Cisco Unified Border Element Protocol-Independent Features and Setup Configuration Guide, Cisco IOS Release 15M&T

Last Modified: November 25, 2015

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

CONTENTS

CHAPTER 1

Cisco Unified Border Element Protocol-Independent Features and Setup

This Cisco Unified Border Element is a special Cisco IOS software image it provides a network-to-network interface point for billing, security, call admission control, quality of service, and signaling interworking. This chapter describes basic gateway functionality, software images, topology, and summarizes supported features.

Note Cisco Product Authorization Key (PAK)--A Product Authorization Key (PAK) is required to configure some of the features described in this guide. Before you start the configuration process, please register your products and activate your PAK at the following URL http://www.cisco.com/go/license .

Finding Feature Information

For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the "Cisco Unified Border Element Features Roadmap".

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Cisco Unified Border Element Protocol-Independent Features and Setup

This chapter contains the following configuration topics:

Cisco UBE Prerequisites and Restrictions

- Prerequisites for Cisco Unified Border Element
- Restrictions for Cisco Unified Border Element

Dial Plan Management

- Dial Peer Configuration on Voice Gateway Routers

http://www.cisco.com/en/US/docs/ios/12_3/vvf_c/dial_peer/dpeer_c.html

- Translation Rules

http://www.cisco.com/en/US/docs/ios/voice/command/reference/vr_t3.html#wp1651612

- ENUM Support
- Configuring Tool Command Language (Tcl)

http://www.cisco.com/en/US/products/sw/voicesw/ps2192/products_programming_reference_guides_list.html

- Cisco Service Advertisement Framework (SAF)

http://www.cisco.com/en/US/prod/collateral/iosswrel/ps8802/ps10587/ps10591/ps10621/product_bulletin_c25-561938.html#wp9000293

Configuring Call Admission Control (CAC)

- VoIP Call Admissions Control

http://www.cisco.com/en/US/docs/ios/solutions_docs/voip_solutions/CAC.html

- VoIP Call Admission Control Using RSVP

http://www.cisco.com/en/US/docs/ios/12_1t/12_1t5/feature/guide/dt4trsvp.html

RSVP

- Configuring RSVP Agent
- Interworking Between RSVP Capable and RSVP Incapable Networks

Dual-Tone Multifrequency (DTMF) Support and Interworking

- SIP--INFO Method for DTMF Tone Generation
- DTMF Events through SIP Signaling
- Configuring SIP DTMF Features

http://www.cisco.com/en/US/docs/ios/12_3/sip/configuration/guide/chapter8.html

- H.323 RFC2833 - SIP NOTIFY

http://www.cisco.com/en/US/docs/ios/voice/sip/configuration/guide/sip_cg-dtmf_ps6441_TSD_Products_Configuration_Guide_Chapter.html#wp1062375

Codec Negotiation

- Support for Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element

Payload Type Interoperability

- Dynamic payload type interworking for DTMF and codec packets for SIP-to-SIP calls

Transcoding

- iLBC Support for SIP and H.323
- Universal Transcoding

Fax/modem Support

- Modem Passthrough
- T.38 Fax Relay

http://www.cisco.com/en/US/docs/ios/12_3/vvf_c/cisco_ios_fax_services_over_ip_application_guide/t38.html

- Cisco Fax Relay

http://www.cisco.com/en/US/docs/ios/12_3/vvf_c/cisco_ios_fax_services_over_ip_application_guide/cisrly.html

SIP Video

- SIP Video Calls with Flow Around Media
- RTP Media Loopback for SIP Calls
- Configuring RTP Media Loopback for SIP Calls

Telepresence

- SIP Video Support for Telepresence Calls

Security Features

- Toll Fraud Prevention

http://www.cisco.com/en/US/docs/ios/ios_xe/voice_cube_-_ent/configuration/guide/vb_ch2_xe.html

- Access lists (ACLs)

http://www.cisco.com/en/US/products/sw/voicesw/ps4625/products_tech_note09186a00809dc487.shtml?

- CAC (call spike)

http://www.cisco.com/en/US/docs/ios/voice/command/reference/vr_c3.html#wp1210005?

- SIP--Ability to Send a SIP Registration Message on a Border Element

- SIP Parameter Modification
- SIP--SIP Stack Portability
- Session Refresh with Reinvites
- CDR

http://www.cisco.com/en/US/docs/ios/voice/cube/configuration/guide/vb-gw-overview_ps5640_TSD_Products_Configuration_Guide_Chapter.html#wp1166707

- Transport Layer Security (TLS)
- Interworking of Secure RTP calls for SIP and H.323
- SIP SRTP Fallback to Nonsecure RTP
- Cisco Unified Communications Trusted Firewall

IPv4 and IPv6 Interworking

- VoIP for IPv6
  - IPv4 to IPv6 Calls (SIP and SIP)
  - IPv6 to IPv6 Calls (SIP and SIP)
  - Support for Dual Stack ANAT

RSVP Interworking

- Support for Interworking Between RSVP Capable and RSVP Incapable Networks

Collocated Services

- Media Termination Point (MTP)
- Cisco Unified SIP Survivable Remote Site Telephony (SRST)
- Cisco IOS Tcl IVR and VoiceXML Application Guide
- Cisco VoiceXML Programmer's Guide
- Cisco Unified Communications Trusted Firewall
- Cisco Unified Border Element with Gatekeeper

http://www.cisco.com/en/US/docs/ios/voice/cubegk/configuration/guide/ve_book/ve_book.html

Toll Fraud Prevention

When a Cisco router platform is installed with a voice-capable Cisco IOS software image, appropriate features must be enabled on the platform to prevent potential toll fraud exploitation by unauthorized users. Deploy these features on all Cisco router Unified Communications applications that process voice calls, such as Cisco Unified Communications Manager Express (CME), Cisco Survivable Remote Site Telephony (SRST), Cisco Unified Border Element (UBE), Cisco IOS-based router and standalone analog and digital PBX and public-switched telephone network (PSTN) gateways, and Cisco contact-center VoiceXML gateways. These features include, but are not limited to, the following:

- Disable secondary dial tone on voice ports--By default, secondary dial tone is presented on voice ports on Cisco router gateways. Use private line automatic ringdown (PLAR) for foreign exchange office (FXO) ports and direct-inward-dial (DID) for T1/E1 ports to prevent secondary dial tone from being presented to inbound callers.
- Cisco router access control lists (ACLs)--Define ACLs to allow only explicitly valid sources of calls to the router or gateway, and therefore to prevent unauthorized Session Initiation Protocol (SIP) or H.323 calls from unknown parties to be processed and connected by the router or gateway.
- Close unused SIP and H.323 ports--If either the SIP or H.323 protocol is not used in your deployment, close the associated protocol ports. If a Cisco voice gateway has dial peers configured to route calls outbound to the PSTN using either time division multiplex (TDM) trunks or IP, close the unused H.323 or SIP ports so that calls from unauthorized endpoints cannot connect calls. If the protocols are used and the ports must remain open, use ACLs to limit access to legitimate sources.
- Change SIP port 5060--If SIP is actively used, consider changing the port to something other than well-known port 5060.
- SIP registration--If SIP registration is available on SIP trunks, turn on this feature because it provides an extra level of authentication and validation that only legitimate sources can connect calls. If it is not available, ensure that the appropriate ACLs are in place.
- SIP Digest Authentication--If the SIP Digest Authentication feature is available for either registrations or invites, turn this feature on because it provides an extra level of authentication and validation that only legitimate sources can connect calls.
- Explicit incoming and outgoing dial peers--Use explicit dial peers to control the types and parameters of calls allowed by the router, especially in IP-to-IP connections used on CME, SRST, and Cisco UBE. Incoming dial peers offer additional control on the sources of calls, and outgoing dial peers on the destinations. Incoming dial peers are always used for calls. If a dial peer is not explicitly defined, the implicit dial peer 0 is used to allow all calls.
- Explicit destination patterns--Use dial peers with more granularity than.T for destination patterns to block disallowed off-net call destinations. Use class of restriction (COR) on dial peers with specific destination patterns to allow even more granular control of calls to different destinations on the PSTN.
- Translation rules--Use translation rules to manipulate dialed digits before calls connect to the PSTN to provide better control over who may dial PSTN destinations. Legitimate users dial an access code and an augmented number for PSTN for certain PSTN (for example, international) locations.
- Tcl and VoiceXML scripts--Attach a Tcl/VoiceXML script to dial peers to do database lookups or additional off-router authorization checks to allow or deny call flows based on origination or destination numbers. Tcl/VoiceXML scripts can also be used to add a prefix to inbound DID calls. If the prefix plus DID matches internal extensions, then the call is completed. Otherwise, a prompt can be played to the caller that an invalid number has been dialed.
- Host name validation--Use the "permit hostname" feature to validate initial SIP Invites that contain a fully qualified domain name (FQDN) host name in the Request Uniform Resource Identifier (Request URI) against a configured list of legitimate source hostnames.
- Dynamic Domain Name Service (DNS)--If you are using DNS as the "session target" on dial peers, the actual IP address destination of call connections can vary from one call to the next. Use voice source groups and ACLs to restrict the valid address ranges expected in DNS responses (which are used subsequently for call setup destinations).

For more configuration guidance, see the " Cisco IOS Unified Communications Toll Fraud Prevention " paper.

CHAPTER 2

Interworking Between RSVP Capable and RSVP Incapable Networks

The Interworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based Resource Reservation Protocol (RSVP) support for basic audio call and supplementary services on Cisco Unified Border Element (UBE). This feature improves the interoperability between RSVP and non-RSVP networks. RSVP functionality added to Cisco UBE helps you to reserve the required bandwidth before making a call.

This feature extends RSVP support to delayed-offer to delayed-offer and delayed-offer to early-offer calls, along with the early-offer to early-offer calls.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Interworking Between RSVP Capable and RSVP Incapable Networks

- RSVP policies allow you to configure separate bandwidth pools with varying limits so that any one application, such as video, can consume all the RSVP bandwidth on a specified interface at the expense of other applications, such as voice, which would be dropped.
- To limit bandwidth per application, you must configure a bandwidth limit before configuring Support for the Interworking Between RSVP Capable and RSVP Incapable Networks feature. See the Configuring RSVP on an Interface, on page 8.

Cisco Unified Border Element

- Cisco IOS Release 15.0(1)XA or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.1S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for Interworking Between RSVP Capable and RSVP Incapable Networks

The Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature has the following restrictions:

- Segmented RSVP is not supported.
- Interoperability between Cisco UBE and Cisco Unified Communications Manager is not available.
- RSVP-enabled video calls are not supported.

How to Configure Interworking Between RSVP Capable and RSVP Incapable Networks

Configuring RSVP on an Interface

You must allocate some bandwidth for the interface before enabling RSVP. Perform this task to configure RSVP on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / port*
4. **ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / port* **Example:** `Device(config)# interface FastEthernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]] **Example:** `Device(config-if)# ip rsvp bandwidth 10000 100000`	Enables RSVP for IP on an interface.
Step 5	**end** **Example:** `Device(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Optional RSVP on the Dial Peer

Perform this task to configure optional RSVP at the dial peer level. This configuration allows you to have uninterrupted call even if there is a failure in bandwidth reservation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**]
5. **req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer 77 voip`	Enters dial peer voice configuration mode.

Step 4	**no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**]
	
	Example:
	`Device(config-dial-peer)# no acc-qos controlled-load`
		Removes any value configured for the **acc-qos** command.
		• Keywords are as follows:
		  • **controlled-load**--Indicates that RSVP guarantees a single level of preferential service, presumed to correlate to a delay boundary. The controlled load service uses admission (or capacity) control to ensure that preferential service is received even when the bandwidth is overloaded.
		  • **guaranteed-delay**--Indicates that RSVP reserves bandwidth and guarantees a minimum bit rate and preferential queueing if the bandwidth reserved is not exceeded.

Step 5	**req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]	Configures the desired quality of service (QoS) to be used. • Calls continue even if there is a failure in bandwidth reservation.

	Command or Action	Purpose
	Example: `Device(config-dial-peer)# req-qos controlled-load`	**Note** Configure the **req-qos** commandusing the same keyword that you used to configure the **acc-qos** command, either **controlled-load** or **guaranteed-delay**. That is, if you configured **acc-qos controlled-load** command in the previous step, then use the **req-qos controlled-load** command here.
Step 6	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring Mandatory RSVP on the Dial Peer

Perform this task to configure Mandatory RSVP on the dial peer. This configuration ensures that the call does not connect if sufficient bandwidth is not allocated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **acc-qos** {**best-effort** | **controlled-load** | **guaranteed-delay**} [**audio** | **video**]
5. **req-qos** {**best-effort** [**audio** | **video**] | {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer 77 voip`	Enters dial peer voice configuration mode.
Step 4	**acc-qos** {**best-effort** | **controlled-load** | **guaranteed-delay**} [**audio** | **video**] **Example:** `Device(config-dial-peer)# acc-qos best-effort`	Configures mandatory RSVP on the dial-peer. • Keywords are as follows: • **best-effort**--Indicates that Resource Reservation Protocol (RSVP) makes no bandwidth reservation. This is the default. • **controlled-load**--Indicates that RSVP guarantees a single level of preferential service, presumed to correlate to a delay boundary. The controlled load service uses admission (or capacity) control to ensure that preferential service is received even when the bandwidth is overloaded. • **guaranteed-delay**--Indicates that RSVP reserves bandwidth and guarantees a minimum bit rate and preferential queueing if the bandwidth reserved is not exceeded.
Step 5	**req-qos** {**best-effort** [**audio** | **video**] | {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]} **Example:** `Device(config-dial-peer)# req-qos controlled-load`	Configures mandatory RSVP on the dial-peer. • Calls continue even if there is a drop in the bandwidth reservation.
Step 6	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring Midcall RSVP Failure Policies

Perform this task to enable call handling policies for a midcall RSVP failure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **voice-class sip rsvp-fail-policy** {**video** | **voice**} **post-alert** {**optional keep-alive** | **mandatory** {**keep-alive** | **disconnect retry** *retry-attempts*}} **interval** *seconds*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.

Step 4	**voice-class sip rsvp-fail-policy** {**video** | **voice**} **post-alert** {**optional keep-alive** | **mandatory** {**keep-alive** | **disconnect retry** *retry-attempts*}} **interval** *seconds*
	
	Example:
	
	`Device(config-dial-peer)# voice-class sip rsvp-fail-policy voice post-alert mandatory keep-alive interval 50`
		Enables call handling policies for a midcall RSVP failure.
		
		• Keywords are as follows:
		  • **optional keep-alive**--The keepalive messages are sent when RSVP fails only if RSVP negotiation is optional.
		  • **mandatory keep-alive**--The keepalive messages are sent when RSVP fails only if RSVP negotiation is mandatory.
		
		Note Keepalive messages are sent at 30-second intervals when a postalert call fails to negotiate RSVP regardless of the RSVP negotiation setting (mandatory or optional).

Step 5	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring DSCP Values

Perform this task to configure different Differentiated Services Code Point (DSCP) values based on RSVP status.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos dscp** {*dscp-value* | *set-af* | *set-cs* | **default** | **ef**} {**signaling** | **media** [**rsvp-pass** | **rsvp-fail**] | **video**[**rsvp-none**| **rsvp-pass** | **rsvp-fail**]}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.

Step 4	**ip qos dscp** {*dscp-value* | *set-af* | *set-cs* | **default** | **ef**} {**signaling** | **media** [**rsvp-pass** | **rsvp-fail**] | **video**[**rsvp-none**| **rsvp-pass** | **rsvp-fail**]}
	
	Example:
	`Device(config-dial-peer)# ip qos dscp af11 media rsvp-pass`
		Configures DSCP values based on RSVP status.
		• Keywords are as follows:
		  • **media rsvp-pass**--Specifies that the DSCP value applies to media packets with successful RSVP reservations.
		  • **media rsvp-fail**--Specifies that the DSCP value applies to packets (media or video) with failed RSVP reservations.

	Command or Action	Purpose
		• The default DSCP value for all media (voice and fax) packets is **ef**. **Note** You must configure the DSCP values for all cases: **media rsvp-pass** and **media rsvp-fail**.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring an Application ID

Perform this task to configure a specific application ID for RSVP establishment.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos policy-locator** {**video** | **voice**} [**app** *app-string*] [**guid** *guid-string*] [**sapp** *subapp-string*] [**ver** *version-string*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**ip qos policy-locator** {**video** | **voice**} [**app** *app-string*] [**guid** *guid-string*] [**sapp** *subapp-string*] [**ver** *version-string*] **Example:** `Device(config-dial-peer)# ip qos policy-locator voice`	Configures a QoS policylocator (application ID) used to deploy RSVP policies for specifying bandwidth reservations on Cisco IOS Session Initiation Protocol (SIP) devices.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring Priority

Perform this task to configure priorities for call preemption.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos defending-priority** *defending-pri-value*
5. **ip qos preemption-priority** *preemption-pri-value*
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**ip qos defending-priority** *defending-pri-value* **Example:** `Device(config-dial-peer)# ip qos defending-priority 66`	Configures the RSVP defending priority value for determining QoS.
Step 5	**ip qos preemption-priority** *preemption-pri-value* **Example:** `Device(config-dial-peer)# ip qos preemption-priority 75`	Configures the RSVP preemption priority value for determining QoS.
Step 6	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer configuration mode and returns to privileged EXEC mode.

Troubleshooting for Interworking Between RSVP Capable and RSVP Incapable Networks Feature

Use the following commands to debug any errors that you may encounter when you configure the Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature.

- **debug call rsvp-sync events**
- **debug call rsvp-sync func-trace**
- **debug ccsip all**
- **debug ccsip messages**
- **debug ip rsvp messages**
- **debug sccp all**

Verifying Interworking Between RSVP Capable and RSVP Incapable Networks

This task explains how to display information to verify the configuration for the Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature. These commands need not be entered in any specific order.

SUMMARY STEPS

1. **enable**
2. **show sip-ua calls**
3. **show ip rsvp installed**
4. **show ip rsvp reservation**
5. **show ip rsvp interface detail** [*interface-type number*]
6. **show sccp connections details**
7. **show sccp connections rsvp**
8. **show sccp connections internal**
9. **show sccp** [**all** | **connections** | **statistics**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show sip-ua calls** **Example:** `Device# show sip-ua calls`	(Optional) Displays active user agent client (UAC) and user agent server (UAS) information on SIP calls.
Step 3	**show ip rsvp installed** **Example:** `Device# show ip rsvp installed`	(Optional) Displays RSVP-related installed filters and corresponding bandwidth information.
Step 4	**show ip rsvp reservation** **Example:** `Device# show ip rsvp reservation`	(Optional) Displays RSVP-related receiver information currently in the database.

	Command or Action	Purpose
Step 5	**show ip rsvp interface detail** [*interface-type number*] **Example:** `Device# show ip rsvp interface detail GigabitEthernet 0/0`	(Optional) Displays the interface configuration for hello.
Step 6	**show sccp connections details** **Example:** `Device# show sccp connections details`	(Optional) Displays SCCP connection details, such as call-leg details.
Step 7	**show sccp connections rsvp** **Example:** `Device# show sccp connections rsvp`	(Optional) Displays information about active SCCP connections that are using RSVP.
Step 8	**show sccp connections internal** **Example:** `Device# show sccp connections internal`	(Optional) Displays the internal SCCP details, such as time-stamp values.
Step 9	**show sccp** [**all** | **connections** | **statistics**] **Example:** `Device# show sccp statistics`	(Optional) Displays SCCP information, such as administrative and operational status.

Feature Information for Interworking Between RSVP Capable and RSVP Incapable Networks

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.
Feature History Table entry for the Cisco Unified Border Element.

Table 1: Feature Information for Interworking Between RSVP Capable and RSVP Incapable Network

Feature Name	Releases	Feature Information
Interworking Between RSVP Capable and RSVP Incapable Networks	15.0(1)XA 15.1(1)T	The Interworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based RSVP support for basic audio call and supplementary services on the Cisco UBE. The following commands were introduced or modified: **acc-qos**, **ip qos defending-priority**, **ip qos dscp**, **ip qos policy-locator**, **ip qos preemption-priority, req-qos**, **voice-class sip rsvp-fail-policy**,
Interworking Between RSVP Capable and RSVP Incapable Networks	Cisco IOS XE Release 3.1S	The nterworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based RSVP support for basic audio call and supplementary services on the Cisco UBE. The following commands were introduced or modified: **acc-qos**, **ip qos defending-priority**, **ip qos dscp**, **ip qos policy-locator**, **ip qos preemption-priority**, **req-qos**, **voice-class sip rsvp-fail-policy**,

CHAPTER 3

SIP INFO Method for DTMF Tone Generation

The SIP: INFO Method for DTMF Tone Generation feature uses the Session Initiation Protocol (SIP) INFO method to generate dual tone multifrequency (DTMF) tones on the telephony call leg. SIP info methods, or request message types, request a specific action be taken by another user agent (UA) or proxy server. The SIP INFO message is sent along the signaling path of the call. Upon receipt of a SIP INFO message with DTMF relay content, the gateway generates the specified DTMF tone on the telephony end of the call.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for SIP INFO Method for DTMF Tone Generation

You cannot configure, enable, or disable this feature. No configuration tasks are required to configure the SIP - INFO Method for DTMF Tone Generation feature. The feature is enabled by default.

Cisco Unified Border Element

- Cisco IOS Release 12.2(11)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for SIP INFO Methods for DTMF Tone Generation

The SIP: INFO Method for DTMF Tone Generation feature includes the following signal duration parameters:

- Minimum signal duration is 100 milliseconds (ms). If a request is received with a duration less than 100 ms, the minimum duration of 100 ms is used by default.
- Maximum signal duration is 5000 ms. If a request is received with a duration longer than 5000 ms, the maximum duration of 5000 ms is used by default.
- If no duration parameter is included in a request, the gateway defaults to a signal duration of 250 ms.

Information About SIP INFO Method for DTMF Tone Generation

The SIP: INFO Method for DTMF Tone Generation feature is always enabled, and is invoked when a SIP INFO message is received with DTMF relay content. This feature is related to the DTMF Events Through SIP Signaling feature, which allows an application to be notified about DTMF events using SIP NOTIFY messages. Together, the two features provide a mechanism to both send and receive DTMF digits along the signaling path. For more information on sending DTMF event notification using SIP NOTIFY messages, refer to the DTMF Events Through SIP Signaling feature.

How to Review SIP INFO Messages

The SIP INFO method is used by a UA to send call signaling information to another UA with which it has an established media session. The following example shows a SIP INFO message with DTMF content:

```
INFO sip:2143302100@172.17.2.33 SIP/2.0
Via: SIP/2.0/UDP 172.80.2.100:5060
From:  <sip:9724401003@172.80.2.100>;tag=43
To:  <sip:2143302100@172.17.2.33>;tag=9753.0207
Call-ID: 984072_15401962@172.80.2.100
CSeq: 25634 INFO
Supported: 100rel
Supported: timer
Content-Length: 26
Content-Type: application/dtmf-relay
Signal= 1
Duration= 160
```

This sample message shows a SIP INFO message received by the gateway with specifics about the DTMF tone to be generated. The combination of the "From", "To", and "Call-ID" headers identifies the call leg. The signal and duration headers specify the digit, in this case 1, and duration, 160 milliseconds in the example, for DTMF tone play.

Configuring for SIP INFO Method for DTMF Tone Generation

You cannot configure, enable, or disable this feature. No configuration tasks are required to configure the SIP - INFO Method for DTMF Tone Generation feature. The feature is enabled by default.

Troubleshooting Tips

You can display SIP statistics, including SIP INFO method statistics, by using the **show sip-ua statistics** and **show sip-ua status** commands in privileged EXEC mode. See the following fields for SIP INFO method statistics:

- OkInfo 0/0, under SIP Response Statistics, Success, displays the number of successful responses to an INFO request.
- Info 0/0, under SIP Total Traffic Statistics, displays the number of INFO messages received and sent by the gateway.

The following is sample output from the **show sip-ua statistics** command:

```
Device# show sip-ua statistics
SIP Response Statistics (Inbound/Outbound)
Informational:
Trying 1/1, Ringing 0/0,
Forwarded 0/0, Queued 0/0,
SessionProgress 0/1
Success:
OkInvite 0/1, OkBye 1/0,
OkCancel 0/0, OkOptions 0/0,
OkPrack 0/0, OkPreconditionMet 0/0
OkSubscibe 0/0, OkNotify 0/0,
OkInfo 0/0, 202Accepted 0/0
Redirection (Inbound only):
MultipleChoice 0, MovedPermanently 0,
MovedTemporarily 0, SeeOther 0,
UseProxy 0, AlternateService 0
Client Error:
BadRequest 0/0, Unauthorized 0/0,
PaymentRequired 0/0, Forbidden 0/0,
NotFound 0/0, MethodNotAllowed 0/0,
NotAcceptable 0/0, ProxyAuthReqd 0/0,
ReqTimeout 0/0, Conflict 0/0, Gone 0/0,
LengthRequired 0/0, ReqEntityTooLarge 0/0,
ReqURITooLarge 0/0, UnsupportedMediaType 0/0,
BadExtension 0/0, TempNotAvailable 0/0,
CallLegNonExistent 0/0, LoopDetected 0/0,
TooManyHops 0/0, AddrIncomplete 0/0,
Ambiguous 0/0, BusyHere 0/0,
BadEvent 0/0
Server Error:
InternalError 0/0, NotImplemented 0/0,
BadGateway 0/0, ServiceUnavail 0/0,
GatewayTimeout 0/0, BadSipVer 0/0
Global Failure:
BusyEverywhere 0/0, Decline 0/0,
NotExistAnywhere 0/0, NotAcceptable 0/0
SIP Total Traffic Statistics (Inbound/Outbound)
    Invite 0/0, Ack 0/0, Bye 0/0,
    Cancel 0/0, Options 0/0,
```

```
    Prack 0/0, Comet 0/0,
    Subscribe 0/0, Notify 0/0,
    Refer 0/0, Info 0/0
Retry Statistics
Invite 0, Bye 0, Cancel 0, Response 0, Notify 0
```

The following is sample output from the **show sip-ua status**command:

```
Device# show sip-ua status
SIP User Agent Status
SIP User Agent for UDP : ENABLED
SIP User Agent for TCP : ENABLED
SIP User Agent bind status(signaling): DISABLED
SIP User Agent bind status(media): DISABLED
SIP max-forwards : 6
SIP DNS SRV version: 2 (rfc 2782)
SDP application configuration:
 Version line (v=) required
 Owner line (o=) required
 Session name line (s=) required
 Timespec line (t=) required
 Media supported: audio image
 Network types supported: IN
 Address types supported: IP4
 Transport types supported: RTP/AVP udptl
```

Feature Information for SIP INFO Method for DTMF Tone Generation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 2: Feature Information for SIP: INFO Method for DTMF Tone Generation

Feature Name	Releases	Feature Information
SIP: INFO Method for DTMF Tone Generation	12.2(11)T 12.3(2)T 12.2(8)YN 12.2(11)YV 12.2(11)T 12.2(15)T	The SIP: INFO Method for DTMF Tone Generation feature uses the Session Initiation Protocol (SIP) INFO method to generate dual-tone multifrequency (DTMF) tones on the telephony call leg. SIP methods, or request message types, request a specific action be taken by another user agent (UA) or proxy server. The SIP INFO message is sent along the signaling path of the call. The following command was introduced: **show sip-ua**.

