

CLI Configuration Guide for Cisco Contact Center SIP Proxy (CCCSP)

First Published: 2025-11-26

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

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

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

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

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

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

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

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

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

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

© 2025 Cisco Systems, Inc. All rights reserved.



CONTENTS

CHAPTER 1 **Overview of Cisco Contact Center SIP Proxy (CCCSP)** 1

About This Document 1

Administration Interfaces 1

Commercial Open Source Licensing 2

Obtain Documentation and Submit Service Request 2

Technical Assistance 2

CHAPTER 2 **Initial Configuration Tasks** 3

Configure SNMP MIB 3

About SNMP MIB Support 3

SIP Proxy SNMP MIBs 5

MIB Objects 6

MIB Notifications (Traps) 11

Configure Community String 14

Configure SNMP Traps 15

Configure CPU Threshold Values for Traps 17

Configure Smart Licensing 18

About Smart Licensing 18

Verify Smart Licensing 19

Set Backup Parameters 20

About Backup Parameters 20

Configure NTP Servers 22

Add NTP Servers 22

Remove NTP Server 24

Display NTP Server Information 25

Configure Sub-interfaces 26

CHAPTER 3 **Configure Cisco Contact Center SIP Proxy (CCCSP)** 29

Configure Logical Networks 29
Configure Trigger Conditions 30
Configure Server Groups 32
Configure Route Tables 34
Configure Normalization Policies 35
Configure Lookup Policies 36
Configure Routing Triggers 38
Configure Normalization Triggers 39
Configure Listen and Record-Route Ports 40
Configure a Hostname 41
Configure Transport Layer Security (TLS) 42
Create and Import a Signed Certificate 42
Create and Import a Self-Signed Certificate 46
Configure TLS on Cisco Contact Center SIP Proxy (CCCSP) 51
Update Web Session with an Imported Signed Certificate 52
Configure Lite Mode 53
Configure Call Rate Limit 54
Commit the Configuration 55

CHAPTER 4 **Configure Users and Groups** 57

Add and Modify a User 57
Exec mode 58
Configuration Mode 59
Add and Modify a Group 61
Exec mode 62
Configuration Mode 63

CHAPTER 5 **Back Up and Restore Data** 65

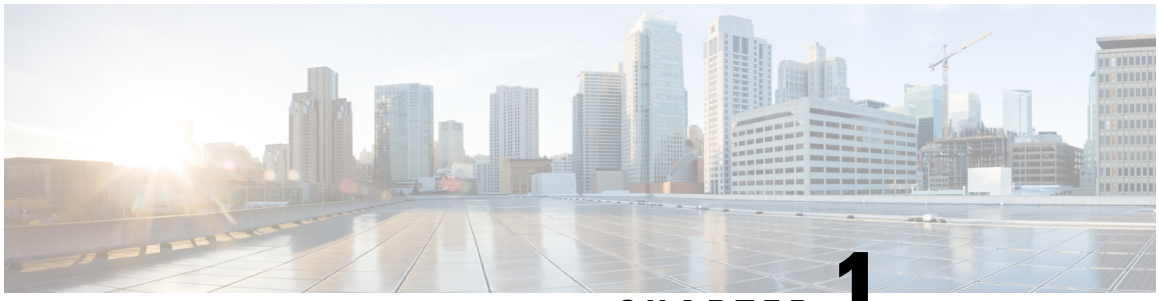
About Back up and Restore Data 65
Restrictions for Back Up and Restore Data 65
Back Up Files 66
About Back Up Files 66

Restore Files 68
About Restore Files 68
Related Topics 70

CHAPTER 6 Maintain Cisco Contact Center SIP Proxy (CCCSP) System 71
Copy Configuration 71
Copy Startup Configuration from Hard Disk to Another Location 71
Copy Startup Configuration from Network SFTP Server to Another Location 72
Copy Running Configuration from Hard Disk to Another Location 72
Copy Running Configuration from Network TFTP Server to Another Location 73

CHAPTER 7 Troubleshooting 75
Using CLI Commands to Troubleshoot the System 75
About Logging 75
Log Commands 76
Example of Log Output 76
Using Trace Commands 76
Using Show Commands 77
Troubleshooting Configuration Changes 77
Related Topics 77

CHAPTER 8 Configuration Example 79
Configuration Example 79



CHAPTER 1

Overview of Cisco Contact Center SIP Proxy (CCCSP)

- About This Document, on page 1
- Administration Interfaces, on page 1
- Commercial Open Source Licensing, on page 2
- Obtain Documentation and Submit Service Request, on page 2
- Technical Assistance, on page 2

About This Document

This document contains information about how to configure the Cisco Contact Center SIP Proxy (CCCSP) system using the CLI. Use it in conjunction with the *CLI Command Reference for Cisco Contact Center SIP Proxy (CCCSP)*, which lists all the CLI commands.

Administration Interfaces

Cisco Contact Center SIP Proxy (CCCSP) utilizes both command-line interface (CLI) and graphical user interface (GUI).

Command-Line Interface

The CLI is a text-based interface that is accessed through SSH session directly to the Cisco Contact Center SIP Proxy (CCCSP) application. Those familiar with Cisco IOS command structure and routers will see similarities.

Graphical User Interface

The application also has a GUI that is used to configure and operate the Cisco Contact Center SIP Proxy (CCCSP) system.

For information on using the GUI, see the online help in the application or the *GUI Configuration Guide for Cisco Contact Center SIP Proxy (CCCSP)*.

Commercial Open Source Licensing

Some components of the application created for Cisco Contact Center SIP Proxy (CCCSP) are provided through open source or commercial licensing. These components and the associated copyright statements can be found at:

https://www.cisco.com/c/en/us/about/legal/open-source-documentation-responsive.html

Obtain Documentation and Submit Service Request

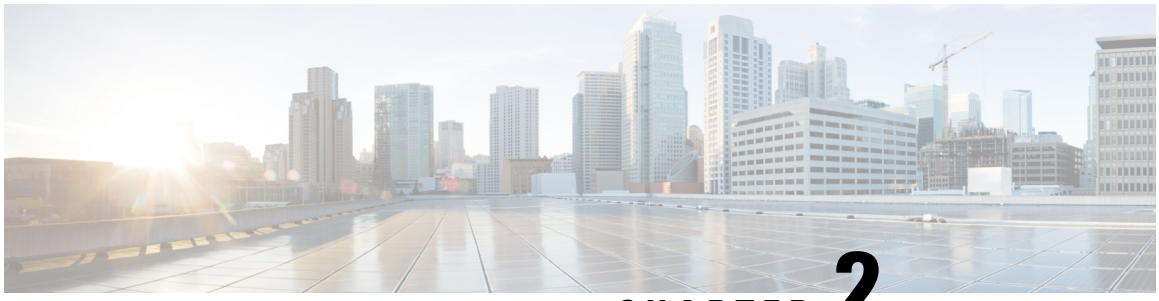
For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and RSS Feeds. Access to most tools on the Cisco Support website requires a Cisco.com username and password.	https://www.cisco.com/c/en_in/support/index.html
Use the Cisco Feature Navigator website to find information about platform support. An account on Cisco.com is not required.	http://www.cisco.com/go/cfn



CHAPTER 2

Initial Configuration Tasks

- Configure SNMP MIB, on page 3
- Configure Smart Licensing, on page 18
- Set Backup Parameters, on page 20
- Configure NTP Servers, on page 22
- Configure Sub-interfaces, on page 26

Configure SNMP MIB

About SNMP MIB Support

The CCCSP supports SNMP integration with Cisco-USP-MIB. This MIB is SNMP version 2 (SNMPv2c) compliant.

SIP Proxy integrates an SNMP agent and SNMP MIBs to monitor the health and to conduct performance monitoring and data collection for SIP Proxy. Cisco-USP-MIB and Cisco-Process-MIB monitor the following data:

- Call Statistics
- Server Group Tables
- License State
- Memory and CPU Utilization
- System State

The SNMP integration sends notifications that helps to effectively monitor and manage performance and all the relevant system-specific data. Cisco-Process-MIB is supported in CCCSP for generating traps on configured CPU thresholds.

You can configure SNMP to send notifications to one or more monitoring systems. The maximum number of SNMP trap hosts that you can configure is limited to five.

Definitions

Table 1: Definition of SNMP MIB Related Terms

Term	Definition
Simple Network Management Protocol (SNMP)	It is a common network protocol that describes information passed between SNMP-enabled applications.
SNMP Agent	An SNMP Agent acts as a client to an SNMP management application by providing data values for registered OIDs.
Management Information Base (MIB)	MIBs are a defined hierarchy of data values managed by an SNMP Agent application.
SNMP Notification (Trap)/Informs	Information shared by a network entity with the management station to monitor a fault, exception, or an attribute value change. Traps do not need acknowledgment, but informs request acknowledgment. From SNMPv2, traps are known as notifications.
Object Identifiers (OID)	It is a unique string of digits representing the value defined in an MIB.
SNMP GET	SNMP GET is an SNMP message used to fetch the value for a particular OID.
SNMP SET	SNMP SET is an SNMP request used to modify information on the target agent (controlling agent behavior or configuration of agent).

Prerequisites

For using Cisco-USP-MIB, you must ensure that the following prerequisites are met:

- Configure Community Strings.
- Administrators of the SIP Proxy must be familiar with the Cisco Command-line Interface (CLI) or the Graphical User Interface (GUI).
- Use a MIB browser or Network Management System (NMS) to interact with the SIP Proxy.
- Upload the CISCO-USP-MIB to the NMS.
- Ensure that MIB browser or NMS provides SNMP v2c compliance.

Restrictions

SNMP MIB support is known to have the following limitations or restrictions:

- No Support for SNMP Version 3 (SNMPv3).

- Certain MIB objects in the SIP Proxy MIB tree are not supported. For a list of MIB objects that are not supported, see MIB Objects.
- If both read-only and read-write community strings are same for SNMP MIBs, then read-only takes preference and SET operations are not allowed.
- If the element table contains nested server group as an element, it does not display the partial state. The element state is shown as either up or down.
- The dropped call MIB objects are not updated if the license is in unidentified state.
- If call rate limit is set to a value lesser than license limit, cuspCallsDroppedExceedingLicense MIB counts calls are dropped due to call rate limit.
- There is no CLI and GUI equivalent for the data retrieved through MIB objects related to Calls Per Second (CPS).

Structure

The SNMP MIB structure for SIP Proxy has the following main considerations:

- The SIP Proxy is uniquely identified within the Cisco management (9) group by the number –.1.3.6.1.4.1.9.9.827.
- Use either of the following methods to identify objects in the CISCO-USP-MIB:
  - The object identifier –.1.3.6.1.4.1.9.9.827.<Cisco-USP- MIB-variable>
  - The object name – iso(1).org(3).dod(6).internet(1).private(4).enterprise(1).cisco(9).ciscoMgmt(9).CISCO-USP-MIB(827).<Cisco-USP-MIB-variable>
- The following traps in Cisco-Process-MIB for CPU utilization monitoring is supported:
  - cpmCPURisingThreshold (.1.3.6.1.4.1.9.9.109.2.0.1)
  - cpmCPUFallingThreshold (.1.3.6.1.4.1.9.9.109.2.0.2)

The SIP Proxy MIB structure has the following groups and subgroups:

- MIBNotifs
- MIBObjects
  - cuspScalar
  - cuspTable
  - cuspNotifControlInfo
- MIBConform

SIP Proxy SNMP MIBs

The CCCSP captures the following in a management information base:

- MIB Objects
- MIB Notifications (Traps)

MIB Objects

The supported SIP Proxy MIB Objects are:

- cuspScalar
  - cuspCallStats
  - cuspMessageStats
  - cuspThresholdValues
- cuspTable
- cuspNotifControlInfo

The CCCSP-15.0(1) release does not support the following MIB objects:

- cuspMemoryThresholdAlert
- cuspDiskSpaceThresholdAlert
- cuspBackupProcessFailAlert
- cuspConnectionExceptionAlert
- cuspThresholdValues
- cuspDiskSpaceThresholdValue
- cuspMemoryThresholdValue
- cuspMessageStats
- cuspStrayMessageCount
- cuspNoOfMessagesRecieved
- cuspMemoryThresholdAlertEnable
- cuspExtensiveLoggingAlertEnable
- cuspDiskSpaceThresholdAlertEnable
- cuspBackupProcessFailAlertEnable
- cuspConnectionExceptionAlertEnable
- cuspDiskSpaceUsed

cuspScalar

This table contains a list of SIP Proxy scalars. An entry in this table represents SIP Proxy information relevant to licenses, system state, and memory.

Table 2: MIB Description for cuspScalar

MIB	OID	Description
cuspLastCounterResetTime	.1.3.6.1.4.1.9.9.827.1.1.1	Gives the timestamps in date and time when the call counter was last reset. All counters related to calls, Calls Per Second (CPS) and messages are reset when the counter is reset.
cuspSystemState	.1.3.6.1.4.1.9.9.827.1.1.2	Gives the SIP Proxy system state as UP or DOWN.
cuspSystemUpTime	.1.3.6.1.4.1.9.9.827.1.1.3	Gives information on the active time of the SIP Proxy system.
cuspLicenseLimit	.1.3.6.1.4.1.9.9.827.1.1.4	Gives the license limit information. Calls are rejected if the license limit is exceeded.
cuspLicenseState	.1.3.6.1.4.1.9.9.827.1.1.5	Gives the current license state of SIP Proxy.
cuspSmartAgentState	.1.3.6.1.4.1.9.9.827.1.1.6	Gives the current license state of the SmartLicense Agent.
cuspConfiguredMemory	.1.3.6.1.4.1.9.9.827.1.1.7	Gives the total memory (RAM) configured on SIP Proxy in Megabytes.
cuspMemoryUsed	.1.3.6.1.4.1.9.9.827.1.1.8	Gives the SIP Proxy current memory (RAM) usage information in Megabytes.
cuspDiskSpaceUsed	.1.3.6.1.4.1.9.9.827.1.1.9	Gives the current disk utilization of CUSP in MB (Megabytes).

cuspCallStats

This SIP Proxy MIB defines data related to calls.

Table 3: MIB Description for cuspCallStats

MIB	OID	Description
cuspTotalCalls	.1.3.6.1.4.1.9.9.827.1.1.10.1	The total number of calls since the last counter reset.
cuspTotalFailedCalls	.1.3.6.1.4.1.9.9.827.1.1.10.2	The total number of failed calls since last counter reset.
cuspCPS	.1.3.6.1.4.1.9.9.827.1.1.10.3	The current running Calls Per Second (CPS) information.

MIB	OID	Description
cuspAvgCPSOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.4	The average CPS in the last one minute.
cuspMaxCPSOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.5	The Maximum value of CPS in the last one minute.
cuspDroppedCallsOneSec	.1.3.6.1.4.1.9.9.827.1.1.10.6	The count on number of calls dropped in the last one second.
cuspAvgDroppedCallsOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.7	The average of dropped calls per second' in the last one minute.
cuspMaxDroppedCallsOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.8	The Maximum of dropped calls per second' in the last one minute.
cuspCallsRoutedOneSec	.1.3.6.1.4.1.9.9.827.1.1.10.9	The number of calls routed through CUSP in one second.
cuspAvgCallsRoutedOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.10	The average of calls routed per second' in last one minute.
cuspMaxCallsRoutedOneMin	.1.3.6.1.4.1.9.9.827.1.1.10.11	The maximum of 'calls routed per second' in the last one minute.
cuspCallsDroppedExceedingLicense	.1.3.6.1.4.1.9.9.827.1.1.10.12	The total calls dropped due to exceeding license limit.

cuspThresholdValues

The SIP proxy MIB object cuspThresholdValues (.1.3.6.1.4.1.9.9.827.1.1.12) provides threshold value information (as configured by user) on disk space and memory utilization.

Table 4: MIB Description for cuspThresholdValues

MIB	OID	Description
cuspDiskSpaceThresholdValue	.1.3.6.1.4.1.9.9.827.1.1.12.1	The percentage threshold value configured by the user. If the percentage disk space utilization exceeds this limit, then cuspDiskSpaceThresholdAlert notification is sent.
cuspMemoryThresholdValue	.1.3.6.1.4.1.9.9.827.1.1.12.2	The percentage threshold value configured by the user. If the percentage memory utilization exceeds this limit, then cuspMemoryThresholdAlert notification is sent.

cuspTable

The SIP proxy MIB object cuspTable (.1.3.6.1.4.1.9.9.827.1.2) consists of two main subgroups of objects:

- cuspServerGroupTable (OID:.1.3.6.1.4.1.9.9.827.1.2.1)
- cuspElementTable (OID:.1.3.6.1.4.1.9.9.827.1.2.2)

Note If data is retrieved from multiple network elements using cuspTable MIBs, the CPU utilization can spike beyond the optimum levels.

cuspServerGroupTable

The MIB cuspServerGroupTable represents a list of server groups that are part of active configuration. Server groups define the elements with which the SIP Proxy system interacts for each network.

