIP gateway 1 sends a SIP CANCEL request to SIP gateway 2. A CANCEL request cancels a pending request with the same Call-ID, To, From, and CSeq header field values.</td>
</tr>
<tr>
<td>14</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>15</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>16</td>
<td>Disconnect—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 sends a Disconnect message to PBX B.</td>
</tr>
<tr>
<td>17</td>
<td>200 OK—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a 200 OK response to SIP gateway 2. The 200 OK response confirms that the CANCEL request has been received.</td>
</tr>
<tr>
<td>18</td>
<td>Release Complete—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release Complete message to SIP gateway 1 and the call session attempt is terminated.</td>
</tr>
<tr>
<td>19</td>
<td>Release—PBX B to SIP gateway 2</td>
<td>PBX B sends a Release message to SIP gateway 2.</td>
</tr>
<tr>
<td>20</td>
<td>Release Complete—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 sends a Release Complete message to PBX B.</td>
</tr>
</tbody>
</table>
SIP Gateway-to-SIP Gateway via SIP Redirect Server—Client, Server, or Global Error

Figure 14 illustrates an unsuccessful call in which User A initiates a call to User B but SIP gateway 2 determines that User B does not exist at the domain specified in the INVITE request sent by SIP gateway 1. SIP gateway 2 refuses the connection.

Figure 14 SIP Gateway-to-SIP Gateway Call via a SIP Redirect Server—Client, Server, or Global Error

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP redirect server | SIP gateway 1 sends an INVITE request to the SIP redirect server. The INVITE request is an invitation to User B to participate in a call session. In the INVITE request:  
  • The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.  
  • PBX A is identified as the initiator in the From field.  
  • A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  • The transaction number within a single call leg is identified in the CSeq field.  
  • The media capability User A is ready to receive is specified.  
  • The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>300 Multiple Choice—SIP redirect server to SIP gateway 1</td>
<td>The SIP redirect server sends a 300 Multiple Choice response to SIP gateway 1. The 300 Multiple Choice response indicates that the SIP redirect server accepted the INVITE request, contacted a location server with all or part of User B’s SIP URL, and the location server provided a list of alternative locations where User B might be located. The SIP redirect server returns these possible addresses to User A in the 300 Multiple Choice response.</td>
</tr>
<tr>
<td>4</td>
<td>ACK—SIP gateway 1 to SIP redirect server</td>
<td>SIP gateway 1 acknowledges the 300 Multiple Choice response with an ACK.</td>
</tr>
<tr>
<td>5</td>
<td>INVITE—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a new INVITE request to User B. The new INVITE request includes a new address for User B, a higher transaction number in the CSeq field, but the same Call-ID as the first INVITE request.</td>
</tr>
<tr>
<td>6</td>
<td>Call Proceeding—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request.</td>
</tr>
<tr>
<td>7</td>
<td>100 Trying—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 sends a 100 Trying response to the INVITE request sent by SIP gateway 1. The 100 Trying message indicates that the INVITE request has been received by SIP gateway 2 but that User B has not yet been located and that some unspecified action, such as a database consultation, is taking place.</td>
</tr>
<tr>
<td>8</td>
<td>Class 4xx/5xx/6xx Failure—SIP gateway 2 to SIP gateway 1</td>
<td>SIP gateway 2 determines that User B does not exist at the domain specified in the INVITE request sent by SIP gateway 1. SIP gateway 2 refuses the connection and sends a 404 Not Found response to SIP gateway 1. The 404 Not Found response is a class 4.xx failure response. The call actions differ, based on the class of failure response. If SIP gateway 2 sends a class 4xx failure response (a definite failure response that is a client error), the request will not be retried without modification. If SIP gateway 2 sends a class 5xx failure response (an indefinite failure that is a server error), the request is not terminated but rather other possible locations are tried. If SIP gateway 2 sends a class 6xx failure response (a global error), the search for User B is terminated because the 6xx response indicates that a server has definite information about User B, but not for the particular instance indicated in the Request-URI field. Therefore, all further searches for this user will fail.</td>
</tr>
<tr>
<td>9</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>10</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>11</td>
<td>ACK—SIP gateway 1 to SIP gateway 2</td>
<td>SIP gateway 1 sends an ACK to SIP gateway 2. The ACK confirms that the 404 Not Found failure response has been received.</td>
</tr>
<tr>
<td>12</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session attempt is terminated.</td>
</tr>
</tbody>
</table>

### SIP Gateway-to-SIP Gateway via SIP Proxy Server— Called User is Busy

Figure B-15 illustrates an unsuccessful call in which User A initiates a call to User B but User B is on the phone and is unwilling or unable to accept another call.
Table: Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP proxy server   | SIP gateway 1 sends an INVITE request to the SIP proxy server. The INVITE request is an invitation to User B to participate in a call session. In the INVITE request:  
  • The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.  
  • PBX A is identified as the call session initiator in the From field.  
  • A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
  • The transaction number within a single call leg is identified in the CSeq field.  
  • The media capability User A is ready to receive is specified.  
  • The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | INVITE—SIP proxy server to SIP gateway 2   | The SIP proxy server checks whether its own address is contained in the Via field (to prevent loops), directly copies the To, From, Call-ID, and Contact fields from the request it received from SIP gateway 1, changes the Request-URI to indicate the server to which it intends to send the INVITE request, and sends a new INVITE request to SIP gateway 2. |
| 4    | Call Proceeding—SIP gateway 1 to PBX A     | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request.                                                                                                                                 |
### Call Flow Scenarios for Failed Calls

**SIP Gateway-to-SIP Gateway via SIP Proxy Server—Client or Server Error**

_Figure B-16_ illustrates an unsuccessful call in which User A initiates a call to User B but there are no more channels available on SIP gateway 2. Therefore, SIP gateway 2 refuses the connection.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Setup—SIP gateway 2 to PBX B</td>
<td>SIP gateway 2 receives the INVITE request from the SIP proxy server and initiates a Call Setup with UserB via PBX B.</td>
</tr>
<tr>
<td>6</td>
<td>100 Trying—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server sends a 100 Trying response to SIP gateway 1.</td>
</tr>
<tr>
<td>7</td>
<td>100 Trying—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 sends a 100 Trying response to the SIP proxy server.</td>
</tr>
<tr>
<td>8</td>
<td>Release Complete (Busy)—PBX B to SIP gateway 2</td>
<td>PBX B sends a Release Complete message to SIP gateway 2. In the Release Complete message, the cause code indicates that User B is busy. The Release Complete message starts the call session termination process.</td>
</tr>
<tr>
<td>9</td>
<td>486 Busy Here—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 maps the Release message cause code (Busy) to the 486 Busy response and sends the response to the SIP proxy server. The 486 Busy Here response is a client error response that indicates that User B’s phone was successfully contacted but User B was either unwilling or unable to take another call.</td>
</tr>
<tr>
<td>10</td>
<td>486 Busy Here—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the 486 Busy response to SIP gateway 1.</td>
</tr>
<tr>
<td>11</td>
<td>Disconnect (Busy)—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>12</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>13</td>
<td>ACK—SIP gateway 1 to SIP proxy server</td>
<td>SIP gateway 1 sends an SIP ACK to the SIP proxy server.</td>
</tr>
<tr>
<td>14</td>
<td>ACK—SIP proxy server to SIP gateway 2</td>
<td>The SIP proxy server forwards the SIP ACK to SIP gateway 2. The ACK confirms that the 486 Busy Here response has been received.</td>
</tr>
<tr>
<td>15</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session attempt is terminated.</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

**Figure B-16  SIP Gateway-to-SIP Gateway Call via a SIP Proxy Server—Client or Server Error**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP proxy server | SIP gateway 1 sends an INVITE request to the SIP proxy server. The INVITE request is an invitation to User B to participate in a call session.  
In the INVITE request:  
- The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.  
- PBX A is identified as the initiator in the From field.  
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.  
- The transaction number within a single call leg is identified in the CSeq field.  
- The media capability User A is ready to receive is specified.  
- The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | INVITE—SIP proxy server to SIP gateway 2 | The SIP proxy server checks whether its own address is contained in the Via field (to prevent loops), directly copies the To, From, Call-ID, and Contact fields from the request it received from SIP gateway 1, changes the Request-URI to indicate the server to which it intends to send the INVITE request, and sends a new INVITE request to SIP gateway 2. |
| 4    | Call Proceeding—SIP gateway 1 to PBX A | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request. |
## Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>100 Trying—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server sends a 100 Trying response to SIP gateway 1.</td>
</tr>
<tr>
<td>6</td>
<td>100 Trying—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 sends a 100 Trying response to the SIP proxy server.</td>
</tr>
<tr>
<td>7</td>
<td>Class 4xx/5xx/6xx Failure—SIP gateway 2 to SIP proxy server</td>
<td>SIP gateway 2 determines that it does not have any more channels available, refuses the connection, and sends a SIP 503 Service Unavailable response to the SIP proxy server.</td>
</tr>
<tr>
<td>8</td>
<td>Class 4xx/5xx/6xx Failure—SIP proxy server to SIP gateway 1</td>
<td>The SIP proxy server forwards the SIP 503 Service Unavailable response to SIP gateway 1.</td>
</tr>
<tr>
<td>9</td>
<td>Disconnect—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Disconnect message to PBX A.</td>
</tr>
<tr>
<td>10</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>11</td>
<td>ACK—SIP gateway 1 to SIP proxy server</td>
<td>SIP gateway 1 sends an ACK to the SIP proxy server.</td>
</tr>
<tr>
<td>12</td>
<td>ACK—SIP proxy server to SIP gateway 2</td>
<td>The SIP proxy server forwards the SIP ACK to SIP gateway 2. The ACK confirms that the 503 Service Unavailable response has been received.</td>
</tr>
<tr>
<td>13</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session attempt is terminated.</td>
</tr>
</tbody>
</table>

### SIP Gateway-to-SIP Gateway via SIP Proxy Server—Global Error

**FigureB-17** illustrates an unsuccessful call in which User A initiates a call to User B and receives a class 6xx response.