Feature Name	Releases	Feature Information
SIP: INFO Method for DTMF Tone Generation	Cisco IOS XE Release 2.5S	The SIP: INFO Method for DTMF Tone Generation feature uses the Session Initiation Protocol (SIP) INFO method to generate dual-tone multifrequency (DTMF) tones on the telephony call leg. SIP methods, or request message types, request a specific action be taken by another user agent (UA) or proxy server. The SIP INFO message is sent along the signaling path of the call. The following command was introduced: **show sip-ua**.

CHAPTER 4

WebEx Telepresence Media Support Over Single SIP Session

The WebEx Telepresence Media Support over Single SIP Session feature provides support for end-to-end negotiation of up to 6 m-lines or media lines over a single Session Initiation Protocol (SIP) session. The media types can be audio, video, or application.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for WebEx Telepresence Media Support Over Single SIP Session

- High availability is not supported with multiple m-lines.
- Only single dynamic payload type in the m-line for H.224 protocol is supported.
- Payload type interworking for Aggregation Service Routers (ASR) is not supported, so dynamic payload type is negotiated end-to-end.

Information About WebEx Telepresence Media Support Over Single SIP Session

The WebEx Telepresence Media Support over Single SIP Session feature provides the following support:

- End-to-end negotiation of multiple m-lines.
- Negotiation of Binary Floor Control Protocol (BFCP), IX, and H.224 protocol m-lines (m=application) and creation of Real-time Transport Protocol (RTP) or UDP streams for the same.
- Early-Offer (EO-EO) and Delayed-Offer (DO-DO) calls' support by the Cisco Unified Border Element (Cisco UBE) with multiple m-lines.
- End-to-end negotiation of multiple m-lines of same media type for video and application (but not audio).
- Mid-call escalation and de-escalation for multiple application and video m-lines.
- Secure RTP (SRTP) passthrough for all RTP streams (audio, video, and application).
- SRTP-RTP interworking for video (ASR only).
- Multiple dynamic payload types in the same m-line for the H.264 codec.

You can use the **show voip rtp connections** and **show call active video compact** commands to see the details about additional video and application streams.

Monitoring WebEx Telepresence Media Support Over Single SIP Session

Perform this task to see the details about additional video and application streams. The **show** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show call active video compact**
3. **show voip rtp connections**
4. **show sip-ua calls**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show call active video compact**

Displays a compact version of call information for Skinny Call Control Protocol (SCCP), SIP, and H.323 video calls in progress. The codec type, negotiated codec, and remote media ports are displayed.

Example:

```
Device# show call active video compact

<callID>  A/O FAX T<sec> Codec       type        Peer Address       IP R<ip>:<udp>
Total call-legs: 2
         1 ANS     T5     H264        VOIP-VIDEO  P332211       9.45.38.39:2448
         6 ORG     T5     H264        VOIP-VIDEO  P1111         9.45.38.39:2438
```

Step 3 **show voip rtp connections**

Displays RTP named event packets. In the following sample output, two RTP connections are displayed for each m-line and a total of 10 RTP connections are displayed for 5 m-lines.

Example:

```
Device# show voip rtp connections

VoIP RTP active connections :
No. CallId     dstCallId  LocalRTP  RmtRTP   LocalIP                     RemoteIP
1     1          6          16384    54024  192.0.2.123                 192.0.2.39
2     2          7          16386    2448   192.0.2.123                 192.0.2.39
3     3          8          16400    5070   192.0.2.123                 192.0.2.39
4     4          9          16388    2450   192.0.2.123                 192.0.2.39
5     5          10         16402    2452   192.0.2.123                 192.0.2.39
6     6          1          16390    58121  192.0.2.123                 192.0.2.39
7     7          2          16392    2438   192.0.2.123                 192.0.2.39
8     8          3          16394    5070   192.0.2.123                 192.0.2.39
9     9          4          16396    2440   192.0.2.123                 192.0.2.39
10    10         5          16398    2442   192.0.2.123                 192.0.2.39
Found 10 active RTP connections
```

Step 4 **show sip-ua calls**

Displays active user agent client (UAC) and user agent server (UAS) information on Session Initiation Protocol (SIP) calls.

Example:

```
Device# show sip-ua calls

Total SIP call legs:2, User Agent Client:1, User Agent Server:1
SIP UAC CALL INFO
Call 1
SIP Call ID                : 72B6C784-753E11E2-FFFFFFFF8008B555-FFFFFFFFE340699E@9.45.47.123
   State of the call       : STATE_ACTIVE (7)
   Substate of the call    : SUBSTATE_NONE (0)
   Calling Number          : 332211
   Called Number           : 1111
   Bit Flags               : 0xC04018 0x10000100 0x80
   CC Call ID              : 6
   Source IP Address (Sig ): 9.45.47.123
   Destn SIP Req Addr:Port : [9.45.38.39]:5267
   Destn SIP Resp Addr:Port: [9.45.38.39]:5267
   Destination Name        : 9.45.38.39
   Number of Media Streams : 5
   Number of Active Streams: 5
   RTP Fork Object         : 0x0
   Media Mode              : flow-through
Media Stream 1
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 6
     Stream Type              : voice-only (0)
```

```
        Stream Media Addr Type   : 1
        Negotiated Codec         : g711ulaw (160 bytes)
        Codec Payload Type       : 0
        Negotiated Dtmf-relay    : inband-voice
        Dtmf-relay Payload Type  : 0
        QoS ID                   : -1
        Local QoS Strength       : BestEffort
        Negotiated QoS Strength  : BestEffort
        Negotiated QoS Direction : NoneLocal QoS Status          : None
        Media Source IP Addr:Port: [9.45.47.123]:16390
        Media Dest IP Addr:Port  : [9.45.38.39]:58121
   Media Stream 2
        State of the stream      : STREAM_ACTIVE
        Stream Call ID           : 7
        Stream Type              : video (7)
        Stream Media Addr Type   : 1
        Negotiated Codec         : h263 (0 bytes)
        Codec Payload Type       : 97
        Negotiated Dtmf-relay    : inband-voice
        Dtmf-relay Payload Type  : 0
        QoS ID                   : -1
        Local QoS Strength       : BestEffort
        Negotiated QoS Strength  : BestEffort
        Negotiated QoS Direction : None
        Local QoS Status         : None
        Media Source IP Addr:Port: [9.45.47.123]:16392
        Media Dest IP Addr:Port  : [9.45.38.39]:2438
   Media Stream 3
        State of the stream      : STREAM_ACTIVE
        Stream Call ID           : 8
        Stream Type              : application (8)
        Stream Media Addr Type   : 1
        Negotiated Codec         : No Codec    (0 bytes)
              Codec Payload Type       : 255 (None)
        Negotiated Dtmf-relay    : inband-voice
        Dtmf-relay Payload Type  : 0
        QoS ID                   : -1
        Local QoS Strength       : BestEffort
        Negotiated QoS Strength  : BestEffort
        Negotiated QoS Direction : None
        Local QoS Status         : None
        Media Source IP Addr:Port: [9.45.47.123]:16394
        Media Dest IP Addr:Port  : [9.45.38.39]:5070
   Media Stream 4
        State of the stream      : STREAM_ACTIVE
        Stream Call ID           : 9
        Stream Type              : video (7)
        Stream Media Addr Type   : 1
        Negotiated Codec         : h263 (0 bytes)
        Codec Payload Type       : 97
        Negotiated Dtmf-relay    : inband-voice
        Dtmf-relay Payload Type  : 0
        QoS ID                   : -1
        Local QoS Strength       : BestEffort
        Negotiated QoS Strength  : BestEffort
        Negotiated QoS Direction : None
        Local QoS Status         : None
        Media Source IP Addr:Port: [9.45.47.123]:16396
        Media Dest IP Addr:Port  : [9.45.38.39]:2440
   Media Stream 5
        State of the stream      : STREAM_ACTIVE
        Stream Call ID           : 10
        Stream Type              : application (8)
        Stream Media Addr Type   : 1
        Negotiated Codec         : H.224 (0 bytes)
        Codec Payload Type       : 107
        Negotiated Dtmf-relay    : inband-voice
        Dtmf-relay Payload Type  : 0
        QoS ID                   : -1
        Local QoS Strength       : BestEffort
        Negotiated QoS Strength  : BestEffort
```

```
     Negotiated QoS Direction : None
     Local QoS Status         : None
     Media Source IP Addr:Port: [9.45.47.123]:16398
     Media Dest IP Addr:Port  : [9.45.38.39]:2442

Options-Ping    ENABLED:NO    ACTIVE:NO
   Number of SIP User Agent Client(UAC) calls: 1
```

Feature Information for WebEx Telepresence Media Support Over Single SIP Session

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 3: Feature Information for WebEx Telepresence Media Support Over Single SIP Session

Feature Name	Releases	Feature Information
WebEx Telepresence Media Support Over Single SIP Session	15.3(2)T	The WebEx Telepresence Media Support over Single SIP Session feature provides support for end-to-end negotiation of up to 6 m-lines or media lines over a single Session Initiation Protocol (SIP) session. The media types can be audio, video, or application.
WebEx Telepresence Media Support Over Single SIP Session	Cisco IOS XE Release 3.9S	The WebEx Telepresence Media Support over Single SIP Session feature provides support for end-to-end negotiation of up to 6 m-lines or media lines over a single Session Initiation Protocol (SIP) session. The media types can be audio, video, or application.

CHAPTER 5

DTMF Events through SIP Signaling

The DTMF Events through SIP Signaling feature provides the following:

- DTMF event notification for SIP messages.
- Capability of receiving hookflash event notification through the SIP NOTIFY method.
- Third-party call control, or other signaling mechanisms, to provide enhanced services, such as calling card and messaging services.
- Communication with the application outside of the media connection.

The DTMF Events through SIP Signaling feature allows telephone event notifications to be sent through SIP NOTIFY messages, using the SIP SUBSCRIBE/NOTIFY method as defined in the Internet Engineering Task Force (IETF) draft, SIP-Specific Event Notification.

The feature also supports sending DTMF notifications based on the IETF draft: Signaled Telephony Events in the Session Initiation Protocol (SIP) (draft-mahy-sip-signaled-digits-01.txt).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for DTMF Events through SIP Signaling

Cisco Unified Border Element

- Cisco IOS Release 12.2(11)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for DTMF Events through SIP Signaling

The DTMF Events through SIP Signaling feature adds support for sending telephone-event notifications via SIP NOTIFY messages from a SIP gateway. The events for which notifications are sent out are DTMF events from the local Plain Old Telephone Service (POTS) interface on the gateway. Notifications are not sent for DTMF events received in the Real-Time Transport Protocol (RTP) stream from the recipient user agent.

DTMF Dialing

DTMF dialing consists of simultaneous voice-band tones generated when a button is pressed on a telephone. The use of DTMF signaling for this feature enables support for advanced telephony services. Currently there are a number of application servers and service creation platforms that do not support media connections. To provide value-added services to the network, these servers and platforms need to be aware of signaling events from a specific participant in the call. Once the server or platform is aware of the DTMF events that are being signaled, it can use third-party call control, or other signaling mechanisms, to provide enhanced services. Examples of the types of services and platforms that are supported by this feature are various voice web browser services, Centrex switches or business service platforms, calling card services, and unified message servers. All of these applications require a method for the user to communicate with the application outside of the media connection. The DTMF Events Through SIP Signaling feature provides this signaling capability.

This feature is related to the SIP INFO Method for DTMF Tone Generation feature, which adds support for out-of-band DTMF tone generation using the SIP INFO method. Together the two features provide a mechanism to both send and receive DTMF digits along the signaling path.

NOTIFY Messages

The SIP event notification mechanism uses NOTIFY messages to signal when certain telephony events take place. In order to send DTMF signals through NOTIFY messages, the gateway notifies the subscriber when DTMF digits are signaled by the originator. The notification contains a message body with a SIP response status line.

The following sample message shows a NOTIFY message from the Notifier letting the Subscriber know that the subscription is completed. The combination of the From, To, and Call-ID headers identifies the call leg. The Events header specifies the event type being signaled, and the Content-Type specifies the Internet media type. The Content-Length header indicates the number of octets in the message body.

```
NOTIFY sip:subscriber@example1.com SIP/2.0
Via: SIP/2.0/UDP example2.com:5060
From: Notifier <sip:notifier@example2.com>;tag=5678-EFGH
To: Subscriber <sip:subscriber@example1.com>;tag=1234-ABCD
Call-ID: 12345@example2.com
CSeq: 104 NOTIFY
Contact: Notifier <sip:notifier@example2.com>
Events: telephone-event;rate=1000
Content-Type: audio/telephone-event
Content-Length: 4
```

Configuring DTMF Events through SIP Signaling

To configure the DTMF Events through SIP Signaling feature, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sip-ua**
4. **timers notify** *number*
5. **retry notify** *number*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode or any other security level set by a system administrator. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sip-ua** **Example:** `Device(config)# sip-ua`	Enters SIP user-agent configuration mode.

	Command or Action	Purpose
Step 4	**timers notify** *number* **Example:** `Device(config-sip-ua)# timers notify 100`	Sets the amount of time that the user agent waits before retransmitting the Notify message. The argument is as follows: • *number* --Time, in milliseconds, to wait before retransmitting. Range: 100 to 1000. Default: 500.
Step 5	**retry notify** *number* **Example:** `Device(config-sip-ua)# retry notify 6`	Sets the number of times that the Notify message is retransmitted to the user agent that initiated the transfer or Refer request. The argument is as follows: • *number* --Number of retries. Range: 1 to 10. Default: 10.
Step 6	**exit** **Example:** `Device(config-sip-ua)# exit`	Exits the current mode.

Verifying SIP DTMF Support

To verify SIP DTMF support, perform the following steps as appropriate (commands are listed in alphabetical order).

SUMMARY STEPS

1. **show running-config**
2. **show sip-ua retry**
3. **show sip-ua statistics**
4. **show sip-ua status**
5. **show sip-ua timers**
6. **show voip rtp connections**
7. **show sip-ua calls**

DETAILED STEPS

Step 1 **show running-config**
Use this command to show dial-peer configurations.

The following sample output shows that the **dtmf-relay sip-notify** command is configured in dial peer 123:

Example:

```
Device# show running-config
.
.
.
dial-peer voice 123 voip
 destination-pattern [12]...
 monitor probe icmp-ping
 session protocol sipv2
 session target ipv4:10.8.17.42
 dtmf-relay sip-notify
```

The following sample output shows that DTMF relay and NTE are configured on the dial peer.

Example:

```
Device# show running-config
!
dial-peer voice 1000 pots
 destination-pattern 4961234
 port 1/0/0
!
dial-peer voice 2000 voip
 application session
 destination-pattern 4965678
 session protocol sipv2
 session target ipv4:192.0.2.34
 dtmf-relay rtp-nte
! RTP payload type value = 101 (default)
!
dial-peer voice 3000 voip
 application session
 destination-pattern 2021010101
 session protocol sipv2
 session target ipv4:192.0.2.34
 dtmf-relay rtp-nte
 rtp payload-type nte 110
! RTP payload type value = 110 (user assigned)
!
```

Step 2 **show sip-ua retry**

Use this command to display SIP retry statistics.

Example:

```
Device# show sip-ua retry
SIP UA Retry Values
invite retry count = 6 response retry count = 1
bye retry count = 1 cancel retry count = 1
prack retry count = 10 comet retry count = 10
reliable 1xx count = 6 notify retry count = 10
```

Step 3 **show sip-ua statistics**

Use this command to display response, traffic, and retry SIP statistics.

Tip To reset counters for the **show sip-ua statistics**display, use the **clear sip-ua statistics** command.

Example:

```
Device# show sip-ua statistics
SIP Response Statistics (Inbound/Outbound)
```

```
Informational:
Trying 4/2, Ringing 2/1,
Forwarded 0/0, Queued 0/0,
SessionProgress 0/0
Success:
OkInvite 1/2, OkBye 0/1,
OkCancel 1/0, OkOptions 0/0,
OkPrack 2/0, OkPreconditionMet 0/0,
OkNotify 1/0, 202Accepted 0/1
Redirection (Inbound only):
MultipleChoice 0, MovedPermanently 0,
MovedTemporarily 0, SeeOther 0,
UseProxy 0, AlternateService 0
Client Error:
BadRequest 0/0, Unauthorized 0/0,
PaymentRequired 0/0, Forbidden 0/0,
NotFound 0/0, MethodNotAllowed 0/0,
NotAcceptable 0/0, ProxyAuthReqd 0/0,
ReqTimeout 0/0, Conflict 0/0, Gone 0/0,
LengthRequired 0/0, ReqEntityTooLarge 0/0,
ReqURITooLarge 0/0, UnsupportedMediaType 0/0,
BadExtension 0/0, TempNotAvailable 0/0,
CallLegNonExistent 0/0, LoopDetected 0/0,
TooManyHops 0/0, AddrIncomplete 0/0,
Ambiguous 0/0, BusyHere 0/0
RequestCancel 1/0, NotAcceptableMedia 0/0
Server Error:
InternalError 0/1, NotImplemented 0/0,
BadGateway 0/0, ServiceUnavail 0/0,
GatewayTimeout 0/0, BadSipVer 0/0,
PreCondFailure 0/0
Global Failure:
BusyEverywhere 0/0, Decline 0/0,
NotExistAnywhere 0/0, NotAcceptable 0/0
SIP Total Traffic Statistics (Inbound/Outbound) /* Traffic Statistics
Invite 3/2, Ack 3/2, Bye 1/0,
Cancel 0/1, Options 0/0,
Prack 0/2, Comet 0/0,
Notify 0/1, Refer 1/0
Retry Statistics        /* Retry Statistics
Invite 0, Bye 0, Cancel 0, Response 0,
Prack 0, Comet 0, Reliable1xx 0, Notify 0
```

Following is sample output verifying configuration of the SIP INFO Method for DTMF Tone Generation feature:

Example:

```
Device# show sip-ua statistics
SIP Response Statistics (Inbound/Outbound)
Informational:
Trying 1/1, Ringing 0/0,
Forwarded 0/0, Queued 0/0,
SessionProgress 0/1
Success:
OkInvite 0/1, OkBye 1/0,
OkCancel 0/0, OkOptions 0/0,
OkPrack 0/0, OkPreconditionMet 0/0
OkSubscibe 0/0, OkNotify 0/0,
OkInfo 0/0, 202Accepted 0/0
Redirection (Inbound only):
MultipleChoice 0, MovedPermanently 0,
MovedTemporarily 0, SeeOther 0,
UseProxy 0, AlternateService 0
Client Error:
BadRequest 0/0, Unauthorized 0/0,
PaymentRequired 0/0, Forbidden 0/0,
NotFound 0/0, MethodNotAllowed 0/0,
NotAcceptable 0/0, ProxyAuthReqd 0/0,
ReqTimeout 0/0, Conflict 0/0, Gone 0/0,
```

```
LengthRequired 0/0, ReqEntityTooLarge 0/0,
ReqURITooLarge 0/0, UnsupportedMediaType 0/0,
BadExtension 0/0, TempNotAvailable 0/0,
CallLegNonExistent 0/0, LoopDetected 0/0,
TooManyHops 0/0, AddrIncomplete 0/0,
Ambiguous 0/0, BusyHere 0/0,
BadEvent 0/0
Server Error:
InternalError 0/0, NotImplemented 0/0,
BadGateway 0/0, ServiceUnavail 0/0,
GatewayTimeout 0/0, BadSipVer 0/0
Global Failure:
BusyEverywhere 0/0, Decline 0/0,
NotExistAnywhere 0/0, NotAcceptable 0/0
SIP Total Traffic Statistics (Inbound/Outbound)
    Invite 0/0, Ack 0/0, Bye 0/0,
    Cancel 0/0, Options 0/0,
    Prack 0/0, Comet 0/0,
    Subscribe 0/0, Notify 0/0,
    Refer 0/0, Info 0/0
Retry Statistics
Invite 0, Bye 0, Cancel 0, Response 0, Notify 0
```

Step 4 **show sip-ua status**

Use this command to display status for the SIP user agent.

Example:

```
Device# show sip-ua status
SIP User Agent Status
SIP User Agent for UDP : ENABLED
SIP User Agent for TCP : ENABLED
SIP User Agent bind status(signaling): DISABLED
SIP User Agent bind status(media): DISABLED
SIP max-forwards : 6
SIP DNS SRV version: 2 (rfc 2782)
SDP application configuration:
 Version line (v=) required
 Owner line (o=) required
 Session name line (s=) required
 Timespec line (t=) required
 Media supported: audio image
 Network types supported: IN
 Address types supported: IP4
 Transport types supported: RTP/AVP udptl
```

The following sample output shows that the time interval between consecutive NOTIFY messages for a telephone event is the default of 2000 ms:

Example:

```
Device# show sip-ua status
SIP User Agent Status
SIP User Agent for UDP : ENABLED
SIP User Agent for TCP : ENABLED
SIP User Agent bind status(signaling): DISABLED
SIP User Agent bind status(media): DISABLED
SIP early-media for 180 responses with SDP: ENABLED
SIP max-forwards : 6
SIP DNS SRV version: 2 (rfc 2782)
NAT Settings for the SIP-UA
Role in SDP: NONE
Check media source packets: DISABLED
Maximum duration for a telephone-event in NOTIFYs: 2000 ms
SIP support for ISDN SUSPEND/RESUME: ENABLED
Redirection (3xx) message handling: ENABLED
```

```
 SDP application configuration:
 Version line (v=) required
 Owner line (o=) required
 Timespec line (t=) required
 Media supported: audio image
 Network types supported: IN
 Address types supported: IP4
 Transport types supported: RTP/AVP udptl
```

The following sample output shows configuration of the SIP INFO Method for DTMF Tone Generation feature:

Example:

```
Device# show sip-ua status
SIP User Agent Status
SIP User Agent for UDP : ENABLED
SIP User Agent for TCP : ENABLED
SIP User Agent bind status(signaling): DISABLED
SIP User Agent bind status(media): DISABLED
SIP max-forwards : 6
SIP DNS SRV version: 2 (rfc 2782)
SDP application configuration:
 Version line (v=) required
 Owner line (o=) required
 Session name line (s=) required
 Timespec line (t=) required
 Media supported: audio image
 Network types supported: IN
 Address types supported: IP4
 Transport types supported: RTP/AVP udptl
```

Step 5 **show sip-ua timers**

Use this command to display the current settings for SIP user-agent timers.

Example:

```
Device# show sip-ua timers
SIP UA Timer Values (millisecs)
trying 500, expires 300000, connect 500, disconnect 500
comet 500, prack 500, rel1xx 500, notify 500
```

Step 6 **show voip rtp connections**

Use this command to show local and remote Calling ID and IP address and port information.

Step 7 **show sip-ua calls**

Use this command to ensure the DTMF method is SIP-KPML.

The following sample output shows that the DTMF method isSIP-KPML.

Example:

```
Device# show sip-ua calls
SIP UAC CALL INFO
Call 1
SIP Call ID                : 57633F68-2BE011D6-8013D46B-B4F9B5F6@172.18.193.251
   State of the call       : STATE_ACTIVE (7)
   Substate of the call    : SUBSTATE_NONE (0)
   Calling Number          :
   Called Number           : 8888
   Bit Flags               : 0xD44018 0x100 0x0
   CC Call ID              : 6
   Source IP Address (Sig ): 192.0.2.1
   Destn SIP Req Addr:Port : 192.0.2.2:5060
```

```
   Destn SIP Resp Addr:Port: 192.0.2.3:5060
   Destination Name        : 192.0.2.4.250
   Number of Media Streams : 1
   Number of Active Streams: 1
   RTP Fork Object         : 0x0
   Media Mode              : flow-through
   Media Stream 1
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 6
     Stream Type              : voice-only (0)
     Negotiated Codec         : g711ulaw (160 bytes)
 Codec Payload Type       : 0
     Negotiated Dtmf-relay    : sip-kpml
     Dtmf-relay Payload Type  : 0
     Media Source IP Addr:Port: 192.0.2.5:17576
     Media Dest IP Addr:Port  : 192.0.2.6:17468
     Orig Media Dest IP Addr:Port : 0.0.0.0:0
   Number of SIP User Agent Client(UAC) calls: 1
SIP UAS CALL INFO
   Number of SIP User Agent Server(UAS) calls: 0
```

Troubleshooting Tips

- To enable debugging for RTP named-event packets, use the **debug voip rtp** command.
- To enable KPML debugs, use the **debug kpml** command.
- To enable SIP debugs, use the **debug ccsip** command.
- Collect debugs while the call is being established and during digit presses.
- If an established call is not sending digits through KPML, use the **show sip-ua calls** command to ensure SIP-KPML is included in the negotiation process.

Feature Information for DTMF Events through SIP Signaling

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 4: Feature Information for Configuring DTMF Events through SIP Signaling

Feature Name	Releases	Feature Information
DTMF Events through SIP Signaling	12.2(11)T 12.2(8)YN 12.2(15)T 12.2(11)YV 12.2(11)T,	The DTMF Events through SIP Signaling feature provides the following: • DTMF event notification for SIP messages. • Capability of receiving hookflash event notification through the SIP NOTIFY method. • Third-party call control, or other signaling mechanisms, to provide enhanced services, such as calling card and messaging services. • Communication with the application outside of the media connection. The following commands were introduced or modified: **timers notify** and **retry notify**.
DTMF Events through SIP Signaling	Cisco IOS XE Release 2.5	The DTMF Events through SIP Signaling feature provides the following: • DTMF event notification for SIP messages. • Capability of receiving hookflash event notification through the SIP NOTIFY method. • Third-party call control, or other signaling mechanisms, to provide enhanced services, such as calling card and messaging services. • Communication with the application outside of the media connection. The following commands were introduced or modified: **timers notify** and **retry notify**.

CHAPTER 6

Call Progress Analysis Over IP-to-IP Media Session

The Call Progress Analysis Over IP-IP Media Session feature enables the detection of automated answering systems and live human voices on outbound calls and communicates the detected information to the external application. Typically, call progress analysis (CPA) is extensively used in contact center deployments in conjunction with the outbound Session Initiation Protocol (SIP) dialer, where CPA is enabled on the Cisco Unified Border Element (Cisco UBE), and digital signal processors (DSP) perform the CPA functionality.

Feature Information for Call Progress Analysis Over IP-IP Media Session

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 5: Feature Information for Call Progress Analysis Over IP-IP Media Session

Feature Name	Releases	Feature Information
Call Progress Analysis Over IP-to-IP Media Session	15.3(2)T	The Call Progress Analysis Over IP-to-IP Media Session feature enables detection of automated answering systems and live human voices on outbound calls and communicates the detected information to an external application. The following command was introduced: **call-progress-analysis**.
Call Progress Analysis Over IP-to-IP Media Session	Cisco IOS XE Release 3.9S	The Call Progress Analysis Over IP-to-IP Media Session feature enables detection of automated answering systems and live human voices on outbound calls and communicates the detected information to an external application. The following command was introduced: **call-progress-analysis**.
Support for additional call flows	15.5(2)T Cisco IOS XE Release 3.15S	Call Progress Analysis feature is enhanced to support the following call-flows: • 180 SIP response received without SDP • Direct call connect (without 18x from Service Provider) • Multiple 18x response to INVITE • Early dialog UPDATE • Dialer-CUBE CPA call record

Restrictions for Call Progress Analysis Over IP-to-IP Media Session

- Only SIP-to-SIP Early Offer (EO-to-EO) call flows are supported.
- Session Description Protocol (SDP) passthrough and flow-around media calls are not supported.
- Only the G711 flavor of codec is supported.
- High Availability (HA) is not supported.
- Skinny Client Control Protocol (SCCP)-based digital signal processor (DSP) farm is not supported.