Table 5: MIB Description for cuspServerGroupTable

MIB	OID	Description
cuspServerGroupEntry	.1.3.6.1.4.1.9.9.827.1.2.1.1	An entry (conceptual row) in the ServerGroup Table.
cuspServerGroupIndex	.1.3.6.1.4.1.9.9.827.1.2.1.1.1	A unique value, greater than zero, for each server group.
cuspServerGroupName	.1.3.6.1.4.1.9.9.827.1.2.1.1.2	The name of the server group.
cuspServerGroupNetwork	.1.3.6.1.4.1.9.9.827.1.2.1.1.3	The network to which the server group belongs.
cuspServerGroupStatus	.1.3.6.1.4.1.9.9.827.1.2.1.1.4	The Server group status is given as up, partial down, and down.
cuspServerGroupPingStatus	.1.3.6.1.4.1.9.9.827.1.2.1.1.5	Server group ping status.
cuspServerGroupLBType	.1.3.6.1.4.1.9.9.827.1.2.1.1.6	The load balancing algorithm for the server group.

Note The CuspservergroupPingStatus MIB object retrieves the information of a group, irrespective of the global ping status.

cuspElementTable

The MIB cuspElementTable provides a list of elements in a server group table. Also, the table contains information on status (up or down) of the element, its Q-value, weight, and transport type.

Table 6: MIB Description for cuspElementTable

MIB	OID	Description
cuspElementEntry	.1.3.6.1.4.1.9.9.827.1.2.2.1	An entry (conceptual row) in the cuspElementTable.
cuspElementIndex	.1.3.6.1.4.1.9.9.827.1.2.2.1.1	A unique value, greater than zero, for each element.
cuspElementName	.1.3.6.1.4.1.9.9.827.1.2.2.1.2	The Server group element ID.
cuspElementStatus	.1.3.6.1.4.1.9.9.827.1.2.2.1.3	The server group element status as up or down.
cuspElementQValue	.1.3.6.1.4.1.9.9.827.1.2.2.1.4	The Q value of the server group element. Q value range is 0.0 to 1.0.
cuspElementWeight	.1.3.6.1.4.1.9.9.827.1.2.2.1.5	The weight of the server group element. Weight is used for load balancing between server group elements.
cuspElementPort	.1.3.6.1.4.1.9.9.827.1.2.2.1.6	Gives the port number of the server group element.
cuspElementTransport	.1.3.6.1.4.1.9.9.827.1.2.2.1.7	The transport type of the server group element. Transport type can be udp, tcp, or tls.
cuspElementTotalCalls	.1.3.6.1.4.1.9.9.827.1.2.2.1.8	The total routed calls to the server group element.
cuspElementFailedCalls	.1.3.6.1.4.1.9.9.827.1.2.2.1.9	The total failed calls on the server group element.

cuspNotifControlInfo

The MIB cuspNotifControlInfo (OID is.1.3.6.1.4.1.9.9.827.1.3) contains object that manages (enabling and disabling) the traps defined in CiscoUspMIBNotifs.

Table 7: MIB Description for cuspNotifControlInfo

MIB	OID	Description
cuspNotifSeverity	.1.3.6.1.4.1.9.9.827.1.3.1	The classification on the event severity.
cuspNotifDetail	.1.3.6.1.4.1.9.9.827.1.3.2	The detailed information on error encountered.
cuspSystemStateAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.3	Controls generation of cuspSystemStateAlert, cuspConnectionExceptionAlert.

MIB	OID	Description
cuspServerGroupAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.4	Controls the generation of cuspServerGroupElementAlert and cuspServerGroupAlert.
cuspServerGroupElementAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.5	Controls the generation of cuspServerGroupElementAlert.
cuspLicenseExceededAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.6	Controls the generation of cuspLicenseExceededAlert.
cuspLicenseStateAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.7	Controls the generation of cuspLicenseStateAlert.
cuspExtensiveLoggingAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.8	Controls the generation of cuspExtensiveLoggingAlert.
cuspDiskSpaceThresholdAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.9	Controls the generation of cuspDiskSpaceThresholdAlert.
cuspMemoryThresholdAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.10	Controls the generation of cuspMemoryThresholdAlert.
cuspBackupProcessFailAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.11	Controls the generation of cuspBackupProcessFailAlert notification.
cuspConnectionExceptionAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.12	Controls the generation of cuspConnectionExceptionAlert.
cuspSIPMessageQueueOverflowAlertEnable	.1.3.6.1.4.1.9.9.827.1.3.13	Controls the generation of cuspSIPMessageQueueOverflowAlert.

MIB Notifications (Traps)

SIP Proxy generates trap notifications when the Network Management Station (NMS) or the administrator has to be informed about an event. The notification describes the operation state information of a service when a condition occurs. Traps provide information on issues that occur in the network element without polling for SNMP objects.

The administrator can control traps using the Command-line Interface (CLI), the Graphical User Interface (GUI), or through SNMP. By default, the traps are set to disabled state.

CCCSP-15.0(1) supports a generic trap and raises SNMP traps on the following events:

- License Limit is exceeded
- System Failure
- Change in Server element state
- Change in Server group element state

CCCSP-15.0(1) does not support SNMP traps on the following events:

- Backup Process Failure
- Memory threshold is exceeded
- Disk space threshold is exceeded
- Extensive Debug level logging
- Connection Exception

Table 8: MIB Description for MIB Traps

MIB	OID	Description
cuspSystemStateAlert	.1.3.6.1.4.1.9.9.827.0.1	Generated when the SIP Proxy system goes up or down. This notification can be enabled or disabled by setting cuspSystemStateAlertEnable. CLI command to configure the trap: **snmp-server enable traps System-State**
cuspServerGroupElementAlert	.1.3.6.1.4.1.9.9.827.0.2	Generated when the status of server group element changes. This notification can be enabled or disabled by setting cuspServerGroupAlertEnable. CLI command to configure the trap: **snmp-server enable traps SG-Element**
cuspServerGroupAlert	.1.3.6.1.4.1.9.9.827.0.3	Generated when all the elements in the server group go down. Also, it is generated when any one element in the server group comes up after all the elements in the group were down. This notification is enabled or disabled by setting cuspServerGroupAlertEnable. CLI command to configure the trap: **snmp-server enable traps Server-Group**
cuspMemoryThresholdAlert	.1.3.6.1.4.1.9.9.827.0.4	Generated when SIP Proxy memory usage exceeds the cuspMemoryThresholdValue. This notification can be enabled or disabled by setting cuspThresholdAlertEnable.

MIB	OID	Description
cuspLicenseExceededAlert	.1.3.6.1.4.1.9.9.827.0.5	Generated when average CPS exceeds cuspLicenseLimit. This notification can be enabled or disabled by setting cuspLicenseExceededAlertEnable. CLI command to configure the trap: **snmp-server enable traps License-Exceeded**
cuspLicenseStateAlert	.1.3.6.1.4.1.9.9.827.0.6	Generated when SIP Proxy license state changes. This notification is enabled or disabled by setting cuspLicenseStateAlertEnable CLI command to configure the trap: **snmp-server enable traps License-State**
cuspExtensiveLoggingAlert	.1.3.6.1.4.1.9.9.827.0.7	Generated when extensive debug level logging is enabled in SIP Proxy. Extensive logging has an impact on performance and system stability. This notification can be enabled or disabled by setting cuspExtensiveLoggingAlertEnable. CLI Command to configure the trap: **snmp-server enable traps Extensive-Logging**
cuspDiskSpaceThresholdAlert	.1.3.6.1.4.1.9.9.827.0.8	Generated when the SIP Proxy Disk usage exceeds the cuspDiskSpaceThresholdValue. This notification can be enabled or disabled by setting cuspThresholdAlertEnable.
cuspBackupProcessFailAlert	.1.3.6.1.4.1.9.9.827.0.9	Generated when backup process fails. This notification is enabled or disabled by setting cuspBackupProcessFailAlertEnabl.
cuspConnectionExceptionAlert	.1.3.6.1.4.1.9.9.827.0.10	Generated when a connection exception occurs. This notification can be enabled or disabled by setting cuspSystemStateAlertEnable.

MIB	OID	Description
cuspSIPMessageQueueOverflowAlert	.1.3.6.1.4.1.9.9.827.0.11	Generated when the SIP Proxy system queue is full. Queue full indicates that either SIP Proxy is overloaded or encountering network issues. The time interval between two successive notifications is 5 minutes. Notification is not sent within this time frame even if the queue is full. This back-off timer of 5 minutes prevents the SIP Proxy overload. This notification can be enabled or disabled by setting cuspSIPMessageQueueOverflowAlertEnable. CLI command to configure the trap: **snmp-server enable traps SIP-Message-Queue-Overflow**
cpmCPURisingThreshold	.1.3.6.1.4.1.9.9.109.2.0.1	Sent when configured rising CPU utilization threshold is reached and CPU utilization remains above the threshold for configured interval, and such a notification is requested. CLI command to configure the trap: **snmp-server enable traps CPU-Rising**
cpmCPUFallingThreshold	.1.3.6.1.4.1.9.9.109.2.0.2	Sent when the configured falling threshold is reached and CPU utilization remains under threshold for configured interval, and such a notification is requested. CLI command to configure the trap: **snmp-server enable traps CPU-Falling**

Note cuspLicenseExceededAlert is not generated if the license is in unidentified state.

Configure Community String

Configure community string to poll data using MIB objects.

SUMMARY STEPS

1. **config**
2. **snmp-server community** *community string {RO| RW}*
3. **end**
4. **write memory**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**config** **Example:** `Hostname# config terminal`	Enables privileged EXEC mode.
Step 2	**snmp-server community** *community string {RO| RW}* **Example:** `Hostname(config)# snmp-server community public RW`	Configures the community string. The access could be read-only or read-write based on the selected configuration.
Step 3	**end** **Example:** `Hostname(config)# end`	Exits the privileged EXEC mode.
Step 4	**write memory** **Example:** `Hostname# write memory`	Stores the configuration in the startup configuration file.

Examples

Example

The following example configures Community Strings on the CCCSP:

```
Hostname# config terminal
Hostname(config)# snmp-server community public RW
Hostname(config)# end
Hostname# write memory
```

Configure SNMP Traps

SUMMARY STEPS

1. **config** *terminal*

2. **snmp-server host** *IP Address community string*
3. **snmp-server enable traps [All | System-State | Server-Group | SG-Element | CPU-Rising | CPU-Falling | License-State | License-Exceeded | Extensive-Logging | SIP-Message-Queue-Overflow]**
4. **snmp-server enable traps**
5. **end**
6. **write memory**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**config** *terminal* **Example:** `Hostname# config terminal`	Enables privileged EXEC mode.
Step 2	**snmp-server host** *IP Address community string* **Example:** `Hostname(config)# snmp-server host 10.104.54.108 public`	Specifies the host that receives SNMP notifications.
Step 3	**snmp-server enable traps [All | System-State | Server-Group | SG-Element | CPU-Rising | CPU-Falling | License-State | License-Exceeded | Extensive-Logging | SIP-Message-Queue-Overflow]** **Example:** `Hostname(config)# snmp-server enable traps SG-Element`	Activates the traps selected. The command **snmp-server enable traps all** activates all traps. To activate a specific trap, follow **snmp-server enable traps** with the subcommand specific to that trap.
Step 4	**snmp-server enable traps** **Example:** `Hostname(config)# snmp-server enable traps`	Enables trap generation from CCCSP to the configured hosts. Traps are sent to the host only when this global command is enabled.
Step 5	**end** **Example:** `Hostname(config)# end`	Exits the privileged EXEC mode.
Step 6	**write memory** **Example:** `Hostname(config)# write memory`	Stores the configuration in the startup configuration file.

Example

The following example configures SNMP Traps on the CCCSP:

```
Hostname# config terminal
 Hostname(config)# snmp-server host 10.104.54.108 public
 Hostname(config)# snmp-server enable traps SG-Element
 Hostname(config)# snmp-server enable traps
 Hostname(config)# end
Hostname# write memory
```

Note The trap body information can only be seen on the trap listener host. However, a generic trap notification will be logged in the vCCCSP atrace.log logs.

Configure CPU Threshold Values for Traps

To define rising and falling CPU threshold values for traps, perform these steps:

SUMMARY STEPS

1. **config** *terminal*
2. **process cpu threshold type {total} rising** *percentage* **interval** *seconds* **falling** *percentage* **interval** *seconds*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**config** *terminal* **Example:** `Hostname# config terminal`	Enables privileged EXEC mode.
Step 2	**process cpu threshold type {total} rising** *percentage* **interval** *seconds* **falling** *percentage* **interval** *seconds* **Example:** `Hostname(config)# process cpu threshold type {total} rising 80 interval 300 falling 5 interval 300`	Sets the CPU threshold notifications types and values. In this example, the CPU utilization threshold is set to 80 percent for a rising threshold notification and 5 percent for a falling threshold notification. The polling interval is set as 300 seconds.

Examples

Example

The following example configures CPU thresholding values for SNMP traps on the CCCSP:

```
Hostname# config terminal
Hostname(config)# process cpu threshold type {total} rising 80 interval 300 falling 5
interval 300
```

Configure Smart Licensing

About Smart Licensing

Cisco Smart Software Licensing is a standardized licensing platform that facilitates you to deploy and manage Cisco software licenses easily and quickly. Cisco Smart Software Licensing establishes a pool of software licenses that can be used across your network in a flexible and automated manner. It also provides visibility to your purchased and deployed licenses in your network. Cisco Smart Software Licensing removes the need for Product Activation Keys (PAKs) and reduces your license activation and registration time.

Note For more information on Smart Licensing, see http://www.cisco.com/go/smartlicensing.

Before you begin

Log in to your Smart Account and generate a valid token and make sure that the required licenses for CCCSP is available in your Account.

SUMMARY STEPS

1. **enable**
2. **license smart activate cccsp** *count*
3. **license smart destinationAddr** *server url*
4. **license smart httpProxyAddr** *ip address / FQDN* **port** *port*
5. **license smart register token_id** *token*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Hostname# enable`	Enables privileged EXEC mode. Enter your password if prompted.

	Command or Action	Purpose
Step 2	**license smart activate cccsp** *count* **Example:** `Hostname# license smart activate cccsp 100`	Activates the request number of licenses. The count must be a multiple of 5. In the example, the count is configured as 100, so the actual license count that is consumed on the Smart Account is 20 (100/5).
Step 3	**license smart destinationAddr** *server url* **Example:** `Hostname# license smart destinationAddr https://smartreceiver.cisco.com/licservice/license`	Connects to the central licensing server.

Step 4	**license smart httpProxyAddr** *ip address / FQDN* **port** *port*
	Example:
	`Hostname# license smart httpProxyAddr http://10.1.1.1 port 3128`
	Example:
	`Hostname# license smart httpProxyAddr https://10.1.1.1 port 3128`
		Sets the HTTP(S) proxy server IP address or FQDN and port address for smart licensing.

Step 5	**license smart register token_id** *token* **Example:** `Hostname# license smart register token_id` [redacted]	Registers the device instance with the Cisco licensing cloud. This step is performed only once per device instance. The license agent registers the product with Cisco and receives back an identity certificate. This certificate is saved and automatically used for all future communications with Cisco. The license agent automatically renews the registration information with Cisco after one year.

Example

The following example configures Smart License on the CCCSP:

```
Hostname# enable
Hostname# license smart activate cccsp 100
Hostname# license smart destinationAddr https://smartreceiver.cisco.com/licservice/license
Hostname# license smart httpProxyAddr https://10.1.1.1 port 3128
Hostname# license smart register token_id
```
[redacted]

Verify Smart Licensing

Use the **show license smart summary** command to verify if the Smart License Agent is successfully configured.

```
Hostname# show license smart summary
Smart Agent is Enabled: true
Current State of the Agent: OUT_OF_COMPLIANCE
Evaluation Mode: NO
Registration Successful: YES
Authorization Successful: YES

CPS Count Requested: 5
Configured destination address: https://smartreceiver.cisco.com/licservice/license
HTTP Proxy Address: http://10.10.10.39:3128
License UDI: CCCSP:JZp7KbrPwS6
Product Serial Number: JZp7KbrPwS6
Product ID: CCCSP
Product License Version: 1.0
Licensing State: OutOfCompliance
Registration expiry period: Tue Aug 11 17:57:33 IST 2026
Latest Failure Reason String Notification: Successful
Auth period: Tue Aug 12 18:02:37 IST 2025
```

Set Backup Parameters

About Backup Parameters

CCCSP backup and restore functions use an SFTP server to store and retrieve data. The backup function copies the files from CCCSP to the SFTP server and the restore function copies the files from the SFTP server to CCCSP. The SFTP server can reside anywhere in the network as long as the backup and restore functions can access it with an IP address or hostname.

All CCCSP backup files are stored on the specified server. You can copy the backup files to other locations or servers, if necessary.