**FigureB-17  SIP Gateway-to-SIP Gateway Call via a SIP Proxy Server—Global Error Response**
## Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Setup—PBX A to SIP gateway 1</td>
<td>Call Setup is initiated between PBX A and SIP gateway 1. The Call Setup includes the standard transactions that take place as User A attempts to call User B.</td>
</tr>
</tbody>
</table>
| 2    | INVITE—SIP gateway 1 to SIP proxy server | SIP gateway 1 sends an INVITE request to the SIP proxy server. The INVITE request is an invitation to User B to participate in a call session. 
In the INVITE request:
- The phone number of User B is inserted in the Request-URI field in the form of a SIP URL.
- PBX A is identified as the call session initiator in the From field.
- A unique numeric identifier is assigned to the call and inserted in the Call-ID field.
- The transaction number within a single call leg is identified in the CSeq field.
- The media capability User A is ready to receive is specified.
- The port on which SIP gateway 1 is prepared to receive the RTP data is specified. |
| 3    | Call Proceeding—SIP gateway 1 to PBX A | SIP gateway 1 sends a Call Proceeding message to PBX A to acknowledge the Call Setup request. |
| 4    | INVITE—SIP proxy server to SIP gateway 2 | The SIP proxy server checks whether its own address is contained in the Via field (to prevent loops), directly copies the To, From, Call-ID, and Contact fields from the request it received from SIP gateway 1, changes the Request-URI to indicate the server to which it intends to send the INVITE request, and sends a new INVITE request to SIP gateway 2. |
| 5    | Setup—SIP gateway 2 to PBX B | SIP gateway 2 receives the INVITE request from the SIP proxy server and initiates a Call Setup with User B via PBXB. |
| 6    | 100 Trying—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a 100 Trying response to the SIP proxy server. The SIP proxy server might or might not forward the 100 Trying response to SIP gateway 1. |
| 7    | 100 Trying—SIP proxy server to SIP gateway 1 | The SIP proxy server forwards the 100 Trying response to SIP gateway 1. |
| 8    | Release Complete—PBX B to SIP gateway 2 | PBX B sends a Release Complete message to SIP gateway 2. The Release Complete message starts the call session termination process. |
| 9    | 6xx Failure—SIP gateway 2 to SIP proxy server | SIP gateway 2 sends a class 6xx failure response (a global error) to the SIP proxy server. A class 6xx failure response indicates that a server has definite information about User B, but not for the particular instance indicated in the Request-URI field. All further searches for this user will fail, therefore the search is terminated. 
The SIP proxy server must forward all class 6xx failure responses to the client. |
| 10   | 6xx Failure—SIP proxy server to SIP gateway 1 | The SIP proxy server forwards the 6xx failure to SIP gateway 1. |
| 11   | Disconnect—SIP gateway 1 to PBX A | SIP gateway 1 sends a Disconnect message to PBX A. |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Release—PBX A to SIP gateway 1</td>
<td>PBX A sends a Release message to SIP gateway 1.</td>
</tr>
<tr>
<td>13</td>
<td>ACK—SIP gateway 1 to SIP proxy server</td>
<td>SIP gateway 1 sends an ACK to the SIP proxy server.</td>
</tr>
<tr>
<td>14</td>
<td>ACK—SIP proxy server to SIP gateway 2</td>
<td>The SIP proxy server sends an ACK to SIP gateway 2. The ACK confirms that the 6xx failure response has been received.</td>
</tr>
<tr>
<td>15</td>
<td>Release Complete—SIP gateway 1 to PBX A</td>
<td>SIP gateway 1 sends a Release Complete message to PBX A and the call session attempt is terminated.</td>
</tr>
</tbody>
</table>
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled

Figure B-18  Call Via SIP Proxy Server with Record-Route disabled
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1    | INVITE-SIP phone to SIP proxy server | SIP UAC sends an INVITE request to the SIP proxy server. INVITE sip:20002@proxy.cisco.com; user=phone;phone-context=000000 SIP/2.0
Via: SIP/2.0/UDP 161.44.3.207:49489
From: "555-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:20002@company.com;user=phone;phone-context=000000>
Date: Thu, 18 Mar 2000 04:48:28 UTC
Call-ID: 23-99990146-0-5894369F@161.44.3.207
Cisco-Guid: 428806444-2576941380-0-1486104925
User-Agent: Cisco IP Phone
CSeq: 1 INVITE
Max-Forwards: 6
Timestamp: 732430108
Contact: <sip:+19195550001@bounty.cisco.com:49489;user=phone>
Expires: 5
Content-Type: application/sdp
v=0
o=CiscoSystemsSIP-UserAgent 8870 5284 IN IP4 172.18.193.101
s=SIP Call
t=0 0
c=IN IP4 172.18.193.101
m=audio 20354 RTP/AVP 0 3
a=rtpmap:0 PCMU/8000
a=rtpmap:3 GSM/8000

In the INVITE request:
The phone number of called party is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of the called party and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to the called party appears as "INVITE sip:20002@proxy.cisco.com; user=phone" The "user=phone" parameter distinguishes that the Request-URI address is a telephone number rather than a user name.

A unique numeric identifier is assigned to the call and is inserted in the Call-ID field. The transaction number within a single call leg is identified in the Cseq field. The media capability the calling party is ready to receive is specified.
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>100-Trying SIP Proxy sends to UAC</td>
<td>SIP proxy server sends 100-Trying response message to the upstream UAC upon receiving the INVITE in step ++SIP/2.0 100 Trying Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:20002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Content-Length: 0</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3    | RAS LRQ - SIP Proxy sends a RAS LRQ message to a DGK | The SIP proxy server expands the "20002" number into "19193920002" number but found no static route to route the request. It then invoke the new routing module, creates a LRQ RAS message from the incoming INVITE SIP message. The LRQ message is sent to one of the DGK configured in the sipd.conf file. The SIP proxy server adds a technology prefix "001#" in front of the expanded number and use it to fill the "destinationInfo" field of the LRQ RAS message. The (decoded) RAS LRQ looks like the following example: value RasMessage ::= locationRequest :

```plaintext
{
    requestSeqNum 2519
    destinationInfo
    {
        e164 : "001#19193920002"
    }
    nonStandardData
    {
        nonStandardIdentifier h221NonStandard :
        {
            t35CountryCode 181
            t35Extension 0
            manufacturerCode 18
        }
    }
    data '8284901100ECAA98A0252200080000000E1963...'H
    }
    replyAddress ipAddress :
    {
        ip 'AC12C247'H
        port 1719
    }
    sourceInfo
    {
        h323-ID : {"genuity-sip1"}
    }
    canMapAlias TRUE
}
```
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 4    | RAS RIP - H.323 DGK sends back a RIP to the SIP proxy server | Upon receiving the RAS LRQ message from the SIP proxy server, the H.323 DGK can send back a RIP with delay timer value. SIP server should adjust timer accordingly.  