- CPA cannot not be detected if Dialer uses Inband as DTMF relay mechanism, that is, Inband to RTP-NTE DTMF inter-working is not supported with CPA.
- CPA call record is not supported for "180 without SDP" and "Direct Call Connect (without 18x)" call flows from Service Provider.

Information About Call Progress Analysis Over IP-IP Media Session

Call Progress Analysis

Call progress analysis (CPA) is a DSP algorithm that analyzes the Real-Time Transport Protocol (RTP) voice stream to look for special information tones (SIT), fax or modem tones, human speech, and answering machine tones. CPA also passes the voice information to Cisco IOS or Cisco Unified Border Element (Cisco UBE).

CPA is initiated on receiving a new SIP INVITE with x-cisco-cpa content. While a call is in progress, the DSP or the Xcoder analyzes the incoming voice or media stream. The DSP identifies the type of voice stream based on statistical voice patterns or specific tone frequencies and provides the information to the Cisco UBE. The Cisco UBE notifies the dialer with a SIP UPDATE with x-cisco-cpa content along with the detected event. Based on the report, the caller (dialer) can decide to either transfer the call or terminate the call.

To use the CPA functionality, you must enable CPA and configure CPA timing and threshold parameters.

Table 6: X-cisco-cpa content meaning

SIP Message	Direction of Message	Meaning
18x or 200	Cisco IOS to dialer	Cisco UBE informs the dialer if CPA is enabled for a call or not.
New INVITE	Dialer to Cisco IOS	Dialer requests Cisco IOS or the Cisco UBE to activate the CPA algorithm for this session.
UPDATE	Cisco IOS to dialer	Cisco IOS or the Cisco UBE notifies the dialer about the detected event.

CPA Events

Table 7: CPA Event Detection List

CPA Event	Definition
Asm	Answer machine

CPA Event	Definition
AsmT	Answer machine terminate tone
CpaS	Start of the Call Progress Analysis
FT	Fax/Modem tone
LS	Live human speech
LV	Low volume or dead air call
SitIC	Special information tone IC -- Intercept -- Vacant number or Automatic Identification System (AIS)
SitNC	SIT tone NC—No Circuit (NC), Emergency, or Trunk Blockage
SitVC	SIT tone VC—Vacant Code
SitRO	SIT tone RO—Reorder Announcement
SitMT	Miscellaneous SIT Tone

How to Configure Call Progress Analysis Over IP-to-IP Media Session

Enabling CPA and Setting the CPA Parameters

Perform the following task to enable CPA and set the CPA timing and threshold parameters:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dspfarm profile** *profile-identifier* **transcode**
4. **call-progress-analysis**
5. **exit**
6. **voice service voip**
7. **cpa timing live-person** *max-duration*
8. **cpa timing term-tone** *max-duration*
9. **cpa threshold active-signal** *signal-threshold*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dspfarm profile** *profile-identifier* **transcode** **Example:** `Device(config)# dspfarm profile 15 transcode`	Enters DSP farm profile configuration mode, defines a profile for DSP farm services, and enables the profile for transcoding.
Step 4	**call-progress-analysis** **Example:** `Device(config-dspfarm-profile)# call-progress-analysis`	Enables call progress analysis (CPA) on Cisco UBE. • You must configure this command to activate the CPA feature and set CPA parameters.
Step 5	**exit** **Example:** `Device(config-dspfarm-profile)# exit`	Exits DSP farm profile configuration mode and enters global configuration mode.
Step 6	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters voice service configuration mode.
Step 7	**cpa timing live-person** *max-duration* **Example:** `Device(conf-voi-serv)# cpa timing live-person 2501`	(Optional) Sets the maximum waiting time (in milliseconds) that the CPA algorithm uses to determine if a call is answered by a live human.
Step 8	**cpa timing term-tone** *max-duration* **Example:** `Device(conf-voi-serv)# cpa timing term-tone 15500`	(Optional) Sets the maximum waiting time (in milliseconds) that the CPA algorithm uses to wait for the answering machine termination tone after the answering machine is detected.

	Command or Action	Purpose
Step 9	**cpa threshold active-signal** *signal-threshold* **Example:** `Device(conf-voi-serv)# cpa threshold active-signal 18db`	(Optional) Sets the threshold (in decibels) of an active signal that is related to the measured noise floor level. • If a signal threshold configured by this command is greater than the measured noise floor level, then the signal is considered as active. The active signal thresholds that you can configure are 9, 12, 15, 18, and 21 decibels.
Step 10	**end** **Example:** `Device(conf-voi-serv)# end`	Exits voice service configuration mode and returns to privileged EXEC mode.

Verifying the Call Progress Analysis Over IP-to-IP Media Session

Perform this task to verify that call progress analysis has been configured for a digital signal processor (DSP) farm profile.

SUMMARY STEPS

1. **enable**
2. **show dspfarm profile** *profile-identifier*

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show dspfarm profile** *profile-identifier*

Displays the configured DSP farm profile information for a selected Cisco Call Manager group. In the following sample output, the Call Progress Analysis field shows that CPA is enabled.

Example:

```
Device# show dspfarm profile 3

 Profile ID = 3, Service =Universal TRANSCODING, Resource ID = 3
 Profile Description :
 Profile Service Mode : Non Secure
 Profile Admin State : UP
 Profile Operation State : ACTIVE
```

```
Application : CUBE   Status : ASSOCIATED
Resource Provider : FLEX_DSPRM   Status : UP
Number of Resource Configured : 4
Number of Resources Out of Service : 0
Number of Resources Active : 0
Codec Configuration: num_of_codecs:4
Codec : g711ulaw, Maximum Packetization Period : 30
Codec : g711alaw, Maximum Packetization Period : 30
Codec : g729ar8, Maximum Packetization Period : 60
Codec : g729abr8, Maximum Packetization Period : 60
Noise Reduction : ENABLED
Call Progress Analysis : ENABLED
```

Troubleshooting Tips

Use the following commands to troubleshoot the call progress analysis for SIP-to-SIP calls:

- **debug ccsip all**
- **debug voip ccapi inout**
- **debug voip hpi all**
- **debug voip ipipgw**
- **debug voip media resource provisioning all**

Configuration Examples for the Call Progress Analysis Over IP-to-IP Media Session

Example: Enabling CPA and Setting the CPA Parameters

The following example shows how to enable CPA and set a few timing and threshold parameters. Depending on your requirements, you can configure more timing and threshold parameters.

```
Device> enable
Device# configure terminal
Device(config)# dspfarm profile 15 transcode
Device(config-dspfarm-profile)# call-progress-analysis
Device(config-dspfarm-profile)# exit
Device(config)# voice service voip
Device(conf-voi-serv)# cpa timing live-person 2501
Device(conf-voi-serv)# cpa timing term-tone 15500
Device(conf-voi-serv)# cpa threshold active-signal 18db
Device(conf-voi-serv)# end
```

CHAPTER 7

Codec Preference Lists

This chapter describes how to negotiate an audio codec from a list of codec associated with a preference. This chapter also describes how to disable codec filtering by configuring CUBE to send an outgoing offer with all configured audio codecs in the list assuming that the dspfarm supports all these codecs.

Feature Information for Negotiation of an Audio Codec from a List of Codecs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 8: Feature Information for Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element

Feature Name	Releases	Feature Information
Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element	15.1(2)T	The Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element feature supports negotiation of an audio codec using the Voice Class Codec and Codec Transparent infrastructure on the Cisco UBE. The following command was introduced or modified: **voice-class codec (dial peer).**
Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element	Cisco IOS XE Release 3.8S	The Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element feature supports negotiation of an audio codec using the Voice Class Codec and Codec Transparent infrastructure on the Cisco UBE. The following command was introduced or modified: **voice-class codec (dial peer)**.
Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element.	15.3(2)T	This feature provides high availability support for negotiation of an audio codec from a list of codecs on each leg of a SIP-to-SIP call on the Cisco Unified Border Element under the Voice Class Codec.

Codecs configured using Preference Lists

SIP-to-SIP calls configured using codecs using preference lists have the following features:

- Incoming and outgoing dial-peers can be configured with different preference lists.
- Both normal transcoding and high-density transcoding are supported with preference lists.
- Mid-call codec changes for supplementary services are supported with preference lists. Transcoder resources are dynamically inserted or deleted when there is a codec or RTP-NTE to inband DTMF interworking required.

- Reinvite-based supplementary services invoked from the Cisco Unified Communications Manager (CUCM), like call hold, call resume, music on hold (MOH), call transfer, and call forward are supported with preference lists.
- T.38 fax and fax passthrough switchover with preference lists are supported.
- Reinvite-based call hold and call resume for Secure Real-Time Transfer protocol (SRTP) and Real-Time Transport Protocol (RTP) interworking on CUBE is supported with preference lists.
- High availability is supported for calls that use codecs with preference lists. But calls requiring the transcoder to be invoked are not checkpointed. During mid-call renegotiation, if the call releases the transcoder, then the call is checkpointed.

Prerequisites for Codec Preference Lists

- Transcoding configuration on the CUBE.
- The digital signal processor (DSP) requirements to support the transcoding feature on the CUBE.

Restrictions for Codecs Preference Lists

For All Calls (SIP-to-SIP, H323-to-H323, SIP-to-H323 calls)

- Video codecs are not supported with preference lists.
- Multiple audio streams are not supported.
- High-density transcoding is not supported when delayed offer to early offer is configured. Only low density transcoding is supported.
- Codec re-packetization feature is not supported when preference lists are configured.

For H323-to-H323 and SIP-to-H323 Calls

The below restrictions do not exist for SIP-to-SIP calls from 15.1(2)T and Cisco IOS XE Release 3.8S onwards.

- You can configure dissimilar preference lists on the incoming and outgoing dial peers.
- Incoming and outgoing dial-peers cannot be configured with the different preference lists.
- Transcoding is not supported when preference lists are used.
- Mid-call codec changes and supplementary services (call-hold / resume, call forward) do not work when a preference list is configured.
- Mid-call insertion or deletion of transcoder is not supported with preference lists.
- Rotary dial peers are not supported when preference lists are used.
- Both incoming and outgoing dial-peers need to be configured with the same codec voice classes.
- The preference of codecs configured in a codec voice classes is not be applied to the outgoing call-leg. Basically codec filtering is applied first and only the filtered codecs will be sent out in the outgoing offer from CUBE.

- T.38 fax, fax-passthru and modem-passthru is not be supported with preference lists.
- SRTP<->RTP is not supported with preference lists.
- When a codec voice class is configured, call establishment is un-predictable when a transcoder is involved in the call. The call succeeds only if the end points choose the first codec in the list of offered codecs.

How to Configure Codec Preference Lists

Configuring Audio Codecs Using a Codec Voice Class and Preference Lists

Preferences can be used to determine which codecs will be selected over others.

A codec voice class is a construct within which a codec preference order can be defined. A codec voice class can then be applied to a dial peer, which then follows the preference order defined in the codec voice class.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class codec** *tag*
4. Do the following for each audio codec you want to configure in the voice class:
    - **codec preference** *value codec-type*[**bytes** *payload-size* **fixed-bytes**]
    - **codec preference** *value* **isac** [**mode {adaptive | independent}** [**bit-rate** *value* **framesize { 30 | 60 } [fixed]**]
    - **codec preference** *value* **ilbc** [**mode** *frame-size* [**bytes** *payload-size*]]
    - **codec preference** *value* **mp4-latm** [**profile** *tag*]
5. **exit**
6. **dial-peer voice** *number* **voip**
7. **voice-class codec** *tag* **offer-all**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device> configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**voice class codec** *tag* **Example:** `Device(config)# voice class codec 10`	Enters voice-class configuration mode for the specified codec voice class.
Step 4	Do the following for each audio codec you want to configure in the voice class: • **codec preference** *value* *codec-type*[**bytes** *payload-size* **fixed-bytes**] • **codec preference** *value* **isac** [**mode {adaptive | independent}** [**bit-rate** *value* **framesize { 30 | 60 }** **[fixed]**] • **codec preference** *value* **ilbc** [**mode** *frame-size* [**bytes** *payload-size*]] • **codec preference** *value* **mp4-latm** [**profile** *tag*]	Configure a codec within the voice class and specifies a preference for the codec. This becomes part of a preference list
Step 5	**exit** **Example:** `Device(config-class)# exit`	Exits the current mode. • Enter your password if prompted.
Step 6	**dial-peer voice** *number* **voip** **Example:** `Device(config)# dial-peer voice 1 voip`	Enters dial peer configuration mode for the specified VoIP dial peer.
Step 7	**voice-class codec** *tag* **offer-all** **Example:** `Device(config-dial-peer)# voice-class codec 10`	Applies the previously configured voice class and associated codecs to a dial peer. • The **offer-all** keyword allows the device to offer all codecs configured in a codec voice class.
Step 8	**end** **Example:** `Device(config-dial-peer)# end`	Returns to privileged EXEC mode.

Disabling Codec Filtering

Cisco UBE is configured to filter common codecs for the subsets, by default. The filtered codecs are sent in the outgoing offer. You can configure the Cisco UBE to offer all the codecs configured on an outbound leg instead of offering only the filtered codecs.

Note This configuration is applicable only for early offer calls from the Cisco UBE. For delayed offer calls, by default all codecs are offered irrespective of this configuration.

Perform this task to disable codec filtering and allow all the codecs configured on an outbound leg.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **voice-class codec** *tag* **offer-all**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 10 voip`	Enters dial peer voice configuration mode.
Step 4	**voice-class codec** *tag* **offer-all** **Example:** `Device(config-dial-peer)# voice-class codec 10 offer-all`	Adds all the configured voice class codec to the outgoing offer from the Cisco UBE.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	Exits the dial peer voice configuration mode.

Troubleshooting Negotiation of an Audio Codec from a List of Codecs

Use the following commands to debug any errors that you may encounter when you configure the Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element feature:

- **debug ccsip all**
- **debug voip ccapi input**
- **debug sccp messages**
- **debug voip rtp session**

For DSP-related debugs, use the following commands:

- **debug voip dsmp all**
- **debug voip dsmp rtp both payload all**
- **debug voip ipipgw**

Verifying Negotiation of an Audio Codec from a List of Codecs

Perform this task to display information to verify Negotiation of an Audio Codec from a List of Codecs on Each Leg of a SIP-to-SIP Call on the Cisco Unified Border Element configuration. These **show** commands need not be entered in any specific order.

SUMMARY STEPS

1. **enable**
2. **show call active voice brief**
3. **show voip rtp connections**
4. **show sccp connections**
5. **show dspfarm dsp active**

DETAILED STEPS

Step 1 **enable**
Enables privileged EXEC mode.

Step 2 **show call active voice brief**
Displays a truncated version of call information for voice calls in progress.

Example:

```
Device# show call active voice brief
```

```
<ID>: <CallID> <start>ms.<index> +<connect> pid:<peer_id> <dir> <addr> <state>
  dur hh:mm:ss tx:<packets>/<bytes> rx:<packets>/<bytes>
 IP <ip>:<udp> rtt:<time>ms pl:<play>/<gap>ms lost:<lost>/<early>/<late>
  delay:<last>/<min>/<max>ms <codec>
 media inactive detected:<y/n> media cntrl rcvd:<y/n> timestamp:<time>
 long duration call detected:<y/n> long duration call duration :<sec> timestamp:<time>
  MODEMPASS <method> buf:<fills>/<drains> loss <overall%> <multipkt>/<corrected>
   last <buf event time>s dur:<Min>/<Max>s
 FR <protocol> [int dlci cid] vad:<y/n> dtmf:<y/n> seq:<y/n>
  <codec> (payload size)
 ATM <protocol> [int vpi/vci cid] vad:<y/n> dtmf:<y/n> seq:<y/n>
  <codec> (payload size)
 Tele <int> (callID) [channel_id] tx:<tot>/<v>/<fax>ms <codec> noise:<l> acom:<l> i/o:<l>/<l> dBm
  MODEMRELAY info:<rcvd>/<sent>/<resent> xid:<rcvd>/<sent> total:<rcvd>/<sent>/<drops>
         speeds(bps): local <rx>/<tx> remote <rx>/<tx>
 Proxy <ip>:<audio udp>,<video udp>,<tcp0>,<tcp1>,<tcp2>,<tcp3> endpt: <type>/<manf>
 bw: <req>/<act> codec: <audio>/<video>
  tx: <audio pkts>/<audio bytes>,<video pkts>/<video bytes>,<t120 pkts>/<t120 bytes>
 rx: <audio pkts>/<audio bytes>,<video pkts>/<video bytes>,<t120 pkts>/<t120 bytes>
Telephony call-legs: 0
SIP call-legs: 2
H323 call-legs: 0
Call agent controlled call-legs: 0
SCCP call-legs: 2
Multicast call-legs: 0
Total call-legs: 4
1243 : 11 971490ms.1 +-1 pid:1 Answer 1230000 connecting
 dur 00:00:00 tx:415/66400 rx:17/2561
 IP 192.0.2.1:19304 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a
1243 : 12 971500ms.1 +-1 pid:2 Originate 3210000 connected
 dur 00:00:00 tx:5/10 rx:4/8
 IP 9.44.26.4:16512 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g729br8 TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a
0    : 13 971560ms.1 +0 pid:0 Originate  connecting
 dur 00:00:08 tx:415/66400 rx:17/2561
 IP 192.0.2.2:2000 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a
0    : 15 971570ms.1 +0 pid:0 Originate  connecting
 dur 00:00:08 tx:5/10 rx:3/6
 IP 192.0.2.3:2000 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g729br8 TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a
Telephony call-legs: 0
SIP call-legs: 2
H323 call-legs: 0
Call agent controlled call-legs: 0
SCCP call-legs: 2
Multicast call-legs: 0
Total call-legs: 4
```

Step 3 **show voip rtp connections**

Displays Real-Time Transport Protocol (RTP) connections.

Example:

```
Device# show voip rtp connections
VoIP RTP active connections :
No. CallId     dstCallId  LocalRTP RmtRTP     LocalIP                                 RemoteIP
1     11         12         16662    19304    192.0.2.1
192.0.2.2
2     12         11         17404    16512    192.0.2.2
192.0.2.3
3     13         14         18422    2000     192.0.2.4
9.44.26.3
4     15         14         16576    2000     192.0.2.6
```

```
192.0.2.5
Found 4 active RTP connections
```

Step 4 **show sccp connections**

Displays information about the connections controlled by the Skinny Client Control Protocol (SCCP) transcoding and conferencing applications.

Example:

```
Device# show sccp connections
sess_id    conn_id      stype mode     codec   sport rport ripaddr
5          5            xcode sendrecv g729b   16576 2000  192.0.2.3
5          6            xcode sendrecv g711u   18422 2000  192.0.2.4
Total number of active session(s) 1, and connection(s) 2
```

Step 5 **show dspfarm dsp active**

Displays active DSP information about the DSP farm service.

Example:

```
Device# show dspfarm dsp active
SLOT DSP VERSION  STATUS CHNL USE   TYPE    RSC_ID BRIDGE_ID PKTS_TXED PKTS_RXED
0    1   27.0.201 UP     1    USED  xcode   1      0x9         5         8
0    1   27.0.201 UP     1    USED  xcode   1      0x8         2558      17
Total number of DSPFARM DSP channel(s) 1
```

CHAPTER 8

AAC-LD MP4A-LATM Codec Support on Cisco UBE

The AAC-LD MP4A-LATM codec is a wideband audio codec used by video endpoints. MP4A-LATM is an MPEG4 audio coding standard, where LATM is Low-Overhead MPEG-4 Audio Transport Multiplex. The Cisco Unified Border Element (Cisco UBE) supports MP4A-LATM to enable call flows involving endpoints that use this codec, especially for media recording.

For basic information on Codecs and how to configure them, refer to Codecs in the Cisco Unified Border Element Fundamentals and Basic Setup.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for AAC-LD MP4A-LATM Codec Support on Cisco UBE

Cisco UBE does not support the following:

- Codec transcoding between MP4A-LATM and other codecs
- Dual-tone Multifrequency (DTMF) interworking with MP4A-LATM codec
- Non-SIP-SIP, that is, SIP to other service provider interface (SPI) interworking with MP4A-LATM codec

AAC-LD MP4A-LATM Codec Support on Cisco UBE

As part of this feature, Cisco UBE supports the following:

- Accept and send MP4A-LATM codec and corresponding FMTP profiles
- Configure MP4A-LATM under dial-peer or under voice-class codec as preferred codec
- Pass across real-time transport protocol (RTP) media for MP4A-LATM codec without any interworking
- Offer pre-configured FMTP profile for MP4A-LATM for DO-EO (Delayed-Offer to Early-Offer) calls
- Offer more than one FMTP profile (each with different payload type number) as mentioned by the offering endpoint, so that the answering endpoint can choose the best option.
- Offer only one instance of MP4A-LATM if media forking is applicable. The offered instance is the first one received in the offer.
- Calculate bandwidth for MP4A-LATM on the basis of either "b=TIAS" attribute or "bitrate" parameter in the FMTP attribute. If none of them are present in the session description protocol (SDP), the default maximum bandwidth, that is, 128 Kbps will be used for calculation.
- The following Cisco UBE features are supported with the MP4A-LATM codec:
    - Basic call (audio and video) flow-around and flow-through (FA and FT).
    - Voice Class Codec support in Cisco UBE with codec filtering
    - SRTP and SRCTP passthrough for SIP-to-SIP calls
    - Supplementary services
    - RSVP
    - Dynamic payload type interworking for DTMF and codec packets for SIP-to-SIP calls
    - Media Anti-Trombone with SIP signaling control on CUBE
    - Support for SIP UPDATE message per RFC 3311
    - RTP Media Loopback
    - Media forking for IP based calls using Zephyr recording server
    - Cisco UBE Mid-call Re-INVITE consumption

- ◦ Signaling forking (Fastweb multile SIP Early Dialog Support, FA and FT)
- ◦ Maximum bandwidth-based CAC
- ◦ Media Policing
- ◦ Box-to-Box High Availability (B2B HA)
- ◦ Inbox High Availability (Inbox HA)

How to Configure the MP4A-LATM Codec

Configuring the MP4A-LATM Codec on a Dial Peer

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **destination-pattern** [+] *string* [**T**]
5. **session protocol sipv2**
6. **session target ipv4:***destination-address*
7. **codec mp4a-latm** [**profile** *tag*]
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode or any other security level set by a system administrator. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 24 voip`	Specifies the method of voice encapsulation and enters dial peer voice configuration mode for the specified dial peer.

	Command or Action	Purpose
Step 4	**destination-pattern** [+] *string* [**T**] **Example:** `Device(config-dial-peer)# destination-pattern 595959`	Specifies either the prefix or the full E.164 telephone number (depending on your dial plan) to be used for a dial peer. Keywords and arguments are as follows: • **+** --(Optional) Character that indicates an E.164 standard number. • *string* --Series of digits that specify the E.164 or private dialing plan telephone number. Valid entries are the digits 0 through 9, the letters A through D, and any special character. • **T** --(Optional) Control character indicating that the **destination-pattern** value is a variable-length dial string.
Step 5	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Configures the VoIP dial peer to use Session Initiation Protocol (SIP).
Step 6	**session target ipv4:***destination-address* **Example:** `Device(config-dial-peer)# session target ipv4:10.42.29.7`	Specifies a network-specific address for a dial peer. Keyword and argument are as follows: • **ipv4:** *destination address* --IP address of the dial peer, in this format: *xxx.xxx.xxx.xxx*
Step 7	**codec mp4a-latm** [**profile** *tag*] **Example:** `Device(config-dial-peer)# codec mp4a-latm profile 5`	Configures the MP4A-LATM codec for the dial peer.
Step 8	**end** **Example:** `Device(config-dial-peer)# end`	Exits dial peer voice configuration mode.

Configuring the MP4A-LATM Codec under Voice Class Codec

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class codec** *tag*
4. **codec preference** *value codec-type* [**profile** *tag*]
5. **exit**
6. **dial-peer voice** *tag* **voip**
7. **destination-pattern** [+] *string* [**T**]
8. **session protocol sipv2**
9. **session target ipv4:***destination-address*
10. **voice-class codec** *tag*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode or any other security level set by a system administrator. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice class codec** *tag* **Example:** `Device(config)# voice class codec 1`	Enters voice-class configuration mode and assigns an identification tag number for a codec voice class.
Step 4	**codec preference** *value codec-type* [**profile** *tag*] **Example:** `Device(config-class)# codec preference 1 mp4a-latm profile 5`	Specifies the preferred codec (or codecs) to use on a dial peer.
Step 5	**exit** **Example:** `Device(config-class)# exit`	Exits voice-class configuration mode.

	Command or Action	Purpose
Step 6	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 24 voip`	Specifies the method of voice encapsulation and enters dial peer voice configuration mode for the specified dial peer.
Step 7	**destination-pattern** [+] *string* [**T**] **Example:** `Device(config-dial-peer)# destination-pattern 595959`	Specifies either the prefix or the full E.164 telephone number (depending on your dial plan) to be used for a dial peer. Keywords and arguments are as follows: • **+** --(Optional) Character that indicates an E.164 standard number. • *string* --Series of digits that specify the E.164 or private dialing plan telephone number. Valid entries are the digits 0 through 9, the letters A through D, and any special character. • **T** --(Optional) Control character indicating that the **destination-pattern** value is a variable-length dial string.
Step 8	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Configures the VoIP dial peer to use Session Initiation Protocol (SIP).
Step 9	**session target ipv4:***destination-address* **Example:** `Device(config-dial-peer)# session target ipv4:10.42.29.7`	Specifies a network-specific address for a dial peer. Keyword and argument are as follows: • **ipv4:** *destination address* --IP address of the dial peer, in this format: *xxx.xxx.xxx.xxx*
Step 10	**voice-class codec** *tag* **Example:** `Device(config-dial-peer)# voice-class codec 1`	Enters voice-class configuration mode and assigns an identification tag number for a codec voice class.

Verifying an Audio Call

SUMMARY STEPS

1. **show call active voice [compact]**

DETAILED STEPS

show call active voice [compact]
Displays a compact version of call information for voice calls in progress.