The backup parameters specify the SFTP server to use for storing CCCSP backup files and the number of backups that are stored before the system overwrites the oldest one.

Before you begin

- Verify that an SFTP administrator or other user who can log in to the SFTP server has full permission on the SFTP server, such as read, write, overwrite, create, and delete permissions for files and directories.
- Gather the SFTP server URL and the username and password of the SFTP server login.
- Make sure that the SFTP server URL is pointing to the absolute path of the backup directory. For example, sftp://<hostname>/full/path/from/root/to/backup_directory.
- Determine the number of revisions to save before the oldest backup is overwritten.

SUMMARY STEPS

1. **configure terminal**
2. **backup server url** *sftp-url* { **username** *sftp-username* **password** *sftp-password* }
3. **backup revisions** *number*
4. **end**
5. **show backup**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Hostname# config terminal`	Enters configuration mode.

Step 2	**backup server url** *sftp-url* { **username** *sftp-username* **password** *sftp-password* }
	Example:
	`Hostname(config)# backup server url sftp://main/backups username "admin" password "wxyz"`
	`Hostname(config)# backup server url sftp://192.0.2.15/backups username "admin" password "wxyz"`
		Sets the backup parameters.
		Note You must configure the backup server before you can configure the backup revisions.
		• **server url**—The *sftp-url* value is the URL to the network SFTP server where the backup files will be stored.
		• The *sftp-username* and *sftp-password* values are the username and password for the network SFTP server.
		In the example, **main** is the hostname of the SFTP server and **backup** is the directory where backup files are stored.
		Note /backups is an absolute path to the backups directory.

Step 3	**backup revisions** *number* **Example:** `Hostname(config)# backup revisions 5`	Sets the number of backup files that will be stored. When the system reaches this number of backups, it deletes the oldest stored file.
Step 4	**end** **Example:** `Hostname(config)# end`	Exists configuration mode.
Step 5	**show backup** **Example:** `Hostname# show backup`	Displays the backup server configuration information, including the SFTP server URL and the maximum number of backup files available.

Example

The following example configures a backup server and displays the **show backup** output:

```
Hostname# enable
```

```
Hostname# configure terminal
Hostname(config)# backup revisions 5
Hostname(config)# backup server url sftp://10.12.0.1/sftp username “admin” password “wxyz”
Hostname(config)# end
Hostname# show backup
Server URL: sftp://10.12.0.1/sftp
User Account on Server:

Number of Backups to Retain: 5
Hostname#
```

What to do next

Related Topics

- For information about the CLI commands, see the CLI Command Reference for .
- For information about back up and restore configuration, see Backing Up and Restoring Data .

Configure NTP Servers

When you install the Cisco Contact Center SIP Proxy (CCCSP) application, the system gives you the option of adding up to five Network Time Protocol (NTP) servers. You can add, modify, or delete up to five NTP servers in CCCSP application using the CLI.

Add NTP Servers

You can specify an NTP server using its IP address or its hostname.

CCCSP uses the DNS server to resolve the hostname to an IP address and stores the IP address as an NTP server. If DNS resolves the hostname to more than one IP address, CCCSP randomly chooses one of the IP addresses that is not already designated as an NTP server. If you do not want to go with the random choice, set the **prefer** attribute for one server.

To configure an NTP server with multiple IP addresses for a hostname, repeat the configuration steps using the same hostname. Each iteration assigns the NTP server to its remaining IP addresses.

SUMMARY STEPS

1. **configure terminal**
2. **ntp server** { *hostname* | *ip-address* } [**prefer**]
3. **end**
4. **show ntp status**
5. **show ntp configuration**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Hostname# config terminal`	Enters configuration mode.

Step 2	**ntp server** { *hostname* | *ip-address* } [**prefer**]
	Example:
	`Hostname(config)# ntp server 192.0.2.14`
	`Hostname(config)# ntp server 192.0.2.17 prefer`
		Specifies the hostname or IP address of the NTP server.
		If more than one server is configured, the server with the **prefer** attribute is used before the others.

Step 3	**end** **Example:** `Hostname(config)# end`	Exists configuration mode.
Step 4	**show ntp status** **Example:** `Hostname# show ntp status`	Displays statistics about the NTP server.
Step 5	**show ntp configuration** **Example:** `Hostname# show ntp configuration`	Displays the configured NTP servers.
Step 6	**copy running-config startup-config** **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

Example

The following commands configure the NTP server:

```
Hostname# configure terminal
Hostname(config)# ntp server 192.0.2.14
Hostname(config)# exit
Hostname#
```

The output from the **show ntp status** command looks similar to the following:

```
Hostname# show ntp status
```

```
NTP reference server 1: 192.0.2.14
Status: sys.peer
Time difference (secs): 3.268110099434328E8
Time jitter (secs): 0.1719226837158203
```

Remove NTP Server

You can remove a NTP server using its IP address or hostname.

SUMMARY STEPS

1. **configure terminal**
2. **no ntp server** { *hostname* | *ip-address*
3. **exit**
4. **show ntp status**
5. **show ntp configuration**
6. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Hostname# configure terminal`	Enters the configuration mode.

Step 2	**no ntp server** { *hostname* | *ip-address*
	Example:
	`Hostname(config)# no ntp server 192.0.2.14`
	`Hostname(config)# no ntp server myhost`
		Specifies the hostname or IP address of the NTP server to remove.

Step 3	**exit** **Example:** `Hostname(config)# exit`	Exits configuration mode.
Step 4	**show ntp status** **Example:** `Hostname# show ntp status`	Displays statistics about the NTP server.
Step 5	**show ntp configuration** **Example:**	Displays the configured NTP servers.

	Command or Action	Purpose
	`Hostname# show ntp configuration`	
Step 6	**copy running-config startup-config** **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

Display NTP Server Information

Commands to Display NTP Server Information

The following commands are available to display NTP server configuration information and status:

- **show ntp associations**
- **show ntp servers**
- **show ntp source**
- **show ntp status**

Examples of Showing NTP Server Information

The following is a sample output for the **show ntp associations** command:

```
Hostname# show ntp associations

MS Name/IP address  Stratum Poll Reach LastRx  Last sample
===============================================================================
^* 10.10.10.39         3     6    377   25     -55us[ -294us] +/-  103ms
```

The following is a sample output for the **show ntp servers** command:

```
Hostname# show ntp servers

Name/IP Address    NP  NR  Span  Frequency  Freq Skew  Offset  Std Dev
===============================================================================
nemv6.cisco.com     6   3   324     -1.449     5.560    -56us   178us
```

The following is a sample output for the **show ntp source** command:

```
Hostname# show ntp source
MS  Name/IP address  Stratum Poll Reach LastRx Last sample
===============================================================================
^*  nemv6.cisco.com   3       6   377     1    +12us[ +174us] +/-  103ms
```

The following is a sample output for the **show ntp status** command:

```
Hostname# show ntp status
```

```
Remote address  : 10.10.10.39 (0A405627)
Remote port     : 123
Local address   : 10.10.10.199 (0A4E68C7)
Leap status     : Normal
Version         : 4
Mode            : Server
Stratum         : 3
Poll interval   : 6 (64 seconds)
Precision       : -24 (0.000000060 seconds)
Root delay      : 0.189423 seconds
Root dispersion : 0.007996 seconds
Reference ID    : 417A8361 ()
Reference time  : Tue Aug 12 06:03:58 2025
Offset          : +0.000048062 seconds
Peer delay      : 0.000625227 seconds
Peer dispersion : 0.000000083 seconds
Response time   : 0.000123992 seconds
Jitter asymmetry: +0.50
NTP tests       : 111 111 1111
Interleaved     : No
Authenticated   : No
TX timestamping : Daemon
RX timestamping : Kernel
Total TX        : 211
Total RX        : 211
Total valid RX  : 211
Total good RX   : 203
```

Related Topics

- For information about the CLI commands, see the *CLI Command Reference Guide*.
- For information about the initial installation of the CCCSP system and the post installation configuration tool, see the *Installation Guide for CCCSP*. For information about copying the configuration, see Copying Configurations.

Configure Sub-interfaces

You can define multiple sub-interfaces on Cisco Contact Center SIP Proxy (CCCSP) and there is no specific restriction on the number of sub-interfaces from CCCSP.

Note Ensure that all the sub-interfaces are configured with IP addresses from the same subnet, as the trunk port config with sub-interfaces are not supported CCCSP-15.0(1).

SUMMARY STEPS

1. **configure terminal**
2. **interface FastEthernet**
3. **ip address**
4. **end**

DETAILED STEPS

Procedure

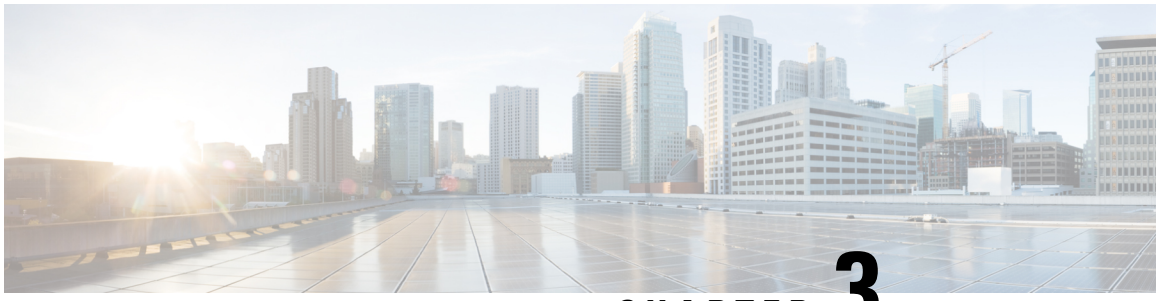
	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Hostname# conf t`	Enters configuration mode.
Step 2	**interface FastEthernet** **Example:** `Hostname(config)# interface FastEthernet 0.11`	Enters interface FastEthernet.
Step 3	**ip address** **Example:** `Hostname(config-subif)# ip address 10.10.10.19 255.255.255.0`	Configures subinterface for Fastethernet 0.11 in configuration mode.
Step 4	**end** **Example:** `Hostname(config-subif)# end`	Exits network command mode.

Examples

Example

The following example configures subinterface for Fastethernet:

```
Hostname# config
Hostname(config)# interface FastEthernet 0.11
Hostname(config-subif)# ip address 10.10.10.19 255.255.255.0
Hostname(config-subif)# end
```



CHAPTER 3

Configure Cisco Contact Center SIP Proxy (CCCSP)

- Configure Logical Networks, on page 29
- Configure Trigger Conditions, on page 30
- Configure Server Groups, on page 32
- Configure Route Tables, on page 34
- Configure Normalization Policies, on page 35
- Configure Lookup Policies, on page 36
- Configure Routing Triggers, on page 38
- Configure Normalization Triggers, on page 39
- Configure Listen and Record-Route Ports, on page 40
- Configure a Hostname, on page 41
- Configure Transport Layer Security (TLS), on page 42
- Update Web Session with an Imported Signed Certificate, on page 52
- Configure Lite Mode, on page 53
- Configure Call Rate Limit, on page 54
- Commit the Configuration, on page 55

Configure Logical Networks

Before you begin

Each interface on the Cisco Contact Center SIP Proxy (CCCSP) is associated with a logical network. Logical networks are used to organize server groups, listen points, and other properties. SIP messages are associated with the network on which they arrive.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **sip network** *network*
4. **end network**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP EXEC mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters Cisco Unified SIP Proxy configuration mode.
Step 3	**sip network** *network* **Example:** `Hostname(cusp-config)# sip network service-provider`	Creates a network and puts you into network command mode. In this case, the network that is being created is called "service provider".
Step 4	**end network** **Example:** `Hostname(cusp-config-network)# end network`	Exits network command mode.

Example

The following example creates a network called **service-provider**:

```
Hostname# cusp
Hostname(cusp)# configure
Hostname(cusp-config)# sip network service-provider
Hostname(cusp-config-network)# end network
```

Configure Trigger Conditions

Before you begin

You create trigger conditions to allow Cisco Contact Center SIP Proxy (CCCSP) to respond with the appropriate action for various call flows. In general, the more complex the call flow is, the more complex the trigger must be.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **trigger condition** *trigger-condition-name*

4. **sequence** *sequence-number*
5. **in-network** *network-name*
6. **mid-dialog**
7. **end sequence**
8. **end trigger condition**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters Cisco Unified SIP Proxy EXEC mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters Cisco Unified SIP Proxy configuration mode.
Step 3	**trigger condition** *trigger-condition-name* **Example:** `Hostname(cusp-config)# trigger condition call-from-service-provider`	Creates a trigger condition and puts you into trigger command mode. In this case, the trigger that is being created is called “call-from-service-provider”.
Step 4	**sequence** *sequence-number* **Example:** `Hostname(cusp-config-trigger)# sequence 1`	Creates a sequence with the specified number and puts you into trigger sequence command mode. The number indicates the order in which triggers are evaluated. In this case, the sequence that is being created is sequence number 1.
Step 5	**in-network** *network-name* **Example:** `Hostname(cusp-config-trigger-seq)# in-network service-provider`	Specifies the incoming network name for the trigger condition. In this case, the incoming network is the “service-provider” network.
Step 6	**mid-dialog** **Example:** `Hostname(cusp-config-trigger-seq)# mid-dialog`	A special trigger that bypasses routing policies on mid-dialog messages.
Step 7	**end sequence** **Example:** `Hostname(cusp-config-trigger-seq)# end sequence`	Exits the trigger sequence command mode.
Step 8	**end trigger condition** **Example:** `Hostname(cusp-config-trigger)# end trigger condition`	Exits the trigger command mode.

Example

In this example, Cisco Contact Center SIP Proxy (CCCSP) only reacts based on the network the call came in on, so the triggers are simple.

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# trigger condition call-from-service-provider
Hostname(cusp-config-trigger)# sequence 1
Hostname(cusp-config-trigger-seq)# in-network service-provider
Hostname(cusp-config-trigger-seq)# end sequence
Hostname(cusp-config-trigger)# end trigger condition

Hostname(cusp-config)# trigger condition mid-dialog
Hostname(cusp-config-trigger)# sequence 1
Hostname(cusp-config-trigger-seq)# mid-dialog
Hostname(cusp-config-trigger-seq)# end sequence
Hostname(cusp-config-trigger)# end trigger condition
```

Configure Server Groups

Server groups define the elements that Cisco Contact Center SIP Proxy (CCCSP) interacts with for each network. The server group name that is used is inserted into the SIP URI of the outgoing request. Some devices, such as Cisco Unified Communications Manager, validate the URI of requests before processing, which means that the end device might need to be configured with a Fully Qualified Domain Name (FQDN) to allow for this.

Two of the fields for each individual element, q-value and weight, are important to use to specify the priorities of elements, and also for load balancing. Calls are routed to specific elements based on q-value. The element with the highest q-value receives all traffic routed to that server group. If multiple elements have the same q-value, traffic is distributed between them based on the load-balancing option used. The default load-balancing is based on call-id, but weight can also be used. If weight is used, the percentage of traffic that an element receives is equal to its weight divided by the sum of up elements with the same q-value's weights. The sum of their weights does not need to equal 100. You can change the weights and q-values to configure a different priority or load-balancing scheme.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **server-group sip group** *server-group-name network*
4. **element ip-address** *ipaddress port* {**udp** | **tcp** | **tls**} [**q-value** *q-value*] [**weight** *weight*]
5. **lbtype {global | highest-q | request-uri | call-id | to-uri | weight}**
6. **end server-group**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**server-group sip group** *server-group-name network* **Example:** `Hostname(cusp-config)# server-group sip group sp.example.com service-provider`	Creates a SIP server group and enters server group command mode. In this case, the server group being created is called "sp.example.com" and it uses the network called "service-provider".
Step 4	**element ip-address** *ipaddress port* {**udp** | **tcp** | **tls**} [**q-value** *q-value*] [**weight** *weight*] **Example:** `Hostname(cusp-config-sg)# element ip-address 192.168.10.3 5060 tls q-value 1.0 weight 100`	Creates an IP element for a SIP server group and determines the characteristics of the SIP server group. **Note** You can enter this command multiple times.
Step 5	**lbtype {global | highest-q | request-uri | call-id | to-uri | weight}** **Example:** `Hostname(cusp-config-sg)# lbtype weight`	Configures the load-balancing algorithm for the SIP server group. In this example, it specifies that the element will be selected proportional to its weight relative to the weights of other elements of the same q-value.
Step 6	**end server-group** **Example:** `Hostname(cusp-config-sg)# end server-group`	Exits the server group command mode.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# server-group sip group sp.example.com service-provider
Hostname(cusp-config-sg)# element ip-address 192.168.10.3 5060 tls q-value 1.0 weight 100
Hostname(cusp-config-sg)# element ip-address 192.168.10.4 5060 tls q-value 1.0 weight 50
Hostname(cusp-config-sg)# element ip-address 192.168.10.5 5060 tls q-value 1.0 weight 50
Hostname(cusp-config-sg)# lbtype weight
Hostname(cusp-config-sg)# end server-group
```

Configure Route Tables

You must configure route tables to direct SIP requests to their appropriate destinations. Each route table consists of a set of keys that are matched based on the lookup policy. For example, each key might represent the prefix of a phone number dialed.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **route table** *table-name*
4. **key** *key* **response** *reponse-code*
5. **key** *key* **target-destination** *target-destination netowrk*
6. **end route table**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**route table** *table-name* **Example:** `Hostname(cusp-config)# route table service-provider-table`	Creates a route table and enters route table command mode. In this case, it creates a route table called "service-provider-table".
Step 4	**key** *key* **response** *reponse-code* **Example:** `Hostname(cusp-config-rt)# key * response 404`	Assigns a response code to a lookup key. In this example, it returns a response of "404" to everything.
Step 5	**key** *key* **target-destination** *target-destination netowrk* **Example:** `Hostname(cusp-config-rt)# key 510 target-destination cube-sp.example.com cube-sp`	Replaces the key part of the target destination with a specified value. **Note** You can enter this command multiple times.
Step 6	**end route table** **Example:**	Exits the route table command mode.