\[
\text{value RasMessage ::= requestInProgress :}  
\{  
\text{requestSeqNum 2519}  
\text{delay 9000}  
\}\n\]
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 5    | RAS LCF - H.323 DGK sends back a LCF to the SIP proxy server | ```
{
  requestSeqNum 2519
  callSignalAddress ipAddress :
  {
    ip 'AC12C250'H
    port 1720
  }
  rasAddress ipAddress :
  {
    ip 'AC12C250'H
    port 56812
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '0002400900630033003600320030002D0032002D...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
          }
        }
      }
    }
  }
``` |
AppendixB   Cisco SIP Proxy Server (CSPS) Call Flows

Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mc FALSE</td>
<td>undefinedNode FALSE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td>value LCFnonStandardInfo ::=</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>termAlias</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>h323-ID : {&quot;c3620-2-gw&quot;}.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>e164 : &quot;001#19193920002&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gkID {&quot;c3620-1-gk&quot;}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gateways</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gwType voip : NULL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gwAlias</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>h323-ID : {&quot;c3620-2-gw&quot;}.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>e164 : &quot;001#19193920002&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sigAddress</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ip 'AC12C250'H</td>
<td></td>
</tr>
<tr>
<td></td>
<td>port 1720</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>resources</td>
<td></td>
</tr>
<tr>
<td></td>
<td>{</td>
<td></td>
</tr>
<tr>
<td></td>
<td>maxDSPs 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>inUseDSPs 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>maxBChannels 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>inUseBChannels 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>activeCalls 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bandwidth 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>inuseBandwidth 0</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Action</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 6    | SIP INVITE - SIP proxy server forwards the INVITE to the gateway | The SIP proxy server receives the RAS LCF message, decode it and obtain the gateway transport address (172.18.194.80) value from the "callSignalAddressipAddress:" field of the LCF message. It then adds the SIP port number (5060) and forwards the INVITE to the gateway. Since the "001#" tech-prefix flag is turned "On" in the sipd.conf file, the "001#" string will not be stripped from the request-uri. The forwarded SIP INVITE message can be similar to the following:  
INVITE sip:001#19193920002@172.18.194.80:5060; user=phone;phone-context=000000 SIP/2.0  
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1  
Via: SIP/2.0/UDP 161.44.3.207:49489  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
Date: Thu, 18 Mar 2000 04:48:28 UTC  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
Cisco-Guid: 428806444-2576941380-0-1486104925  
User-Agent: Cisco IP PhoneCSeq:1  
INVITEMax-Forwards: 6  
Timestamp: 732430108  
Contact: <sip:+19195550001@bounty.cisco.com:49489;user=phone>  
Expires: 5  
Content-Type: application/sdp  
v=0  
o=CiscoSystemsSIP-UserAgent 8870 5284 IN IP4 172.18.193.101  
s=SIP Callt=0 0  
c=IN IP4 172.18.193.101  
m=audio 20354 RTP/AVP 0 3  
a=rtpmap:0 PCMU/8000  
a=rtpmap:3 GSM/8000 |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 7    | SIP 180 Ringing - Gateway sends 180 Ringing back to the SIP proxy server | The SIP/H.323 gateway receives the forwarded SIP INVITE message from the SIP proxy server and sends it downstream. Assume the call signal reaches the end-point and a SIP 180 Ringing is sent from the gateway up to the SIP proxy server. The SIP 180-Ringing message can be similar to the following:  
SIP/2.0 180 Ringing  
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369f@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 INVITE  
Content-Length: 0 |
| 8    | SIP 180 Ringing - SIP proxy server forwards to the UAC | The SIP proxy server receives the 180 Ringing from the gateway, it found the record in TCB and forwards the 180 Ringing upstream to the UAC.  
SIP/2.0 180 Ringing  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369f@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 INVITE  
Content-Length: 0 |
## Call Flow Scenarios for Failed Calls

### Step 9: SIP 200 OK - Gateway sends 200 OK to upstream SIP proxy server

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 9    | SIP 200 OK | The called party picks up the phone … The gateway sends a 200 OK to the SIP proxy server. The SIP 200 OK message can be similar to the following:  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
Contact: <sip:001#19195550002@172.18.194.80>  
CSeq: 1 INVITE  
Content-Length: 0  
v=0  
o=CiscoSystemsSIP- Gateway 537556 235334 IN IP4 172.18.194.80  
s=SIP Call  
t=0 0  
c=IN IP4 gateway.cisco.com  
m=audio 5004 RTP/AVP 0 3  
a=rtpmap:0 PCMU/8000  
a=rtpmap:3 GSM/8000 |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 10   | SIP 200 OK - The SIP proxy server forward the 200 OK to the calling UAC | The SIP proxy server receives the 200 OK from the gateway. It forwards it upstream to the calling UAC.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
Contact: <sip:001#19195550002@172.18.194.80>  
CSeq: 1 INVITE  
Content-Length: 0  
v=0  
o=CiscoSystemsSIP- Gateway 537556 235334 IN IP4 172.18.194.80  
s=SIP Call  
t=0 0  
c=IN IP4 gateway.cisco.com  
m=audio 5004 RTP/AVP 0 3  
a=rtpmap:0 PCMU/8000  
a=rtpmap:3 GSM/8000 |
| 11   | SIP ACK - The calling UAC sends ACK directly to the gateway | Upon receiving the 200 OK message, the UAC opens the media port and responds with ACK directly to the gateway.  
SIP/2.0 ACK  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 ACK |
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 12   | SIP BYE - The gateway sends BYE to the calling UAC                   | The callee hang up the phone … the gateway sends a BYE to the calling UAC.  
SIP/2.0 BYE  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 BYE |
| 13   | SIP 200 OK - The calling UAC sends back a 200 OK to the gateway     | The calling UAC receives the BYE from the gateway, it sends back a 200 OK.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 BYE |
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route enabled

Figure B-19  Call Via SIP Proxy Server with Record-Route enabled

- SIP Phone/UAC
- SIP Proxy Server
- Directory Gatekeeper
- SIP/H.323 Gateway

1. SIP INVITE
2. SIP 100 Trying
3. RAS LRQ
4. RAS RSP
5. RAS LCF
6. SIP INVITE
7. SIP 180 Ringing
8. SIP 180 Ringing
9. SIP 200 OK
10. SIP 200 OK
11. SIP ACK
12. SIP ACK
13. SIP BYE
14. SIP BYE
15. SIP 200 OK
16. Media out through
17. 2-way RTP Channel
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1     | INVITE-SIP phone to SIP proxy server | SIP UAC sends an INVITE request to the SIP proxy server. INVITE sip:20002@proxy.cisco.com;user=phone;phone-context=000000 SIP/2.0 Via: SIP/2.0/UDP 161.44.3.207:49489 From: "555-0001" <sip:+19195550001@bounty.cisco.com> To: <sip:20002@company.com;user=phone;phone-context=000000> Date: Thu, 18 Mar 2000 04:48:28 UTC Call-ID: 23-99990146-0-5894369F@161.44.3.207 Cisco-Guid: 428806444-2576941380-0-1486104925 User-Agent: Cisco IP Phone CSeq:1 INVITE Max-Forwards: 6 Timestamp: 732430108 Contact: <sip:+19195550001@bounty.cisco.com:49489;user=phone> Expires: 5 Content-Type: application/sdp

v=0
o=CiscoSystemsSIP-UserAgent 8870 5284 IN IP4 172.18.193.101
s=SIP Call
t=0 0
c=IN IP4 172.18.193.101
m=audio 20354 RTP/AVP 0 3
a=rtpmap:0 PCMU/8000
a=rtpmap:3 GSM/8000

In the INVITE request:
- The phone number of called party is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of the called party and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to the called party appears as "INVITE sip:20002@proxy.cisco.com; user=phone" The "user=phone" parameter distinguishes that the Request-URI address is a telephone number rather than a user name.
- A unique numeric identifier is assigned to the call and is inserted in the Call-ID field.

- The transaction number within a single call leg is identified in the Cseq field. The media capability the calling party is ready to receive is specified.
## Call Flow Scenarios for Failed Calls

### 2 100-Trying SIP Proxy sends to UAC

SIP proxy server sends 100-Trying response message to the upstream UAC upon receiving the INVITE in step 1.

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP 161.44.3.207:49489
Call-ID: 23-99990146-0-5894369F@161.44.3.207
From: "255-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:20002@company.com;user=phone;phone-context=000000>
CSeq: 1 INVITE
Content-Length: 0
```
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3     | RAS LRQ - SIP Proxy sends a RAS LRQ message to a DGK | The SIP proxy server expands the "20002" number into "9193920002" number but found no static route to route the request. It then invoke the new routing module, creates a LRQ RAS message from the incoming INVITE SIP message. The LRQ message is sent to one of the DGK configured in the sipd.conf file.

The SIP proxy server adds a technology prefix "001#" in front of the expanded number and use it to fill the "destinationInfo" field of the LRQ RAS message. The (decoded) RAS LRQ looks like the following example:

```plaintext
value RasMessage ::= locationRequest :
{
    requestSeqNum 2519
    destinationInfo
    {
        e164 : "001#19193920002"
    }
    nonStandardData
    {
        nonStandardIdentifier h221NonStandard : 
        {
            t35CountryCode 181
            t35Extension 0
            manufacturerCode 18
        }
    }
    data '8284901100ECAA98A02522000800000000E1963...'H
    }
    replyAddress ipAddress :
    {
        ip 'AC12C247'H
        port 1719
    }
    sourceInfo
    {
        h323-ID : "{genuity-sip1}"
    }
    canMapAlias TRUE
}
```
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 4     | RAS RIP - H.323 DGK sends back a RIP to the SIP proxy server | Upon receiving the RAS LRQ message from the SIP proxy server, the H.323 DGK can send back a RIP with delay timer value. SIP server should adjust timer accordingly. 

RasMessage ::= requestInProgress :
{
  requestSeqNum 2519
  delay 9000
} |
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 5     | RAS LCF - H.323 DGK sends back a LCF to the SIP proxy server | The H.323 DGK forwards the request to H.323 network and finds a SIP/H.323 gateway that can handle this particular call. It then sends back to the SIP proxy server a RAS LCF message. Value RasMessage ::= locationConfirm : {
  requestSeqNum 2519
  callSignalAddress ipAddress :
  {
    ip 'AC12C250'H
    port 1720
  }
  rasAddress ipAddress :
  {
    ip 'AC12C250'H
    port 56812
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '00024009006300360032003D0032002D...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          ...
Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>supportedPrefixes</td>
<td>{ }</td>
</tr>
<tr>
<td></td>
<td>mc FALSE</td>
<td>undefinedNode FALSE</td>
</tr>
<tr>
<td></td>
<td>value LCFnonStandardInfo ::= { }</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>termAlias</td>
<td>{h323-ID : {&quot;c3620-2-gw&quot;}, e164 : &quot;001#19193920002&quot; }</td>
</tr>
<tr>
<td></td>
<td>gkID {&quot;c3620-1-gk&quot;}</td>
<td>gateways</td>
</tr>
<tr>
<td></td>
<td>gwType voip : NULL</td>
<td>gwAlias</td>
</tr>
<tr>
<td></td>
<td>sigAddress</td>
<td>{ ip 'AC12C250'H, port 1720 }</td>
</tr>
<tr>
<td></td>
<td>resources</td>
<td></td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
|       | SIP INVITE - SIP proxy server forwards the INVITE to the gateway | To: <sip:20002@company.com;user=phone;phone-context=000000>
Date: Thu, 18 Mar 2000 04:48:28 UTC
Call-ID: 23-99990146-0-5894369F@161.44.3.207
Cisco-Guid: 428806444-2576941380-0-1486104925
User-Agent: Cisco IP Phone
CSeq:1 INVITE
Max-Forwards: 6
Timestamp: 732430108
Contact: <sip:+19193920001@bounty.cisco.com:49489;user=phone>
Expires: 5
Content-Type: application/sdp

v=0
go=CiscoSystemsSIP- UserAgent 8870 5284 IN IP4 172.18.193.101
s=SIP Call
t=0 0
c=IN IP4 172.18.193.101
m=audio 20354 RTP/AVP 0 3
a=rtpmap:0 PCMU/8000
a=rtpmap:3 GSM/8000 |
## Call Flow Scenarios for Failed Calls

### 7 SIP 180 Ringing - Gateway sends 180 Ringing back to the SIP proxy server

The SIP/H.323 gateway receives the forwarded SIP INVITE message from the SIP proxy server and sends it downstream. Assume the call signal reaches the end-point and a SIP 180 Ringing is sent from the gateway up to the SIP proxy server.