Example:
```
Device# show call active voice compact

 <callID>  A/O FAX T<sec> Codec       type        Peer Address       IP R<ip>:<udp>
Total call-legs: 2
        23 ANS     T3     mp4a-latm   VOIP        Psipp          9.45.33.11:57210
        24 ORG     T3     mp4a-latm   VOIP        P123           9.45.33.11:57210
```

Example:
```
Device# show call active voice compact

<callID>   A/O FAX T<sec> Codec    type    Peer Address    IP R<ip>:<udp>
Total call-legs: 2
58 ANS     T11         g711ulaw   VOIP     Psipp 2001:......:230A:6080
59 ORG     T11         g711ulaw   VOIP     P5000110011       10.13.37.150:6090
```

Configuration Examples for AAC-LD MP4A-LATM Codec Support on Cisco UBE

Example: Configuring the MP4A-LATM Codec under a Dial Peer

```
Device> enable
Device# configure terminal
Device(config)# dial-peer voice 24 voip
Device(config-dial-peer)# destination-pattern 595959
Device(config-dial-peer)# session protocol sipv2
Device(config-dial-peer)# session target ipv4:10.42.29.7
Device(config-dial-peer)# codec mp4a-latm profile 5
Device(config-dial-peer)# end
```

Example: Configuring the MP4A-LATM Codec under Voice Class Codec

```
Device> enable
Device# configure terminal
Device(config)# voice class codec 1
Device(config-class)# codec preference 1 mp4a-latm profile 5
Device(config-class)# exit
Device(config)# dial-peer voice 24 voip
Device(config-dial-peer)# destination-pattern 595959
Device(config-dial-peer)# session protocol sipv2
Device(config-dial-peer)# session target ipv4:10.42.29.7
Device(config-dial-peer)# voice-class codec 1
```

Feature Information for AAC-LD MP4A-LATM Codec Support on Cisco UBE

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 9: Feature Information for AAC-LD MP4A-LATM Codec Support on Cisco UBE

Feature Name	Releases	Feature Information
AAC-LD MP4A-LATM Codec Support on Cisco UBE	15.4(1)T	The AAC-LD MP4A-LATM codec is a wideband audio codec used by video endpoints. MP4A-LATM is an MPEG4 audio coding standard, where LATM is Low-Overhead MPEG-4 Audio Transport Multiplex. The Cisco Unified Border Element (Cisco UBE) supports MP4A-LATM to enable call flows involving endpoints that use this codec, especially for media recording. The following commands were introduced or modified: **codec mp4a-latm**, **codec preference** *tag* **mp4a-latm**

CHAPTER 9

Multicast Music-on-Hold Support on Cisco UBE

First Published: July 22, 2011

Last Updated: July 22, 2011

The Multicast Music-on-Hold (MMOH) feature enables you to subscribe to a music streaming service when you are using a Cisco Unified Border Element. Music streams from an MMOH server to the interface of Cisco UBE, which then converts it into unicast. To play the MMOH to customers using Cisco UBE, you must enable the MMOH feature on Cisco UBE.

Prerequisites for Multicast Music-on-Hold Support on Cisco UBE

Cisco Unified Border Element

- Cisco IOS Release 15.2(1)T or a later release must be installed and running on your Cisco Unified Border Element.

Restrictions for Multicast Music-on-Hold Support on Cisco UBE

- The Multicast Music-on-Hold (MMOH) feature will not work when the Session Description Protocol (SDP) Passthrough feature is enabled on Cisco UBE.

- The MMOH feature will work for Low Density Transcoded calls but not for High Density Transcoded calls.
- MMOH is supported only on SIP-to-SIP call flows on Cisco UBE.
- MMOH with RTCP is not supported.
- MMOH is not supported for SRTP trunk.
- MMOH with media flow-around is not supported.

Information About Multicast Music-on-Hold Support onCisco UBE

Multicast Music-on-Hold

To play Multicast Music-on-Hold (MMOH) to customers using Cisco UBE, you must enable the MMOH feature on Cisco UBE. When Cisco UBE receives an MMOH call, it converts the multicast address received on the inbound leg into a unicast address and sends the address on the outbound leg.

Cisco UBE uses preconfigured CLIs to "listen" for Real-Time Transport Protocol (RTP) packets that are broadcast from an MMOH server in the network and converts them to unicast. When a call is placed on hold, the MOH server streams the RTP packets to the Cisco UBE interface. This interface converts the RTP packets to unicast and relays the packets to the appropriate voice interfaces that have been placed on hold.

Note MMOH is already supported on SIP-TDM gateways.

How to Enable Multicast Music-on-Hold on Cisco UBE

Enabling MMOH on Cisco UBE

Perform this task to enable the MMOH feature on Cisco UBE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip multicast-routing distributed**
4. **interface gigabitethernet** *router-shelf/slot/port*
5. **ip address** *ip-address subnet-mask*
6. **ip pim dense-mode**
7. **negotiation auto**
8. **exit**
9. **ccm-manager music-on-hold**
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing distributed** **Example:** `Device(config)# ip multicast-routing distributed`	Enables distributed IP multicast routing.
Step 4	**interface gigabitethernet** *router-shelf/slot/port* **Example:** `Device(config)# interface gigabitethernet 0/0/0`	Configures a Gigabit Ethernet interface and enters interface configuration mode.
Step 5	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 9.40.1.140 255.255.0.0`	Configures the IP address and the subnet mask on the interface.

	Command or Action	Purpose
Step 6	**ip pim dense-mode** **Example:** `Device(config-if)# ip pim dense-mode`	Enables protocol-independent multicast (PIM) dense-mode operation.
Step 7	**negotiation auto** **Example:** `Device(config-if)# negotiation auto`	Performs link auto-negotiation.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 9	**ccm-manager music-on-hold** **Example:** `Device(config)# ccm-manager music-on-hold`	Enables the multicast music-on-hold feature on a voice gateway.
Step 10	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.

Verifying the MMOH Support on Cisco UBE

Perform this task to verify the MMOH support on Cisco UBE. The **show** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show ccm-manager music-on-hold**
3. **show voip rtp connections**
4. **show call active voice compact**
5. **show platform hardware qfp active feature sbc mmoh global**
6. **show platform hardware qfp active feature sbc mmoh group**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show ccm-manager music-on-hold**

Displays information about all the multicast music-on-hold (MOH) sessions in the gateway at any given time.

Example:

```
Device# show ccm-manager music-on-hold
Current active multicast sessions: 1
Multicast Address   RTP port number    Packets in/out    CallId   Codec    Incoming Interface
239.1.1.1           16386              614/614           132     g711ulaw
Gi0/0
```

Step 3 **show voip rtp connections**

Displays RTP-named event packets.

Example:

```
Device# show voip rtp connections

VoIP RTP Port Usage Information:
Max Ports Available: 20000, Ports Reserved: 101, Ports in Use: 2
Port range not configured, Min: 8000, Max: 48200
 Ports       Ports       Ports
Media-Address Range                      Available   Reserved    In-use
Default Address-Range                    20000       101         2

VoIP RTP active connections:
No. CallId     dstCallId       LocalRTP RmtRTP     LocalIP                                 RemoteIP
1     140        141            18792    18638     9.42.30.10                             9.42.30.32
2     141        140           19256    26184     9.42.30.10                            9.42.30.189
Found 2 active RTP sessions
```

Step 4 **show call active voice compact**

Displays a compact version of voice calls in progress.

Example:

```
Device# show call active voice compact
<callID>  A/O FAX T<sec> Codec       type        Peer Address       IP R<ip>:<udp>
Total call-legs: 3
       140 ANS     T644   g711ulaw    VOIP        P10000       9.42.30.32:18638
       141 ORG     T644   g711ulaw    VOIP        P708090      9.42.30.189:26184
       145 ORG     T643   g711ulaw    VOIP        P595959      9.42.29.7:3852
```

Step 5 **show platform hardware qfp active feature sbc mmoh global**

Displays SBC multicast Music-on-Hold global statistics.

Example:

```
Device# show platform hardware qfp active feature sbc mmoh global

SBC multicast Music-on-Hold Global Statistics
-------------------------------------------------------------------
 Total MMOH groups                           = 1
 Total RTP packets received                  = 6311
 Total RTP octects received                  = 1262200
 Total RTP packets replicated                = 6311
 Total RTP octects replicated                = 1262200
 Total RTP packets dropped                   = 0
 Total RTP octects dropped                   = 0
```

Step 6 **show platform hardware qfp active feature sbc mmoh group**
Displays SBC multicast Music-on-Hold group structure.

Example:

```
Device# show platform hardware qfp active feature sbc mmoh group

SBC multicast Music-on-Hold group structure:
---------------------------------------
 VRF                                   = 0
 IP                                    = 239.1.1.1
 Port                                  = 16384
 Protocol                              = 1
 Calls in group                        = 1

SBC MMOH group Statistics
---------------------------------------
  Total RTP packets received                  = 406
  Total RTP octects received                  = 81200
  Total RTP packets replicated                = 406
  Total RTP octects replicated                = 81200
  Total RTP packets dropped                   = 0
  Total RTP octects dropped                   = 0
```

Troubleshooting Tips

The following commands can help troubleshoot MMOH:

- **debug ccm-manager music-on-hold [all | errors | events]**
- **debug voip rtp**
- **debug ccsip all**

Configuration Examples for Multicast Music-on-Hold Support on Cisco UBE

Example: Enabling MMOH on Cisco UBE

```
Device> enable
Device# configure terminal
Device(config)# ip multicast-routing distributed
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# ip address 9.40.1.140 255.255.0.0
Device(config-if)# ip pim dense-mode
Device(config-if)# negotiation auto
Device(config-if)# exit
Device(config)# ccm-manager music-on-hold
Device# show running-config
Building configuration...
Current configuration : 2375 bytes
!
! Last configuration change at 11:01:36 UTC Wed Jan 5 2011
!
version 15.1
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname carbon-1
!
boot-start-marker
boot system flash usbflash0:c2951-universalk9-mz.SSA.MMOH-carbon_dev
boot-end-marker
!
!
!
no aaa new-model
!
no ipv6 cef
ip source-route
ip cef
!
!
!
ip multicast-routing
!
!
no ip domain lookup
multilink bundle-name authenticated
!
!
!
!
!
crypto pki token default removal timeout 0
!
!
voice-card 0
!
!
!
voice service voip
 mode border-element license capacity 1200
 allow-connections sip to sip
 sip
!
```

```
!
!
!
!
license udi pid CISCO2951/K9 sn FHK1433F39H
hw-module pvdm 0/0
!
!
!
!
redundancy inter-device
!
!
redundancy
!
!
!
!
!
!
interface GigabitEthernet0/0
 ip address 9.42.30.12 255.255.0.0
 duplex auto
 speed auto
!
interface GigabitEthernet0/1
no ip address
 shutdown
 duplex auto
 speed auto
!
interface GigabitEthernet0/2
 no ip address
 shutdown
 duplex auto
 speed auto
!
ip forward-protocol nd
!
no ip http server
no ip http secure-server
!
ip route 0.0.0.0 0.0.0.0 9.42.0.1
!
!
nls resp-timeout 1
cpd cr-id 1
!
!
control-plane
!
!
ccm-manager music-on-hold
!
!
mgcp profile default
!
!
dial-peer voice 100 voip
 destination-pattern 878767
 session protocol sipv2
 session target ipv4:9.42.30.5
 codec g711ulaw
!
gatekeeper
 shutdown
!
!
!
line con 0
 speed 115200
line aux 0
line vty 0 4
```

```
 login
 transport input all
!
exception data-corruption buffer truncate
scheduler allocate 20000 1000
end
```

Feature Information for Multicast Music-on-Hold Support on Cisco UBE

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Table 10: Feature Information for Multicast Music-on-Hold Support on Cisco UBE

Feature Name	Releases	Feature Information
Multicast Music-on-Hold Support on Cisco UBE	15.2(1)T Cisco IOS XE Release 3.11S	The Multicast Music-on-Hold (MMOH) feature enables you to subscribe to a music streaming service when you are using a Cisco Unified Border Element. To play MMOH to customers using Cisco UBE, you must enable the MMOH feature on Cisco UBE. No new commands were introduced or modified.

CHAPTER 10

Network-Based Recording

The Network-Based Recording feature supports software-based forking for Real-time Transport Protocol (RTP) streams. Media forking provides the ability to create midcall multiple streams (or branches) of audio and video associated with a single call and then send the streams of data to different destinations. To enable network-based recording using Cisco Unified Border Element (CUBE), you can configure specific commands or use a call agent. CUBE acts as a recording client and MediaSense Session Initiation Protocol (SIP) recorder acts a recording server.

Feature Information for Network-Based Recording

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11: Feature Information for Network-Based Recording

Feature Name	Releases	Feature Information
Audio-only Stream Forking of Video Call	Cisco IOS 15.4(3)M Cisco IOS XE 3.13S	The Audio-only Stream Forking of Video Call feature supports CUBE-based forking and recording of only audio calls in a call that includes both audio and video. The following commands were introduced: **media-type audio**.
Network-Based Recording of Video Calls Using CUBE	Cisco IOS 15.3(3)M Cisco IOS XE 3.10S	The Network-Based Recording of Video Calls using CUBE feature supports forking and recording of video calls.
Network-Based Recording of Audio Calls Using CUBE	Cisco IOS 15.2(1)T Cisco IOS XE 3.8S	The Network-Based Recording of Audio Calls using CUBE feature supports forking for RTP streams. The following commands were introduced or modified: **media class**, **media profile recorder**, **media-recording**, **recorder parameter**, **recorder profile**, **show voip recmsp session**.

Restrictions for Network-Based Recording

- Network-based recording is not supported for the following calls:
    - Calls that do not use Session Initiation Protocol (SIP). Must be a SIP-to-SIP call flow
    - Flow-around calls
    - Session Description Protocol (SDP) pass-through calls
    - Real-time Transport Protocol (RTP) loopback calls
    - High-density transcoder calls
    - IPv6-to-IPv6 calls
    - IPv6-to-IPv4 calls with IPv4 endpoint.
    - Secure Real-time Transport Protocol (SRTP) passthrough calls
    - SRTP-RTP calls with forking for SRTP leg (forking is supported for the RTP leg)
    - Resource Reservation Protocol (RSVP)
    - Multicast music on hold (MOH)

- Any media service parameter change via Re-INVITE or UPDATE from Recording server is not supported Midcall renegotiation and supplementary services can be done through the primary call only.
- Media service parameter change via Re-INVITE or UPDATE message from the recording server is not supported
- Recording is not supported if CUBE is running a TCL IVR application.
- Media mixing on forked streams is not supported

Restrictions for Video Recording

- If the main call has multiple video streams (m-lines), the video streams other than the first video m-line are not forked.
- Application media streams of the primary call are not forked to the recording server.
- Forking is not supported if the anchor leg or recording server is on IPv6.
- High availability is not supported on forked video calls.

Information About Network-Based Recording Using CUBE

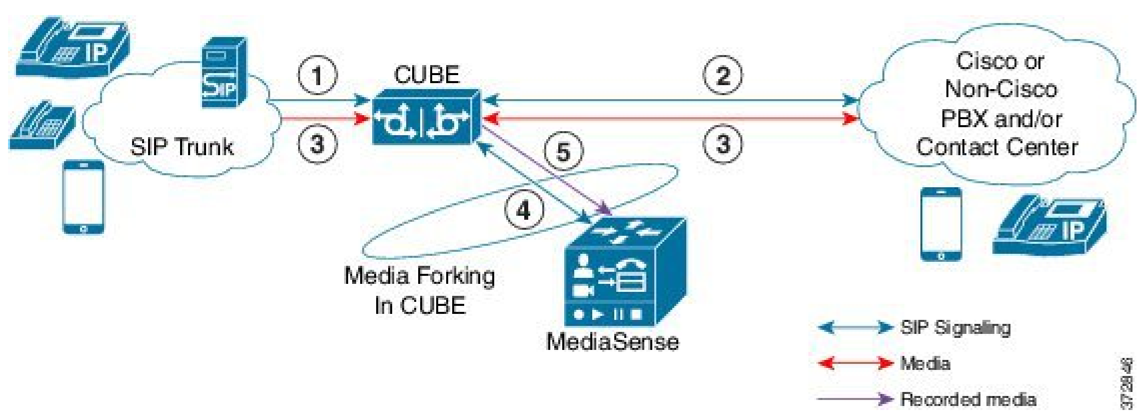
Deployment Scenarios for CUBE-based Recording

CUBE as a recording client has the following functions:

- Acts as a SIP user agent and sets up a recording session (SIP dialog) with the recording server.
- Acts as the source of the recorded media and forwards the recorded media to the recording server.
- Sends information to a server that helps the recording server associate the call with media streams and identifies the participants of the call. This information sent to the recording server is called metadata.

Given below is a typical deployment scenario of a CUBE-based recording solution. The information flow is described below:

Figure 1: Deployment Scenario for CUBE-based Recording Solution



1 Incoming call from SIP trunk.
2 Outbound call to a Contact Centre
3 Media between endpoints flowthrough CUBE
4 CUBE sets up a new SIP session with MediaSense based on policy.
5 CUBE forks RTP media to MediaSense. For an audio call, audio is forked. For a video call, both audio and video are .forked. For an audio-only configuration in a audio-video call, only audio is forked. There will be two or four m-lines to the recording server, based on the type of recording

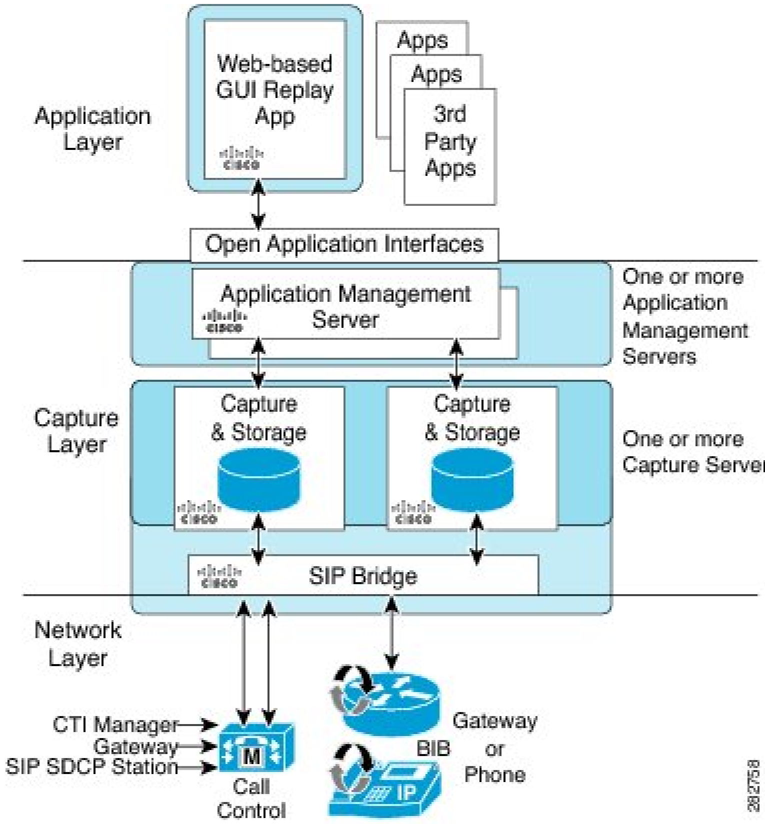
The metadata carried in the SIP session between the recording client and the recording server is to:

- Carry the communication session data that describes the call.
- Send the metadata to the recording server. The recording server uses the metadata to associate communication sessions involving two or more participants with media streams.

The call leg that is created between the recording client and the recording server is known as the recording session.

Open Recording Architecture

The Open Recording Architecture (ORA) comprises of elements, such as application management server and SIP bridge, to support IP-based recording. The ORA IP enables recording by solving topology issues, which accelerates the adoption of Cisco unified communication solutions.



Following are the three layers of the ORA architecture:

Network Layer

The ORA network layer is comprises call control systems, media sources, and IP foundation components, such as routers and switches.

Capture and Media Processing Layer

The ORA capture and media processing layer includes core functions of ORA—terminating media streams, storage of media and metadata, and speech analytics that can provide real-time events for applications.

Application Layer

The ORA application layer supports in-call and post-call applications through open programming interfaces.

In-call applications include applications that make real-time business decisions (for example, whether to record a particular call or not), control pause and resume from Interactive Voice Response (IVR) or agent desktop systems, and perform metadata tagging and encryption key exchange at the call setup.

Post-call applications include the following:

- Traditional compliance search, replay, and quality monitoring.
- Advanced capabilities, such as speech analytics, transcription, and phonetic search.

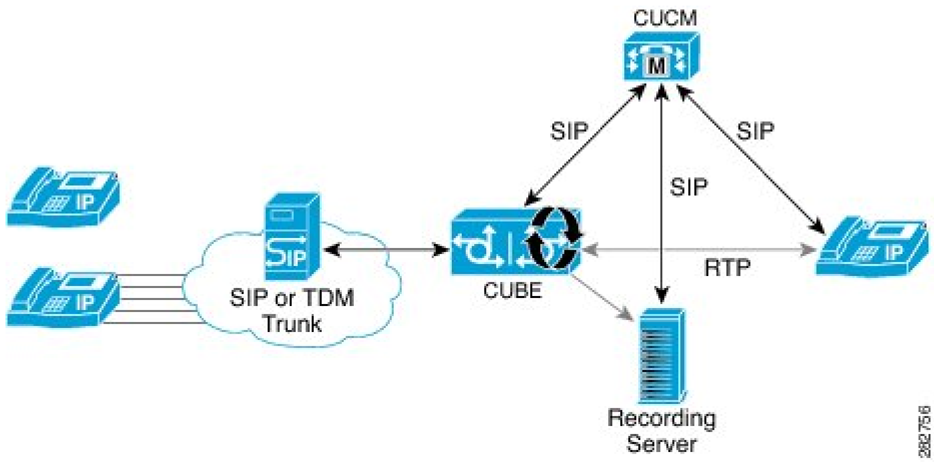
- Custom enterprise integration.
- Enterprise-wide policy management.

Media Forking Topologies

The following topologies support media forking:

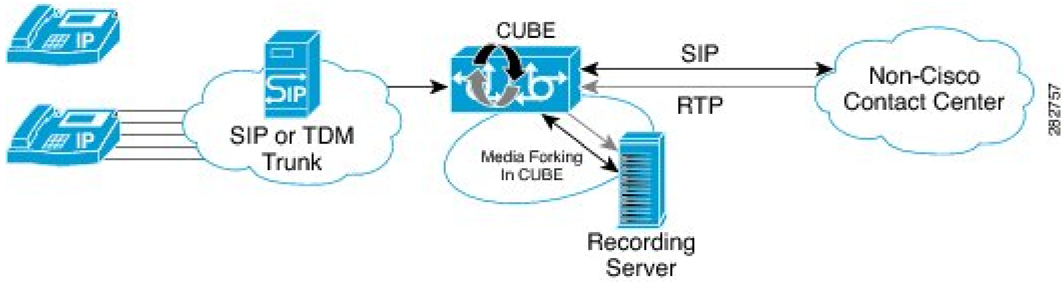
Media Forking with Cisco UCM

The figure below illustrates media forking with Cisco Unified CallManager (Cisco UCM) topology. This topology supports replication of media packets to allow recording by the caller agent. It also enables CUBE to establish full-duplex communication with the recording server. In this topology, SIP recording trunk is enhanced to have additional call metadata.



Media Forking without Cisco UCM

The topology below shows media forking without the Cisco UCM topology. This topology supports static configuration on CUBE and the replication of media packets to allow recording caller-agent and full-duplex interactions at an IP call recording server.



SIP Recorder Interface

SIP is used as a protocol between CUBE and the MediaSense SIP server. Extensions are made to SIP to carry the recording session information needed for the recording server. This information carried in SIP sessions between the recording client and the recording server is called metadata.

Metadata

Metadata is the information that is passed by the recording client to the recording server in a SIP session. Metadata describes the communication session and its media streams.

Metadata is used by the recording server to:

- Identify participants of the call.
- Associate media streams with the participant information. Each participant can have one or more media streams, such as audio and video.
- Identify the participant change due to transfers during the call.

The recording server uses the metadata information along with other SIP message information, such as dialog ID and time and date header, to derive a unique key. The recording server uses this key to store media streams and associate the participant information with the media streams.

How to Configure Network-Based Recording

Configuring Network-Based Recording (with Media Profile Recorder)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile recorder** *profile-tag*
4. (Optional) **media-type audio**
5. **media-recording** *dial-peer-tag* [*dial-peer-tag2...dial-peer-tag5*]
6. **exit**
7. **media class** *tag*
8. **recorder profile** *tag*
9. **exit**
10. **dial-peer voice** *dummy-recorder-dial-peer-tag* **voip**
11. **media-class** *tag*
12. **destination-pattern** [+] *string* [**T**]
13. **session protocol sipv2**
14. **session target ipv4:**[*recording-server-destination-address* | *recording-server-dns*]
15. **session transport tcp**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile recorder** *profile-tag* **Example:** `Device(config)# media profile recorder 100`	Configures the media profile recorder and enters media profile configuration mode.

	Command or Action	Purpose
Step 4	**media-type audio** **Example:** `Device(cfg-mediaprofile)# media-type audio`	(Optional) Configures recording of audio only in a call with both audio and video. If this configuration is not done, both audio and video are recorded.
Step 5	**media-recording** *dial-peer-tag* [*dial-peer-tag2...dial-peer-tag5*] **Example:** `Device(cfg-mediaprofile)# media-recording 8000 8001 8002`	Configures the dial-peers that need to be configured **Note** You can specify a maximum of five dial-peer tags.
Step 6	**exit** **Example:** `Device(cfg-mediaprofile)# exit`	Exits media profile configuration mode.
Step 7	**media class** *tag* **Example:** `Device(config)# media class 100`	Configures a media class and enters media class configuration mode.
Step 8	**recorder profile** *tag* **Example:** `Device(cfg-mediaclass)# recorder profile 100`	Configures the media profile recorder.
Step 9	**exit** **Example:** `Device(cfg-mediaclass)# exit`	Exits media class configuration mode.
Step 10	**dial-peer voice** *dummy-recorder-dial-peer-tag* **voip** **Example:** `Device(config)# dial-peer voice 8000 voip`	Configures a recorder dial peer and enters dial peer voice configuration mode.
Step 11	**media-class** *tag* **Example:** `Device(config-dial-peer)# media-class 100`	Configures media class on a dial peer.
Step 12	**destination-pattern** [+] *string* [**T**] **Example:** `Device(config-dial-peer)# destination-pattern 595959`	Specifies either the prefix or the full E.164 telephone number (depending on your dial plan) to be used for a dial peer.

	Command or Action	Purpose
Step 13	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Configures the VoIP dial peer to use Session Initiation Protocol (SIP).
Step 14	**session target ipv4:**[*recording-server-destination-address* | *recording-server-dns*] **Example:** `Device(config-dial-peer)# session target ipv4:10.42.29.7`	Specifies a network-specific address for a dial peer. Keyword and argument are as follows: • **ipv4:** *destination address* --IP address of the dial peer, in this format: *xxx.xxx.xxx.xxx*
Step 15	**session transport tcp** **Example:** `Device(config-dial-peer)# session transport tcp`	Configures a VoIP dial peer to use Transmission Control Protocol (TCP).
Step 16	**end** **Example:** `Device(config-dial-peer)# end`	Returns to privileged EXEC mode.

Configuring Network-Based Recording (without Media Profile Recorder)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media class** *tag*
4. **recorder parameter**
5. (Optional) **media-type audio**
6. **media-recording** *dial-peer-tag*
7. **exit**
8. **exit**
9. **dial-peer voice** *dummy-recorder-dial-peer-tag* **voip**
10. **media-class** *tag*
11. **destination-pattern** [+] *string* [**T**]
12. **session protocol sipv2**
13. **session target ipv4:**[*recording-server-destination-address* | *recording-server-dns*]
14. **session transport tcp**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media class** *tag* **Example:** `Device(config)# media class 100`	Configures the media class and enters media class configuration mode.
Step 4	**recorder parameter** **Example:** `Device(cfg-mediaclass)# recorder parameter`	Enters media class recorder parameter configuration mode to enable you to configure recorder-specific parameters.