	Command or Action	Purpose
	`Hostname(cusp-config-rt)# end route table`	

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# route table service-provider-table
Hostname(cusp-config-rt)# key * response 404
Hostname(cusp-config-rt)# key 510 target-destination cube-sp.example.com cube-sp
Hostname(cusp-config-rt)# end route table
```

Configure Normalization Policies

Normalization policies modify SIP messages to account for incompatibilities between networks. In this case, the service provider cannot handle phone numbers with the escape sequence “91,” so the sequence must be removed from the request-uri and TO header.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **policy normalization** *policy-name*
4. **uri-component update request-uri {user | host | host-port | phone | uri}** {**all** | *match-string*} *replace-string*
5. **uri-component update header { first | last | all} {user | host | host-port | phone | uri }** {**all** | *match-string*} *replace-string*
6. **end policy**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.

	Command or Action	Purpose
Step 3	**policy normalization** *policy-name* **Example:** `Hostname(cusp-config)# policy normalization outgoing-norm-policy`	Creates a normalization policy and enters policy normalization command mode. In this example, the normalization policy is called “outgoing-norm-policy”.
Step 4	**uri-component update request-uri {user | host | host-port | phone | uri}** {**all** | *match-string*} *replace-string* **Example:** `Hostname(cusp-config-norm)# uri-component update request-uri user ^91 ""`	Configures a normalization policy step that updates a URI component field within a request URI.
Step 5	**uri-component update header { first | last | all} {user | host | host-port | phone | uri }** {**all** | *match-string*} *replace-string* **Example:** `Hostname(cusp-config-norm)# uri-component update TO all user ^91 ""`	Configures a normalization policy step that updates a URI component field within a header of the source message.
Step 6	**end policy** **Example:** `Hostname(cusp-config-norm)# end policy`	Exits policy normalization command mode.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# policy normalization outgoing-norm-policy
Hostname(cusp-config-norm)# uri-component update request-uri user ^91 ""
Hostname(cusp-config-norm)# uri-component update TO all user ^91 ""
Hostname(cusp-config-norm)# end policy
```

Configure Lookup Policies

Lookup policies decide how the keys in the route tables are used. Each key represents the beginning of the phone number dialed because each policy states to match the user component of the request-uri against the keys in its route table. The user component of the request-uri is the phone number called. The rule used to match is prefix, which means that the longest prefix match in the route table is used. So if the dialed number is 510-1XX-XXXX, the call is sent to the cme.example.com server group. If the dialed number is 510-XXX-XXXX, the call is sent to the cucm.example.com server group. The four policies in the following example are identical, except that they each refer to their specific table.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **policy lookup** *policy-name*
4. **sequence** *sequence-number table-name* **field {in-network | local-ip-address | local-ip-port | remote-ip-address | remote-ip-port} | header {p-asserted identity| from | to | diversion | remote-party-id} | request uri [uri component {param| user | phone | host| host-port| uri}]**
5. **rule {exact | prefix | subdomain | subnet | fixed** *length}* [**case-insensitive**]
6. **end sequence**
7. **end policy**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**policy lookup** *policy-name* **Example:** `Hostname(cusp-config)# policy lookup service-provider-policy`	Creates a policy with the specified name and enters policy lookup command mode. In this case, creates a policy called "service-provider-policy".
Step 4	**sequence** *sequence-number table-name* **field {in-network	local-ip-address	local-ip-port	remote-ip-address	remote-ip-port}	header {p-asserted identity| from	to	diversion	remote-party-id}	request uri [uri component {param| user	phone	host| host-port| uri}]** **Example:** `Hostname(cusp-config-lookup)# sequence 1`	Creates a sequence with the specified number and enters policy lookup sequence command mode. Sequences are performed according to the order of their number.
Step 5	**rule {exact	prefix	subdomain	subnet	fixed** *length}* [**case-insensitive**] **Example:** `Hostname(cusp-config-lookup-seq)# rule prefix`	Creates a rule that determines the routing algorithm for the lookup policy. In this case, it creates a rule that specifies that the lookup policy searches for the longest prefix match.

	Command or Action	Purpose
Step 6	**end sequence** **Example:** `Hostname(cusp-config-lookup-seq)# end sequence`	Exits policy lookup sequence command mode.
Step 7	**end policy** **Example:** `Hostname(cusp-config-lookup)# end policy`	Exits policy lookup command mode.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# policy lookup service-provider-policy
Hostname(cusp-config-lookup)# sequence 1 service-provider-table request-uri uri-component
user
Hostname(cusp-config-lookup-seq)# rule prefix
Hostname(cusp-config-lookup-seq)# end sequence
Hostname(cusp-config-lookup)# end policy
```

Configure Routing Triggers

Routing triggers correlate trigger conditions with lookup policies. A single policy is chosen based on which corresponding condition is matched. The conditions are evaluated in ascending order based on sequence number. The mid-dialog condition is the first one so that the policy step is skipped for mid-dialog messages. Based on the following configuration, after the INVITE message is successfully routed, all subsequent messages (which are mid-dialog) bypass routing policies.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **trigger routing sequence** *sequence-number* { **by-pass | policy** *policy*} [**condition** *trigger-condition*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.

	Command or Action	Purpose
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**trigger routing sequence** *sequence-number* { **by-pass** | **policy** *policy*} [**condition** *trigger-condition* **Example:** `Hostname(cusp-config)# trigger routing sequence 2 policy service-provider-policy condition call-from-service-provider`	Associates a routing policy with a trigger condition. In this example, the second sequence follows the previously-created policy called “service-provider-policy” and the previously-created trigger called “call-from-service-provider”.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# trigger routing sequence 1 by-pass condition mid-dialog
Hostname(cusp-config)# trigger routing sequence 2 policy service-provider-policy condition
 call-from-service-provider
Hostname(cusp-config)# trigger routing sequence 3 policy cube-sp-policy condition
call-from-cube-sp
Hostname(cusp-config)# trigger routing sequence 4 policy cube-es-policy condition
call-from-cube-es
Hostname(cusp-config)# trigger routing sequence 5 policy enterprise-policy condition
call-from-enterprise
```

Configure Normalization Triggers

Normalization triggers correlate trigger conditions with normalization policies. There are two types of triggers: pre-normalization, which occurs before routing, and post-normalization, which occurs after routing. Similar to routing policies, a special policy bypasses normalization on mid-dialog messages.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **trigger pre-normalization sequence** *sequence-number* { **by-pass | policy** *policy*} [**condition** *trigger-condition*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**trigger pre-normalization sequence** *sequence-number* { **by-pass | policy** *policy*} [**condition** *trigger-condition*] **Example:** `Hostname(cusp-config)# trigger pre-normalization sequence 2 policy outgoing-norm-policy condition call-from-cube-sp`	Configures a pre-normalization algorithm for incoming SIP messages to a normalization policy. In this example, the second sequence follows the previously-created policy called "outgoing-norm-policy" and the previously-created trigger called "call-from-cube-sp".

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# trigger pre-normalization sequence 1 by-pass condition mid-dialog
Hostname(cusp-config)# trigger pre-normalization sequence 2 policy outgoing-norm-policy
condition call-from-cube-sp
```

Configure Listen and Record-Route Ports

You must configure listen and record-route ports for each network. For the listen and record-route ports, the actual addresses of the Cisco Contact Center SIP Proxy (CCCSP) application are used. The **sip record-route** command inserts the record-route header into outgoing requests. The **sip listen** command allows for CCCSP to accept incoming requests on that port.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **sip record-route** *network_name* { **tcp | tls | udp** } *ip_addess* [*port*]
4. **sip listen** *network_name* { **tcp | tls | udp** } *ip_addess* [*port*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**sip record-route** *network_name* { **tcp** | **tls** | **udp** } *ip_addess* [*port*] **Example:** `Hostname(cusp-config)# sip record-route service-provider udp 10.10.10.99 5060`	Enables record-routing for a SIP network. In this example, the "service-provider" network is associated with a record-route configuration and the IP address that populates the record-route header field is "10.10.10.99" and the port that populates the record-route header is 5060.
Step 4	**sip listen** *network_name* { **tcp** | **tls** | **udp** } *ip_addess* [*port*] **Example:** `Hostname(cusp-config)# sip listen service-provider udp 10.10.10.99 5060`	Creates a listener that listens for SIP traffic on a specific SIP network, host, and port.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# sip record-route service-provider udp 10.10.10.99 5060
Hostname(cusp-config)# sip listen service-provider udp 10.10.10.99 5060
```

Configure a Hostname

If the upstream element is using DNS SRV for routing to the two CCCSPs in a network, you must configure the two CCCSPs to have the same FQDN by entering the **sip alias** command in the configuration mode on both CCCSPs.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **sip alias** *hostname*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**sip alias** *hostname* **Example:** `Hostname(cusp-config)# sip alias myhost`	Configures the hostname of this instance.

Example

```
Hostname# cusp

Hostname(cusp)# configure
Hostname(cusp-config)# sip alias myhost
```

Configure Transport Layer Security (TLS)

Create and Import a Signed Certificate

Cisco Contact Center SIP Proxy (CCCSP) supports TLS, Transmission Control Protocol (TCP), and User Datagram Protocol (UDP). Establishing TLS connections requires some extra steps because the connections require authentication using signed certificates.

Before you begin

You need an SFTP server or HTTP to import certificate requests.

Use the **crypto key delete all** command and delete all the existing certificates before importing new certificates.

Note Use the **show crypto key all** command to verify the key generation. Note down the **Label** from the output displayed.

Note When CCCSP operates as a SIP TLS client, it prioritizes the use of RSA ciphers and signature algorithms. If CCCSP connects to applications such as CUCM that present both RSA and ECDSA certificates, these applications favors the RSA certificate.

Note SHA1 is deprecated and not secure. When handling externally generated certificates, ensure that SHA1 is not used for certificate generation or signing. We recommend you to use stronger algorithms such as SHA256 or higher.

SUMMARY STEPS

1. **configure terminal**
2. **crypto key generate** [**rsa** { **label** *label-name* | **modulus** *modulus-size* } | **default**]
3. **crypto key certreq label** *label-name* **url** { **sftp:** | **http:** }
4. Use your CA infrastructure or use a 3rd party CA to get the CSR signed (external) before importing it in the next step.
5. **crypto key import trustcacert label** *label-name* { **der url**{ **sftp: | http:** } | **pem { terminal | url {sftp: | http: }} [default]**
6. **crypto key import cer label** *label-name* { **der url**{ **sftp: | http:** } | **pem { terminal | url {sftp: | http: }} [default]**
7. **offline**
8. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`Hostname# configure terminal`
	`Enter configuration commands, one per line. End with CNTL/Z.`
	`Hostname(config)#`
		Enters configuration mode.

Step 2	**crypto key generate** [**rsa** { **label** *label-name* | **modulus** *modulus-size* } | **default**]
	Example:
	`Hostname(config)# crypto key generate rsa label mykey modulus 2048 default`
	`Key generation in progress. Please wait...`
	`The label name for the key is Hostname`
	`Hostname(config)#`
		Creates an 2048-bit RSA private key.

	Command or Action	Purpose
	Hostname(config)# **exit** Hostname# **write**	
Step 3	**crypto key certreq label** *label-name* **url** { **sftp:** | **http:** } **Example:** Hostname(config)# **crypto key certreq label mykey url sftp:** Address or name of remote host? 10.64.x.y Username (ENTER if none)? user Password (not shown)? Destination path? /tmp/ Uploading CSR file succeed	Creates a certificate request to be signed.
Step 4	Use your CA infrastructure or use a 3rd party CA to get the CSR signed (external) before importing it in the next step.	
Step 5	**crypto key import trustcacert label** *label-name* { **der url**{ **sftp: | http:** } | **pem { terminal | url {sftp: | http: }} [default]** **Example:** Hostname(config)# **crypto key import trustcacert label rootCA terminal** Enter certificate... End with a blank line or "quit" on a line by itself -----BEGIN CERTIFICATE----- MIIFszCCA5ugAwIBAgIUNh1Jsm5FO6QkWXoAHzQQJBJ6eT4wDQYJKoZIhvcNAQEL BQAwaTELMAkGA1UEBhMCSU4xEjAQBgNVBAgMCUthcm5hdGFrYTESMBAGA1UEBwwJ QmFuZ2Fsb3JlMRQwEgYDVQQKDAtDb2xsYWJhdGlvbjEcMBoGA1UEAwwTQ29sbGFi YXRpb24gUm9vdCBDQTAeFw0yNTA3MTYwOTA4MTJaFw0zNTA3MTQwOTA4MTJaMGkx CzAJBgNVBAYTAklOMRIwEAYDVQQIDAlLYXJuYXRha2ExEjAQBgNVBAcMCUJhbmdh bG9yZTEUMBIGA1UECgwLQ29sbGFiYXRpb24xHDAaBgNVBAMME0NvbGxhYmF0aW9u IFJvb3QgQ0EwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQDB1d/DJi9q KFjG0McBCWsPQIojQE6j61R2pqgqZdKGfuoi68B/zWctspMck6DvvQ/KNCQ8LYoa Cr3QL6dJAxRkLGBu2FBV5NattAybaEoJNZRlwnCKfPTksUNnL/U/PkooQU+Bn8WN v+dh8zTL+ffN9uRuDJPq7rptn2fZfk+2IT3x2XNQMYOZuwIcvG/urB6sSbUFJu0x JUl9Kv//HX9COyOrnRW3g7tq/CT0eObAoIddqf12H7woCL3PiR/YwsmtfwGXL/sk xLONfbP1wxq/WdjgX/QBc07vZF9uUjsIupzEKLqvjBK1CsE4bYxKkgrreik320wK ySvSgdFfmnhA+G2V7FqlHAV9s1LBrhaqnCWDQFMOzEDqkKn+UAcATD3u9b7NdcPn NKKxKMtvC7A+JDbI2naMPASPI99shTE0Pr0ZwaYL1TVN+UlDCRYYqtTDHm1PEi+R GNgeW99/LeW49n0qHtqw8xb5OtQbGUAEQaOztnKWUR+nQ19pCHuNflBDtuKGSvzk iE+vHFSHo9L66kU+oVIMziiqJ1jRnnWMaYiQy077XWB1/jkC5593t1cmai6DUEeP 91INDbXiIcYNnJGK/JKSIp0JJ1N5gyyPAoXWv7sRG2Cu0VNaXtkAk10KHnnTIHJp fP1oCiJtZ8RdJmRthztLUs13MBHy1+fEmwIDAQABo1MwUTAdBgNVHQ4EFgQUksR4 KNEr9C+AeTd4z5sagT2Y+NAwHwYDVR0jBBgwFoAUksR4KNEr9C+AeTd4z5sagT2Y +NAwDwYDVR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQsFAAOCAgEACu84XDsAO3EU tyhVIvSKSyOXJen1iqw6vK1idnMEbre00a4hinOPIVbi+O9+o1vL4KIstoPBkM7+ YffWXhRewX2HyHu/W4d7aNowVaXB8S2IGjisQf3vdrA2IAIU65DdsZwDcNCsg8HA BDE6XkrA0FOD1oW3ge6z5anIThwqp3mofi3jcT6RacmibRAf8cKPi3xfKAxdgSNM 0ZOmoR8ahhINszo6puLoYdVlGDBR4UMPl+qhktd4t/mW5nXfcHfIKbCUecjFP4td 5rOIpJOxHJQBlksxUPul7FXKfOPkiDyDRVnJzz+yPpqukmimxx2KQUyEkX/z24SR uRVr77VIKi6ZQBi5Lq3+rPstrTdNtet4o1InJl3yjJqlottDvmaSAsbqN9z85/Dz MjsIHHGA63aevJo7/3R7v1o9qYigwMd1Tz2Qhz7rXVLFZikHn2ZKEYeIzA3sCxgI p08qIvwyERrAoAYrSBmtMyDy7MtESplty3gJ7jcmSnm6uAF3JV8j7vjooKVIT7Ne Kng+7R7GqnnJYeezRPAgwyEcHt50ng4B98I98q+AHFaWyMtpW9zo4kYkd1G/Nntx nMqoDo6QwICy5GbvAaXbzsGfQm8ULDeS9Zk0AdmSpZalRsplym0mRTfnTcsRkwjk	After the certificate request is signed, imports the trusted certificate authority (CA) certificate that you used to sign the request. **Note** Trusted CA validity must be between 5 to 30 years, else the import will fail.