The SIP 180 Ringing message can be similar to the following:

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1
Via: SIP/2.0/UDP 161.44.3.207:49489
Record-Route: <sip:001#9195550002@proxy.cisco.com; maddr=proxy.cisco.com>
Call-ID: 23-99990146-0-5894369F@161.44.3.207
From: "555-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:20002@company.com; user=phone; phone-context=000000>
CSeq: 1 INVITE
Content-Length: 0
```

### 8 SIP 180 Ringing - SIP proxy server forwards to the UAC

The SIP proxy server receives the 180 Ringing from the gateway, it found the record in TCB and forwards the 180 Ringing upstream to the UAC.

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP 161.44.3.207:49489
Record-Route: <sip:001#9193920002@proxy.cisco.com; maddr=proxy.cisco.com>
Call-ID: 23-99990146-0-5894369F@161.44.3.207
From: "555-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:20002@company.com; user=phone; phone-context=000000>
CSeq: 1 INVITE
Content-Length: 0
```
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 9     | SIP 200 OK - Gateway sends 200 OK to upstream SIP proxy server | The called party picks up the phone … The gateway sends a 200 OK to the SIP proxy server. The SIP 200 OK message can be similar to the following:
```
SIP/2.0 200 OK
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1
Via: SIP/2.0/UDP 161.44.3.207:49489
Record-Route: < sip:001#9193920002@proxy.cisco.com; maddr=proxy.cisco.com>
Call-ID: 23-99990146-0-5894369F@161.44.3.207
From: "555-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:20002@company.com;user=phone;phone-context=000000>
CSeq: 1 INVITE
Contact: <sip:001#19193920002@172.18.194.80>
Content-Length: 0
v=0
o=CiscoSystemsSIP- Gateway 537556 235334 IN IP4 172.18.194.80
s=SIP Call
t=0 0
c=IN IP4 gateway.cisco.com
m=audio 5004 RTP/AVP 0 3
a=rtpmap:0 PCMU/8000
a=rtpmap:3 GSM/8000
``` |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>SIP 200 OK - The SIP proxy server forward the 200 OK to the calling UAC</td>
<td>The SIP proxy server receives the 200 OK from the gateway. It forwards it upstream to the calling UAC. <strong>SIP/2.0 200 OK</strong>&lt;br&gt;Via: SIP/2.0/UDP 161.44.3.207:49489&lt;br&gt;Record-Route: &lt;sip:001#<a href="mailto:19193920002@proxy.cisco.com">19193920002@proxy.cisco.com</a>; maddr=proxy.cisco.com&gt; Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:20002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Contact: <a href="">sip:001#19193920002@172.18.194.80</a> Content-Length: 0</td>
</tr>
<tr>
<td>11</td>
<td>SIP ACK - The calling UAC sends ACK to the SIP proxy</td>
<td>Upon receiving the 200 OK message, the caller UAC opens the media port and responds with an ACK to the SIP proxy. <strong>SIP/2.0 ACK</strong>&lt;br&gt;Via: SIP/2.0/UDP 161.44.3.207:49489 Route: <a href="">sip:001#19193920002@172.18.194.80</a> Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:20002@company.com;user=phone;phone-context=000000</a> CSeq: 1 ACK</td>
</tr>
<tr>
<td>Steps</td>
<td>Action</td>
<td>Description</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
| 12 | SIP ACK - The SIP proxy forwards an ACK to the gateway | Upon receiving the SIP ACK message, the SIP proxy server forwards the ACK to the downstream gateway.  
SIP/2.0 ACK  
Via: SIP/2.0/UDP 172.18.194.80:48987  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:20002@company.com;user=phone;phone-context=000000>  
CSeq: 1 ACK |
| 13 | SIP BYE - The gateway sends BYE to the SIP proxy | The callee hang up the phone … the gateway sends a BYE to the SIP proxy.  
SIP/2.0 BYE sip: +19195550001@bounty.cisco.com  
Via: SIP/2.0/UDP 172.18.194.80:5060  
Route: < sip: +19195550001@ bounty.cisco.com >  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: <sip:+19193920002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
| 14 | SIP BYE - The SIP proxy forwards they BYE to the calling party | The SIP proxy server receives the BYE from the gateway, it forwards it upstream to the calling user agent.  
SIP/2.0 BYE sip: +19195550001@bounty.cisco.com:5060  
Via: SIP/2.0/UDP 172.18.194.80:5060  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Record-Route: <sip: +19195550001@proxy.cisco.com>  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: <sip:+19193920002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 15    | SIP 200 OK - The calling UAC sends back a 200 OK to the SIP proxy | The calling UAC receives the BYE from the gateway, it sends back a 200 OK to the SIP proxy.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: <sip:+19193920002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
| 16    | SIP 200 OK - The SIP proxy forwards the 200 OK to the gateway | The SIP proxy receives the 200 OK from the calling UAC, it forwards it to the gateway.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP proxy.cisco.com:5060  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: <sip:+19193920002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
SIP Phone to SIP/H.323 Gateway— Call via SIP Redirect Server

Figure B-20 Call via SIP Redirect Server
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1     | INVITE-SIP phone to SIP redirect server | SIP UAC sends an INVITE request to the SIP redirect server. INVITE sip:50002@redirect.cisco.com;user=phone;phone-context=000000 SIP/2.0 Via: SIP/2.0/UDP 161.44.3.207:49489 From: "555-0001" <sip:+19195550001@bounty.cisco.com> To: <sip:50002@company.com;user=phone;phone-context=000000> Date: Thu, 18 Mar 2000 04:48:28 UTC Call-ID: 23-99990146-0-5894369F@161.44.3.207 Cisco-Guid: 428806444-2576941380-0-1486104925 User-Agent: Cisco IP Phone CSeq:1 INVITE Max-Forwards: 6 Timestamp: 732430108 Contact: <sip:+19195550001@bounty.cisco.com:49489;user=phone> Expires: 5 Content-Type: application/sdp v=0 o=CiscoSystemsSIP-UserAgent 8870 5284 IN IP4 172.18.193.101 s=SIP Call t=0 0 c=IN IP4 172.18.193.101 m=audio 20354 RTP/AVP 0 3 a=rtpmap:0 PCMU/8000 a=rtpmap:3 GSM/8000

In the INVITE request:

- The phone number of called party is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of the called party and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to the called party appears as "INVITE sip:50002@redirect.cisco.com; user=phone" The "user=phone" parameter distinguishes that the Request-URI address is a telephone number rather than a user name.
- A unique numeric identifier is assigned to the call and is inserted in the Call-ID field.
- The transaction number within a single call leg is identified in the Cseq field.
- The media capability the calling party is ready to receive is specified.
## Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>100-Trying SIP redirect server sends back 100 Trying to UAC</td>
<td>SIP redirect server sends 100-Trying response message to the upstream UAC upon receiving the INVITE in step 1. SIP/2.0 100 Trying Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Content-Length: 0</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>RAS LRQ - SIP redirect server sends a RAS LRQ message to a DGK</td>
<td>The SIP redirect server expands the &quot;50002&quot; number into &quot;9193650002&quot; number but found no static route. It then invoke the new routing module, creates a LRQ RAS message from the incoming INVITE SIP message. The LRQ message is sent to one of the DGK configured in the sipd.conf file. The SIP redirect server adds a technology prefix &quot;002#&quot; in front of the expanded number and use it to fill the &quot;destinationInfo&quot; field of the LRQ RAS message. The (decoded) RAS LRQ looks like the following example:</td>
</tr>
</tbody>
</table>

```plaintext
value RasMessage ::= locationRequest :
{
    requestSeqNum 2519
    destinationInfo
    {
        e164 : "002#19193650002"
    }
    nonStandardData
    {
        nonStandardIdentifier h221NonStandard :
        {
            t35CountryCode 181
            t35Extension 0
            manufacturerCode 18
        }
    }
data '8284901100ECAA98A02522000800000000E1963...'H
    replyAddress ipAddress :
    {
        ip 'AC12C247'H
        port 1719
    }
    sourceInfo
    {
        h323-ID : {"genuity-sip1"}
    }
    canMapAlias TRUE
}
```
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>RAS RIP - H.323 DGK sends back a RIP to the SIP redirect server</td>
<td>Upon receiving the RAS LRQ message from the SIP redirect server, the H.323 DGK can send back a RIP with delay timer value. SIP server should adjust timer accordingly.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>value RasMessage ::= requestInProgress :</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requestSeqNum 2519</td>
</tr>
<tr>
<td></td>
<td></td>
<td>delay 9000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>RAS LCF - H.323</td>
<td>DGK sends back a LCF to the SIP redirect server. The H.323 DGK forwards the request to H.323 network and finds a SIP/H.323 gateway that can handle this particular call. It then sends back to the SIP redirect server a RAS LCF message.</td>
</tr>
</tbody>
</table>

```
value RasMessage ::= locationConfirm :
{
  requestSeqNum 2519
  callSignalAddress ipAddress :
  {
    ip 'AC12C250'H
    port 1720
  }
  rasAddress ipAddress :
  {
    ip 'AC12C250'H
    port 56812
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '000240090063003600320030002D0032002D...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
          }
        }
      }
    }
  }
```
Call Flow Scenarios for Failed Calls