	Command or Action	Purpose
Step 5	**media-type audio** **Example:** `Device(cfg-mediaprofile)# media-type audio`	(Optional) Configures recording of audio only in a call with both audio and video. **Note** If this configuration is not done, both audio and video are recorded.
Step 6	**media-recording** *dial-peer-tag* **Example:** `Device(cfg-mediaclass-recorder)# media-recording 8000, 8001, 8002`	Configures voice-class recording parameters. **Note** You can specify a maximum of five dial-peer tags.
Step 7	**exit** **Example:** `Device(cfg-mediaclass-recorder)# exit`	Exits media class recorder parameter configuration mode.
Step 8	**exit** **Example:** `Device(cfg-mediaclass)# exit`	Exits media class configuration mode.
Step 9	**dial-peer voice** *dummy-recorder-dial-peer-tag* **voip** **Example:** `Device(config)# dial-peer voice 8000 voip`	Configures a recorder dial peer and enters dial peer voice configuration mode.
Step 10	**media-class** *tag* **Example:** `Device(config-dial-peer)# media-class 100`	Configures media class on a dial peer.
Step 11	**destination-pattern** [+] *string* [**T**] **Example:** `Device(config-dial-peer)# destination-pattern 595959`	Specifies either the prefix or the full E.164 telephone number (depending on your dial plan) to be used for a dial peer. Keywords and arguments are as follows:
Step 12	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Configures the VoIP dial peer to use Session Initiation Protocol (SIP).
Step 13	**session target ipv4:**[*recording-server-destination-address* | *recording-server-dns*]	Specifies a network-specific address for a dial peer. Keyword and argument are as follows:

	Command or Action	Purpose
	Example: `Device(config-dial-peer)# session target ipv4:10.42.29.7`	• **ipv4:** *destination address* --IP address of the dial peer, in this format: *xxx.xxx.xxx.xxx*
Step 14	**session transport tcp** **Example:** `Device(config-dial-peer)# session transport tcp`	Configures a VoIP dial peer to use Transmission Control Protocol (TCP).
Step 15	**end** **Example:** `Device(config-dial-peer)# end`	Returns to privileged EXEC mode.

Verifying the Network-Based Recording Using CUBE

Perform this task to verify the configuration of the Network-Based Recording Using CUBE. The **show** and **debug** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show voip rtp connections**
3. **show voip recmsp session**
4. **show voip recmsp session detail call-id** *call-id*
5. **show voip rtp forking**
6. **show call active voice compact**
7. **show call active video compact**
8. **show sip-ua calls**
9. **show call active video brief**
10. **debug ccsip messages** (for audio calls)
11. **debug ccsip messages** (for video calls)
12. **debug ccsip messages** (for audio-only recording in a call with both audio and video)
13. Enter one of the following:
    - **debug ccsip all**
    - **debug voip recmsp all**
    - **debug voip ccapi all**
    - **debug voip fpi all** (for ASR devices only)

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show voip rtp connections**

Displays Real-Time Transport Protocol (RTP) connections. Two extra connections are displayed for forked legs.

Example:

```
Device# show voip rtp connections

VoIP RTP Port Usage Information:
Max Ports Available: 8091, Ports Reserved: 101, Ports in Use: 8
Port range not configured, Min: 16384, Max: 32767

                                                Ports       Ports       Ports
Media-Address Range                             Available   Reserved    In-use

Default Address-Range                           8091        101         8

VoIP RTP active connections :
No. CallId     dstCallId  LocalRTP RmtRTP LocalIP                                       RemoteIP

1     1          2          16384    20918  10.104.45.191                               10.104.8.94

2     2          1          16386    17412  10.104.45.191                               10.104.8.98

3     3          4          16388    29652  10.104.45.191                               10.104.8.98

4     4          3          16390    20036  10.104.45.191                               10.104.8.94

5     6          5          16392    58368  10.104.45.191                               10.104.105.232

6     7          5          16394    53828  10.104.45.191                               10.104.105.232

7     8          5          16396    39318  10.104.45.191                               10.104.105.232

8     9          5          16398    41114  10.104.45.191                               10.104.105.232

Found 8 active RTP connections
```

Step 3 **show voip recmsp session**

Displays active recording Media Service Provider (MSP) session information internal to CUBE.

Example:

```
Device# show voip recmsp session

RECMSP active sessions:
MSP Call-ID              AnchorLeg Call-ID        ForkedLeg Call-ID
143                      141                              145
Found 1 active sessions
```

Step 4 **show voip recmsp session detail call-id** *call-id*

Displays detailed information about the recording MSP Call ID.

Example:

```
Device# show voip recmsp session detail call-id 145
RECMSP active sessions:
Detailed Information
=========================
Recording MSP Leg Details:
Call ID: 143
GUID : 7C5946D38ECD

AnchorLeg Details:
Call ID: 141
Forking Stream type: voice-nearend
Participant: 708090

Non-anchor Leg Details:
Call ID: 140
Forking Stream type: voice-farend
Participant: 10000

Forked Leg Details:
Call ID: 145
Near End Stream CallID 145
Stream State ACTIVE
Far End stream CallID 146
Stream State ACTIVE
Found 1 active sessions
Device# show voip recmsp session detail call-id 5

RECMSP active sessions:
Detailed Information
=========================
Recording MSP Leg Details:
Call ID: 5
GUID : 1E01B6000000

AnchorLeg Details:
Call ID: 1
Forking Stream type: voice-nearend
Forking Stream type: video-nearend
Participant: 1777

Non-anchor Leg Details:
Call ID: 2
Forking Stream type: voice-farend
Forking Stream type: video-farend
Participant: 1888

Forked Leg Details:
Call ID: 6
Voice Near End Stream CallID 6
Stream State ACTIVE
Voice Far End stream CallID 7
Stream State ACTIVE
Video Near End stream CallID 8
Stream State ACTIVE
Video Far End stream CallID 9
Stream State ACTIVE
Found 1 active sessions
```

Output Field	Description
Stream State	Displays the state of the call. This can be ACTIVE or HOLD.

Output Field	Description
Msp Call-Id	Displays an internal Media service provider call ID and forking related statistics for an active forked call.
Anchor Leg Call-id	Displays an internal anchor leg ID, which is the dial peer where forking enabled. The output displays the participant number and stream type. Stream type voice-near end indicates the called party side.
Non-Anchor Call-id	Displays an internal non-anchor leg ID, which is the dial peer where forking is not enabled. The output displays the participant number and stream type. Stream type voice-near end indicates the called party side.
Forked Call-id	This forking leg call-id will show near-end and far-end stream call-id details with state of the Stream . Displays an internal foked leg ID. The output displays near-end and far-end details of a stream.

Step 5 **show voip rtp forking**
Displays RTP media-forking connections.

Example:

```
Device# show voip rtp forking
VoIP RTP active forks :
 Fork 1
   stream type voice-only (0): count 0
   stream type voice+dtmf (1): count 0
   stream type dtmf-only (2): count 0
   stream type voice-nearend (3): count 1
     remote ip 10.42.29.7,  remote port 38526,  local port 18648
       codec g711ulaw,  logical ssrc 0x53
       packets sent 29687,  packets received 0
   stream type voice+dtmf-nearend (4): count 0
   stream type voice-farend (5): count 1
     remote ip 10.42.29.7,  remote port 50482,  local port 17780
       codec g711ulaw,  logical ssrc 0x55
       packets sent 29686,  packets received 0
   stream type voice+dtmf-farend (6): count 0
   stream type video (7): count
```

Output Field	Description
remote ip 10.42.29.7, remote port 38526, local port 18648	Recording server IP, recording server port, and local CUBE device port where data for stream 1 was first sent from.
remote ip 10.42.29.7, remote port 50482, local port 17780	Recording server IP, recording server port, and local CUBE device port where data for stream 2 was first sent from.
packets sent 29686	Number of packets sent to the recorder

Output Field	Description
codec g711ulaw	Codec negotiated for the recording leg.

Step 6 **show call active voice compact**

Displays a compact version of voice calls in progress. An additional call leg is displayed for media forking.

Example:

```
Device# show call active voice compact
<callID>  A/O FAX T<sec> Codec       type        Peer Address       IP R<ip>:<udp>
Total call-legs: 3
       140 ANS     T644   g711ulaw    VOIP        P10000       10.42.30.32:18638
       141 ORG     T644   g711ulaw    VOIP        P708090      10.42.30.189:26184
       145 ORG     T643   g711ulaw    VOIP        P595959      10.42.29.7:38526
```

Step 7 **show call active video compact**

Displays a compact version of video calls in progress.

Example:

```
Device# show call active video compact

<callID>  A/O FAX T<sec> Codec       type        Peer Address       IP R<ip>:<udp>
Total call-legs: 3
         1 ANS     T14    H264        VOIP-VIDEO  P1777      10.104.8.94:20036
         2 ORG     T14    H264        VOIP-VIDEO  P1888      10.104.8.98:29652
         6 ORG     T13    H264        VOIP-VIDEO  P1234   10.104.105.232:39318
```

Step 8 **show sip-ua calls**

Displays active user agent client (UAC) and user agent server (UAS) information on SIP calls.

Example:

```
Device# show sip-ua calls
Total SIP call legs:3, User Agent Client:2, User Agent Server:1
SIP UAC CALL INFO
Call 1
SIP Call ID                : 99EA5118-506211E0-80C6E01B-4C27AA62@10.42.30.10
   State of the call       : STATE_ACTIVE (7)
   Substate of the call    : SUBSTATE_NONE (0)
   Calling Number          : 10000
   Called Number           : 708090
   Bit Flags               : 0xC04018 0x10000100 0x80
   CC Call ID              : 141
   Source IP Address (Sig ): 10.42.30.10
   Destn SIP Req Addr:Port : [10.42.30.5]:5060
   Destn SIP Resp Addr:Port: [10.42.30.5]:5060
   Destination Name        : 10.42.30.5
   Number of Media Streams : 1
   Number of Active Streams: 1
   RTP Fork Object         : 0x0
   Media Mode              : flow-through
   Media Stream 1
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 141
     Stream Type              : voice+dtmf (1)
     Stream Media Addr Type   : 1
     Negotiated Codec         : g711ulaw (160 bytes)
     Codec Payload Type       : 0
     Negotiated Dtmf-relay    : rtp-nte
```

```
      Dtmf-relay Payload Type  : 101
      QoS ID                   : -1
      Local QoS Strength       : BestEffort
      Negotiated QoS Strength  : BestEffort
      Negotiated QoS Direction : None
      Local QoS Status         : None
      Media Source IP Addr:Port: [10.42.30.10]:19256
      Media Dest IP Addr:Port  : [10.42.30.189]:26184
Options-Ping    ENABLED:NO    ACTIVE:NO
Call 2
SIP Call ID                : 9A6D8922-506211E0-80CEE01B-4C27AA62@10.42.30.10
   State of the call       : STATE_ACTIVE (7)
   Substate of the call    : SUBSTATE_NONE (0)
   Calling Number          :
   Called Number           : 595959                              Recoding server number
   Bit Flags               : 0xC04018 0x10800100 0x80
   CC Call ID              : 145
   Source IP Address (Sig ): 10.42.30.10
   Destn SIP Req Addr:Port : [10.42.29.7]:5060
   Destn SIP Resp Addr:Port: [10.42.29.7]:5060
   Destination Name        : 10.42.29.7
   Number of Media Streams : 2
   Number of Active Streams: 2
   RTP Fork Object         : 0x0
   Media Mode              : flow-through
   Media Stream 1
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 145
     Stream Type              : voice-nearend (3)
     Stream Media Addr Type   : 1
     Negotiated Codec         : g711ulaw (160 bytes)
     Codec Payload Type       : 0
     Negotiated Dtmf-relay    : inband-voice
     Dtmf-relay Payload Type  : 0
     QoS ID                   : -1
     Local QoS Strength       : BestEffort
     Negotiated QoS Strength  : BestEffort
     Negotiated QoS Direction : None
     Local QoS Status         : None
     Media Source IP Addr:Port: [10.42.30.10]:18648
     Media Dest IP Addr:Port  : [10.42.29.7]:38526
   Media Stream 2
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 146
     Stream Type              : voice-farend (5)
     Stream Media Addr Type   : 1
     Negotiated Codec         : g711ulaw (160 bytes)
     Codec Payload Type       : 0
     Negotiated Dtmf-relay    : inband-voice
     Dtmf-relay Payload Type  : 0
     QoS ID                   : -1
     Local QoS Strength       : BestEffort
     Negotiated QoS Strength  : BestEffort
     Negotiated QoS Direction : None
     Local QoS Status         : None
     Media Source IP Addr:Port: [10.42.30.10]:17780
     Media Dest IP Addr:Port  : [10.42.29.7]:50482
Options-Ping    ENABLED:NO    ACTIVE:NO
   Number of SIP User Agent Client(UAC) calls: 2
SIP UAS CALL INFO
Call 1
SIP Call ID                : 7CF44DF3-506611E0-8ED2B9D4-CA68C314@10.42.30.32
   State of the call       : STATE_ACTIVE (7)
   Substate of the call    : SUBSTATE_NONE (0)
   Calling Number          : 10000
   Called Number           : 708090
   Bit Flags               : 0x8C4401C 0x10000100 0x4
   CC Call ID              : 140
   Source IP Address (Sig ): 10.42.30.10
   Destn SIP Req Addr:Port : [10.42.30.32]:5060
   Destn SIP Resp Addr:Port: [10.42.30.32]:52757
```

```
   Destination Name        : 10.42.30.32
   Number of Media Streams : 1
   Number of Active Streams: 1
   RTP Fork Object         : 0x0
   Media Mode              : flow-through
   Media Stream 1
     State of the stream      : STREAM_ACTIVE
     Stream Call ID           : 140
     Stream Type              : voice+dtmf (0)
     Stream Media Addr Type   : 1
     Negotiated Codec         : g711ulaw (160 bytes)
     Codec Payload Type       : 0
     Negotiated Dtmf-relay    : rtp-nte
     Dtmf-relay Payload Type  : 101
     QoS ID                   : -1
     Local QoS Strength       : BestEffort
     Negotiated QoS Strength  : BestEffort
     Negotiated QoS Direction : None
     Local QoS Status         : None
     Media Source IP Addr:Port: [10.42.30.10]:18792
     Media Dest IP Addr:Port  : [10.42.30.32]:18638
Options-Ping    ENABLED:NO    ACTIVE:NO
   Number of SIP User Agent Server(UAS) calls: 1
```

Step 9 **show call active video brief**

Displays a truncated version of video calls in progress.

Example:

```
Device# show call active video brief

Telephony call-legs: 0
SIP call-legs: 3
H323 call-legs: 0
Call agent controlled call-legs: 0
SCCP call-legs: 0
Multicast call-legs: 0
Total call-legs: 3

0    : 1 87424920ms.1 (*12:23:53.573 IST Wed Jul 17 2013) +1050 pid:1 Answer 1777 active
 dur 00:00:46 tx:5250/1857831 rx:5293/1930598 dscp:0 media:0 audio tos:0xB8 video tos:0x88
 IP 10.104.8.94:20036 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms H264 TextRelay: off
Transcoded: No
…
0    : 2 87424930ms.1 (*12:23:53.583 IST Wed Jul 17 2013) +1040 pid:2 Originate 1888 active
 dur 00:00:46 tx:5293/1930598 rx:5250/1857831 dscp:0 media:0 audio tos:0xB8 video tos:0x88
 IP 10.104.8.98:29652 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms H264 TextRelay: off
Transcoded: No
…
0    : 6 87425990ms.1 (*12:23:54.643 IST Wed Jul 17 2013) +680 pid:1234 Originate 1234 active
 dur 00:00:46 tx:10398/3732871 rx:0/0 dscp:0 media:0 audio tos:0xB8 video tos:0x0
 IP 10.104.105.232:39318 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms H264 TextRelay: off
Transcoded: No
…
```

Step 10 **debug ccsip messages** (for audio calls)

```
Sent:
INVITE sip:22222@10.42.29.7:5060 SIP/2.0
Via: SIP/2.0/TCP 10.42.30.10:5060;branch=z9hG4bKB622CF
X-Cisco-Recording-Participant: sip:708090@10.42.30.5;media-index="0"
X-Cisco-Recording-Participant: sip:10000@10.42.30.32;media-index="1"
From: <sip:10.42.30.10>;tag=5096700-1E1A
To: <sip:595959@10.42.29.7>
Date: Fri, 18 Mar 2011 07:01:50 GMT
Call-ID: 6E6CF813-506411E0-80EAE01B-4C27AA62@10.42.30.10
Supported: 100rel,timer,resource-priority,replaces,sdp-anat
Min-SE:  1800
Cisco-Guid: 1334370502-1348997600-2396699092-3395863316
```

```
User-Agent: Cisco-SIPGateway/IOS-15.2(0.0.2)PIA16
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, UPDATE, REFER, SUBSCRIBE, NOTIFY, INFO, REGISTER
CSeq: 101 INVITE
Max-Forwards: 70
Timestamp: 1300431710
Contact: <sip:10.42.30.10:5060;transport=tcp>
Expires: 180
Allow-Events: telephone-event
Content-Type: application/sdp
Content-Disposition: session;handling=required
Content-Length: 449
v=0
o=CiscoSystemsSIP-GW-UserAgent 3021 3526 IN IP4 10.42.30.10
s=SIP Call
c=IN IP4 10.42.30.10
t=0 0
m=audio 24544 RTP/AVP 0 101 19
c=IN IP4 10.42.30.10
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:19 CN/8000
a=ptime:20
a=sendonly
m=audio 31166 RTP/AVP 0 101 19
c=IN IP4 10.42.30.10
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:19 CN/8000
a=ptime:20
a=sendonly
Received:
SIP/2.0 200 Ok
Via: SIP/2.0/TCP 10.104.46.198:5060;branch=z9hG4bK13262B
To: <sip:23232323@10.104.46.201>;tag=ds457251f
From: <sip:10.104.46.198>;tag=110B66-1CBC
Call-ID: 7142FB-9A5011E0-801EF71A-59B4D258@10.104.46.198
CSeq: 101 INVITE
Content-Length: 206
Contact: <sip:23232323@10.104.46.201:5060;transport=tcp>
Content-Type: application/sdp
Allow: INVITE, BYE, CANCEL, ACK, NOTIFY, INFO, UPDATE
Server: Cisco-ORA/8.5
v=0
o=CiscoORA 2187 1 IN IP4 10.104.46.201
s=SIP Call
c=IN IP4 10.104.46.201
t=0 0
m=audio 54100 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=recvonly
m=audio 39674 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=recvonly

Sent:
ACK sip:23232323@10.104.46.201:5060;transport=tcp SIP/2.0
Via: SIP/2.0/TCP 10.104.46.198:5060;branch=z9hG4bK141B87
From: <sip:10.104.46.198>;tag=110B66-1CBC
To: <sip:23232323@10.104.46.201>;tag=ds457251f
Date: Mon, 20 Jun 2011 08:42:01 GMT
Call-ID: 7142FB-9A5011E0-801EF71A-59B4D258@10.104.46.198
Max-Forwards: 70
CSeq: 101 ACK
Allow-Events: telephone-event
Content-Length: 0
```

Output Field	Description
INVITE sip:22222@10.42.29.7:5060 SIP/2.0	22222 is the destination pattern or the number of recording server and is configured under the recorder dial peer.
X-Cisco-Recording-Participant: sip:708090@10.42.30.5;media-index="0"	Cisco proprietary header with originating and terminating participant number and IP address used to communicate to the recording server
Cisco-Guid: 1334370502-1348997600-2396699092-3395863316	GUID is the same for the primary call and forked call .
m=audio 24544 RTP/AVP 0 101 19	First m-line of participant with payload type and codec information .
m=audio 31166 RTP/AVP 0 101 19	Second m- line of another participant with codec info and payload type.
a=sendonly	CUBE is always in send only mode towards Recording server.
a=recvonly	Recording server is in receive mode only.

Step 11 **debug ccsip messages** (for video calls)

```
Sent: INVITE sip:575757@9.45.38.39:7686 SIP/2.0

.
.
Via: SIP/2.0/UDP 9.41.36.41:5060;branch=z9hG4bK2CC2408
X-Cisco-Recording-Participant: sip:1777@10.104.45.207;media-
index="0 2"
X-Cisco-Recording-Participant: sip:1888@10.104.45.207;media-   index="1 3"
.
.
Cisco-Guid: 0884935168-0000065536-0000000401-3475859466
.
.
v=0
.
.
.
m=audio 17232 RTP/AVP 0 19
.
.
a=sendonly
m=audio 17234 RTP/AVP 0 19
.
.
a=sendonly

m=video 17236 RTP/AVP 126
.
.
```

```
.

a=fmtp:126 profile-level-id=42801E;packetization-mode=1
a=sendonly
m=video 17238 RTP/AVP 126
.
.

.
a=fmtp:126 profile-level-id=42801E;packetization-mode=1
a=sendonly
```

Output Field	Description
Sent: INVITE sip:575757@9.45.38.39:7686 SIP/2.0	22222 is the destination pattern or the number of recording server and is configured under the recorder dial peer.
X-Cisco-Recording-Participant: sip:1777@10.104.45.207;media- index="0 2“ X-Cisco-Recording-Participant: sip:1888@10.104.45.207;media- index="1 3“	Cisco proprietary header with originating and terminating participant number and IP address used to communicate to the recording server
Cisco-Guid: 0884935168-0000065536-0000000401-3475859466	GUID is the same for the primary call and forked call .
m=audio 17232 RTP/AVP 0 19	First m-line of participant with payload type and audio codec.
m=audio 17234 RTP/AVP 0 19	Second m-line of another participant with payload type and audio codec.
m=video 17236 RTP/AVP 126	Third m-line of participant with video payload type and codec info .
m=video 17238 RTP/AVP 126	Fourth m-line of another participant with video payload type and codec info .
a=sendonly	CUBE is always in send only mode towards Recording server.

```
Receive:
SIP/2.0 200 OK
.
.
.

v=0
.
.
m=audio 1592 RTP/AVP 0
.
.
a=recvonly
m=audio 1594 RTP/AVP 0
.
.
a=recvonly
```

```
m=video 1596 RTP/AVP 126
.
.
a=fmtp:97 profile-level-id=420015
a=recvonly
m=video 1598 RTP/AVP 126
.
.
a=fmtp:126 profile-level-id=420015
a=recvonly
Sent:
ACK sip:9.45.38.39:7686;transport=UDP SIP/2.0

Via: SIP/2.0/UDP 9.41.36.41:5060;branch=z9hG4bK2CD7

From: <sip:9.41.36.41>;tag=1ECFD128-24DF

To: <sip:575757@9.45.38.39>;tag=16104SIPpTag011

Date: Tue, 19 Mar 2013 11:40:01 GMT

Call-ID: FFFFFFFF91E00FE6-FFFFFFFF8FC011E2-FFFFFFFF824DF469-FFFFFFFFB6661C06@9.41.36.41

Max-Forwards: 70

CSeq: 101 ACK

Allow-Events: telephone-event

Content-Length: 0
```

Output Field	Description
m=audio 1592 RTP/AVP 0	First m-line of recording server after it started listening.
m=audio 1594 RTP/AVP 0	Second m-line of recording server after it started listening.
m=video 1596 RTP/AVP 126	Third m-line of recording server after it started listening.
m=video 1598 RTP/AVP 126	Fourth m-line of recording server after it started listening.
a=recvonly	Recording server in receive only mode.

Step 12 **debug ccsip messages** (for audio-only recording in a call with both audio and video)
Displays offer sent to MediaSense having only audio m-lines, when the **media-type audio** command is configured.

```
Sent:
INVITE sip:54321@9.45.38.39:36212 SIP/2.0
Via: SIP/2.0/UDP 9.41.36.15:5060;branch=z9hG4bK2216B
X-Cisco-Recording-Participant: sip:4321@9.45.38.39;media-index="0"
X-Cisco-Recording-Participant: sip:1111000010@9.45.38.39;media-index="1"
From: <sip:9.41.36.15>;tag=A2C74-5D9
To: <sip:54321@9.45.38.39>……
Content-Type: application/sdp
Content-Disposition: session;handling=required
Content-Length: 337

v=0
o=CiscoSystemsSIP-GW-UserAgent 9849 5909 IN IP4 9.41.36.15
s=SIP Call
c=IN IP4 9.41.36.15
t=0 0
```

```
m=audio 16392 RTP/AVP 0 19
c=IN IP4 9.41.36.15
a=rtpmap:0 PCMU/8000
a=rtpmap:19 CN/8000
a=ptime:20
a=sendonly
m=audio 16394 RTP/AVP 0 19
c=IN IP4 9.41.36.15
a=rtpmap:0 PCMU/8000
a=rtpmap:19 CN/8000
a=ptime:20
a=sendonly
```

Response from CUBE has inactive video m-lines.

```
Received:
SIP/2.0 200 OK
Via: SIP/2.0/UDP 9.41.36.15:5060;branch=z9hG4bK2216B
…..
v=0
…
m=audio 36600 RTP/AVP 0
c=IN IP4 9.45.38.39
a=rtpmap:0 PCMU/8000
a=ptime:20
a=recvonly
m=audio 36602 RTP/AVP 0
c=IN IP4 9.45.38.39
a=rtpmap:0 PCMU/8000
a=ptime:20
a=recvonly
m=video 0 RTP/AVP 98
c=IN IP4 9.45.38.39
b=TIAS:1500000
a=rtpmap:98 H264/90000
a=fmtp:98 profile-level-id=420015
a=inactive
m=video 0 RTP/AVP 98
c=IN IP4 9.45.38.39
b=TIAS:1500000
a=rtpmap:98 H264/90000
a=fmtp:98 profile-level-id=420015
a=inactive
```

Step 13 Enter one of the following:

- **debug ccsip all**
- **debug voip recmsp all**
- **debug voip ccapi all**
- **debug voip fpi all** (for ASR devices only)

Displays detailed debug messages.