	Command or Action	Purpose
	KOmRkBwUNMu2DdByY9fcuVZu+RcXgkw= -----END CERTIFICATE----- Certificate info ********** Owner: CN=Collabation Root CA,O=Collabation,L=Bangalore,ST=Karnataka,C=IN Issuer: CN=Collabation Root CA,O=Collabation,L=Bangalore,ST=Karnataka,C=IN Valid from: Wed Jul 16 02:08:12 PDT 2025 until: Sat Jul 14 02:08:12 PDT 2035 Certificate fingerprint (MD5): 9D:D4:F6:0C:08:12:2A:7E:EE:64:16:93:D9:28:AF:55 Do you want to continue to import this certificate, additional validation will be perfomed? [y/n]: y Hostname(config)#	
Step 6	**crypto key import cer label** *label-name* { **der url**{ **sftp: | http:** } | **pem { terminal | url {sftp: | http: }} [default]** **Example:** Hostname(config)# **crypto key import cer label mykey terminal** Enter certificate... End with a blank line or "quit" on a line by itself -----BEGIN CERTIFICATE----- MIIESjCCAjICFC9ZC6cO5IXQSmjDeP7z3uWMgmLPMA0GCSqGSIb3DQEBCwUAMGkx CzAJBgNVBAYTAklOMRIwEAYDVQQIDAlLYXJuYXRha2ExEjAQBgNVBAcMCUJhbmdh bG9yZTEUMBIGA1UECgwLQ29sbGFiYXRpb24xHDAaBgNVBAMME0NvbGxhYmF0aW9u IFJvb3QgQ0EwHhcNMjUwNzI5MDcyMDIxWhcNMjYwNzI5MDcyMDIxWjBaMQswCQYD VQQGEwInJzELMAkGA1UECAwCJycxCzAJBgNVBAcMAicnMQswCQYDVQQKDAInJzEL MAkGA1UECwwCJycxFzAVBgNVBAMMDnZtNTYuY2lzY28uY29tMIIBIjANBgkqhkiG 9w0BAQEFAAOCAQ8AMIIBCgKCAQEAoyi9yh/+/W3F+SyJNebEDeouCItlba+mSv9n ztADtMKsxDRPd+4gO4xPFeoVGPtuixuVtlxIGXB9lzfrie0Byk3v/kmwkCPDLaBY OWXQAzPqf/4EDBQZRQ9+7yGqtfFnn5aiLojFefGvkniR2Z7YXzCMHDdjDic9ygUm /Hg6qvpOwRrBCCOrLGOb+giG0c2vvjtSq9fYqI6VyJftjRqE5CTODWpXMxlpxGS8 XUATJcXaJkdqDWbeHFOlqhNCBYSQEiSsP/gJOrg06lIAEApnrBHUrXRf+fKNGPmJ en54KqR37TiONPBy0i6YNnXbFDXexETJZxx6/dNcQSZVoDId8QIDAQABMA0GCSqG SIb3DQEBCwUAA4ICAQB2g2i43xaFgBtYxeATxY+rqkYuRUex0x5ojzZsEdlCWV9b wxyRQ5G7p3a81/bm8tnzlN2sUWb86Xjf37mVP7xY0bYM1A/kPNEwpkPVivGHycXQ cX9G98Nl3c1vHB5/v94tb8tcvnpVWs5uemSWDsRbh9x729YcWGQ+ZO7D9sEXklaF xuvN+5ufx6V7ARu7RVDC0d5Ti5du37gB6731R/E68hN2XHdGMSvCVfnTSnNUbyr8 qhwaArRVTCMqPqSKv0PkhQHgINZQ2p1hXPf38NBdrVnh/2wE0ib0zP/fnY7AsxuW jBJyDcdciXkbDBXzMHGsH/vf0yvj5DAgh33mN/YI9q0f3b9WZUyh/NrFUpeV6Jn/ 5570WJ2XEDFb39L3ok590yHY0NqewPfXYfnJvo2doHltm88V4KKvPzGWAvJ6+8MY IO/8mlA3E2Or5IFjUkOTr/yvX67vlqSzFwArC9Hjxw4VG23mi4fZQ0Qnpj2Mnwrs YEDQ/oP7uRkHjtPwkq35Pyug4ieyWIumKbBvMr+iCdS4sFmKlmKmYteT6s4SIDfQ ryZv6MgqhOhDfEvu/F4lbdmW9SjFxSWd8FDHPgkS8iVmDfbM/sOyVfC6cDVdL0ba JOOmr/uzFBk5smVepZg1fALYgsGeiFR9tItDmouk9fYYbq9jcjHEFqCLWryu6Q== -----END CERTIFICATE----- Import succeeded Hostname(config)#	After the root CA is imported, imports the signed certificate.
Step 7	**offline** **Example:**	Initiates CCCSP offline mode.

	Command or Action	Purpose

	`Hostname# offline`
	
	`!!!WARNING!!!: Putting the system offline will terminate all active calls.`
	
	`Do you wish to continue[n]?: y`	

Step 8	**reload** **Example:** `Hostname(offline)# reload`	Restarts the CCCSP system and enables CCCSP to verify the imported trusted certificate.

Example

Example for creating a signed certificate:

```
Hostname# configure terminal
Hostname(config)# crypto key generate rsa label mykey modulus 2048 default
Key generation in progress. Please wait...
The label name for the key is mykey

Hostname(config)# crypto key certreq label mykey url sftp:
Address or name of remote host? test:test123@192.168.202.216
Username (ENTER if none)? anonymous
Password (not shown)?
Destination path? netmod/mykey.csr
Uploading CSR file succeed

Hostname(config)# crypto key import trustcacert label rootCA url sftp:
Import certificate file...
Address or name of remote host? test:test123@192.168.202.216
Source filename? netmod/rootCA/cacert.pem
1212 bytes received.

Hostname(config)# crypto key import cer label mykey url sftp:
Import certificate file...
Address or name of remote host? test:test123@192.168.202.216
Source filename? netmod/mycert.cer
952 bytes received.
Import succeeded
```

What to do next

- Import the trusted CA certificates for any of the TLS peer elements.

Create and Import a Self-Signed Certificate

Self signed certificates are generated using SSL utilities on a Linux or Windows machine and are imported to CCCSP.

SUMMARY STEPS

1. **vim** *filename*

2. **openssl req -new -newkey rsa:2048 -nodes -keyout** *key* **-out** *csr* **-config** *configuration file name*
3. **openssl x509 -req -days** <*days*> **-in** <*csr*>**-signkey** <*keys*> **-out** <*certificate*>
4. **configure terminal**
5. **crypto key import trustcacert label** <*label_name*> **terminal**
6. **crypto key import rsa label** *label-name* **pem terminal**
7. **offline**
8. **reload**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**vim** *filename*
	Example:
	This is an example only. You can use any text editor.
	
	`Linux-server-test$ vim abc`
	`distinguished_name = req_distinguished_name`
	`[ req_distinguished_name ]`
	`countryName = Country Name (2 letter code)`
	`countryName_default = US`
	`countryName_min = 2`
	`countryName_max = 2`
	
	`stateOrProvinceName = State or Province Name (full name)`
	`stateOrProvinceName_default = California`
	
	`localityName = Locality Name (eg, city)`
	`localityName_default = San Jose`
	
	`organizationName = Organization Name (eg, company)`
	`organizationName_default = Cisco Systems, Inc.`
	
	`organizationalUnitName = Organizational Unit Name (eg, section)`
	`organizationalUnitName_default = Cisco Webex`
	
	`commonName = Common Name (eg, YOUR name)`
	`commonName_max = 64`
	`emailAddress = Email Address`
	`emailAddress_default = csg-avops@cisco.com`
	`emailAddress_max = 40`
		On a Linux server, create a configuration file.

Step 2	**openssl req -new -newkey rsa:2048 -nodes -keyout** *key* **-out** *csr* **-config** *configuration file name* **Example:** `openssl req -new -newkey rsa:2048 -nodes -keyout me90sjvce001.webex.com.key -out me90sjvce001.webex.com.csr -config abc`	Generate self signed certificate and csr pair on Linux server. **Note** Use the **crypto key delete all** command to delete all the existing certificates before importing new certificates.

	Command or Action	Purpose
Step 3	**openssl x509 -req -days** <*days*> **-in** <*csr*>**-signkey** <*keys*> **-out** <*certificate*> **Example:** `openssl x509 -req -days 720 -in me90sjvce001.webex.com.csr -signkey me90sjvce001.webex.com.key -out me90sjvce001.webex.com.cer`	Sign the CSR file with your own key.
Step 4	**configure terminal** **Example:** `Hostname# configure terminal`	Log in to CCCSP and enter the configuration mode.
Step 5	**crypto key import trustcacert label** <*label_name*> **terminal** **Example:** `Hostname(config)# crypto key import trustcacert label sample_cert terminal` `% Self-signed CA certificate:` `-----BEGIN CERTIFICATE-----` `MIIDLjCCAhagAwIBAgIBATANBgkqhkiG9w0BAQUFADAwMS4wLAYDVQQDEyVJT1Mt U2VsZi1TaWduZWQtQ2VydGlmaWNhdGUtMzc4ODk3MTYzMB4XDTE3MDExODA2MzYy N1oXDTIwMDEwMTAwMDAwMFowMDEuMCwGA1UEAxM1SU9TLVNlbGYtU2lnbmVkLUNl cnRpZmljYXRlLTM3ODg5NzE2MzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoC ggEBAI/k+J1/RdXkUu3aBp8qIMVA7ifpRehG9AXJKlqOafc9Ly92hwNxeLGV/U8k Xlo/fuoyaNyLIu9GwS1BfvM3yHOthhX+T5RHgcj3s1Yctl6HUW93M/EJYluo5RDE NAXJ2UXa/Utl9ZGjCvat8h3N4QduP2ulIsK1IqyYLDRwD1fiSNFrdZB2zzIE1M7g eeitn4n1INHiVtH0jQmO4En/FjUa3YPCFEyB1/U17YGWN/GOHguCsZluL8WywAT5 PqluaipVxWoCzXCb74BSxTJiHs/tmPGkIHi57RvLKxgqr5vHXCOsWsQ6/C9z6My3 tvE6dtLHuP2RgR6r+3xOhKdqcHECAwEAAaNIMEEwDwYDVR0TAQH/BAUwAwEB/zAf BgNVHSMEGDAWgBSIzQOOrJrnxzR8LEQ2VIIfVFpO2DAdBgNVHQ4EFgQUiM0Djqya 58c0fCxENlSCH1RaTtgwDQYJKoZIhvcNAQEFBQADggEBABhYrhWv9DZ0sZZt7Smc o5pgIIFFOtGQYc+ei7H6QNzW5iNSZbSPBAIpmVMQWHVS6cOvJ/N63ayQ+1TN3rZm wmOU9tFExBzjge0nX+Go+0KdWNNQG4XO8SU7BKwM8iWTsM1jT1j6cb9Bv1kMgXW0 5K5AzVYIbaTP/OMoMCsuOJts+GI/Q82H7tlIbdJFbbu3iVEN+gf3coUrHa4X2jLr K3EVLniCLedkcXdy5TppTvQM9j1FzkGMIrWAlFlp/Vh2CTigJy8GZ4pWt5QzjO6m KuP6FZxGPNe8F5BsFCWNM5aHPa8MUqlFKZMuUb50w43SZRT3xfI2WLv1yd49f65T mBA=` `-----END CERTIFICATE-----` `Certificate info` `**********` `Owner: CN=cusp.test.cisco.com,OU=Cisco,O=Example Company,L=San Francisco,ST=Cal ifornia,C=US` `Issuer: CN=cusp.test.cisco.com,OU=Cisco,O=Example Company,L=San Francisco,ST=Ca lifornia,C=US` `Valid from: Wed Jul 16 06:02:54 PDT 2025 until: Sat Jul 14 06:02:54 PDT 2035` `Certificate fingerprint (MD5): 8F:62:94:9C:87:09:5E:B7:91:15:8B:94:63:FD:17:B5`	Import the self-signed CA certificate to CCCSP. Copy the content of the certificate created and paste it when prompted. **Note** Trusted CA validity must be between 5 to 30 years.

	Command or Action	Purpose
	`Do you want to continue to import this certificate, additional validation will be perfomed? [y/n]: y`	

Step 6	**crypto key import rsa label** *label-name* **pem terminal**
	Example:
	`Hostname# configure terminal`
	`Hostname(config)# crypto key import rsa label testtls pem terminal`
	`Enter certificate...`
	`End with a blank line or "quit" on a line by itself`
	`-----BEGIN CERTIFICATE-----`
	`MIIDjTCCAnUCFFg2NWDvvdpXKLwV4hZZ/H9JeAqjMA0GCSqGSIb3DQEBCwUAMIGC`
	`MQswCQYDVQQGEwJVUzETMBEGA1UECAwKQ2FsaWZvcm5pYTEWMBQGA1UEBwwNU2Fu`
	`IEZyYW5jaXNjbzEYMBYGA1UECgwPRXhhbXBsZSBDb21wYW55MQ4wDAYDVQQLDAVD`
	`aXNjbzEcMBoGA1UEAwwTY3VzcC50ZXN0LmNpc2NvLmNvbTAeFw0yNTA3MTYxMzAy`
	`NTRaFw0zNTA3MTQxMzAyNTRaMIGCMQswCQYDVQQGEwJVUzETMBEGA1UECAwKQ2Fs`
	`aWZvcm5pYTEWMBQGA1UEBwwNU2FuIEZyYW5jaXNjbzEYMBYGA1UECgwPRXhhbXBs`
	`ZSBDb21wYW55MQ4wDAYDVQQLDAVDaXNjbzEcMBoGA1UEAwwTY3VzcC50ZXN0LmNp`
	`c2NvLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMXIJCYqLZEd`
	`ZNwFKmvtxNCUgjLIQ9HBa+ivfqqv2fQPIGNv0+y++Io+PDjrXbUhC53EwZ0OtJ/A`
	`TSsuVb1/pXZgoes9eHSK7gA8ZujDi1Oj66EONcmDN9JF1ujUzHm5IUQG5SHntK1H`
	`WrGQRWPRSPEJTxrvt2H0Vp/3Iwb/7wKGpd/Gdor9v6c60bALNN8XoNs+u5UnC1eM`
	`YP4EzkFiou6zE4pI79vmVgHtEaWpCEMAdIbccToaOU/VmVu0C4pfmDdwmX1Yj2eQ`
	`tfWupUEdEbYb0wgvaBTIJ1xeL0Zd+38b8uwPuljjcsSQbPS6lgGMRN+6CNb3IIkS`
	`QB2rFUUs3XECAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAYDw0t81LSNpIDJfJDVPU`
	`Bvc1qe7g08dT+eoWFN6m8s1IZoBs0y3yk6P75IKWCLOY+wbCC2NII2Gg04Odr7b9`
	`3OhsIsDESa/sqOHIA0V+67sXpV5TimpJqUzcimHhvHSI764SSLU8Wf98oKlUVp7Z`
	`OHCFRYnh6brnGyrslIqR5YiNYuy7e1cH6OXAG706D14Ck63qvzF7rOttfeGVTfcf`
	`Hv3HzS+8elNtKX0TON7ijAWv3Jlntu74T9ZaqZZv15qe8i36dIM7j6g5bjOej2Ki`
	`a5xPEqahkYZhpvXzrRD3swhcB7vfsjdvM9NfSBEjagxO8yZb2GfdJ44wSFcNjf/V`
	`Zw==`
	`-----END CERTIFICATE-----`
	`Enter private key...`
	`Private key passphrase?`
	`End with a blank line or "quit" on a line by itself`
	`-----BEGIN PRIVATE KEY-----`
	`MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDFyCQmKi2RHWIc`
	`BSpr7cTQlIIyyEPRwWvor36qr9n0DyBjb9PsvviKPjw46121IQudxMGdDrSfwE0r`
	`LlW9f6V2YHHrPXh0iu4APGbow4tTo+uhDjXJgzfSRdbo1Mx5uSFEBuUh57StR1qx`
	`kEVj0UjxCU8a77dh9Faf9yMG/+8ChqXfxnaK/b+nOtGwCzTfF6DbPruVJwtXjGD+`
	`BM5BYqIusxOKSO/b51YB7RGlqQhDAHU23HE6GjlP1ZlbtAuKX5g3cJ19WI9nkIX1`
	`rqVBHRG2G9MIL2gU0ydcXi9GXft/G/LsD7pY43LEkGz0upYBjETfugjW9yCJEkAd`
	`qxVFLNlxAgMBAAECggEACoIN81+r47th+Vem/W18WbGESSogfT/hutLWlEJc/59s`
	`VA2HAmDXvBZ6Qna2GaODyCIGPVGi2pqmIAVLVxQ6WxTfCilPRjd7V/TebkO0HdMq`
	`jXrBfpGlft/rAaALre+cJs+NESQPFmYkmyv2nMvtKrCZje+w25tQnhpZOQ6UihhC`
	`CdFIzNDd/Sx6N+HOoE/+NU+ERUb/yZQi0a2OyJm5cDvCOBLUWV35VRBxM5j6yJ+r`
	`zu/LWTq3duuwGg6nAzqoAfvs34bIYcLk6cqSIrYHIq0455czdog0EWJpfMG5E077`
	`9kMQTQwO00d0MAZsu5aio08L3BFfT3rToniaRk7ZqQKBgQD/rHYZ9hYLuVHwKV0P`
	`BTLHSIoreKcUtgOS/G1xyw/RUljNdSZhYbYfmv5qHs2/8JeFw+/GBjItVC4+LZR/`
	`pNFJHtiMMg3po6LAYHw3jNc0xfPaDUpYDFhU+704IzOb4A7/XZTedxHzqJ2bn4gA`
	`kSEf6uvOGDKpbHSzviO4lQW9JQKBgQDGCMOqPqEPoAV/HgQnYn8ykZZL967aye+Z`
	`+4p5Wh6Fhco1778wMYZ/jXjJ+HdK/sHGT4Py7BIB8oj78B18HUDOCgUFcFWPp48E`
	`HJkIhvGqy6Brf3WHek+EcY9PjimdWe61QVvJQU9biTej3t2yVvErNBEXOaChy6E/`
	`vvS08P9bXQKBgH1vhk28XxqHZDzYflB1M3F0FF25hvdxSRWfBK70eiEvImQ2mori`
	`XnDpIPK9hCIsDprRA1z7Azkb+MgGorlxCkC+gm1dzVeYMawCak7yqU6546ZSKOxD`
	`uj6Pd0kIdDSAW7tLlVMDxenuCs0kGpx0l3TKU8A8PPbEQAlDOtE/Q56FAoGBAJ0w`
	`j78uf0Gbz0EbXki00iodjaM3Mr/P0jPzifLRKbDqhZtSdfuXFB9i+CG461fH55+1`
	`thK4d4K8NSdlbMH7pljlcx7iD8g7RDYLITSZG0sSzZN5vCadD2qJSUWLggP+5RtTw`
	`0B5rjMYnoNaeLHNGt5+HcJly0KIctB2ZQTvun4KxAoGAfoiJThJpklsIL/PbFBJE`
		Import the self-signed certificate. For each prompt provide the required details.