Steps | Action | Description
--- | --- | ---
| | | 

```
mc FALSE
undefinedNode FALSE
}
}
value LCFnonStandardInfo ::= 
{
  termAlias 
  
  h323-ID : 
    {"c3620-2-gw"},
  e164 : "4056701000"
}
gkID {
  "c3620-1-gk"
} gateways 
{

  
  gwType voip : NULL
  gwAlias 
  
  h323-ID : 
    {"c3620-2-gw"},
  e164 : "002#19193650002"
} sigAddress 
{
  ip 'AC12C250'H
  port 1720
} ```
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>SIP 302 Moved Temporarily - SIP redirect server sends a 302 Moved Temporarily to the UAC</td>
<td>The SIP redirect server receives the RAS LCF message, decode it and obtain the gateway transport address (172.18.194.80) value from the &quot;callSignalAddress ipAddress&quot; field of the LCF message. It then add the SIP port number (5060) and return the 302 Moved Temporarily message back to the UAC. Since the &quot;002#&quot; tech-prefix flag is turned &quot;Off&quot; in the sipd.conf file, the &quot;002#&quot; string will be stripped from the contact header. The returned SIP 302 Moved Temporarily message can be similar to the following example: SIP/2.0 302 MovedTemporarily Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Contact: <a href="">sip:19193650002@172.18.194.80:5060</a> Content-Length: 0</td>
</tr>
<tr>
<td>7</td>
<td>SIP ACK - UAC sends back a SIP ACK to the redirect server</td>
<td>Upon receiving of the 302 response message, the UAC sends back a SIP ACK to the redirect server. SIP/2.0 ACK Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 ACK</td>
</tr>
<tr>
<td>Steps</td>
<td>Action</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>8</td>
<td>SIP INVITE - UAC sends directly to the gateway</td>
<td>The UAC sends a new INVITE directly to the gateway. The SIP INVITE message can be similar to the following example.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INVITE sip:19193650002@172.18.194.80:5060; user=phone;phone-context=000000 SIP/2.0 Via: SIP/2.0/UDP 161.44.3.207:49489 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> Date: Thu, 18 Mar 2000 04:48:28 UTC Call-ID: 23-99990146-0-5894369F@161.44.3.207 Cisco-Guid: 428806444-2576941380-0-1486104925 User-Agent: Cisco IP Phone CSeq: 2 INVITE Max-Forwards: 6 Timestamp: 732430108 Contact: <a href="">sip:+19195550001@bounty.cisco.com:49489;user=phone</a> Expires: 5 Content-Type: application/sdp</td>
</tr>
<tr>
<td>9</td>
<td>SIP 180 Ringing - Gateway sends 180 Ringing back to the UAC</td>
<td>The SIP/H.323 gateway receives the SIP INVITE message from the UAC and sends it downstream. Assume the call signal reaches the end-point and a SIP 180 Ringing is sent from the gateway to the UAC. The SIP 180-Ringing message can be similar to the following example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIP/2.0 180 Ringing Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 2 INVITE Content-Length: 0</td>
</tr>
</tbody>
</table>
## Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 10    | SIP 200 OK - The gateway sends 200 OK to the calling UAC | The called party picks up the phone and the gateway sends 200 OK to the calling UAC.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 2 INVITE  
Content-Length: 0  
v=0  
o=CiscoSystemsSIP- Gateway 537556 235334 IN IP4 172.18.194.80  
s=SIP Call  
t=0 0  
c=IN IP4 gateway.cisco.com  
m=audio 5004 RTP/AVP 0 3  
a=rtpmap:0 PCMU/8000  
a=rtpmap:3 GSM/8000 |
| 11    | SIP ACK - The calling UAC sends ACK to the gateway | Upon receiving the 200 OK message, the UAC opens the media port and responds with ACK to the gateway.  
SIP/2.0 ACK  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 2 ACK |
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 12    | SIP BYE - The gateway sends BYE to the calling UAC | User hang up the phone … the gateway sends a BYE to the calling UAC.  
SIP/2.0 BYE sip:+19195550001@bounty.cisco.com  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369@161.44.3.207  
From: <sip:+1913650002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
| 13    | SIP 200 OK - The calling UAC sends back a 200 OK to the gateway | The calling UAC receives the BYE from the gateway, it sends back a 200 OK.  
SIP/2.0 200 OK  
Via: SIP/2.0/UDP 172.18.194.80:43576  
Call-ID: 23-99990146-0-5894369@161.44.3.207  
From: <sip:+19193650002@company.com;user=phone>  
To: "555-0001" <sip:+19195550001@bounty.cisco.com>  
CSeq: 1 BYE |
SIP phone-to-SIP/H.323 Gateway—Call via SIP Proxy Server with Record-Route disabled (Call failed with a 503 Service Unavailable response)

Figure B-21: Call via SIP Proxy Server with Record-Route disabled (Call failed with a 503 Service Unavailable response)
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INVITE-SIP phone to SIP proxy</td>
<td>SIP UAC sends an INVITE request to the SIP proxy server.</td>
</tr>
<tr>
<td></td>
<td>server</td>
<td><strong>INVITE sip:<a href="mailto:50002@proxy.cisco.com">50002@proxy.cisco.com</a>;user=phone;phone-context=000000 SIP/2.0</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Via: SIP/2.0/UDP 161.44.3.207:49489</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a></strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a></strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Date: Thu, 18 Mar 2000 04:48:28 UTC</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Call-ID: 23-99990146-0-5894369F@161.44.3.207</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cisco-Guid: 428806444-2576941380-0-1486104925</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>User-Agent: Cisco IP Phone</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>CSeq:1 INVITE</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Max-Forwards: 6</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Timestamp: 732430108</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Contact: <a href="">sip:+19195550001@bounty.cisco.com:49489:user=phone</a></strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Expires: 5</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Content-Type: application/sdp</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>v=0</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>o=CiscoSystemsSIP-UserAgent 8870 5284 IN IP4 172.18.193.101</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>s=SIP Call</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>t=0</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>c=IN IP4 172.18.193.101</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>m=audio 20354 RTP/AVP 0 3</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>a=rtpmap:0 PCMU/8000</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>a=rtpmap:3 GSM/8000</strong></td>
</tr>
</tbody>
</table>

In the INVITE request:

- The phone number of called party is inserted in the Request-URI field in the form of a SIP URL. The SIP URL identifies the address of the called party and takes a form similar to an email address (user@host where user is the telephone number and host is either a domain name or a numeric network address). For example, the Request-URI field in the INVITE request to the called party appears as "INVITE sip:50002@proxy.cisco.com; user=phone". The "user=phone" parameter distinguishes that the Request-URI address is a telephone number rather than a user name.

- A unique numeric identifier is assigned to the call and is inserted in the Call-ID field.

- The transaction number within a single call leg is identified in the Cseq field.

The media capability the calling party is ready to receive is specified.
### Call Flow Scenarios for Failed Calls

**Steps** | **Action** | **Description**
--- | --- | ---
2 | 100-Trying SIP Proxy sends to UAC | SIP proxy server sends 100-Trying response message to the upstream UAC upon receiving the INVITE in step 1.  
SIP/2.0 100 Trying  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 1 INVITE  
Content-Length: 0
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3     | RAS LRQ - SIP Proxy sends a RAS LRQ message to a DGK | The SIP proxy server expands the "50002" number into "9193650002" number but found no static route to route the request. It then invoke the new routing module, creates a LRQ RAS message from the incoming INVITE SIP message. The LRQ message is sent to one of the DGK configured in the sipd.conf file.

The SIP proxy server adds a technology prefix "002#" in front of the expanded number and use it to fill the "destinationInfo" field of the LRQ RAS message. The (decoded) RAS LRQ looks like the following example:

```plaintext
value RasMessage ::= locationRequest :
{
  requestSeqNum 2519
  destinationInfo
  {
    e164 : "002#19193650002"
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
  }
  data '8284901100ECAA98A025220008000000000E1963...'H
  }
  replyAddress ipAddress :
  {
    ip 'AC12C247'H
    port 1719
  }
  sourceInfo
  {
    h323-ID : {"genuity-sip1"}
  }
  canMapAlias TRUE
}
```
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 4     | RAS RIP - H.323 DGK sends back a RIP to the SIP proxy server | Upon receiving the RAS LRQ message from the SIP proxy server, the H.323 DGK can send back a RIP with delay timer value. SIP server should adjust timer accordingly. Value RasMessage ::= requestInProgress : \{
  requestSeqNum 2519
  delay 9000
\} |
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>RAS LCF - H.323 DGK sends back a LCF to the SIP proxy server</td>
<td>The H.323 DGK forwards the request to H.323 network and finds a SIP/H.323 gateway that can handle this particular call. It then sends back to the SIP proxy server a RAS LCF message.</td>
</tr>
</tbody>
</table>