For Audio:
Media forking initialized:

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_trigger_media_forking: MF: Recv Ack..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_trigger_media_forking: MF: Recv Ack & it's
Anchor leg. Start MF.
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_preprocess_event: MF:
initial-call. State = 1 & posting the event E_IPIP_MEDIA_FORKING_CALLSETUP_IND
```

Media forking started:

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_service_get_event_data: Event id
 = 30
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/sipSPIUisValidCcb:
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/ccsip_is_valid_ccb:
```

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking: MF: Current State = 1,
event =30
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking: MF: State & Event
combination is cracked..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/sipSPIGetMainStream:
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/sipSPIGetMainStream:
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_precondition: MF: Can be
 started with current config.
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecParticipant:
 MF: Populate rec parti header from this leg.
```

Forking header populated:

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_get_recording_participant_header: MF: X-Cisco
 header is RPID..
```

Media forking setup record session is successful:

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_get_recording_participant_header: MF: Building
 SIP URL..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_get_recording_participant_header: MF: Sipuser
 = 98459845
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_get_recording_participant_header: MF: Host
= 9.42.30.34
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/sipSPIGetFirstStream:
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Function/voip_media_dir_to_cc_media_dir:
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecStream: MF:
 direction type =3 3
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecStream: MF:
 callid 103 set to nearend..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecStream: MF:
 dtmf is inband
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecStream: MF:
 First element..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecParticipant:
 MF: First element..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_BuildMediaRecParticipant:
 MF: Populate rec parti header from peer leg.
*Jun 15 10:37:55.404: //104/3E7E90AE8006/SIP/Info/ccsip_get_recording_participant_header: MF: X-Cisco
 header is RPID..
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_write_to_TDContainer: MF:
 Data written to TD Container..
*Jun 15 10:37:55.404: //-1/xxxxxxxxxxxx/Event/recmsp_api_setup_session: Event: E_REC_SETUP_REQ anchor
 call ID:103, msp call ID:105 infunction recmsp_api_setup_session
*Jun 15 10:37:55.404: //-1/xxxxxxxxxxxx/Inout/recmsp_api_setup_session:  Exit with Success
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/act_sip_mf_idle_callsetup_ind: MF:
setup_record_session is success..
```

Media forking forked stream started:

```
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/sipSPIMFChangeState: MF: Prev state = 1 & New state
 = 2
*Jun 15 10:37:55.404: //103/3E7E90AE8006/SIP/Info/ccsip_gen_service_process_event: MF: 30 event
handled.
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_call_setup_request: Set Protocol information
*Jun 15 10:37:55.406: //106/xxxxxxxxxxxx/CCAPI/cc_set_post_tagdata:
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_ipip_media_forking_read_from_TDContainer:
MF: Data read from TD container..
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_ipip_media_forking_forked_leg_config: MF:
MSP callid = 105
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_ipip_media_forking_forked_leg_config: MF:
Overwriting the GUID with the value got from MSP.
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_iwf_handle_peer_event:
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_iwf_map_ccapi_event_to_iwf_event: Event
Category: 1, Event Id: 179
*Jun 15 10:37:55.406: //106/000000000000/SIP/Info/ccsip_iwf_process_event:
*Jun 15 10:37:55.406: //106/000000000000/SIP/Function/sipSPIUisValidCcb:
*Jun 15 10:37:55.406: //106/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_add_forking_stream: MF:
Forked stream added..
*Jun 15 10:37:55.406: //106/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_read_from_TDContainer:
MF: Data read from TD container..
*Jun 15 10:37:55.406: //106/3E7E90AE8006/SIP/Function/sipSPIGetFirstStream:
*Jun 15 10:37:55.406: //106/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_Display_TDContainerData:
 ** DISPLAY REC PART ***
```

```
*Jun 15 10:37:55.406: //106/3E7E90AE8006/SIP/Info/ccsip_ipip_media_forking_Display_TDContainerData:
 recorder tag = 5
```

For Video:

Media Forking Initialized:

```
*Mar 19 16:40:01.784 IST: //522/34BF0A000000/SIP/Info/notify/32768/ccsip_trigger_media_forking: MF:
 Recv Ack & it's Anchor leg. Start MF.
*Mar 19 16:40:01.784 IST:
//522/34BF0A000000/SIP/Info/info/32768/ccsip_ipip_media_forking_preprocess_event: MF: initial-call.
 State = 1 & posting the event E_IPIP_MEDIA_FORKING_CALLSETUP_IND
```

Media forking started:

```
 *Mar 19 16:40:01.784 IST: //522/34BF0A000000/SIP/Info/info/36864/ccsip_ipip_media_forking: MF:
Current State = 1, event =31
*Mar 19 16:40:01.784 IST: //522/34BF0A000000/SIP/Info/info/36864/ccsip_ipip_media_forking: MF: State
 & Event combination is cracked..
*Mar 19 16:40:01.784 IST: //522/34BF0A000000/SIP/Function/sipSPIGetMainStream:
 *Mar 19 16:40:01.784 IST: //522/34BF0A000000/SIP/Function/sipSPIGetMainStream:
*Mar 19 16:40:01.787 IST: //522/34BF0A000000/SIP/Info/info/34816/ccsip_ipip_media_forking_precondition:
 MF: Can be started with current config.
*Mar 19 16:40:01.787 IST: //-1/xxxxxxxxxxxx/Event/recmsp_api_create_session: Event:
E_REC_CREATE_SESSION anchor call ID:522, msp call ID:526
*Mar 19 16:40:01.787 IST: //-1/xxxxxxxxxxxx/Inout/recmsp_api_create_session:  Exit with Success
```

Recording participant for anchor leg:

```
//522/34BF0A000000/SIP/Info/verbose/32768/ccsip_ipip_media_forking_BuildMediaRecParticipant: MF:
Populate rec parti header from this leg.
*Mar 19 16:40:01.788 IST:
//522/34BF0A000000/SIP/Info/info/33792/ccsip_get_recording_participant_header: MF: X-Cisco header
is PAI..
```

Adding an audio stream:

```
*Mar 19 16:40:01.788 IST: //522/34BF0A000000/SIP/Function/sipSPIGetFirstStream:
*Mar 19 16:40:01.788 IST:
//522/34BF0A000000/SIP/Info/verbose/32768/ccsip_ipip_media_forking_BuildMediaRecStream: MF: Adding
a Audio stream..
*Mar 19 16:40:01.789 IST: //522/34BF0A000000/SIP/Function/voip_media_dir_to_cc_media_dir:
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/info/32768/ccsip_ipip_media_forking_BuildAudioRecStream: MF: direction
type =3 3
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/info/32768/ccsip_ipip_media_forking_BuildAudioRecStream: MF: callid 522
 set to nearend..
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/info/32768/ccsip_ipip_media_forking_BuildAudioRecStream: MF: This rcstream
 has 522 callid
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/verbose/32768/ccsip_ipip_media_forking_BuildAudioRecStream: MF: Setting
 data for audio stream..
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/info/32800/ccsip_ipip_media_forking_BuildAudioRecStream: MF: dtmf is
inband
.
```

Video forking:

```
*Mar 19 16:40:01.789 IST: //522/34BF0A000000/SIP/Function/sipSPIGetVideoStream:
*Mar 19 16:40:01.789 IST:
//522/34BF0A000000/SIP/Info/verbose/32772/ccsip_ipip_media_forking_BuildMediaRecStream: MF: video_codec
 present,Continue with Video Forking..
```

For Video

Additional References for Network-Based Recording

Related Documents

Related Topic	Document Title
MediaSense Installation and Administration Guide	Cisco MediaSense Installation and Administration Guide

Standards and RFCs

RFCs	Title
RFC 3984	*RTP Payload Format for H.264 Video*
RFC 5104	*Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)*
RFC 5168	*XML Schema for Media Control*

CHAPTER 11

Video Recording - Additional Configurations

This module describes the following additional configurations that can be done for Video Recording:

- Request a Full-Intra Frame using RTCP or SIP INFO methods.
- Configure an H.264 Packetization mode.
- Monitor Intra-Frames and Reference Frames

Feature Information for Video Recording - Additional Configurations

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 12: Feature Information for Network-Based Recording of Video Calls Using Cisco Unified Border Element

Feature Name	Releases	Feature Information
Network-Based Recording of Video Calls Using Cisco Unified Border Element	15.3(3)M Cisco IOS XE Release 3.10S	The Network-Based Recording of Video Calls Using Cisco Unified Border Element feature supports software-based forking and recording of video calls. The following commands were introduced or modified: **media profile video**, **ref-frame-req rtcp**, **ref-frame-req sip-info**, **video profile**, **h264-packetization-mode**, **monitor-ref-frames**.

Information About Additional Configurations for Video Recording

Full Intra-Frame Request

Full Intra-Frame Request is a request sent for an I-frame. An I-frame is an entire key or reference frame that is compressed without considering preceding or succeeding video frames. Succeeding video frames are differences to the original I-frame (what has moved) instead of entire video frame information.

The call between Cisco Unified Border Element and the Cisco MediaSense server is established after the call between the endpoints is established. As a result, the Real-Time Transport Protocol (RTP) channel between the endpoints gets established first and the RTP channel with the recording server gets established later. The impact of this delay is more on video recording because the initial I-frame from the endpoint may not get forked, and frames that follow cannot get decoded. To mitigate the impact of the lost RTP video packets, Cisco Unified Border Element generates Full Intra-Frame Request (FIR) using either Real-Time Transport Control Protocol (RTCP) or SIP INFO, or both, requesting the endpoint to send a fully encoded video frame in the subsequent RTP packet.

The following types of FIR are supported on network-based recording of video calls using Cisco Unified Border Element:

- RTCP FIR (based on RFC 5104).
- SIP INFO FIR (based on RFC 5168).
- Both RTCP FIR and SIP INFO FIR (Cisco Unified Border Element can be configured to send both RTCP FIR and SIP INFO requests at the same time).

How to Configure Additional Configurations for Video Recording

Enabling FIR for Video Calls (Using RTCP of SIP INFO)

Perform this task to enable Full Intra-Frame Request (FIR) during the network-based recording of a video call using Real-Time Transport Control Protocol (RTCP) or using the Session Initiation Protocol (SIP) INFO method.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile video** *media-profile-tag*
4. Do one of the following:
   - **ref-frame-req rtcp retransmit-count** *retransmit-number*
   - **ref-frame-req sip-info**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile video** *media-profile-tag* **Example:** `Device(config)# media profile video 1`	Configures a video media profile and enters media profile configuration mode.
Step 4	Do one of the following: • **ref-frame-req rtcp retransmit-count** *retransmit-number* • **ref-frame-req sip-info** **Example:** `Device(cfg-mediaprofile)# ref-frame-req rtcp retransmit-count 4`	Enables FIR using the RTCP or SIP INFO method.

	Command or Action	Purpose
	Example: `Device(cfg-mediaprofile)# ref-frame-req sip-info`	
Step 5	**end** **Example:** `Device(cfg-mediaprofile)# end`	Exits media profile configuration mode.

Configuring H.264 Packetization Mode

When a device configured as CUBE is offered more than one H.264 packetization mode on an inbound video call leg, the device offers all received modes to the outbound call leg, allowing dynamic change of mode during a call. However when a call is forked, the MediaSense recording server is not able to support this dynamic change of the packetization mode.

This feature restricts the device and allows it to offer only the configured packetization mode to the outbound call leg when media forking is configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile video** *media-profile-tag*
4. **h264-packetization-mode** *packetization mode*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile video** *media-profile-tag* **Example:** `Device(config)# media profile video 1`	Configures a video media profile and enters media profile configuration mode.

	Command or Action	Purpose
Step 4	**h264-packetization-mode** *packetization mode* **Example:** `Device(cfg-mediaprofile)# h264-packetization-mode 2`	Configures the H.264 packetization mode offered by a device on the outbound call leg of a forked call when multiple H.264 packetization modes are present in the offer received by the device on the inbound call leg.
Step 5	**end** **Example:** `Device(cfg-mediaprofile)# end`	Exits media profile configuration mode.

Monitoring Reference files or Intra Frames

Perform this task to configure device to perform deep packet inspection (DPI) of RTP packets received from an endpoint and keep track of how many instantaneous decoder refresh (IDR) frames have been received and the timestamp of the IDRs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile video** *media-profile-tag*
4. **monitor-ref-frames**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile video** *media-profile-tag* **Example:** `Device(config)# media profile video 1`	Configures a video media profile and enters media profile configuration mode.

	Command or Action	Purpose
Step 4	**monitor-ref-frames** **Example:** `Device(cfg-mediaprofile)# monitor-ref-frames`	Monitors reference frames or intra-frames.
Step 5	**end** **Example:** `Device(cfg-mediaprofile)# end`	Exits media profile configuration mode.

Verifying Additional Configurations for Video Recording

Perform this task to verify the additional configurations of the video recording. The **show** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show call active video called-number** *number* **| include VideoRtcpIntraFrameRequestCount**
3. **show call active video called-number** *number* **| include VideoSipInfoIntraFrameRequestCount**
4. **show call active video | include VideoTimeOfLastReferenceFrame**
5. **show call active video | include VideoReferenceFrameCount**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show call active video called-number** *number* **| include VideoRtcpIntraFrameRequestCount**

Displays the number of RTCP FIR requests sent on each leg.

Example:

```
Device# show call active video called-number 990057 | include VideoRtcpIntraFrameRequestCount

! Main call legs
VideoRtcpIntraFrameRequestCount=1
VideoRtcpIntraFrameRequestCount=1

!CUBE does not generate FIR request on forked leg
VideoRtcpIntraFrameRequestCount=0
```

Step 3 **show call active video called-number** *number* **| include VideoSipInfoIntraFrameRequestCount**

Displays the number of SIP INFO FIR requests sent on each leg.

Example:

```
Device# show call active video called-number 990062 | include VideoSipInfoIntraFrameRequestCount

! Main call legs
VideoSipInfoIntraFrameRequestCount=1
VideoSipInfoIntraFrameRequestCount=1

!CUBE does not generate FIR request on forked leg
VideoSipInfoIntraFrameRequestCount=0
```

Step 4 **show call active video | include VideoTimeOfLastReferenceFrame**

Displays the timestamp of latest IDR frame.

Step 5 **show call active video | include VideoReferenceFrameCount**

Djsplays the number of IDR frames received on that call leg.

CHAPTER 12

TDoS Attack Mitigation

The TDoS Attack Mitigation feature enables Cisco Unified Border Element (Cisco UBE) to not respond to Session Initiation Protocol (SIP) requests from IP addresses that are not listed in a trusted IP address list. Cisco UBE validates only out-of-dialog SIP requests against IP addresses in the trusted IP address list. It does not validate in-dialog SIP requests because such requests usually arrive from trusted entities. The TDoS Attack Mitigation feature is supported both on IPv4 and IPv6 networks.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About TDoS Attack Mitigation

The TDoS Attack Mitigation feature prevents Cisco Unified Border Element (Cisco UBE) from responding to Session Initiation Protocol (SIP) requests arriving from untrusted IP addresses, which leads to an improvement in performance. The SIP stack authenticates the source IP address of an incoming SIP request and blocks the response if the source IP address does not match any IP address in the trusted IP address list. To create a trusted IP address list, you may configure a list of IP addresses or use the IP addresses that have been configured using the **session target** command in dial-peer configuration mode.

Cisco UBE does not respond to REGISTER requests and consumes REGISTER requests if you configure it only for Telephony Denial-of-Service (TDoS) Attack Mitigation and not as a registrar server.

If you configure Cisco UBE as a registrar server for TDoS attack mitigation, it consumes responses for REGISTER requests that do not belong to any application. Cisco UBE does not consume responses to REGISTER requests that belong to a registrar application.

Note A SIP registrar is a server that accepts REGISTER requests and is typically collocated with a proxy or redirect server.

Syslogs are printed on the device console every 60 minutes after Cisco UBE consumes a threshold value of 1000 SIP requests.

How to Configure TDoS Attack Mitigation

Configuring a Trusted IP Address List for Toll-Fraud Prevention

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **ip address trusted list**
5. **ipv4** *ipv4-address* [*network-mask*]
6. **ipv6** *ipv6-address*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters global VoIP configuration mode.

	Command or Action	Purpose
Step 4	**ip address trusted list** **Example:** `Device(conf-voi-serv)# ip address trusted list`	Enters IP address trusted list mode and enables the addition of valid IP addresses.
Step 5	**ipv4** *ipv4-address* [*network-mask*] **Example:** `Device(cfg-iptrust-list)# ipv4 192.0.2.1 255.255.255.0`	Allows you to add up to 100 IPv4 addresses in the IP address trusted list. Duplicate IP addresses are not allowed. • The *network-mask* argument allows you to define a subnet IP address.
Step 6	**ipv6** *ipv6-address* **Example:** `Device(cfg-iptrust-list)# ipv6 2001:DB8:0:ABCD::1/48`	Allows you to add IPv6 addresses to the trusted IP address list.
Step 7	**end** **Example:** `Device(cfg-iptrust-list)# end`	Returns to privileged EXEC mode.

Configuring TDoS Attack Mitigation

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **ip address trusted authenticate**
5. **allow-connections** *from-type* **to** *to-type*
6. **sip**
7. **no registrar server**
8. **silent-discard untrusted**
9. **end**
10. **show sip-ua statistics**
11. **clear sip-ua statistics**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters voice service configuration mode.
Step 4	**ip address trusted authenticate** **Example:** `Device(conf-voi-serv)# ip address trusted authenticate`	Enables IP address authentication on incoming H.323 or Session Initiation Protocol (SIP) trunk calls for toll fraud prevention support.
Step 5	**allow-connections** *from-type* **to** *to-type* **Example:** `Device(conf-voi-serv)# allow-connections sip to sip`	Allows connections between specific types of endpoints in a Cisco UBE.
Step 6	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters SIP configuration mode.
Step 7	**no registrar server** **Example:** `Device(conf-serv-sip)# no registrar server`	Disables the local SIP registrar.
Step 8	**silent-discard untrusted** **Example:** `Device(conf-serv-sip)# silent-discard untrusted`	Discards SIP requests from untrusted sources on an incoming SIP trunk.
Step 9	**end** **Example:** `Device(conf-serv-sip)# end`	Returns to privileged EXEC mode.
Step 10	**show sip-ua statistics** **Example:** `Device# show sip-ua statistics`	(Optional) Displays response, traffic, and retry SIP statistics.
Step 11	**clear sip-ua statistics** **Example:** `Device# clear sip-ua statistics`	(Optional) Resets the SIP user agent (UA) statistical counters to zero.

Verifying TDoS Attack Mitigation

Sample output for the show sip-ua statistics command

To display response, traffic, and retry Session Initiation Protocol (SIP) statistics, use the **show sip-ua statistics** command in privileged EXEC mode.

```
Device# show sip-ua statistics

SIP Response Statistics (Inbound/Outbound)
    Informational:
      Trying 0/0, Ringing 0/0,
      Forwarded 0/0, Queued 0/0,
      SessionProgress 0/0
    Success:
      OkInvite 0/0, OkBye 0/0,
      OkCancel 0/0, OkOptions 0/0,
      OkPrack 0/0, OkRegister 0/0
      OkSubscribe 0/0, OkNotify 0/0, OkPublish 0/0
      OkInfo 0/0, OkUpdate 0/0,
      202Accepted 0/0, OkOptions 0/0
    Redirection (Inbound only except for MovedTemp(Inbound/Outbound)) :
      MultipleChoice 0, MovedPermanently 0,
      MovedTemporarily 0/0, UseProxy 0,
      AlternateService 0
    Client Error:
      BadRequest 0/0, Unauthorized 0/0,
      PaymentRequired 0/0, Forbidden 0/0,
      NotFound 0/0, MethodNotAllowed 0/0,
      NotAcceptable 0/0, ProxyAuthReqd 0/0,
      ReqTimeout 0/0, Conflict 0/0, Gone 0/0,
      ConditionalRequestFailed 0/0,
      ReqEntityTooLarge 0/0, ReqURITooLarge 0/0,
      UnsupportedMediaType 0/0, UnsupportedURIScheme 0/0,
      BadExtension 0/0, IntervalTooBrief 0/0,
      TempNotAvailable 0/0, CallLegNonExistent 0/0,
      LoopDetected 0/0, TooManyHops 0/0,
      AddrIncomplete 0/0, Ambiguous 0/0,
      BusyHere 0/0, RequestCancel 0/0,
      NotAcceptableMedia 0/0, BadEvent 0/0,
      SETooSmall 0/0, RequestPending 0/0,
      UnsupportedResourcePriority 0/0,
      Total untrusted Request Consumed 1500,//This counter increments (+1) on reception of
 an untrusted SIP request.//
      Untrusted Request Consumed in last lap 300,//This counter is updated after every 60
minutes.//
      Last Threshold for Untrusted Request Consumed 1000//This counter activates when the
router boots up. Counter value is the number of untrusted requests that are consumed (after
 crossing 1000 SIP requests) in each interval of 60 minutes after the router boots up.//
    Server Error:
      InternalError 0/0, NotImplemented 0/0,
      BadGateway 0/0, ServiceUnavail 0/0,
      GatewayTimeout 0/0, BadSipVer 0/0,
      PreCondFailure 0/0
    Global Failure:
      BusyEverywhere 0/0, Decline 0/0,
      NotExistAnywhere 0/0, NotAcceptable 0/0
    Miscellaneous counters:
      RedirectRspMappedToClientErr 0

SIP Total Traffic Statistics (Inbound/Outbound)
    Invite 0/0, Ack 0/0, Bye 0/0,
    Cancel 0/0, Options 0/0,
```

```
    Prack 0/0, Update 0/0,
    Subscribe 0/0, Notify 0/0, Publish 0/0
    Refer 0/0, Info 0/0,
    Register 0/0

Retry Statistics
    Invite 0, Bye 0, Cancel 0, Response 0,
    Prack 0, Reliable1xx 0, Notify 0, Info 0
    Register 0 Subscribe 0 Update 0 Options 0
    Publish 0

SDP application statistics:
 Parses: 0,  Builds 0
 Invalid token order: 0,  Invalid param: 0
 Not SDP desc: 0,  No resource: 0

Last time SIP Statistics were cleared: <never>
```

Configuration Examples for TDoS Attack Mitigation

Example: Trusted IP Address List Configuration

The following example shows how to configure a Trusted IP Address list.

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
Device(conf-voi-serv)# ip address trusted list
Device(cfg-iptrust-list)# ipv4 192.0.2.1
Device(cfg-iptrust-list)# ipv6 2001:DB8:0:ABCD::1/48
```

Example: TDoS Attack Mitigation Configuration

The following example shows how to configure TDoS Attack Mitigation.