	Command or Action	Purpose
	wvKI6qJa7UdErubEuSedrarD72C2OOm9GBKgWxeJR3Jdzi6n9nSQoKBLJ/qV+AIV xb8HzNFwHPElsQR1Q2TPWoOx1SRseEtzueAZAvuHhgDXN/cLs5fb4tT88qPGxtKp w1z9CxJT96OQFn6OireQSp8= -----END PRIVATE KEY----- Import succeeded.	
Step 7	**offline** **Example:** Hostname# **offline** !!!WARNING!!!: Putting the system offline will terminate all active calls. Do you wish to continue[n]?: y	Initiates CCCSP offline mode.
Step 8	**reload** **Example:** Hostname(offline)# **reload**	Restarts the CCCSP system and enables CCCSP to verify the imported trusted certificate.

Example

On a Linux server, execute the following commands:

```
Linux-server-test$ vim abc
distinguished_name = req_distinguished_name
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = US
countryName_min = 2
countryName_max = 2

stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = California

localityName = Locality Name (eg, city)
localityName_default = San Jose

organizationName = Organization Name (eg, company)
organizationName_default = Cisco Systems, Inc.

organizationalUnitName = Organizational Unit Name (eg, section)
organizationalUnitName_default = Cisco Webex

commonName = Common Name (eg, YOUR name)
commonName_max = 64
emailAddress = Email Address
emailAddress_default = csg-avops@cisco.com
emailAddress_max = 40

openssl req -new -newkey rsa:2048 -nodes -keyout me90sjvce001.webex.com.key -out
me90sjvce001.webex.com.csr -config abc
openssl x509 -req -days 720 -in me90sjvce001.webex.com.csr -signkey me90sjvce001.webex.com.key
 -out me90sjvce001.webex.com.cer
```

Configure TLS on Cisco Contact Center SIP Proxy (CCCSP)

After you import the certificates, you must enable TLS connections. If you want more security, you can create a list of trusted peers. If you create such a list, only connections from those peers are accepted. The peer's hostname entry must be the peer's subjectAltName in its certificate. If subjectAltName is not used in the certificate, the peer's hostname entry must be CN.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **sip tls**
4. **sip tls trusted-peer** { *peer's hostname* }
5. **sip tls connection-setup-timeout** { *value in seconds* }
6. **sip tls** {*v1.1* | *v1.2* | *v1.3*}

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**sip tls** **Example:** `Hostname(cusp-config)# sip tls`	Enables the use of SIP TLS connections with other SIP entities, providing secure communication over the Internet.
Step 4	**sip tls trusted-peer** { *peer's hostname* } **Example:** `Hostname(cusp-config)# sip tls trusted-peer example.com`	Creates a list of trusted peers.
Step 5	**sip tls connection-setup-timeout** { *value in seconds* } **Example:** `Hostname(cusp-config)# sip tls connection-setup-timeout <1-60>`	The user specifies the time in SIP Proxy to establish connection with the trusted peer. The default value is 1 second. The range of values is 1 to 60 seconds.
Step 6	**sip tls** {*v1.1* | *v1.2* | *v1.3*} **Example:** `Hostname(cusp-config)# sip tls v1.2`	Enables SIP TLS versions (supports both TLS v1.2, and TLS v1.3). By default, TLS is not enabled for SIP networks. When TLS is enabled for a given network, the connection between the endpoints is established using the highest

	Command or Action	Purpose
		common protocol version negotiated during handshake. For instance, configuring **sip tls** *v1.2* sets "Min TLS 1.2", then the connection is established using TLS v1.2 or TLS v1.3. **Note** Starting from CCCSP-15.0(1) release, CCCSP supports minimum TLS 1.1, 1.2, and 1.3 versions. CCCSP no longer supports TLS v1.0.

Example

Example for TLS configuration:

```
Hostname# cusp
Hostname(cusp)# configure
Hostname(cusp-config)# sip tls
Hostname(cusp-config)# sip tls trusted-peer example.com
Hostname(cusp-config)# sip tls connection-setup-timeout <1-60>
Hostname(cusp-config)# sip tls v1.3
```

Update Web Session with an Imported Signed Certificate

HTTPS is enabled by default on CCCSP for web services. You need not manually generate a crypto key and pass it to the web session security to enable HTTPS. However, you should be able to import a signed certificate that you generated externally, and update the web session with this new key label.

SUMMARY STEPS

1. **configure**
2. **crypto key import rsa label** *label-name* { **der url {sftp: | http: } | pem { terminal | url {sftp: | http:**}} [**default**]
3. **web session security keylabel** *labelname*
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `Hostname# configure terminal`	Enters configuration mode.
Step 2	**crypto key import rsa label** *label-name* { **der url {sftp: | http: } | pem { terminal | url {sftp: | http:**}} [**default**]	Imports the signed certificate.

	Command or Action	Purpose
	Example: `Hostname(config)# crypto key import cer label mykey url sftp:`	
Step 3	**web session security keylabel** *labelname* **Example:** `Hostname(config)# web session security keylabel mykey`	Associates a security key for HTTPS.
Step 4	**end** **Example:** `Hostname(config)# end`	Exits to privileged EXEC mode.

Example

Updating web session with an imported signed certificate:

```
Hostname# configure terminal
Hostname(config)# crypto key import cer label mykey url sftp:
Import certificate file...
Address or name of remote host? 192.0.2.2
Source filename? netmod/mycert.cer
952 bytes received.
Import succeeded
Hostname(config)# web session security keylabel mykey
Hostname(config)# end
```

Note HTTPS is enabled by default on CCCSP for web services. You need not manually generate a crypto key and pass it to the web session security to enable HTTPS. The application supports only TLS v1.2 for HTTPS. If you delete the certificate from the web session security and try to log in through HTTP, you will be redirected to HTTPS. Only the latest connection is retained and the remaining connections are logged out.

Note CCCSP supports Min TLS v1.2 (supports both TLS v1.2, and TLS v1.3) for HTTP protocol.

Configure Lite Mode

You can to switch to lite mode to improve the performance of the Cisco Contact Center SIP Proxy (CCCSP) application. In Lite Mode, which requires you to disable record-route, the performance of the application is boosted. In standard mode, the application processes calls up to the licensed limit.

By default, the application is in standard mode.

For information on the performance difference when using Lite Mode versus standard mode, see *Release Notes for Cisco Contact Center SIP Proxy (CCCSP)*.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **lite-mode**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**lite-mode** **Example:** `Hostname(cusp-config)# lite-mode`	The CCCSP application is switched to Lite mode.

Example

The following example shows how to configure to switch to Lite mode:

```
Hostname# cusp
Hostname(cusp)# configure
Hostname(cusp-config)# lite-mode
```

Configure Call Rate Limit

One of the ways you can control the calls on CCCSP is to restrict the number of calls that the CCCSP can handle.

SUMMARY STEPS

1. **cusp**
2. **configure**
3. **call-rate-limit** *limit*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**cusp** **Example:** `Hostname# cusp`	Enters CCCSP Management mode.
Step 2	**configure** **Example:** `Hostname(cusp)# configure`	Enters CCCSP Configuration mode.
Step 3	**call-rate-limit** *limit* **Example:** `Hostname(cusp-config)# call-rate-limit 50`	Sets the maximum call rate that the CCCSP can handle.

Example

The following example limits the number of calls that the system can process to 50:

```
Hostname# cusp

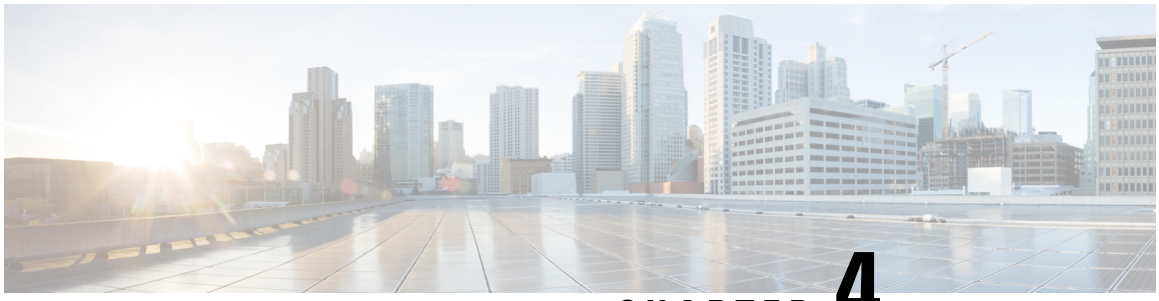
Hostname(cusp)# configure
Hostname(cusp-config)# call-rate-limit 50
```

Commit the Configuration

Now you must commit the configuration. Committing the configuration serves two purposes: the configuration becomes active, and is persisted.

- To see the current active configuration, enter the **show configuration active** command.
- To see what the active configuration will be after you commit your changes, enter the **show configuration candidate** command.
- To commit the configuration for this example, enter the following command:

  `Hostname(cusp-config)# commit`



CHAPTER 4

Configure Users and Groups

All configuration and administration functions for Cisco Contact Center SIP Proxy (CCCSP) are available through the graphical user interface (GUI). However, you may find using the command-line interface (CLI) is more efficient than using the GUI. For example, you may want to create a script to configure a large number of users for a specific system. In this case, the CLI can be more efficient.

- Add and Modify a User, on page 57
- Add and Modify a Group, on page 61

Add and Modify a User

Users, or users, configured in Cisco Unified Communications Manager can be imported to the Cisco Contact Center SIP Proxy (CCCSP) database.

The procedure described in this section allows you to create a new user in the system. Use the same procedures to modify an existing user's properties.

The application license determines the maximum number of users.

Note Ensure not to use reserved keywords or any name that may conflict with the regular users or process names in Linux (root, bin, daemon, adm, lp, sync, shutdown, halt, mail, operator, games, and so on.) while creating the users.

Required Data for this Procedure

The following information is required for adding or modifying a user:

- Username—The user ID. The username must be at least 3 and no more than 32 characters. CCCSP allows only letters, numbers, underscore (_), dot (.), and dash (-) in user IDs. User IDs must start with a letter. Do not use spaces in the username.
- (Optional) Full name—First and last name of the user. It must start and end with quotation marks (“ ”).
- (Optional) Group—Name of an existing group in which this user is a member.
- (Optional) Password—Password for logging into the Cisco Contact Center SIP Proxy (CCCSP) GUI. The password must include a minimum length ranging from 8 through 64 characters. There is no limit

on the maximum number of characters. Spaces are not allowed. A valid password should have at least one uppercase letter, one lowercase letter, one number, and a symbol.

- (Optional) PIN—No PIN is required for vCCCSP users.

Exec mode

SUMMARY STEPS

1. **username** *userid* [**create | delete | fullname** [**firstname** *"first-name"* | **last** *"last-name"* | **display** *"full-name"*] | **group** *group-name* | **language** *"language"* | **password** *"password"* | **pin** *number*]
2. **show users** or **show user detail username** *userid*
3. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**username** *userid* [**create	delete	fullname** [**firstname** *"first-name"*	**last** *"last-name"*	**display** *"full-name"*]	**group** *group-name*	**language** *"language"*	**password** *"password"*	**pin** *number*]
									Example:
									`Hostname# username user1 create`
									`Hostname# username user2 fullname display "User 2"`
									`Hostname# username user2 group sales`
									`Hostname# username user2 password "Green123!"`
									`Hostname# username user2 pin 4444`
									`Hostname# username user2 delete`
										Creates the user with the specified user ID. The optional parameters configure more information for the user:
										• *userid* —User ID of the user. The user ID must be at least 2 and no more than 31 characters. Cisco Unified SIP Proxy allows only letters, numbers, underscore (_), dot (.), and dash (-) in user IDs. Do not use spaces in the username. User IDs must start with a letter.
										• **create** —C reates the user with no other information.
										• **delete** —Deletes an existing user.
										• **fullname** —Specifies a full name for this user. This full name appears on telephone displays.
										• **group** —Associates this user with an existing group.
										• **language** —Specifies the default language used for the specified user. See the Release Notes for Cisco Unified SIP Proxy for a list of available languages.
										• **password** —Specifies a password for this user. The *password* value must be entered within quotation marks (“ ”). Spaces are not allowed. Acceptable password characters are lowercase letters a to z, uppercase letters A to Z, digits 0 to 9, and the following symbols: -,. + = _ ! @ # $ ^ * () ? / ~ < > & %.
										• **pin** —Specifies a personal identification number (PIN) for this user. The user enters this number from the telephone when accessing the voice-mail system. The

	Command or Action	Purpose
		PIN can contain a maximum number of 16 digits. The asterisk (*) and pound sign (#) cannot be used.
Step 2	**show users** or **show user detail username** *userid* **Example:** `Hostname# show user detail username user2`	Displays a list of user IDs for all users configured on the system. or Displays the detailed information configured for the specified user.
Step 3	**copy running-config startup-config** **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

Example

The following output illustrates the **show users** and **show user detail username** commands:

```
Hostname# show users
user1
user2

Hostname# show user detail username user2
Full Name: User 2
First Name:
Last Name: user2
Nickname: user2
Phone:
Phone(E.164):
Language: en_ENU
Hostname#
```

Configuration Mode

SUMMARY STEPS

1. **config t**
2. **username** *userid* [**create | phonenumber** *phone-number* | **phonenumberE164** *full-number*]
3. **exit**
4. **show users** or **show user detail username** *userid*
5. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**config t** **Example:** `Hostname# config t`	Enters configuration mode.

Step 2	**username** *userid* [**create | phonenumber** *phone-number* | **phonenumberE164** *full-number*]
	Example:
	`Hostname(config)# username user3 create`
	`Hostname(config)# username user3 password Usr@50180`
		Creates the user with the specified user ID. The optional parameters configure more information for the user:
		• *userid*—User ID of the user. The user ID must be at least 2 and no more than 31 characters. Cisco Unified SIP Proxy allows only letters, numbers, underscore (_), dot (.), and dash (-) in user IDs. Do not use spaces in the username. User IDs must start with a letter.
		• **create**—Creates the user with no other information.
		• **password**—Specifies a password for this user.