```plaintext
value RasMessage ::= locationConfirm :
{
  requestSeqNum 2519
  callSignalAddress ipAddress :
  {
    ip 'AC12C250'H
    port 1720
  }
  rasAddress ipAddress :
  {
    ip 'AC12C250'H
    port 56812
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '0002400900630033003600320030002D0032002D...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
        }
      }
    }
  }
```
## Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
value LCFnonStandardInfo ::= {
  termAlias {
    h323-ID : {"c3620-2-gw"},
    e164 : "002#19193650002"
  }
  gkID {"c3620-1-gk"}
  gateways {
    {
      gwType voip : NULL
      gwAlias {
        h323-ID : {"c3620-2-gw"},
        e164 : "002#19193650002"
      }
      sigAddress {
        ip 'AC12C250'H
        port 1720
      }
    }
  }
}
```

### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>resources</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{</td>
</tr>
<tr>
<td></td>
<td></td>
<td>maxDSPs 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inUseDSPs 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>maxBChannels 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inUseBChannels 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>activeCalls 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bandwidth 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inuseBandwidth 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
</tr>
</tbody>
</table>
Appendix B Cisco SIP Proxy Server (CSPS) Call Flows

Call Flow Scenarios for Failed Calls

### Step 6: SIP INVITE - SIP proxy server forwards the INVITE to the gateway

The SIP proxy server receives the RAS LCF message, decode it and obtain the gateway transport address (172.18.194.80) value from the "callSignalAddress ipAddress" field of the LCF message. It then add the SIP port number (5060) and forward the INVITE to the gateway. Since the "002#" tech-prefix flag is turned "Off" in the sipd.conf file, the "002#" string will be stripped from the request-uri.

The forwarded SIP INVITE message can be similar to the following:

```
INVITE sip: 19193650002@172.18.194.80:5060; user=phone;phone-context=000000 SIP/2.0
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1
Via: SIP/2.0/UDP 161.44.3.207:49489
From: "555-0001" <sip:+19195550001@bounty.cisco.com>
To: <sip:50002@company.com;user=phone;phone-context=000000>
Date: Thu, 18 Mar 2000 04:48:28 UTC
Call-ID: 23-99990146-0-5894369F@161.44.3.207
Cisco-Guid: 428806444-2576941380-0-1486104925
User-Agent: Cisco IP Phone
CSeq: 1 INVITE
Max-Forwards: 6
Timestamp: 732430108
Contact: <sip:+19195550001@bounty.cisco.com:49489;user=phone>
Expires: 5
Content-Type: application/sdp

v=0
o=Cisco Systems SIP-UserAgent 8870 5284 IN IP4 172.18.193.101
s=SIP Call
t=0 0
c=IN IP4 172.18.193.101
m=audio 20354 RTP/AVP 0 3
a=rtpmap:0 PCMU/8000
a=rtpmap:3 GSM/8000
```
<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>SIP 100 Trying - Gateway sends 100 Trying back to the SIP proxy server</td>
<td>The SIP/H.323 gateway receives the forwarded SIP INVITE message from the SIP proxy server and sends 100-Trying back to the SIP proxy server. The SIP 100-Trying message can be similar to the following: SIP/2.0 100 Trying Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1 Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Content-Length: 0</td>
</tr>
<tr>
<td>8</td>
<td>SIP 100-Trying - SIP proxy server forwards to the UAC</td>
<td>The SIP proxy server receives the 100-Trying from the gateway, it found the record in TCB and forwards the 100-Trying upstream to the UAC. SIP/2.0 100-Trying Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1 Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Content-Length: 0</td>
</tr>
<tr>
<td>9</td>
<td>SIP 503 Service Unavailable—Gateway sends 503 Service Unavailable to upstream SIP proxy server</td>
<td>The gateway overloaded and sends a 503 Service Unavailable to the upstream SIP proxy server. The SIP 503 Service Unavailable message can be similar to the following: SIP/2.0 503 Service Unavailable Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1 Via: SIP/2.0/UDP 161.44.3.207:49489 Call-ID: 23-99990146-0-5894369F@161.44.3.207 From: &quot;555-0001&quot; <a href="">sip:+19195550001@bounty.cisco.com</a> To: <a href="">sip:50002@company.com;user=phone;phone-context=000000</a> CSeq: 1 INVITE Content-Length: 0</td>
</tr>
</tbody>
</table>
### Call Flow Scenarios for Failed Calls

<table>
<thead>
<tr>
<th>Steps</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
</table>
| 10    | SIP 503 Service Unavailable - The SIP proxy server forward the 503 Service Unavailable to the calling UAC | The SIP proxy server receives the 503 Service Unavailable from the gateway. It forwards it upstream to the calling UAC.  
SIP/2.0 503 Service Unavailable  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 1 INVITE  
Content-Length: 0 |
| 11    | SIP ACK - The calling UAC sends ACK to the SIP proxy server | Upon receiving the 503 Service Unavailable message, the UAC responds with ACK to the SIP proxy server.  
SIP/2.0 ACK  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 1 ACK |
| 12    | SIP ACK - The SIP proxy server sends ACK to the downstream gateway | Upon receiving the ACK from UAC, the SIP proxy server forwards the ACK to the downstream gateway.  
SIP/2.0 ACK  
Via: SIP/2.0/UDP proxy.cisco.com:48754; branch=1  
Via: SIP/2.0/UDP 161.44.3.207:49489  
Call-ID: 23-99990146-0-5894369F@161.44.3.207  
From: "555-0001" <sip:+19195550001@bounty.cisco.com>  
To: <sip:50002@company.com;user=phone;phone-context=000000>  
CSeq: 1 ACK |
Accounting Services Record Attributes

This appendix describes the standard and request-specific attributes that are logged in the Call Detail Records generated by the Accounting Services module of the CSPS.

## Standard Attributes

All requests contain the following standard radius attributes. For more details on standard Radius accounting attributes, refer to [RFC 2866] RADIUS Accounting.

### TableC-1 Standard Radius Attributes

<table>
<thead>
<tr>
<th>Radius Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 NAS IP</td>
<td>IP of the proxy</td>
</tr>
<tr>
<td>5 NAS Port</td>
<td>5</td>
</tr>
<tr>
<td>1 Use</td>
<td>User part of the URL in the &quot;From&quot; SIP header or user in &quot;Proxy-authorization&quot; SIP header, if present. This is controlled by a configuration directive (OrigUserNameSource). If the RadiusUserNameAttrAddDomain directive is On, the domain in the &quot;From&quot; header will be appended to the UserName and the User Name attribute will be in the Username@domain format. The UserName can be expanded to a fully expanded E164 number. If the user name is taken from the &quot;From&quot; header, the expansion is dependent on the user type and the NumericUserNameInterpretation directive. If the user name is taken from the &quot;Authorization&quot; or &quot;Proxy-authorization&quot; header, the expansion is dependent on the NumExpandAuthUserName directive. Expansion is based on the number expansion rules which are defined in the number expansion module.</td>
</tr>
<tr>
<td>6 Service Type</td>
<td>1</td>
</tr>
</tbody>
</table>
Vendor-specific Attributes

All requests contain the following vendor-specific attributes. For details on Cisco VSAs, refer to URL http://www.cisco.com/univercd/cc/td/doc/product/access/acs_serv/vapp_dev/vsaig3.htm#68597.

Table C-2  Vendor-specific Attributes

<table>
<thead>
<tr>
<th>Vendor-specific Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>24 h323-conf-id</td>
<td>Value of cisco-GUID header (if present)</td>
</tr>
<tr>
<td>Note: This attribute is only included if a cisco-GUID header is found in the received SIP message.</td>
<td></td>
</tr>
<tr>
<td>27 h323-call-type</td>
<td>VoIP</td>
</tr>
</tbody>
</table>

The following attributes are defined using the Cisco VSA 1 AVPair mechanism.

Table C-3  Vendor-specific Attributes—Cisco VSA 1 AVPair

<table>
<thead>
<tr>
<th>Vendor-specific Attributes</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 session-protocol</td>
<td>sip</td>
</tr>
<tr>
<td>1 call-id</td>
<td>Value of Call-Id header</td>
</tr>
<tr>
<td>1 method</td>
<td>Method name given in request line</td>
</tr>
<tr>
<td>1 incoming-req-uri</td>
<td>Request-URI given in incoming request-line</td>
</tr>
<tr>
<td>1 prev-hop-ip</td>
<td>Previous hop IP address as seen by the proxy</td>
</tr>
<tr>
<td>1 prev-hop-via</td>
<td>The &quot;sent-by&quot; portion of the topmost via when the request arrived at the proxy</td>
</tr>
<tr>
<td>1 outgoing-req-uri</td>
<td>Request-URI used in outgoing request-line</td>
</tr>
<tr>
<td>1 next-hop-ip</td>
<td>IP where the request is forwarded</td>
</tr>
<tr>
<td>1 next-hop-dn</td>
<td>Domain Name or Fully Qualified Domain Name where the request is forwarded</td>
</tr>
<tr>
<td>1 sip-hdr</td>
<td>An arbitrary message header that was found in the SIP message received by the proxy (complete header line). Inclusion of any given header is controlled by a configuration directive (AcctIncludeSIPHeader).</td>
</tr>
</tbody>
</table>
When the sipd.conf directive StatefulServer is Off, these vendor specific attributes are logged in the following order: session-protocol, call-id, method, prev-hop-via, prev-hop-ip, sip-hdr. When the sipd.conf directive StatefulServer is On, these vendor specific attributes are logged in the following order: session-protocol, call-id, method, prev-hop-via, prev-hop-ip, incoming-req-uri, outgoing-req-uri, next-hop-ip, next-hop-fqdn, sip-hdr.