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
Device(conf-voi-serv)# ip address trusted authenticate
Device(conf-voi-serv)# allow-connections sip to sip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# no registrar server
Device(conf-serv-sip)# silent-discard untrusted
```

Feature Information for TDoS Attack Mitigation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 13: Feature Information for TDoS Mitigation

Feature Name	Release	Feature Information
TDoS Attack Mitigation	15.3(3)M	The TDoS Attack Mitigation feature enables Cisco UBE to not respond to Session Initiation Protocol (SIP) requests from IP addresses that are not listed in a trusted IP address list.
TDoS Attack Mitigation	Cisco IOS XE Release 3.10S	The TDoS Attack Mitigation feature enables Cisco UBE to not respond to Session Initiation Protocol (SIP) requests from IP addresses that are not listed in a trusted IP address list.

CHAPTER 13

Cisco Unified Communications Gateway Services--Extended Media Forking

The Cisco Unified Communications (UC) Services API provides a unified web service interface for the different services in IOS gateway thereby facilitating rapid service development at application servers and managed application service providers.

This chapter explains the Extended Media Forking (XMF) provider that allows applications to monitor calls and trigger media forking on Real-time Transport Protocol (RTP) and Secure RTP calls.

Feature Information for Cisco Unified Communications Gateway Services—Extended Media Forking

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Feature Name	Releases	Feature Information
Cisco Unified Communications Gateway Services	Cisco IOS 15.3(3)M Cisco IOS XE 3.10S	The Cisco Unified Communications (UC) Services API provides a unified web service interface for the different services in IOS gateway thereby

Feature Name	Releases	Feature Information
		facilitating rapid service development at application servers and managed application service providers.
Cisco UC Gateway Services API support for Secure RTP Forking	Cisco IOS 15.4(3)M Cisco IOS XE 3.13S	This feature provides support for Extended Media Forking (XMF) provider to monitor calls and trigger media forking on RTP and SRTP calls.
Support for Cisco UC Services API Media Forking with Survivability TCL	Cisco IOS 15.6(1)T Cisco IOS XE 3.17S	This feature allows media forking for the calls controlled by CVP Survivability TCL script with Cisco Unified Communication Services API.

Restrictions for Unified Communications Gateway Services—Extended Media Forking

- Media renegotiation is not supported.
- Media mixing on forked media streams is not supported.
- recordTone insertion is not supported with SRTP calls.
- mediaForkingReason tag is only to notify midcall stream events; notification for events such as codec change is not supported.
- Only voice media stream is supported.
- Supplementary services are not supported.
- High Availability is not supported.

Information About Cisco Unified Communications Gateway Services

Extended Media Forking (XMF) Provider and XMF Connection

The XMF provider allows applications to monitor calls and trigger media forking on the calls and has the capability to service up to 32 applications. The XMF provider can invoke a call-based or a connection-based media forking using the Unified Communications (UC) API. After the media forking is invoked, it can preserve the media forking initiated by the web application if the WAN connection to the application is lost. The XMF provider also provides the recording tone to the parties involved in the call.

The XMF connection describes the relationship between an XMF call and the endpoint (or trunk) involved in the call. A connection abstraction maintained in the gateway has the following connection states:

- IDLE: This state is the initial state for all new connections. Such connections are not actively part of a telephone call, yet their references to the Call and Address objects are valid. Connections typically do not stay in the IDLE state for long and quickly transition to other states. The application may choose to be notified at this state using the event filters and if done, call/connection at the gateway provider will use the NotifyXmfConnectionData(CREATED) message to notify the application listener that a new connection is created.

- ADDRESS_COLLECT: In this state the initial information package is collected from the originating party and is examined according to the “dialing plan” to determine the end of collection of addressing information. In this state, the call in the gateway collects digits from the endpoint. No notification is provided.
- CALL_DELIVERY: On the originating side, this state involves selecting of the route as well as sending an indication of the desire to set up a call to the specified called party. On the terminating side, this state involves checking the busy/idle status of the terminating access and also informing the terminating message of an incoming call. The application may choose to be notified at this state using the event filters and if done, the call or connection at the gateway provider will use the NotifyXmfConnectionData (CALL_DELIVERY) message to notify the application listener.
- ALERTING: This state implies that the Address is being notified of an incoming call. The application may choose to be notified at this state using the event filters and if done, the call or connection at the gateway provider will use the NotifyXmfConnectionData (ALERTING) message to notify the application listener.
- CONNECTED: This state implies that a connection and its Address is actively part of a telephone call. In common terms, two parties talking to one another are represented by two connections in the CONNECTED state. The application may choose to be notified at this state using the event filters and if done, the call or connection at the gateway provider will use the NotifyXmfConnectionData (CONNECTED) message to notify the application listener.
- DISCONNECTED: This state implies it is no longer part of the telephone call. A Connection in this state is interpreted as once previously belonging to this telephone call. The application may choose to be notified at this state using the event filters and if done, the call or connection at the gateway provider will use the NotifyXmfConnectionData (DISCONNECTED) message to notify the application listener.

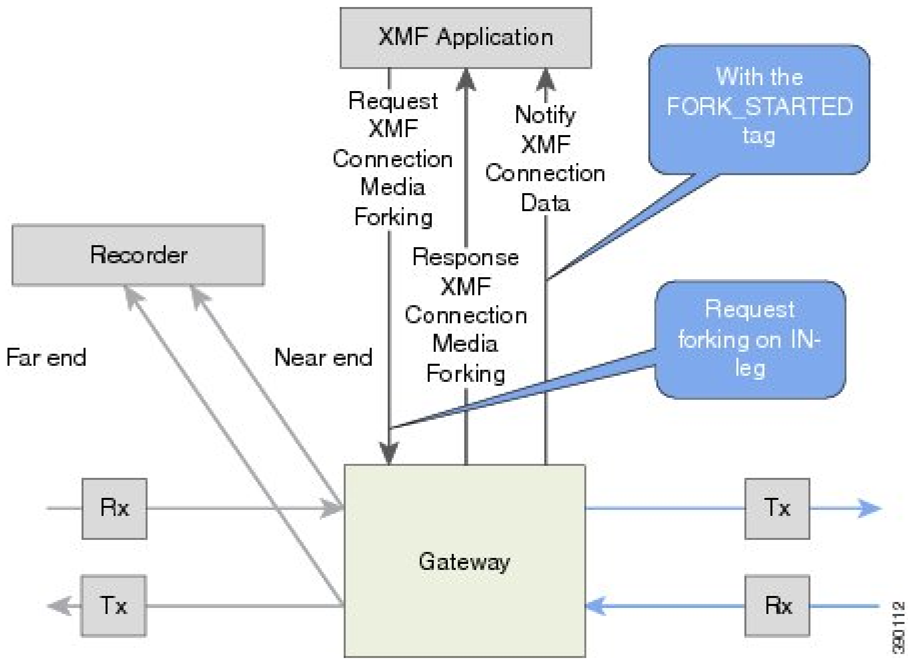
XMF Call-Based Media Forking

In call-based media forking of the gateway, the stream from the calling party is termed as near-end stream and the stream from the called party is termed as far-end stream. The XMF provider actively handles single media forking request per session. Any new media forking request from the external application will override or stop the current forking instance and would start a new forking instance (to the appropriate target IP address or ports). After the media forking request is accepted, the XMF provider returns a response message and starts to fork media streams of a connection to the target forked streams. A NotifyXmfCallData message will be notified to the application for the updated media forking status, that is, FORK-FAILED, FORK_STARTED, or FORK_DONE.

XMF Connection-Based Media Forking

In connection-based media forking of the gateway, the incoming stream to the connection is termed as near-end stream and the outgoing stream of the connection is termed as far-end stream. The XMF provider actively handles single media forking request per session. Any new media forking request from the external application will override or stop the current forking instance and would start a new forking instance (to the appropriate target IP address or ports). After the media forking request is accepted, the XMF provider returns a response message and starts to fork media streams of a connection to the target forked streams.

Figure 2: XMF Connection-Based Media Forking



A NotifyXmfConnectionData message will be notified to the application for the updated media forking status:

- FORK_FAILED—Media forking is setup failure. No forked RTP connections can be established to target RTP addresses.
- FORK_STARTED—Media forking is set up successfully. Both Tx (transmit) and Rx (receive) forked RTP connections are established and connected to target (farEnd and nearEnd) RTP addresses.
- FORK_DONE—Media forking is completed. Both Tx and Rx forked RTP connections are released.

Cisco UC Gateway Services Media Forking API with Survivability TCL

Cisco Unified Border Element (CUBE) supports Survivability TCL Script to co-exist with Cisco Unified Communication (UC) Services API.

Cisco UC Services API XMF interface supports media forking for all the calls controlled by survivability TCL script including the survivability re-attempted calls. Thus, all the calls controlled by survivability TCL script can be recorded when requested by Cisco UC Services XMF API.

Cisco Unified Communications Manager controlled Gateway recording utilizes XMF to trigger media forking on CUBE or SIP based PSTN gateways in the supported call flows.

Note Media forking is allowed only for survivability TCL script supported by Cisco Unified Customer Voice Portal (CVP). CVP survivability TCL script is not supported in High Availability mode.

The following call scenarios are supported:

- Basic comprehensive call
- Calls with Refer Consume
- Calls with Mid-call failure
- Calls with alternative route with initial call failure

There are no configuration changes required for enabling CVP survivability TCL support with Cisco UC Gateway Services API.

Media Forking for SRTP Calls

- SRTP forking is supported in XCC and XMF application service providers and the supported APIs are RequestCallMediaForking, RequestCallMediaSetAttributes, and RequestConnectionMediaForking.
- SRTP forking is supported for SRTP-to-SRTP, SRTP-to-RTP, and RTP-to-SRTP calls.
    - For SRTP-to-SRTP calls, media forking on either leg would result in SRTP streams being forked.
    - For SRTP fallback calls, after the initial offer, CUBE will fall back to RTP. Media forking either call legs would result in RTP streams being forked.
    - For SRTP-to-RTP interworking calls, a digital signal processor (DSP) is required and involves transcoding. In this case, one leg would be SRTP and the other leg RTP.
- SRTP Crypto keys are notified over the API.
- Supports automatic stopping of media forking when stream changes from SRTP or to SRTP.
    - The optional mediaForkingReason tag in XMF or XCC Notify messages indicates that the forking has been stopped internally.
    - mediaForkingReason tag is only present when the connection changes state, such as mid-call re-INVITE. SRTP stream can change to RTP or SRTP stream can change keys mid-call.
    - mediaForkingReason tag is always accompanied by FORK_DONE.

Crypto Tag

For SRTP forking, the optional Crypto tag in NotifyXmfConnectionData or NotifyXmfCallData message indicates the context of an actively forked SRTP connection.

Note The Crypto tag is only present in the notification message where FORK_STARTED tag is present.

The optional Crypto tag specifies the following:

- The Crypto suite used for encryption and authentication algorithm.
- The base64 encoded mastery key and salt used for encryption.

Crypto suite can be one of the two suites supported in IOS:

- AES_CM_128_HMAC_SHA1_32
- AES_CM_128_HMAC_SHA1_80

Example of SDP Data sent in an SRTP Call

Original SIP SDP Crypto Offer	SIP SDP Crypto Answer
v=0 o=CiscoSystemsSIP-GW-UserAgent 7826 3751 IN IP4 172.18.193.98 s=SIP Call c=IN IP4 172.18.193.98 t= 0 0 m=audio 51372 RTP/SAVP 0 a=rtpmap:0 PCMU/8000 **a=crypto:1 AES_CM_128_HMAC_SHA1_32** **inline:d0RmdmcmVCspEc3QGZiNWpVLFJhQX1cfHAwJSoj**	v=0 o=CiscoSystemsSIP-GW-UserAgent 7826 3751 IN IP4 172.18.193.98 s=SIP Call c=IN IP4 172.18.193.98 t=0 0 m=audio 49170 RTP/SAVP 0 **a=crypto:1 AES_CM_128_HMAW_SHA1_32** **inline:NzB4d1BINUAvLEw6UzF3WSJ+PSdFcGdUJShpX1Zj**

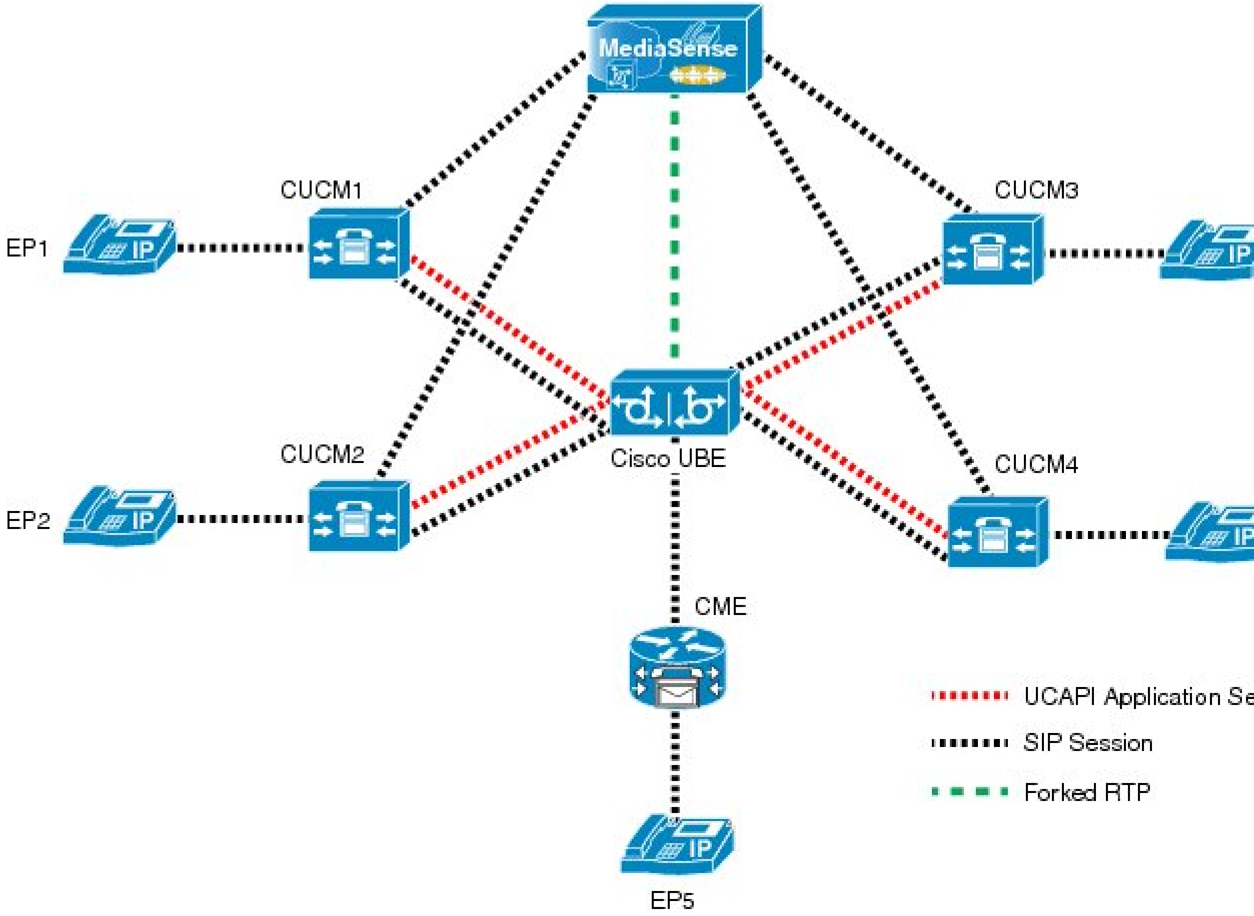
Note The application is notified of the content in Crypto and inline SDP lines.

Multiple XMF Applications and Recording Tone

Multiple XMF allows multiple (maximum 32) web applications to register with the XMF provider as separate XMF applications and provide redundancy for the voice calls recording. Recording tone provides recording tone capability to the recording sessions. Recording tone is supported for IP to IP, IP to TDM, and TDM to TDM trunks.

An example topology is as shown below where 4 CUCM applications are deployed. CUCM triggers media forking request to Cisco UBE. Recording tone is played to the parties involved in the call based on the recordTone parameter set in the media forking request.

Figure 3: Multiple XMF Applications and Recording Tone



Media forking can be invoked using any of the following APIs:

- RequestXmfConnectionMediaForking
- RequestXmfCallMediaForking
- RequestXmfCallMediaSetAttributes

The “recordTone” parameter can be enabled in any of the above requests and recording tone will be played for the parties involved in the call. The “recordTone” parameter in the API request can have the following values:

- COUNTRY_US
- COUNTRY_AUSTRALIA
- COUNTRY_GERMANY
- COUNTRY_RUSSIA

- COUNTRY_SPAIN
- COUNTRY_SWITZERLAND

There is no difference in the recording tone beep when any country value is chosen. Recording tone beep is played at an interval of every 15 seconds. Digital signal processors and other resources are not utilized for playing recording tone even for transcoded calls. No specific configuration is required to enable or disable recording tone. By default, no recording tone is enabled.

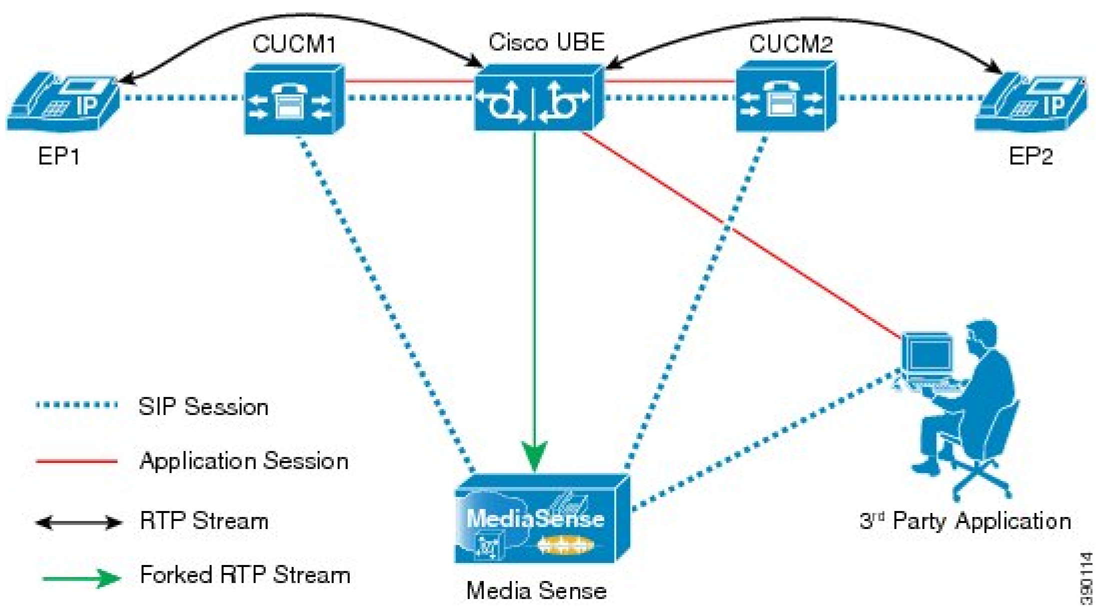
If “recordTone” parameter is enabled only on the farEndAddr, then this tone is played only on the outgoing leg. Likewise, if enabled only on the nearEndAddr, then the tone is played only on the incoming leg. When enabled in both the far and near end, then recording tone is played on both the legs.

The RequestXmfConnectionMediaForking API allows insertion of recording tone on a per connection basis. There could be scenarios where one leg receives two recordTone insertion requests. When a leg receives recordTone insertion request, the nearEnd request always takes precedence over the farEnd request.

Forking Preservation

After media forking is initiated by the web application, the forking can be preserved to continue the recording, even if the WAN connection to the application is lost or if the application is unregistered.

Figure 4: Forking Preservation



The “preserve” parameter value can be set to TRUE or FALSE in any of the 3 forking requests (RequestXmfConnectionMediaForking, RequestXmfCallMediaForking, or RequestXmfCallMediaSetAttributes) from the application to Cisco UBE.

- If the “preserve” parameter received is TRUE, then forking will continue the recording, even if the WAN connection to application is lost or application is unregistered.
- If the “preserve” parameter received is FALSE, then forking will not continue the recording.

- If the "preserve" parameter is not received in the media forking request, then forking will not continue the recording.

How to Configure UC Gateway Services

Configuring Cisco Unified Communication IOS Services on the Device

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **ip http max-connections** *value*
5. **ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value*
6. **http client connection idle timeout** *seconds*
7. **uc wsapi**
8. **message-exchange max-failures** *number*
9. **probing max-failures** *number*
10. **probing interval keepalive** *seconds*
11. **probing interval negative** *seconds*
12. **source-address** *ip-address*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http server** **Example:** `Device(config)# ip http server`	Enables the HTTP server (web server) on the system.

	Command or Action	Purpose
Step 4	**ip http max-connections** *value* **Example:** `Device(config)# ip http max-connection 100`	Sets the maximum number of concurrent connections to the HTTP sever that will be allowed. The default value is 5.
Step 5	**ip http timeout-policy idle** *seconds* **life** *seconds* **requests** *value* **Example:** `Device(config)# ip http timeout-policy idle 600 life 86400 requests 86400`	Sets the characteristics that determine how long a connection to the HTTP server should remain open. The characteristics are: • **idle—**The maximum number of seconds the connection will be kept open if no data is received or response data can not be sent out on the connection. Note that a new value may not take effect on any already existing connections. If the server is too busy or the limit on the life time or the number of requests is reached, the connection may be closed sooner. The default value is 180 seconds (3 minutes). • **life—**The maximum number of seconds the connection will be kept open, from the time the connection is established. Note that the new value may not take effect on any already existing connections. If the server is too busy or the limit on the idle time or the number of requests is reached, it may close the connection sooner. Also, since the server will not close the connection while actively processing a request, the connection may remain open longer than the specified life time if processing is occurring when the life maximum is reached. In this case, the connection will be closed when processing finishes. The default value is 180 seconds (3 minutes). The maximum value is 86400 seconds (24 hours). • **requests—**The maximum limit on the number of requests processed on a persistent connection before it is closed. Note that the new value may not take effect on any already existing connections. If the server is too busy or the limit on the idle time or the life time is reached, the connection may be closed before the maximum number of requests are processed. The default value is 1. The maximum value is 86400.
Step 6	**http client connection idle timeout** *seconds* **Example:** `Device(config)# http client connection idle timeout 600`	Sets the number of seconds that the client waits in the idle state until it closes the connection.
Step 7	**uc wsapi** **Example:** `Device(config)# uc wsapi`	Enters Cisco Unified Communication IOS Service configuration mode.

	Command or Action	Purpose
Step 8	**message-exchange max-failures** *number* **Example:** `Device(config-uc-wsapi)# message-exchange max-failures 2`	Configures the maximum number of failed message exchanges between the application and the provider before the provider stops sending messages to the application. Range is 1 to 3. Default is 1.
Step 9	**probing max-failures** *number* **Example:** `Device(config-uc-wsapi)# probing max-failures 5`	Configures the maximum number of failed probing messages before the router unregisters the application. Range is 1 to 5. Default is 3.
Step 10	**probing interval keepalive** *seconds* **Example:** `Device(config-uc-wsapi)# probing interval keepalive 255`	Configures the maximum number of failed probing messages before the router unregisters the application. Range is 1 to 5. Default is 3.
Step 11	**probing interval negative** *seconds* **Example:** `Device(config-uc-wsapi)# probing interval negative 10`	Configures the interval between negative probing messages, in seconds.
Step 12	**source-address** *ip-address* **Example:** `Device(config-uc-wsapi)# source-address 192.1.12.14`	Configures the IP address (hostname) as the source IP address for the UC IOS service. **Note** The source IP address is used by the provider in the NotifyProviderStatus messages.
Step 13	**end** **Example:** `Device(config-uc-wsapi)# end`	Returns to privileged EXEC mode.

Configuring the XMF Provider

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **uc wsapi**
4. **source-address** *ip address*
5. **provider xmf**
6. **no shutdown**
7. **remote-url** *index url*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**uc wsapi** **Example:** `Device(config)# uc wsapi`	Enters Cisco Unified Communication IOS Service configuration mode.
Step 4	**source-address** *ip address* **Example:** `Device(config)# source-address 172.156.19.38`	Configures the source ip address.
Step 5	**provider xmf** **Example:** `Device(config-uc-wsapi)# provider xmf`	Enters XMF provider configuration mode.

	Command or Action	Purpose
Step 6	**no shutdown** **Example:** `Device(config-uc-wsapi)# no shutdown`	Activates XMF provider.
Step 7	**remote-url** *index url* **Example:** `Device(config-uc-wsapi)# remote-url 1 http://test.com:8090/ucm_xmf`	Specifies the URL (IP address and port number) that the application uses to communicate with XMF provider. The XMF provider uses the IP address and port to authenticate incoming requests.
Step 8	**end** **Example:** `Device(config-uc-wsapi)# end`	Returns to privileged EXEC mode.

Verifying the UC Gateway Services

The **show** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show wsapi registration all**
3. **show wsapi registration xmf** *remote-url-index*
4. **show call media-forking**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show wsapi registration all**

Displays the details of applications registered. Each registered application is identified by a different ID.

Example:

```
Device# show wsapi registration all

 Provider XMF
=====================================================
registration index: 11
  id: 2E7C3034:XMF:myapp:26
  appUrl:http://pascal-lnx.cisco.com:8094/xmf
  appName: myapp
  provUrl: http://9.45.46.16:8090/cisco_xmf
  prober state: STEADY
  connEventsFilter:
CREATED|REDIRECTED|ALERTING|CONNECTED|TRANSFERRED|CALL_DELIVERY|DISCONNECTED|HANDOFF_JOIN|HANDOFF_LEAVE

  mediaEventsFilter: DTMF|MEDIA_ACTIVITY|MODE_CHANGE|TONE_DIAL|TONE_OUT_OF_SERVICE|TONE_SECOND_DIAL

registration index: 1
  id: 2E7C304A:XMF:myapp:27
  appUrl:http://pascal-lnx.cisco.com:8092/xmf
  appName: myapp
  provUrl: http://9.45.46.16:8090/cisco_xmf
  prober state: STEADY
  connEventsFilter:
CREATED|REDIRECTED|ALERTING|CONNECTED|TRANSFERRED|CALL_DELIVERY|DISCONNECTED|HANDOFF_JOIN|HANDOFF_LEAVE

  mediaEventsFilter: DTMF|MEDIA_ACTIVITY|MODE_CHANGE|TONE_DIAL|TONE_OUT_OF_SERVICE|TONE_SECOND_DIAL

registration index: 21
  id: 2E7C6423:XMF:myapp:28
  appUrl:http://pascal-lnx.cisco.com:8096/xmf
  appName: myapp
  provUrl: http://9.45.46.16:8090/cisco_xmf
  prober state: STEADY
  connEventsFilter:
CREATED|REDIRECTED|ALERTING|CONNECTED|TRANSFERRED|CALL_DELIVERY|DISCONNECTED|HANDOFF_JOIN|HANDOFF_LEAVE

  mediaEventsFilter: DTMF|MEDIA_ACTIVITY|MODE_CHANGE|TONE_DIAL|TONE_OUT_OF_SERVICE|TONE_SECOND_DIAL

registration index: 31
  id: 2E7C69E8:XMF:myapp:29
  appUrl:http://pascal-lnx.cisco.com:8098/xmf
  appName: myapp
  provUrl: http://9.45.46.16:8090/cisco_xmf
  prober state: STEADY
  connEventsFilter:
CREATED|REDIRECTED|ALERTING|CONNECTED|TRANSFERRED|CALL_DELIVERY|DISCONNECTED|HANDOFF_JOIN|HANDOFF_LEAVE

  mediaEventsFilter: DTMF|MEDIA_ACTIVITY|MODE_CHANGE|TONE_DIAL|TONE_OUT_OF_SERVICE|TONE_SECOND_DIAL
```

Step 3 **show wsapi registration xmf** *remote-url-index*

Displays the details of only a particular XMF registered application with any ID ranging from 1 to 32.

Example:

```
Device# show wsapi registration xmf 1

Provider XMF
=====================================================
registration index: 1
  id: 2E7C6423:XMF:myapp:28
  appUrl:http://pascal-lnx.cisco.com:8096/xmf
  appName: myapp
  provUrl: http://9.45.46.16:8090/cisco_xmf
  prober state: STEADY
  connEventsFilter:
CREATED|REDIRECTED|ALERTING|CONNECTED|TRANSFERRED|CALL_DELIVERY|DISCONNECTED|HANDOFF_JOIN|HANDOFF_LEAVE
```

```
mediaEventsFilter: DTMF|MEDIA_ACTIVITY|MODE_CHANGE|TONE_DIAL|TONE_OUT_OF_SERVICE|TONE_SECOND_DIAL
```

Step 4 **show call media-forking**

Displays the forked stream information.

Example:

```
Device# show call media-forking

Warning: Output may be truncated if sessions are added/removed concurrently!

Session    Call     n/f  Destination (port address)
187        BA       near 45864 10.104.105.232
188        BA       far  54922 10.104.105.232
189        B9       near 45864 10.104.105.232
190        B9       far  54922 10.104.105.232

FORK _DONE Notifications

//WSAPI/INFRA/wsapi_send_outbound_message_by_provider_info:
*Dec 21 10:31:21.016 IST: //WSAPI/INFRA/0/9/546CF8:25:tx_contextp 15898C1C tx_id 19 context1 (0 0)
context2 (9 9):
out_url http://gauss-lnx.cisco.com:8081/xmf*Dec 21 10:31:21.020 IST:
wsapi_send_outbound_message_by_provider_info:
<?xml version="1.0" encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.w3.org/2003/05/soap-envelope"><SOAP:Body>
<NotifyXmfConnectionData xmlns="http://www.cisco.com/schema/cisco_xmf/v1_0"><msgHeader><transactionID>
546CF8:25</transactionID><registrationID>4CA5E4:XMF:myapp:4</registrationID></msgHeader><callData><callID>25</callID><state>
ACTIVE</state></callData><connData><connID>132</connID><state>ALERTING</state></connData><event><mediaForking>
<mediaForkingState>FORK_DONE</mediaForkingState></mediaForking></event></NotifyXmfConnectionData></SOAP:Body></SOAP:Envelope>

FORK_FAILED Notification

//WSAPI/INFRA/wsapi_send_outbound_message_by_provider_info:
*Dec 21 10:31:21.016 IST: //WSAPI/INFRA/0/9/546CF8:25:tx_contextp 15898C1C tx_id 19 context1 (0 0)
context2 (9 9):
out_url http://gauss-lnx.cisco.com:8081/xmf*Dec 21 10:31:21.020 IST:
wsapi_send_outbound_message_by_provider_info:
<?xml version="1.0" encoding="UTF-8"?><SOAP:Envelope
xmlns:SOAP="http://www.w3.org/2003/05/soap-envelope"><SOAP:Body>
<NotifyXmfConnectionData xmlns="http://www.cisco.com/schema/cisco_xmf/v1_0"><msgHeader><transactionID>
546CF8:25</transactionID><registrationID>4CA5E4:XMF:myapp:4</registrationID></msgHeader><callData><callID>25</callID><state>
ACTIVE</state></callData><connData><connID>132</connID><state>ALERTING</state></connData><event><mediaForking>
<mediaForkingState>FORK_FAILED</mediaForkingState></mediaForking></event></NotifyXmfConnectionData></SOAP:Body>
</SOAP:Envelope>
```

Troubleshooting Tips

You can use the following **debug** commands to troubleshoot the UC Gateway Services configurations.

- **debug wsapi infrastructure all**
- **debug wsapi xcc all**
- **debug wsapi xmf all**
- **debug wsapi xmf messages**
- **debug wsapi infrastructure detail**
- **debug voip application**

- **debug voip application media forking**

Configuration Examples for UC Gateway Services

Example: Configuring Cisco Unified Communication IOS Services

The following example shows how to configure the device for Cisco Unified Communication IOS Services and enable the HTTP server:

```
Device> enable
Device# configure terminal
Device(config)# ip http server
Device(config)# ip http max-connection 100
Device(config)# ip http timeout-policy idle 600 life 86400 requests 86400
Device(config)# http client connection idle timeout 600
Device(config)# uc wsapi
Device(config-uc-wsapi)# message-exchange max-failures 2
Device(config-uc-wsapi)# probing max-failures 5
Device(config-uc-wsapi)# probing interval keepalive 255
Device(config-uc-wsapi)# probing interval negative 10
Device(config-uc-wsapi)# source-address 192.1.12.14
Device(config-uc-wsapi)# end
```

Example: Configuring the XMF Provider

The following example shows how to enable the XMF providers. The configuration specifies the address and port that the application uses to communicate with the XMF provider:

```
Device> enable
Device# configure terminal
Device(config)# uc wsapi
Device(config-uc-wsapi)# provider xmf
Device(config-uc-wsapi)# no shutdown
Device(config-uc-wsapi)# remote-url 1 http://test.com:8090/ucm_xmf
Device(config-uc-wsapi)# end
```

Example: Configuring UC Gateway Services