Step 3	**exit** **Example:** `Hostname(config)# exit`	Exits configuration mode.
Step 4	**show users** or **show user detail username** *userid* **Example:** `Hostname# show user detail username user2`	Displays a list of user IDs for all users configured on the system. or Displays the detailed information configured for the specified user.
Step 5	**copy running-config startup-config** **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

Example

The following example illustrates configuring a user and the output from the **show** commands:

```
Hostname(config)# username user3 create
Hostname(config)# username user3 password User@5521
Hostname(config)# exit
Hostname# show users
user1
user2
```

```
user3
Hostname# show user detail username user3
Full Name: User 3
First Name:
Last Name: user3
Nickname: user3
Password: ******
Language: en_ENU
```

Add and Modify a Group

A group is a collection of users, usually with a common function or purpose, such as sales, main office, customer service, or technicians. A group has the following characteristics:

- Members of the group can be individual users or other groups.
- The group is assigned an extension.
- A group can have zero or more users as owners. An owner of a group can add and delete members. Additionally, an owner can add and delete other owners to the group.
- Members can belong to more than one group.
- Members can be added to the group using the configuration mode **groupname** command or using the EXEC mode **username** command. For more information on **username** command, see Adding and Modifying a User.

Note Users must exist before being added to a group. See Adding and Modifying a User to configure the user's detailed information.

Required Data for this Procedure

The following information is required to define a group:

- EXEC mode:
  - Name of group
  - Description of group
  - Full name of group
- Configuration mode:
  - Name of group
  - One or more existing user or group IDs to be added as members
  - One or more existing user IDs to be added as owners
  - Extension or telephone number of the group
  - Full E.164 telephone number of the group

Exec mode

SUMMARY STEPS

1. **groupname** *userid* [**create | delete | description** *"description"* | **fullname** *"full-name"*
2. **show groups** or **show group detail groupname** *groupid*
3. **copy running-config startup-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**groupname** *userid* [**create	delete	description** *"description"*	**fullname** *"full-name"*
				Example:
				`Hostname# groupname sales fullname "Sales Department"`
				`Hostname# groupname sales description "Retail Sales Department"`
				`Hostname# groupname sales delete`
					Creates the group with the *groupid* value. The optional parameters configure more information for the group:
					• **create** —C reates the group with no other information.
					• **delete** —Deletes an existing group.
					• **description** —Specifies a description of the group.
					• **fullname** —Specifies a long name for the group.

Step 2	**show groups** or **show group detail groupname** *groupid* **Example:** `Hostname# show group detail groupname sales`	Displays a list of group IDs for all configured groups. This command does not display the details for the groups. or Displays the detailed configuration information for the group *groupid* value.
Step 3	**copy running-config startup-config** **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

Example

The following example creates a group and displays the output of the **show** commands:

```
Hostname# groupname sales fullname "Sales Department"
Hostname# groupname sales description "CA office"

Hostname# show groups
Administrators
sales

Hostname# show group detail groupname sales
Full Name: Sales Department
Description: CA office
Phone:
```

```
Phone(E.164):
Language: en_ENU
Owners:
Members:
Hostname#
```

Configuration Mode

SUMMARY STEPS

1. **config t**
2. **groupname** *groupid* [**member username | owner ownername | phonenumber phone-number | phonenumberE164 full-number**
3. **exit**
4. **show groups** or **show group detail groupname** *groupid*
5. **show groups** or **show group detail groupname** *groupid*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**config t** **Example:** `Hostname# config t`	Enters configuration mode.

Step 2	**groupname** *groupid* [**member username | owner ownername | phonenumber phone-number | phonenumberE164 full-number**
	Example:
	`Hostname(config)# groupname sales member user1`
	`Hostname(config)# groupname sales owner user2`
	`Hostname(config)# groupname sales phonenumber 50163`
	`Hostname(config)# groupname sales phonenumberE164 14445550163`
		Creates the group with the *groupid* value. The optional parameters configure more information for the user:
		• **member** —Associates an existing user as a member of this group. Repeat this command to assign multiple users to the group
		• **owner** —Specifies the owner of the group. The owner is not considered a member. If the owner is to have access to the group's voice mailbox, also assign the owner as a member.
		• **phonenumber** —Associates a number or extension with this group. No spaces or dashes are allowed.
		• **phonenumberE164** —Associates a telephone number and area code with this group. No spaces or dashes are allowed.

Step 3	**exit** **Example:** `Hostname(config)# exit`	Exits configuration mode.

	Command or Action	Purpose
Step 4	**show groups** or **show group detail groupname** *groupid* **Example:** `Hostname# show group detail groupname sales`	Displays a list of group IDs for all configured groups. This command does not display the details for the groups. Displays the detailed configuration information for the group *groupid* value
Step 5	**show groups** or **show group detail groupname** *groupid* **Example:** `Hostname# copy running-config startup-config`	Copies the configuration changes to the startup configuration.

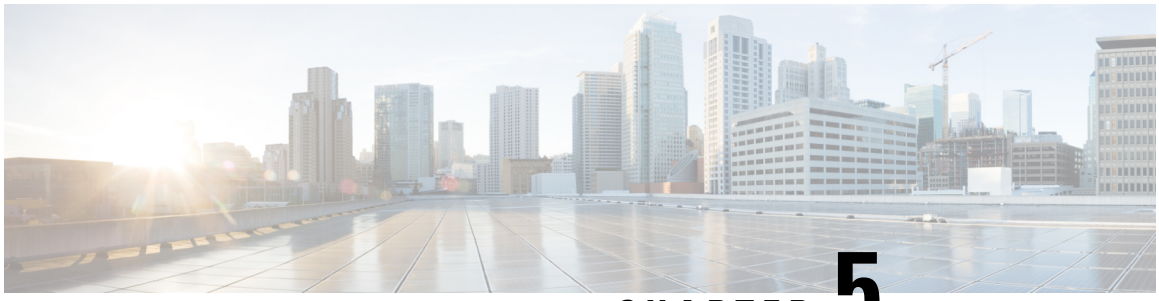
Example

The following example adds an owner and two members to the group sales and assigns sales a phone number:

```
Hostname# config t
Hostname(config)# groupname sales member user1
Hostname(config)# groupname sales member user2
Hostname(config)# groupname sales owner user1
Hostname(config)# groupname sales phonenumber 50163
Hostname(config)# groupname sales phonenumberE164 12225550163
sHostname(config)# exit

Hostname(# show groups
Administrators
sales

Hostname# show group detail groupname sales
Full Name: Sales Department
Description: CA office
Phone: 50163
Phone(E.164): 12225550163
Language: en_ENU
Owners: user1
Members: user1 user2
Hostname(#
```



CHAPTER 5

Back Up and Restore Data

- About Back up and Restore Data, on page 65
- Restrictions for Back Up and Restore Data, on page 65
- Back Up Files, on page 66
- Restore Files, on page 68
- Related Topics, on page 70

About Back up and Restore Data

Cisco Contact Center SIP Proxy (CCCSP) backup and restore functions use an SFTP server to store and retrieve data. The backup function copies the files from the CCCSP application to the SFTP server and the restore function copies the files from the SFTP server to the CCCSP application. The SFTP server can reside anywhere in the network as long as the backup and restore functions can access it with an IP address or hostname.

We recommend that you back up your configuration files whenever you make changes to the system or application files. Do backups regularly to preserve configuration data.

The system supports the following types of backup:

- All—Backs up all files and data.
- Configuration—Backs up only system and application settings.
- Data—Backs up only routes and application data.

Note You must be in offline mode when you back up or restore the system, so we recommend performing these tasks when call traffic is least impacted. Offline mode terminates all calls.

Restrictions for Back Up and Restore Data

- Cisco Contact Center SIP Proxy (CCCSP) does not support the following backup and restore capabilities:
  - Scheduled backup and restore operations. The backup and restore procedures begin when the appropriate command is entered.

- Centralized message storage arrangement. CCCSP backup files cannot be used or integrated with other message stores.
- Selective backup and restore. Only full backup and restore functions are available. Individual messages or other specific data can be neither stored nor retrieved.

Back Up Files

About Back Up Files

Cisco Contact Center SIP Proxy (CCCSP) automatically assigns a backup ID to each backup. Although there are the three different types of backups, the system does not take into account the type of backup when generating the backup ID. Therefore, you will never have two backups with the same backup ID, even if one is a configuration file and the other a data file.

To determine the backup ID of the file you want to restore, use the **show backup server** or **show backup history** commands in either EXEC or offline mode. Those commands list all available backup copies on the remote backup server and their respective backup IDs.

Before you begin

Make sure that the backup server is configured before backup. Follow the steps mentioned in Set Backup Parameters.

SUMMARY STEPS

1. **offline**
2. **backup category** { **all** | **configuration** | **data**}
3. **continue**
4. **show backup history**
5. **show backup server**

DETAILED STEPS

Procedure

	Command or Action	Purpose

Step 1	**offline**
	Example:
	`Hostname# offline`
	`!!!WARNING!!!: Putting the system offline will terminate all active calls.`
	`Do you wish to continue[n]? : y`
		Enters offline mode. All calls are terminated.
		Note Cisco Contact Center SIP Proxy (CCCSP) still routes calls in offline mode.

Step 2	**backup category** { **all**	**configuration**	**data**} **Example:**	Specifies the type of data to be backed up and stored.

	Command or Action	Purpose

	`Hostname(offline)# backup category all`
	`Hostname(offline)# backup category configuration`
	`Hostname(offline)# backup category data`	

Step 3	**continue** **Example:** `Hostname(offline)# continue`	Exits offline mode and returns the system to the previous online mode. The system begins processing new calls and voice messages.
Step 4	**show backup history** **Example:** `Hostname# show backup history`	Displays each backup file, its backup ID, the type of data stored in the file, and the success or failure of the backup procedure.
Step 5	**show backup server** **Example:** `Hostname# show backup server`	Displays a list of the backup files available on the backup server. The files are grouped by category, with the date of each backup and the backup file ID.

Example

The following examples display the output from the **show backup history** and **show backup server** commands:

```
Hostname# show backup history
#Start Operation
Category: Configuration
Backup Server: sftp://192.168.1.35/pub/cusp_backup
Operation: Backup
Backupid: 1
Date: Tue Sep 24 06:14:30 EDT 2019
Result: Success
Reason:
#End Operation

#Start Operation
Category: Configuration
Backup Server: sftp://192.168.1.35/pub/cusp_backup
Operation: Restore
Backupid: 1
Restoreid: 1
Date: Tue Sep 24 06:17:21 EDT 2019
Result: Success
Reason:
#End Operation

Hostname# show backup server

Category: Data
Details of last 5 backups
Backupid: 1
Date: Tue Aug 21 10:55:52 PDT 2019
Description:
```

```
Backupid: 2
Date: Tue Aug 21 18:06:33 PDT 2019
Description:
Backupid: 3
Date: Tue Aug 21 19:10:32 PDT 2019
Description:
Category: Configuration
Details of last 5 backups
Backupid: 1
Date: Tue Aug 22 10:55:48 PDT 2019
Description:
Backupid: 2
Date: Tue Aug 29 18:06:27 PDT 2019

Description:
Backupid: 3
Date: Tue Aug 29 19:10:29 PDT 2019
Description:

Hostname#
```

Restore Files

About Restore Files

After you create the backup files, you can restore them when needed. Restore is done in offline mode, which terminates all calls. You should therefore consider restoring files when call traffic is least impacted.

To determine the backup ID of the file you want to restore, use the **show backup server** or **show backup history** commands in either EXEC or offline mode.

SUMMARY STEPS

1. **show backup server**
2. **offline**
3. **restore id** *backup_id* **category** { **all | configuration | data**}
4. **show restore history**
5. Configure Smart license
6. **reload**
7. Reset the password for all the restored users (administrators, pfs-privusers, and pfs-readonly).

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**show backup server** **Example:** `Hostname# show backup server`	Lists the data and configuration backup files. Look in the backup ID field for the revision number of the file that you want to restore.

	Command or Action	Purpose

Step 2	**offline**
	Example:
	`Hostname# offline`
	`!!!WARNING!!!: Putting the system offline will terminate all active calls.`
	`Do you wish to continue[n]? : y`
		Enters offline mode. All calls are terminated.
		Note
		CCCSP still routes calls in offline mode.

Step 3	**restore id** *backup_id* **category** { **all** | **configuration** | **data**} **Example:** `Hostname(offline)# restore id 22 category all`	Specifies the backup ID value and the file type to be restored.
Step 4	**show restore history** **Example:** `Hostname# show restore history`	Prints backup restore details.
Step 5	Configure Smart license	Configure Smart software license which is a standardized licensing platform that facilitates you to deploy and manage Cisco software licenses easily and quickly. For more information, refer to *GUI Configuration Guide for Cisco Contact Center SIP Proxy (CCCSP)* and *CLI Configuration Guide for Cisco Contact Center SIP Proxy (CCCSP)*.
Step 6	**reload** **Example:** `Hostname(offline)# reload`	Activates the uploaded file information and restarts the CCCSP system.
Step 7	Reset the password for all the restored users (administrators, pfs-privusers, and pfs-readonly).	Refer to *Change your password* section in the *Configure Users* chapter in *GUI Admin Guide for Cisco Contact Center SIP Proxy (CCCSP)*.

Example

The following examples display the output for the **show restore history** and **show backup server** commands:

```
Hostname# show restore history
#Start Operation
Category:      Configuration
Backup Server: sftp://10.10.10.39/home/test/oct13
Operation:     Restore
Backupid:      1
Restoreid:     1
Date:          Sun Oct 12 21:10:41 PDT 2025
Result:        Success
Reason:
```

```
Version:       15.0.1.10000-6
#End Operation

#Start Operation
Category:      Data
Backup Server: sftp://10.10.10.39/home/test/oct13
Operation:     Restore
Backupid:      1
Restoreid:     1
Date:          Sun Oct 12 21:10:42 PDT 2025
Result:        Success
Reason:
Version:       15.0.1.10000-6
#End Operation


Hostname# show backup server
Category:        Data
Details of last 5 backups
Backupid:      1
Date:          Sun Oct 12 21:08:06 PDT 2025
Software Ver:  15.0.1.10000-6

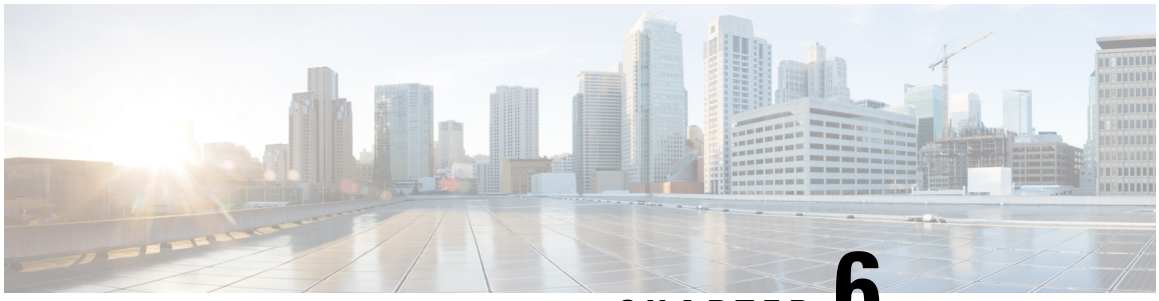
Backupid:      2
Date:          Mon Oct 13 09:12:39 PDT 2025
Software Ver:  15.0.1.10000-6

Category:        Configuration
Details of last 5 backups
Backupid:      1
Date:          Sun Oct 12 21:08:05 PDT 2025
Software Ver:  15.0.1.10000-6

Backupid:      2
Date:          Mon Oct 13 09:12:39 PDT 2025
Software Ver:  15.0.1.10000-6
```

Related Topics

For information on the CLI commands used for backup and restore configuration, see *CLI Command Reference for Cisco Contact Center SIP Proxy (CCCSP)*.



CHAPTER 6

Maintain Cisco Contact Center SIP Proxy (CCCSP) System

• Copy Configuration, on page 71

Copy Configuration

Use CCCSP EXEC commands to copy the startup configuration and running configuration to and from the hard disk on the CCCSP application, the network SFTP server, and the network TFTP server.

Note Depending on the specific TFTP server you are using, you might need to create a file with the same name on the TFTP server and verify that the file has the correct permissions before transferring the running configuration to the TFTP server.

Copy Startup Configuration from Hard Disk to Another Location

Starting in CCCSP EXEC mode, use the following command to copy the startup configuration on the hard disk to another location:

copy startup-config { **sftp:** *user-id:password@sftp-server-url* | **tftp:** *tftp-server-url* }

Syntax Description

sftp: *user-id:password@*	Username and password for the SFTP server. Include the colon (:) and the at sign (@) in your entry.
tftp-server-url	Absolute URL of the SFTP server including directory and filename. An example is userid:password@sftp-server-address/directory/filename.
tftp: *tftp-server-url*	URL of the TFTP server including directory and filename. An example is tftp://server/dir/filename.