Request-specific Attributes

Start Record

The start record is sent to the radius server when a 200 is received by the CSPS.

<table>
<thead>
<tr>
<th>Start Record Attributes</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 Status Type</td>
<td>start</td>
</tr>
<tr>
<td>26 h323-call-origin (VSA)</td>
<td>origin</td>
</tr>
<tr>
<td>25 h323-setup-time (VSA)</td>
<td>timestamp*</td>
</tr>
<tr>
<td>28 h323-connect-time (VSA)</td>
<td>timestamp*</td>
</tr>
</tbody>
</table>


Stop Record

The stop record is written when BYE is received by the CSPS.

<table>
<thead>
<tr>
<th>Stop Record Attributes</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 Status Type</td>
<td>stop</td>
</tr>
<tr>
<td>26 h323-call-origin (VSA)</td>
<td>answer</td>
</tr>
<tr>
<td>29 h323-disconnect-time (VSA)</td>
<td>timestamp*</td>
</tr>
</tbody>
</table>


The following section describes the disconnect cause attributes for the stop record.
### Table C-6  Disconnect Cause Attributes

<table>
<thead>
<tr>
<th>Disconnect Cause Attributes (VSA)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 BYE</td>
<td>normal</td>
</tr>
<tr>
<td>30 3xx responses</td>
<td>redirection</td>
</tr>
<tr>
<td>30 4xx responses</td>
<td>client error</td>
</tr>
<tr>
<td>30 5xx responses</td>
<td>network error</td>
</tr>
<tr>
<td>30 6xx responses</td>
<td>global error</td>
</tr>
<tr>
<td>30 proxy timeout</td>
<td>timeout</td>
</tr>
<tr>
<td>30 cancel (caller hung up)</td>
<td>user abandon</td>
</tr>
<tr>
<td><strong>A</strong></td>
<td><strong>authentication, authorization, and accounting. The network security services that provide the primary framework through which you set up access control on your router or access server.</strong></td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>AAA</strong></td>
<td>Generally, a method for resolving differences between computer addressing schemes. Address resolution usually specifies a method for mapping network layer (Layer 3) addresses to data link layer (Layer 2) addresses.</td>
</tr>
<tr>
<td><strong>address resolution</strong></td>
<td>An object or application that can be a server, a client, or both.</td>
</tr>
<tr>
<td><strong>agent</strong></td>
<td>American Standard Code for Information Interchange. 8-bit code for character representation (7 bits plus parity).</td>
</tr>
<tr>
<td><strong>ASCII</strong></td>
<td>Pattern scanning and processing language</td>
</tr>
<tr>
<td><strong>awk</strong></td>
<td>Bourne Again Shell sh-compatible command language interpreter. It executes commands from standard input or a file.</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>Establishment of (or attempt to establish) a voice or data connection between two endpoints, or between two points which provide a partial link (e.g. a trunk) between two endpoints.</td>
</tr>
<tr>
<td><strong>CDR</strong></td>
<td>Call Record Detail. A term used to describe log records for calling services. This includes information such as where the call originated, the start time, to whom the call was made, time the call ended, etc.</td>
</tr>
<tr>
<td><strong>Codec</strong></td>
<td>Coder-Decoder. Transforms analog voice into digital bit stream and vice-versa.</td>
</tr>
<tr>
<td><strong>cron</strong></td>
<td>Clock daemon that starts processes that execute commands at certain date and time as specified by crontab.</td>
</tr>
<tr>
<td><strong>D</strong></td>
<td>Dynamic Host Control Protocol. A protocol that is used to dynamically allocate and assign IP addresses. DHCP allows you to move network devices from one subnet to another without administrative attention. RFC 2131 and RFC 2132</td>
</tr>
<tr>
<td><strong>dial peer</strong></td>
<td>An addressable call endpoint. In Voice over IP (VoIP), there are two types of dialpeers: POTS and VoIP.</td>
</tr>
<tr>
<td><strong>dial plan</strong></td>
<td>A description of the dialing arrangements for customer use on a network.</td>
</tr>
<tr>
<td><strong>DNIS</strong></td>
<td>Dialed Number Identification Service. A feature of 800 and 900 lines that provides the number the caller dialed. DNIS allows one trunk group to service multiple applications, thus requiring fewer phone lines. For example, you could give one 800 number to callers in New York, one to callers in Chicago, and one to callers in LA. With DNIS, one trunk could be used to answer all those calls, playing a different, customized recording for each number called.</td>
</tr>
<tr>
<td><strong>DNS</strong></td>
<td>Domain Name System. System used in the Internet for translating names of network nodes into addresses.</td>
</tr>
<tr>
<td><strong>DSL</strong></td>
<td>Digital Subscriber Line. Public network technology that delivers high bandwidth over conventional copper wiring at limited distances. There are four types of DSL: ADSL, HDSL, SDSL, and VDSL. All are provisioned via modem pairs, with one modem located at a central office and the other at the customer site. Because most DSL technologies do not use the whole bandwidth of the twisted pair, there is room remaining for a voice channel.</td>
</tr>
<tr>
<td><strong>DTMF</strong></td>
<td>Dual Tone Multi Frequency: The paired, high- and low-frequency tones which make up touch tone dialing.</td>
</tr>
<tr>
<td><strong>E</strong></td>
<td></td>
</tr>
<tr>
<td><strong>E1</strong></td>
<td>Wide-area digital transmission scheme. E1 is the European equivalent of a T1 line. The E1's higher clock rate (2.048 MHz) allows for 32 64 Kbps channels, which include one channel for framing and one channel for D-channel information.</td>
</tr>
<tr>
<td><strong>E.164</strong></td>
<td>ITU-T recommendation for international telecommunication numbering, especially in ISDN, BISDN, and SMDS. An evolution of standard telephone numbers.</td>
</tr>
<tr>
<td><strong>end point</strong></td>
<td>SIP or H.323 terminal or gateway. An end point can call and be called. It generates and terminates the information stream.</td>
</tr>
<tr>
<td><strong>G</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Gateway</strong></td>
<td>The server that connects the VoIP network with PBXs and PSTN devices.</td>
</tr>
<tr>
<td><strong>GKTMP</strong></td>
<td>GateKeeper Transaction Message Protocol (GKTMP). It is a message protocol used as an interface between a gatekeeper and a back-end server such as Cisco NAM. This protocol is text-based and is a logical subset of H.225.0 RAS.</td>
</tr>
<tr>
<td><strong>H</strong></td>
<td></td>
</tr>
<tr>
<td><strong>H.323</strong></td>
<td>An ITU standard for transmitting data on audio, video and data conferencing on an IP-based internetwork. The H.323 standard provides for the following types of endpoints in the network: H.323 terminals, gatekeepers, MCUs, gateways.</td>
</tr>
<tr>
<td><strong>HTTP</strong></td>
<td>Hypertext Transfer Protocol. The protocol used by Web browsers and Web servers to transfer files, such as text and graphic files.</td>
</tr>
<tr>
<td><strong>HTTP Digest</strong></td>
<td>A password-based authentication method supported by LDAP servers.</td>
</tr>
</tbody>
</table>

| **ICMP** | Internet Control Message Protocol. A network-layer Internet protocol that reports errors and provides other information relevant to IP packet processing. RFC792 |
| **IETF** | Internet Engineering Task Force. Task force consisting of over 80 working groups responsible for developing Internet standards. The IETF operates under the auspices of ISOC. |
| **JRE** | Java Runtime Environment |
| **IP** | Internet Protocol. A network-layer protocol in the TCP/IP stack that offers a connectionless internetwork service. IP provides features for addressing, type-of-service (ToS) specification, fragmentation and reassembly, and security. RFC791 |
| **IPSec** | IP Security. An IETF standard that is used to provide security for transmission of sensitive information over unprotected networks such as the Internet. IPSec acts at the network layer, protecting and authenticating IP packets between participating IPSec devices (“peers”), such as Cisco routers. |
| **ISDN** | Integrated Services Digital Network. A communications protocol, offered by telephone companies, that permits telephone networks to carry data, voice, and other traffic. |
| **ISP** | Internet Service Provider. Company that provides Internet access to other companies and individuals. |
| **ITU** | International Telecommunications Union. Established by the United Nations, with membership from virtually every world government. Three primary goals are: defining and adopting telecommunications standards; regulating use of the radio frequency spectrum; and furthering world-wide telecommunications development. |

| **LCF** | Location Confirm. Registration, Admission, Status (RAS) message that the gatekeeper uses to respond to an LRQ (See LRQ in this section). It contains the transport address of the destination endpoint. |
| **LDAP** | Lightweight Directory Access Protocol. An emerging software protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network, whether on the Internet or on a corporate intranet. LDAP is a “lightweight” (smaller amount of code) version of DAP (Directory Access Protocol), which is part of X.500, a standard for directory services in a network. |
| **LEC** | Local Exchange Carrier. Local or regional telephone company that owns and operates a telephone network and the customer lines that connect to it. |
| **lm** | License Manager. It is automatically installed when the pserver is installed. It handles the storage of license keys |
| **location server** | A device that processes requests (typically from a redirect or proxy server) to provide information about the possible location of a target end user. |
**LRJ**
Location Reject. Registration, Admission, Status (RAS) message that a gatekeeper uses to reject the request from a LRQ (See LRQ in this section).

**LRQ**
Location Request. Registration, Admission, Status (RAS) message that is sent from an endpoint to request the gatekeeper to provide address translation.