This command is interactive and prompts you for the information. You cannot enter the parameters in one line. In this example, the startup configuration is copied to the SFTP server, which requires a username and password to transfer files. The startup configuration file is saved on the SFTP server with the filename “start”.

```
Hostname# copy startup-config sftp
Address or name of remote host? test:test123@10.3.61.16/home/test/
Source filename? startup-config
```

The following example shows the startup configuration copied to the TFTP server, which does not require a username and password. The command saves the startup configuration in the TFTP directory called "configs" as a file called "temp_start".

```
Hostname# copy startup-config tftp
Address or name of remote host? tftp://server/dir/temp_start
Source filename? temp_start
```

Copy Startup Configuration from Network SFTP Server to Another Location

Starting in module EXEC mode, use the following command to copy the startup configuration on the network SFTP server to another location:

copy sftp: { **nvram:startup-config** | **running-config** | **startup-config** | **system:running-config** }

For a description of this command, see *CLI Command Reference for Cisco Contact Center SIP Proxy (CCCSP)*.

This command is interactive and prompts you for the information. You cannot enter the parameters in one line. The following example illustrates this process. In this example, the SFTP server requires a username and password. This command copies the file called "start" that resides in the SFTP server directory called "configs" to the startup configuration.

```
Hostname# copy sftp: startup-config
!!!WARNING!!! This operation will overwrite your startup configuration.
Do you wish to continue[y]? y
Address or name or remote host? admin:messaging@tftp://server/configs
Source filename? start
```

Copy Running Configuration from Hard Disk to Another Location

Starting in module EXEC mode, use the following command to copy the running configuration on the hard disk to another location:

copy running-config { **sftp:** *user-id:password* **@sftp://** *server/dir/filename* | **startup-config** **tftp:tftp://** *server/dir/filename* }

For a description of this command, see *CLI Command Reference for Cisco Contact Center SIP Proxy (CCCSP)*.

The command works in two ways, depending on where you are copying the command:

- If you copy the running configuration to the startup configuration, enter the command on one line, like in the following example:

  ```
  Hostname# copy running-config startup-config
  ```

- If you copy the running configuration to the SFTP or TFTP server, this command becomes interactive and prompts you for the information. You cannot enter the parameters in one line. In the following example, the running configuration is copied to the SFTP server, which requires a username and password. The running configuration is copied to the directory called "configs" as a file called "saved_start".

```
Hostname# copy running-config sftp:
Address or name of remote host? admin:messaging@sftp://server/configs
Source filename? saved_start
```

Copy Running Configuration from Network TFTP Server to Another Location

Starting in module EXEC mode, use the following command to copy the running configuration from the network TFTP server to another location:

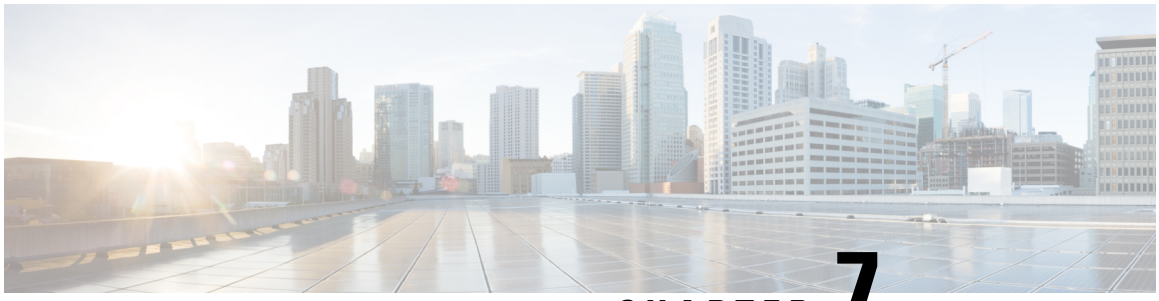
copy t ftp: { **running-config** | **startup-config** } **tftp:** *//server/dir/filename*

Syntax Description

running-config	Active configuration on hard disk.
startup-config	Startup configuration on hard disk.
tftp-server-url	URL of the TFTP server.

This command is interactive and prompts you for the information. You cannot enter the parameters in one line. The following example illustrates this process. In this example, the file called “start” that resides in the directory called “configs” on the TFTP server is copied to the startup configuration.

```
Hostname# copy tftp: startup-config
!!!WARNING!!! This operation will overwrite your startup configuration.
Do you wish to continue[y]? y
Address or name of remote host? tftp://server/configs
Source filename? start
```



CHAPTER 7

Troubleshooting

- Using CLI Commands to Troubleshoot the System, on page 75
- Troubleshooting Configuration Changes, on page 77
- Related Topics, on page 77

Using CLI Commands to Troubleshoot the System

Cisco technical support personnel may request that you run one or more of these commands when troubleshooting a problem. Cisco technical support personnel provides additional information about the commands at that time.

Note Some of these commands may impact performance of your system. We strongly recommend that you do not use these commands unless directed to do so by Cisco Technical Support.

About Logging

You can use log messages to help you debug system problems. Log messages are saved to the messages.log file.

Logging and tracing to the hard disk is turned off by default. Executing the log trace **boot** command starts the log and trace functions immediately.

To check the log and trace files on the hard disk, use the **show logs** command in CCCSP EXEC mode. It displays the list of logs available, their size and their dates of most recent modification.

Each file has a fixed length of 10 MB, and tracing or logging stops automatically when the file reaches this length. New files overwrite the old files.

Note If you cannot view the contents of the log files, copy the log files from CCCSP to an external server and use a text editor, such as **vi**, to display the content.

Log Commands

CCCSP has the following log commands:

- **log console monitor** command
- **log trace boot** command
- **log trace buffer save** command
- **show logs** command

Example of Log Output

The following is an example of the log output:

```
Hostname(exec-mping)# show logs

SIZE            LAST_MODIFIED_TIME                          NAME
      298   Thu Sep 04 22:23:14 IST 2025          gls-registration.log
     2253   Wed Sep 03 20:36:06 IST 2025                      kdump.log
      776   Thu Sep 04 22:23:14 IST 2025                 cisco.amp.log
    33408   Thu Sep 04 22:32:01 IST 2025               dnf.librepo.log
     8352   Thu Sep 04 22:32:01 IST 2025                   dnf.rpm.log
     3116   Wed Sep 03 20:35:10 IST 2025       vmware-vgauthsvc.log.0
   106399   Thu Sep 04 22:32:01 IST 2025                       dnf.log
        0   Wed Sep 03 21:01:02 IST 2025                      boot.log
     5853   Wed Sep 03 20:35:10 IST 2025          vmware-vmsvc-root.log
     1949   Wed Sep 03 20:35:10 IST 2025      vmware-vmtoolsd-root.log
      189   Tue Aug 26 23:29:20 IST 2025          vmware-network.2.log
      187   Wed Sep 03 20:35:10 IST 2025            vmware-network.log
        0   Tue Aug 26 23:39:25 IST 2025                  messages.log
      241   Wed Sep 03 20:34:46 IST 2025          vmware-network.1.log
```

Using Trace Commands

To troubleshoot network configuration in CCCSP, use the **trace enable** command in CCCSP EXEC mode.

CCCSP has the following trace commands:

- **log trace boot** command
- **log trace buffer save** command
- **show trace log** command
- **show trace options** command
- **trace enable** command
- **trace disable** command
- **trace level** command
- **trace logsize** command

Using Show Commands

In addition to the standard show commands, use the following commands to troubleshoot your CCCSP configuration:

- **show status queue**
- **show status server-group radius** [*server-group-name*]
- **show status server-group sip** [*server-group-name*]
- **show status sip**

Troubleshooting Configuration Changes

Problem: You lost some configuration data.

Recommended Action: Copy your changes to the running configuration at frequent intervals. See Copying Configurations.

Problem: You lost configuration data when you rebooted the system.

Explanation: You did not save the data before the reboot.

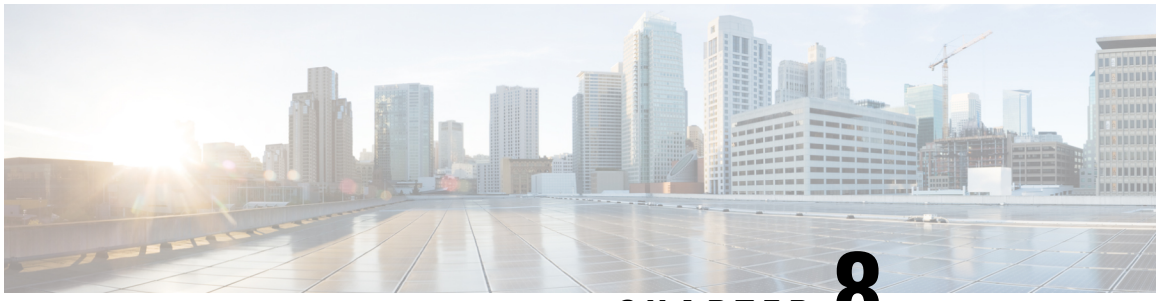
Recommended Action: Use the copy running-config **startup-config** command to copy your changes from the running configuration to the startup configuration. When CCCSP reboots, it reloads the startup configuration. See Copying Configurations.



Note Messages are considered application data and are saved directly to the disk in the startup configuration. (They should be backed up on another server in case of a power outage or a new installation). All other configuration changes require an explicit “save configuration” operation to preserve them in the startup configuration.

Related Topics

- For information about the CLI commands, see *CLI Command Reference for Cisco Contact Center SIP Proxy (CCCSP)*.
- For information about copying configurations, see Copying Configurations.



CHAPTER 8

Configuration Example

• Configuration Example, on page 79

Configuration Example

The following is an example of what you will see after you have configured your CCCSP system and then enter the **show configuration active verbose** command.

```
Hostnaame(cusp-config)# show configuration active verbose
Building CUSP configuration...
!
server-group sip global-load-balance call-id
server-group sip retry-after 0
server-group sip element-retries udp 3
server-group sip element-retries tls 1
server-group sip element-retries tcp 1
sip alias myhostname
sip dns-srv
enable
no naptr
end dns
!
no sip header-compaction
no sip logging
!
sip max-forwards 70
sip network cube-es standard
no non-invite-provisional
allow-connections
retransmit-count invite-server-transaction 9
retransmit-count non-invite-client-transaction 9
retransmit-count invite-client-transaction 5
retransmit-timer clientTn 64000
retransmit-timer serverTn 64000
retransmit-timer T4 5000
retransmit-timer T2 4000
retransmit-timer T1 500
retransmit-timer TU2 32000
retransmit-timer TU1 5000
end network
!
sip network cube-sp standard
no non-invite-provisional
allow-connections
```

```
retransmit-count invite-server-transaction 9
retransmit-count invite-client-transaction 5
retransmit-count non-invite-client-transaction 9
retransmit-timer T4 5000
retransmit-timer T2 4000
retransmit-timer T1 500
retransmit-timer TU2 32000
retransmit-timer TU1 5000
retransmit-timer clientTn 64000
retransmit-timer serverTn 64000
end network
!
sip network enterprise standard
no non-invite-provisional
allow-connections
retransmit-count invite-client-transaction 5
retransmit-count invite-server-transaction 9
retransmit-count non-invite-client-transaction 9
retransmit-timer serverTn 64000
retransmit-timer T4 5000
retransmit-timer T2 4000
retransmit-timer T1 500
retransmit-timer TU2 32000
retransmit-timer TU1 5000
retransmit-timer clientTn 64000
end network
!
sip network service-provider standard
no non-invite-provisional
allow-connections
retransmit-count invite-server-transaction 9
retransmit-count non-invite-client-transaction 9
retransmit-count invite-client-transaction 5
retransmit-timer serverTn 64000
retransmit-timer TU1 5000
retransmit-timer TU2 32000
retransmit-timer T1 500
retransmit-timer T2 4000
retransmit-timer T4 5000
retransmit-timer clientTn 64000
end network
!
sip overload reject retry-after 0
!
no sip peg-counting
!
sip privacy service
sip queue message
drop-policy head
low-threshold 80
size 2000
thread-count 20
end queue
!
sip queue radius
drop-policy head
low-threshold 80
size 2000
thread-count 20
end queue
!
sip queue request
drop-policy head
low-threshold 80
```

```
size 2000
thread-count 20
end queue
!
sip queue response
drop-policy head
low-threshold 80
size 2000
thread-count 20
end queue
!
sip queue st-callback
drop-policy head
low-threshold 80
size 2000
thread-count 10
end queue
!
sip queue timer
drop-policy none
low-threshold 80
size 2500
thread-count 8
end queue
!
sip queue xcl
drop-policy head
low-threshold 80
size 2000
thread-count 2
end queue
!
route recursion
!
sip tcp connection-timeout 240
sip tcp max-connections 256
sip tls
!
trigger condition call-from-cube-es
sequence 1
in-network cube-es
end sequence
end trigger condition
!
trigger condition call-from-cube-sp
sequence 1
in-network cube-sp
end sequence
end trigger condition
!
trigger condition call-from-enterprise
sequence 1
in-network enterprise
end sequence
end trigger condition
!
trigger condition call-from-service-provider
sequence 1
in-network service-provider
end sequence
end trigger condition
!
trigger condition mid-dialog
sequence 1
```

```
mid-dialog
end sequence
end trigger condition
!
accounting
no enable
no client-side
no server-side
end accounting
!
server-group sip group cme.example.com enterprise
element ip-address 192.168.10.6 5060 tls q-value 1.0 weight 0
failover-resp-codes 503
lbtype global
ping
end server-group
!
server-group sip group cube-es.example.com cube-es
element ip-address 192.168.20.4 5060 tls q-value 1.0 weight 0
element ip-address 192.168.20.3 5060 tls q-value 1.0 weight 0
failover-resp-codes 503
lbtype global
ping
end server-group
!
server-group sip group cube-sp.example.com cube-sp
element ip-address 10.10.20.3 5060 tls q-value 1.0 weight 0
element ip-address 10.10.20.4 5060 tls q-value 1.0 weight 0
failover-resp-codes 503
lbtype global
ping
end server-group
!
server-group sip group cucm.example.com enterprise
element ip-address 192.168.10.4 5060 tls q-value 1.0 weight 50
element ip-address 192.168.10.5 5060 tls q-value 1.0 weight 50
element ip-address 192.168.10.3 5060 tls q-value 1.0 weight 100
failover-resp-codes 503
lbtype weight
ping
end server-group
!
server-group sip group sp.example.com service-provider
element ip-address 10.10.10.3 5060 udp q-value 1.0 weight 0
failover-resp-codes 503
lbtype global
ping
end server-group
!
route table cube-es-table
key * response 404
key 5101 target-destination cme.example.com enterprise
key 510 target-destination cucm.example.com enterprise
end route table
!
route table cube-sp-table
key * target-destination sp.example.com service-provider
end route table
!
route table enterprise-table
key * response 404
key 5101 target-destination cme.example.com enterprise
key 91 target-destination cube-es.example.com cube-es
key 510 target-destination cucm.example.com enterprise
```

```
end route table
!
route table service-provider-table
key * response 404
key 510 target-destination cube-sp.example.com cube-sp
end route table
!
policy normalization outgoing-norm-policy
uri-component update TO all user ^91 ""
uri-component update request-uri user ^91 ""
end policy
!
policy lookup cube-es-policy
sequence 1 cube-es-table request-uri uri-component user
rule prefix
end sequence
end policy
!
policy lookup cube-sp-policy
sequence 1 cube-sp-table request-uri uri-component user
rule prefix
end sequence
end policy
!
policy lookup enterprise-policy
sequence 1 enterprise-table request-uri uri-component user
rule prefix
end sequence
end policy
!
policy lookup service-provider-policy
sequence 1 service-provider-table request-uri uri-component user
rule prefix
end sequence
end policy
!
trigger routing sequence 5 policy enterprise-policy condition call-from-enterpri
se
trigger routing sequence 4 policy cube-es-policy condition call-from-cube-es
trigger routing sequence 3 policy cube-sp-policy condition call-from-cube-sp
trigger routing sequence 2 policy service-provider-policy condition call-from-service-provider
trigger routing sequence 1 by-pass condition mid-dialog
trigger pre-normalization sequence 2 policy outgoing-norm-policy condition call-from-cube-sp
trigger pre-normalization sequence 1 by-pass condition mid-dialog
!
no server-group sip global-ping
!
sip listen service-provider udp 10.10.10.99 5060
sip listen cube-sp tls 10.10.20.99 5060
sip listen cube-es tls 192.168.20.99 5060
sip listen enterprise tls 192.168.10.99 5060
!
sip record-route cube-es tls 192.168.20.99 5060
sip record-route service-provider udp 10.10.10.99 5060
sip record-route cube-sp tls 10.10.20.99 5060
sip record-route enterprise tls 192.168.10.99 5060
!
end
Hostname(cusp-config)#
```

Configuration Example