**M**
**MGC**
Media gateway controller. A device that provides control of media and signaling gateways.

**MGCP**
Media Gateway Control Protocol. Protocol that helps bridge the gap between circuit-switched and IP networks. A combination of Internet Protocol Device Control (IPDC) and Simple Gateway Control Protocol (SGCP). MGCP allows external control and management of data communications devices, or “media gateways” at the edge of multiservice packet networks by software programs.

**MIB**
Management Information Base. It is a formal description of a set of network objects that can be managed by using SNMP (Simple Network Management Protocol). The format of the MIB is defined as part of the SNMP. See SNMP

**MySQL**
Database used to store and access provisioning system and subscriber feature data.

**N**
**name mapping**
Generally, the process of associating a name with a network location.

**P**
**PBX**

**POTS**
Plain Old Telephone Service. Basic telephone service supplying standard single line telephones, telephone lines, and access to the Public Switched Telephone Network.

**Provisioning GUI**
Client-provisioning client of the CSPS GUI-based provisioning system. It can be installed independent of the pserver. It requires installation of the correct version of JRE -1.3.1 (Java Runtime Environment 1.3.1).

**proxy server**
An intermediate device that receives SIP requests from a client and then initiates requests on the client’s behalf.

**PSTN**
Public Switched Telephone Network. PSTN refers to the local telephone company.

**R**
**RADIUS**
Remote Authentication Dial-In User Service. An authentication and accounting system used by many Internet Service Providers (ISPs).

**RAS**
redirect server  A device that receives SIP requests, strips out the address in the request, checks its address tables for any other addresses that may be mapped to the one in the request, and then returns the results of the address mapping to the client.

registrar server  A device that processes requests from UACs for registration of their current location. Registrar servers are often co-located with a redirect or proxy server.

RFC  Request For Comments. Document series used as the primary means for communicating information about the Internet. Some RFCs are designated by the IAB as Internet standards. Most RFCs document protocol specifications such as Telnet and FTP, but some are humorous or historical. RFCs are available online from numerous sources.

RPC  Remote Procedure Call. An external form of communication that allows objects to communicate with each other over the network. The RPC programming interface is built into each server's Client and Server subsystems to provide external communication among servers.

RPMS  Resource Pool Management Server

RSVP  IETF specification that allows applications to request dedicated bandwidth.

RTP/RTCP  Real-time Transport Protocol/Real-time Control Protocol. An IETF specification for audio and video signal management. Allows applications to synchronize and spoil audio and video information. RTP connections are established between DAP servers across the Internet after voice has been converted to IP format.


S

SAP  Session Announcement Protocol. A protocol used to assist in the advertisement of multicast multimedia conferences and other multicast sessions, and to communicate the relevant session setup information to prospective participants.

SDP  Session Description Protocol. A protocol used to describe the characteristics of multimedia sessions for the purpose of session announcement, session invitation, and other forms of multimedia session initiation. RCS 2327

sed  Stream editor. It reads text files and makes editing changes according to a script of editing commands.

signaling  Process of sending a transmission signal over a physical medium for purposes of communication.

sipd  SIP Proxy Server. Handles all call processing and SIP messages.

SIP  Session Initialization Protocol. Offers many of the same architectural features as H.323, but relies on IP-specific technologies, such as DNS. It also incorporates the concept of fixed port numbers for all devices and allows for the use of proxy servers.

SNMP  Simple Network Management Protocol
| **spa** | SIP Provisioning Agent. It is the provisioning client for sipd. It is installed if the CSPS GUI-based provisioning system is used. |
| **pserver** | Provisioning Server. It is the main server used by the CSPS GUI-based provisioning system. |
| **T** | **T1** Digital WAN carrier facility. T1 transmits DS-1 formatted data at 1.544 Mbps through the telephone-switching network, using AMI or B8ZS coding. T1 is the North American equivalent of an E1 line. |
| **TCL** | Tool command language. |
| **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. Allows files to be transferred from one computer to another over a network. |
| **U** | **UA** User Agent. See **UAC** and **UAS**. |
| **UAC** | User Agent Client. In SIP, a client application that initiates the SIP request. |
| **UAS** | User Agent Server. In SIP, a server application that contacts the user when a SIP request is received, then returns a response on behalf of the user. The response accepts, rejects or redirects the request. |
| **UCS** | Unified call services. |
| **UDP** | A 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. RFC768 |
| **URL** | Uniform Resource Locator. An identifier used to locate content that is transported via the HTTP protocol. |
| **V** | **VFC** Voice feature card. |
| **VNM** | Voice network module. |
| **VoIP** | Voice over IP. The ability to carry normal telephony-style voice over an IP-based internet with POTS-like functionality, reliability, and voice quality. |
| **VSA** | Vendor-specific attribute |
XML  eXtensible Markup Language.
Numerics

1xx responses A-3
2xx responses A-3
3xx responses A-4
4xx responses A-4
5xx responses A-6
6xx responses A-6

A

Accept header field A-6
Accept-Encoding header field A-6
Accept-Language header field A-6
Access control 3-26, 4-12
Accounting 3-24, 4-11, C-1
Agent
  master 7-1
  MIB subagent 7-11
  subagent 7-1
Allow header field A-6
Also header field A-6
Apache 4-1, 4-2, 4-3
authentication 3-13, 4-14
  services 1-2
Authorization header field A-6

B

billing services 1-2

C

Call forward 3-15, 4-16, 6-9, 6-13
Call-ID header field A-6
clients
  gateways 1-3
  phones 1-3
  SIP 1-3
compliance information A-1
Contact header field A-7
Content-Encoding header field A-7
Content-Length header field A-7
Content-Type header field A-7
conventions, documentation x
Cseq header field A-7

D

Database
  routing and registry 6-1
Date header field A-7
Debug 3-11
directory services 1-2
DNS
  description 1-6
document
  audience x
  conventions x
  objectives ix
  organization x
documentation
related xi
**E**

Encryption header field A-7  
endpoint, SIP 1-2  
ENUM 3-17, 4-18  
Error  
  log 2-1, 4-4, 5-11  
  messages 6-9, 6-23  
Expires header field A-7

**F**

Farm  
  configuring 3-3  
  multi-member 3-34  
  server 4-25  
From header field A-7  
functions  
  proxy server A-1  
  redirect server A-1

**G**

gateways 1-3  
GKTMP 3-24, 4-11

**H**

h.323 1-9  
header fields A-6  
Host  
  virtual 4-1, 4-3, 4-19

**I**

IP  
  description 1-6

**L**

Log file 5-10  
LRQ 1-9

**M**

Max-Forwards header field A-7  
methods  
  ACK A-1  
  BYE A-2  
  CANCEL A-2  
  INVITE A-1  
  OPTIONS A-2  
  REGISTER A-2  
MIB 7-1, 7-8, 7-10  
MySQL 3-23  
  see also chapter on System Administration Tools

**N**

Next hop routing 1-7, 4-21  
Number expansion 3-16, 4-17

**O**

Organization header field A-7

**P**

Packet telephony network 1-1  
Password  
  CHAP 4-14  
phone  
  supported 1-6, 1-7  
supported protocols  
  DNS 1-6  
  IP 1-6
UDP 1-6
Priority header field A-7
protocols
  DNS 1-6
  IP 1-6
  SDP 1-7
  TCP 1-6
  UDP 1-6
proxy server 1-3
Proxy-Authenticate header field A-7
Proxy-Authorization header field A-7
Proxy-Required header field A-7

R

RAS 3-9, 4-24
Record-Route header field A-7
redirect server 1-3
registrar server 1-3
Registries
  subscribers 3-27
Registry 4-20
  static 3-27
related documentation xi
request methods B-1
Require header field A-7
Response-Key header field A-7
responses A-2
  global (6xx) A-6
  information (1xx) A-3
  redirection (3xx) A-4
  request failure (4xx) A-4
  server failure (5xx) A-6
  successful (2xx) A-3
Restart 5-4
  graceful 5-1
Retry-After header field A-7
RFC
  2543 1-1

768 1-6
791 1-6
793 1-6
Route header field A-7
Routes
  static 3-30
RPM 3-11

S

SDP
description 1-7
server
directives 3-18
proxy 1-3
redirect 1-3
registrar 1-3
SIP server core 3-20
Server header field A-7
Session management 1-1
Signaling 1-1
SIP
  architecture 1-3
  clients 1-2, 1-3
    gateways 1-3
    phones 1-3
  compliance information A-1
  components 1-2
    UAC 1-2
    user agent server 1-2
  end point 1-2
  functions A-1
  gateways 1-3
  header fields A-6
  methods A-1, A-2
  request methods B-1
  responses A-2
    global (6xx) A-6
    information (1xx) A-3
Index

redirection (3xx) A-4
request failure (4xx) A-4
server failure (5xx) A-6
successful (2xx) A-3

servers
proxy 1-3
redirect 1-3
registrar 1-3

services
authentication 1-2
billing 1-2
directory 1-2

SNMP 7-1
trap 7-6

Subject header field A-7
Subscriber 6-18

T

TCP
  description 1-6
Timestamp header field A-7
To header field A-7
Translation 1-7

Trap
  SNMP 7-6
  trap sink 7-14

U

UAC 1-2
UDP
  description 1-6
Unsupported header field A-7
user
  agent server 1-2
User-Agent header field A-7

V

Via header field A-7
Virtual proxy host 1-11, 3-9
VSA C-2

W

Warning header field A-7
WWW-Authenticate header field A-7