```
uc wsapi
 message-exchange max-failures 5
 response-timeout 10
 source-address 192.1.12.14
 probing interval negative 20
 probing interval keepalive 250
 !
 provider xmf
  remote-url 1 http://pascal-lnx.cisco.com:8050/ucm_xmf
```

CHAPTER 14

Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls

The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature provides dynamic payload type interworking for dual tone multifrequency (DTMF) and codec packets for Session Initiation Protocol (SIP) to SIP calls.

Based on this feature, the Cisco Unified Border Element (Cisco UBE) interworks between different dynamic payload type values across the call legs for the same codec. Also, Cisco UBE supports any payload type value for audio, video, named signaling events (NSEs), and named telephone events (NTEs) in the dynamic payload type range 96 to 127.

Feature Information for Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 14: Feature Information for Dynamic Payload Interworking for DTMF and Codec Packets Support

Feature Name	Releases	Feature Information
Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls	15.0(1)XA 15.1(1)T	The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature provides dynamic payload type interworking for DTMF and codec packets for SIP-to-SIP calls. The following commands were introduced or modified: **asymmetric payload** and **voice-class sip asymmetric payload**.
Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls	Cisco IOS Release XE 3.1S	The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature provides dynamic payload type interworking for DTMF and codec packets for SIP-to-SIP calls. The following commands were introduced or modified: **asymmetric payload** and **voice-class sip asymmetric payload**.
High Availability Checkpointing Support for Asymmetric Payload	15.4(2)T	High availability support for asymmetric payload type interworking was added.

Restrictions for Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls

The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature is not supported for the following:

- H323-to-H323 and H323-to-SIP calls.
- All transcoded calls.
- Secure Real-Time Protocol (SRTP) pass-through calls.
- Flow-around calls.
- Asymmetric payload types are not supported on early-offer (EO) call legs in a delayed-offer to early-offer (DO-EO) scenario.
- Cisco fax relay.

- Multiple *m* lines with the same dynamic payload types, where *m* is:

m = audio <media-port1> RTP/AVP XXX m = video <media-port2> RTP/AVP XXX

Symmetric and Asymmetric Calls

Cisco UBE supports dynamic payload type negotiation and interworking for all symmetric and asymmetric payload type combinations. A call leg on Cisco UBE is considered as symmetric or asymmetric based on the payload type value exchanged during the offer and answer with the endpoint:

- A symmetric endpoint accepts and sends the same payload type.
- An asymmetric endpoint can accept and send different payload types.

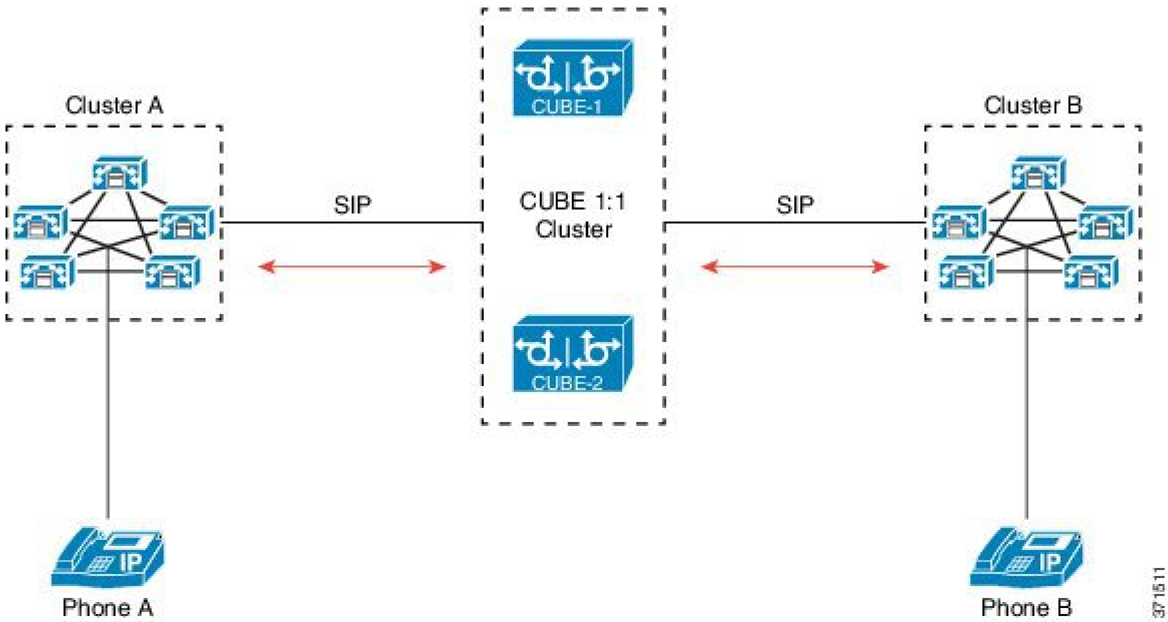
The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature is enabled by default for a symmetric call. An offer is sent with a payload type based on the dial-peer configuration. The answer is sent with the same payload type as was received in the incoming offer. When the payload type values negotiated during the signaling are different, the Cisco UBE changes the Real-Time Transport Protocol (RTP) payload value in the VoIP to RTP media path.

To support asymmetric call legs, you must enable The Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature. The dynamic payload type value is passed across the call legs, and the RTP payload type interworking is not required. The RTP payload type handling is dependent on the endpoint receiving them.

High Availability Checkpointing Support for Asymmetric Payload

High availability for a call involving asymmetric payloads is supported. In case of fail-over from active to stand-by, the asymmetric payload interworking will be continued as new active CUBE passes across the payload type values according to the negotiation and call establishment.

Figure 5: Sample High-Availability Topology



How to Configure Dynamic Payload Type Passthrough for DTMF and Codec Packets for SIP-to-SIP Calls

Configuring Dynamic Payload Type Passthrough at the Global Level

Perform this task to configure the pass through of DTMF or codec payload to the other call leg (instead of performing dynamic payload type interworking) feature at the global level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **asymmetric payload** {**dtmf** | **dynamic-codecs** | **full** | **system**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable` **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters voice service configuration mode.
Step 4	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters voice service SIP configuration mode.
Step 5	**asymmetric payload** {**dtmf** | **dynamic-codecs** | **full** | **system**} **Example:** `Device(conf-serv-sip)# asymmetric payload full`	Configures global SIP asymmetric payload support. **Note** The **dtmf** and **dynamic-codecs** keywords are internally mapped to the **full** keyword to provide asymmetric payload type support for audio and video codecs, DTMF, and NSEs.
Step 6	**end** **Example:** `Device(conf-serv-sip)# end`	Exits voice service SIP configuration mode and enters privileged EXEC mode.

Configuring Dynamic Payload Type Passthrough for a Dial Peer

Perform this task to configure the pass through of DTMF or codec payload to the other call leg (instead of performing dynamic payload type interworking) feature at the dial-peer level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **voice-class sip asymmetric payload** {**dtmf** | **dynamic-codecs** | **full** | **system**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 77 voip`	Enters dial peer voice configuration mode.
Step 4	**voice-class sip asymmetric payload** {**dtmf** | **dynamic-codecs** | **full** | **system**} **Example:** `Device(config-dial-peer)# voice-class sip asymmetric payload full`	Configures the dynamic SIP asymmetric payload support. **Note** The **dtmf** and **dynamic-codecs** keywords are internally mapped to the **full** keyword to provide asymmetric payload type support for audio and video codecs, DTMF, and NSEs.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and enters privileged EXEC mode.

Verifying Dynamic Payload Interworking for DTMF and Codec Packets Support

This task shows how to display information to verify Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls configuration feature. These **show** commands need not be entered in any specific order.

SUMMARY STEPS

1. **enable**
2. **show call active voice compact**
3. **show call active voice**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show call active voice compact** **Example:** `Device# show call active voice compact`	(Optional) Displays a compact version of call information.
Step 3	**show call active voice** **Example:** `Device# show call active voice`	(Optional) Displays call information for voice calls in progress.

Troubleshooting Tips

Use the following commands to debug any errors that you may encounter when you configure the Dynamic Payload Type Interworking for DTMF and Codec Packets for SIP-to-SIP Calls feature:

- **debug ccsip all**
- **debug voip ccapi inout**
- **debug voip rtp**

Use the following debug commands to troubleshoot HA Checkpointing for Asymmetric Payload:

- **debug voip ccapi all**

- **debug voice high-availability all**
- **debug voip rtp error**
- **debug voip rtp inout**
- **debug voip rtp packet**
- **debug voip rtp high-availability**
- **debug voip rtp function**
- **debug ccsip all**

Use the following **show** commands to troubleshoot HA Checkpointing for Asymmetric Payload:

- **show redundancy state**
- **show redundancy inter-device**
- **show standby brief**
- **show voice high-availability summary**
- **show voip rtp stats**
- **show voip rtp high-availability stats**
- **show voip rtp connection detail**
- **show call active voice brief**
- **show call active voice [summary]**
- **show call active video brief**
- **show call active video [summary]**
- **show align**
- **show memory debug leak**

Configuration Examples for Assymetric Payload Interworking

Example: Asymmetric Payload Interworking—Passthrough Configuration

```
!
voice service voip
 allow-connections sip to sip
sip
  rel1xx disable
  asymmetric payload full
  midcall-signaling passthru
!
dial-peer voice 1 voip
 voice-class sip asymmetric payload full
 session protocol sipv2
 rtp payload-type cisco-codec-fax-ind 110
 rtp payload-type cisco-codec-video-h264 112
```

```
 session target ipv4:9.13.8.23
!
```

Example: Asymmetric Payload Interworking—Interworking Configuration

```
!
voice service voip
 allow-connections sip to sip
!
dial-peer voice 1 voip
 session protocol sipv2
 rtp payload-type cisco-codec-fax-ind 110
 rtp payload-type cisco-codec-video-h264 112
 session target ipv4:9.13.8.23
!
```

CHAPTER 15

Acoustic Shock Protection

Acoustic Shock Protection (ASP) is a voice circuit-breaker feature that is designed to protect users, especially those wearing headsets, from exposure to loud, sustained, and piercing tones, such as those produced by a fax machine. It is a workplace-safety feature for voice calls. When the tone is present at the input of the ASP module, the audio path in the affected direction is muted to protect the listener, and a gentle alert tone is played out for as long as the tone persists. ASP may be inserted in either or both directions of a call, that is, applied to incoming packets to protect the ears of a listener on the Time-Division Multiplexing (TDM) gateway, applied to incoming PSTN calls (microphone signal) to protect the ears of listeners at the other end of the call, or applied to both simultaneously.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for ASP

- Supported on PVDM3 only.
- Supported only on flex codec complexity.

- No support for H.32x video call, complex forking calls, and fax and modem calls.
- No support for TDM hairpin call.
- The configuration under dial peer has higher priority than the configuration at the global level.
- No support for conference calls, IP/SIP phones, and the Skinny Client Control Protocol (SCCP).
- CLI supports enabling ASP but not disabling ASP.
- No support for dynamically enabling or disabling ASP during a call.

Information About ASP

Acoustic Shock Protection

Acoustic Shock Protection (ASP) is an adaptive signal processing algorithm on the Digital Signal Processor (DSP) that analyzes incoming audio for the presence of offending tones that might harm humans. Offending tones include signals that are:

- Loud
- Tonal (energy concentrated around a single frequency)
- Persistent (lasts longer than a few tens of milliseconds)

If an offending tone is present, the audio path in that direction is muted temporarily, and a quiet, alerting signal is played out to the listener side. The call is never dropped; only the audio is muted temporarily. If or when the tone disappears from the input, the mute is removed. ASP does not disrupt low-frequency tones (below 650 Hz) such as ringback, dial, and so forth. Since ASP is designed to mute only single-frequency tones, it allows multi-tone signals such as Dual Tone Multi-Frequency (DTMF) to pass unhindered. ASP is supported on TDM gateways (TDM-VoIP and TDM-TDM) and on the Cisco Unified Border Element (Cisco UBE).

Note ASP is for voice calls only and not for faxes and modems.

Some of the best practices for ASP are as follows:

- Use default values
- Use ASP on dial peers where you are certain that people (not faxes) are listening.
- Do not use ASP on dial peers associated with fax machines, modems, or TTY/TDD devices. Use fax-relay or modem-relay modes on dial peers dedicated to such devices.
- ASP is designed for deployment in situations where customers have experienced acoustic shock safety issues. If there are issues like false triggering (for example, ASP alerts on regular voices), then you must turn off ASP. You can choose from three detector sensitivity modes: slow, auto, or fast. Fast mode is a highly sensitive hair-trigger. Auto mode is recommended. Slow mode lets more tone leak through, but has better rejection of false triggers.

How to Configure ASP

Creating the Media Profile for ASP

Perform this task to create a media profile to configure acoustic shock protection.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile asp** *tag*
4. **mode** *mode*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile asp** *tag* **Example:** `Device(config)# media profile asp 5`	Creates the media profile to configure ASP and enters media profile configuration mode. The range for the media profile tag is from 1 to 10000.
Step 4	**mode** *mode* **Example:** `Device(cfg-mediaprofile)# mode auto`	Sets the ASP sensitivity mode to preset = auto (which is default). Auto mode provides a good tradeoff between ASP speed and false trigger rejection. The other modes are: • slow—Presets ASP sensitivity mode to 1. This mode provides slower detection speed for reduced chance of false triggers. • fast—Presets ASP sensitivity mode to 2. This mode provides faster detection speed but higher chance of false triggers. • expert—This mode exposes direct control of individual ASP parameters and is recommended for test use only.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Creating the Media Profile to Enable ASP

After the media profile is created, you must create a media class to enable acoustic shock protection. Perform this task to create a media class.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media class** *tag*
4. **asp profile** *tag*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media class** *tag* **Example:** `Device(config)# media class 2`	Creates the media class to enable the acoustic shock protection feature and enters media class configuration mode. The range for the media class tag is from 1 to 10000.

	Command or Action	Purpose
Step 4	**asp profile** *tag* **Example:** `Device(cfg-mediaclass)# asp profile 200`	Applies the media profile to the media class. The range for the media profile ASP tag is from 1 to 10000.
Step 5	**end** **Example:** `Device(cfg-mediaclass)# end`	Returns to privileged EXEC mode.

Configuring the Media Class at a Dial Peer Level for ASP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **pots**
4. **media-class** *tag*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **pots** **Example:** `Device(config)# dial-peer voice 20 pots`	Defines a particular dial peer and enters dial-peer voice configuration mode. The range for the dial-peer voice tag is from 1 to 1073741823.

	Command or Action	Purpose
Step 4	**media-class** *tag* **Example:** `Device(config-dial-peer)# media-class 2`	Applies the media class to the specific dial peer. The range for the media class tag number is from 1 to 10000.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	Returns to privileged EXEC mode.

Configuring the Media Class Globally for ASP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media service**
4. **enhancement**
5. **tdm** *tag*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media service** **Example:** `Device(config)# media service`	Enters media service configuration mode.

	Command or Action	Purpose
Step 4	**enhancement** **Example:** `Device(cfg-mediaservice)# enhancement`	Enters the submode enhance of media service.
Step 5	**tdm** *tag* **Example:** `Device(cfg-service-enhance)# tdm 2`	Applies the TDM call globally. The range for the media class tag number is from 1 to 10000.
Step 6	**end** **Example:** `Device(config-dial-peer)# end`	Returns to privileged EXEC mode.

Verifying ASP

Perform this task to verify the voice quality metrics.

SUMMARY STEPS

1. **enable**
2. **show call active voice stats | b pid:**

DETAILED STEPS

Step 1 **enable**

Example:

`Device>` **enable**

Enables privileged EXEC mode.

Step 2 **show call active voice stats | b pid:**

Example:

`Device#` **show call active voice stats | b pid:1300**

```
11EC : 5 09:14:25.971 PDT Thu Jul 28 2011.1 +1130 pid:1300 Answer 1300 active dur 00:01:36 tx:17/321
 rx:17/321 dscp:0 media:0
DSP/TX: PK=17, SG=0, NS=1, DU=90570, VO=320
DSP/RX: PK=17, SG=0, CF=1, RX=90570, VO=320, BS=0, BP=0, LP=0, EP=0
….
```

```
DSP/DL: RT=0, ED=0
MIC Direction:
DSP/NR: NR=1, ND=0, LV=257, IN=1, PN=0, ON=0
DSP/AS: AE=1, AD=0, AV=0, AM=0, NT=0, DT=0, TT=0, TD=0, LF=0, LD=0
EAR Direction:
DSP/NR: NR=0, ND=0, LV=0, IN=0, PN=0, ON=0
DSP/AS: AE=0, AD=0, AV=0, AM=0, NT=0, DT=0, TT=0, TD=0, LF=0, LD=0
11EC : 6 09:14:25.973 PDT Thu Jul 28 2011.2 +1130 pid:2300 Originate 2300 active dur 00:01:36 tx:17/457
 rx:17/321 dscp:0 media:0
Telephony call-legs: 1
SIP call-legs: 0
H323 call-legs: 1
```

Displays information about digital signal processing (DSP) voice quality metrics.

Troubleshooting Tips

The following commands can help troubleshoot ASP:

- **debug voip hpi all**
- **debug voip dsmp all**
- **debug voip dsm all**
- **debug voip vtsp all**
- **debug vpm dsp all**

Configuration Examples for the Acoustic Shock Protection Feature

Example: Enabling ASP Globally

```
media profile asp 6
!
media class 1
  asp profile 6
!
media service
  enhancement
    tdm 1
```

Example: Enabling ASP on a Dial Peer

```
media profile asp 4
!
media class 1
  asp profile 4
!
dial-peer voice 2100 pots
  destination-pattern 2100
  incoming called-number 1100
```

```
 media-class 1
 port 0/2/0:1
 forward-digits all
dial-peer voice 1300 voip
destination-pattern 1300 session target ipv4:1.2.146.102 media-class 1
```

Feature Information for Acoustic Shock Protection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 15: Feature Information for Acoustic Shock Protection

Feature Name	Releases	Feature Information
Acoustic Shock Protection	15.2(2)T, 15.2(3)T	Acoustic Shock Protection (ASP) is a voice circuit-breaker feature that is designed to protect users, especially those wearing headsets, from exposure to loud, sustained, and piercing tones, such as those produced by a fax machine. It is a workplace-safety feature for voice calls. ASP is supported on TDM gateways and on Cisco UBE. The following commands were introduced or modified: **media profile asp**, **media service**.
Acoustic Shock Protection	Cisco IOS XE Release 3.6S	Acoustic Shock Protection (ASP) is a voice circuit-breaker feature that is designed to protect users, especially those wearing headsets, from exposure to loud, sustained, and piercing tones, such as those produced by a fax machine. It is a workplace-safety feature for voice calls. ASP is supported on TDM gateways and on Cisco UBE. In Cisco IOS XE Release 3.6S, this feature was implemented on the Cisco Unified Border Element (Enterprise) The following commands were introduced or modified: **media profile asp**, **media service**.

CHAPTER 16

Noise Reduction

Noise Reduction (NR) is a voice enhancement process that improves the quality of incoming speech that has already been corrupted with background noise; for example, a voice conference participant speaking on a cell-phone in a car. NR works best with steady state broadband noises like engine noise but not as well with impulsive noises like nearby chatter.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Noise Reduction

Cisco Unified Border Element

- Cisco IOS Release 15.2(2)T, or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.6S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for NR

- Supported only on PVDM3.
- Supported only on flex codec complexity.
- No support for H.32x video call, complex forking calls, and fax and modem calls.
- No support for Time-Division Multiplexing (TDM) hairpin call.
- Configurations under POTS dial peer has higher priority over VoIP dial peer for NR.
- Configurations under the dial peer has higher priority than configurations at the global level.
- No support for conference calls, IP/SIP phones, and the Skinny Client Control Protocol (SCCP).
- CLI supports enabling NR but not disabling NR.
- No support for dynamically enabling or disabling NR during a call.

Information About NR

Noise Reduction

Noise Reduction (NR) is an adaptive signal processing algorithm on the Digital Signal Processor (DSP) that analyzes incoming audio, extracts a fingerprint of the background noise during talker pauses, and then performs ongoing spectral subtraction of this noise after a short training period (a few seconds). NR constantly adapts to changes in background noises over time.

NR can affect music on hold signals by making the music quieter. NR may disrupt fax/modem/TDD devices, although it is designed to self-disable in those cases. Use modem-relay mode for reliable fax/modem transmission. NR is supported on TDM gateways (TDM-VoIP and TDM-TDM) and on the Cisco Unified Border Element (Cisco UBE).

Some of the best practices for NR are as follows:

- Use default values.
- Do not use NR on dial peers associated with fax machines. Use fax or modem-relay modes for those dial peers.
- NR, when used without dynamic user control of intensity (as is the case with gateways), must be used at a low intensity (default or lower) since it is always on. High intensity is dramatic for demonstrations with loud background noises, but the NR process itself will degrade “normal” calls if NR is run at high intensity.

How to Configure NR

Creating the Media Profile for NR

Perform this task to create a media profile to configure noise reduction parameters.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media profile nr** *tag*
4. **intensity** *level*
5. **noisefloor** *level*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media profile nr** *tag* **Example:** `Device(config)# media profile nr 2`	Creates the media profile to configure noise reduction parameters and enters media profile configuration mode. The range for the media profile tag is from 1 to 10000.
Step 4	**intensity** *level* **Example:** `Device(cfg-mediaprofile)# intensity 2`	Configures the intensity level or depth of the noise reduction process. The range is from 0 to 6.
Step 5	**noisefloor** *level* **Example:** `Device(cfg-mediaprofile)# noisefloor -50`	Configures the noise level, in dBm, above which NR will operate. NR will allow noises quieter than this level to pass without processing. The range is from -58 to -20.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config)# end`	Returns to the privileged EXEC mode.

Creating the Media Class to Enable NR

After the media profile is created, you must create a media class to enable noise reduction. Perform this task to create a media class.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media class** *tag*
4. **nr profile** *tag*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**media class** *tag* **Example:** `Device(config)# media class 2`	Creates the media class to enable the noise reduction feature and enters media class configuration mode. The range for the media class tag is from 1 to 10000.

	Command or Action	Purpose
Step 4	**nr profile** *tag* **Example:** `Device(cfg-mediaclass)# nr profile 200`	Applies the media profile to the media class. The range for the media profile NR tag is from 1 to 10000.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuring the Media Class at a Dial Peer Level for NR

Perform this task to configure the media class for a dial peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **pots**
4. **media-class** *tag*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **pots** **Example:** `Device(config)# dial-peer voice 20 pots`	Defines a particular dial peer and enters the dial-peer voice configuration mode. The range for the dial-peer voice tag is from 1 to 1073741823.

	Command or Action	Purpose
Step 4	**media-class** *tag* **Example:** `Device(config-dial-peer)# media-class 2`	Applies the media class to the specific dial peer. The range for the media class tag number is from 1 to 10000.
Step 5	**end** **Example:** `Device(config-dial-peer)# end`	Returns to the privileged EXEC mode.

Configuring the Media Class Globally for NR

Perform this task to configure a media class globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **media service**
4. **enhancement**
5. **tdm** *tag*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**media service** **Example:** `Device(config)# media service`	Enters media service configuration mode.
Step 4	**enhancement** **Example:** `Device(cfg-mediaservice)# enhancement`	Enters the submode enhance of media service.
Step 5	**tdm** *tag* **Example:** `Device(cfg-service-enhance)# tdm 2`	Applies the TDM call globally. The range for the media class tag number is from 1 to 10000.
Step 6	**end** **Example:** `Device(config-dial-peer)# end`	Returns to the privileged EXEC mode.

Verifying NR

Perform this task to verify the voice quality metrics.

SUMMARY STEPS

1. **enable**
2. **show call active voice stats | b pid:**

DETAILED STEPS

Step 1 **enable**

Example:
```
Device> enable
```

Enables privileged EXEC mode.

Step 2 **show call active voice stats | b pid:**

Example:

Device# **show call active voice stats | b pid:1300**

```
11EC : 5 09:14:25.971 PDT Thu Jul 28 2011.1 +1130 pid:1300 Answer 1300 active dur 00:01:36 tx:17/321
 rx:17/321 dscp:0 media:0
DSP/TX: PK=17, SG=0, NS=1, DU=90570, VO=320
DSP/RX: PK=17, SG=0, CF=1, RX=90570, VO=320, BS=0, BP=0, LP=0, EP=0
….
DSP/DL: RT=0, ED=0
MIC Direction:
DSP/NR: NR=1, ND=0, LV=257, IN=1, PN=0, ON=0
DSP/AS: AE=1, AD=0, AV=0, AM=0, NT=0, DT=0, TT=0, TD=0, LF=0, LD=0
EAR Direction:
DSP/NR: NR=0, ND=0, LV=0, IN=0, PN=0, ON=0
DSP/AS: AE=0, AD=0, AV=0, AM=0, NT=0, DT=0, TT=0, TD=0, LF=0, LD=0
11EC : 6 09:14:25.973 PDT Thu Jul 28 2011.2 +1130 pid:2300 Originate 2300 active dur 00:01:36 tx:17/457
 rx:17/321 dscp:0 media:0
Telephony call-legs: 1
SIP call-legs: 0
H323 call-legs: 1
```

Displays information about digital signal processing (DSP) voice quality metrics.

Troubleshooting Tips

The following commands can help troubleshoot NR:

- **debug voip hpi all**
- **debug voip dsmp all**
- **debug voip dsm all**
- **debug voip vtsp all**
- **debug vpm dsp all**

Configuration Examples for the NR feature

Example: Enabling NR globally

```
media profile nr 1
 intensity 1
!
media profile nr 2
!
media profile nr 3
 intensity 2
!
media profile nr 4
 intensity 3
!
media profile nr 5
 intensity 2
```

```
!
media profile nr 7
 intensity 2
!
media profile asp 6
!
media class 1
 nr profile 5
 asp profile 6
!
media service
 enhancement
  tdm 1
```

Example: Enabling NR on a Dial Peer

```
media profile nr 1
 intensity 1
!
media profile nr 2
 intensity 2
!
media profile nr 3
 intensity 2
!
media profile asp 4
!
media class 1
 nr profile 2
 asp profile 4
!
dial-peer voice 2100 pots
 destination-pattern 2100
 incoming called-number 1100
 media-class 1
 port 0/2/0:1
 forward-digits all

dial-peer voice 1300 voip
 destination-pattern 1300
 session target ipv4:1.2.146.102
 media-class 1
```

Feature Information for Noise Reduction

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 16: Feature Information for Noise Reduction

Feature Name	Releases	Feature Information
Noise Reduction	15.2(2)T, 15.2(3)T	Noise Reduction (NR) is a voice enhancement or restoration process that improves the quality of incoming speech that has already been corrupted with background noise. NR is supported on TDM gateways and on the Cisco UBE. The following commands were introduced or modified: **intensity**, **media profile nr**, **media service**, and **noisefloor**.
Noise Reduction	Cisco IOS XE Release 3.6S	Noise Reduction (NR) is a voice enhancement or restoration process that improves the quality of incoming speech that has already been corrupted with background noise. NR is supported on TDM gateways and on Cisco UBE. In Cisco IOS XE Release 3.6S, this feature was implemented on the Cisco Unified Border Element (Enterprise). The following commands were introduced or modified: **intensity**, **media profile nr**, **media service**, **noisefloor**.

CHAPTER 17

iLBC Support for SIP and H.323

The internet Low Bitrate Codec (iLBC) is a standard, high-complexity speech codec suitable for robust voice communication over IP. The iLBC has built-in error correction functionality that helps the codec perform in networks with high-packet loss. This codec is supported on both Session Initiation Protocol (SIP) and H.323.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for iLBC Support for SIP and H.323

Cisco Unified Border Element

- Cisco IOS Release 12.2(11)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for iLBC Support for SIP and H.323

The iLBC Support for SIP and H.323 feature is supported on the following:

- IP-to-IP gateways with no transcoding and conferencing
- All c5510 DSP-based platforms

Information About iLBC Support for SIP and H.323

The internet Low Bit Rate Codec (iLBC) is designed for narrow band speech and results in a payload bit rate of 13.33 kbits per second for 30-millisecond (ms) frames and 15.20 kbits per second for 20 ms frames.

When the codec operates at block lengths of 20 ms, it produces 304 bits per block, which is packetized as defined in RFC 3952. Similarly, for block lengths of 30 ms it produces 400 bits per block, which is packetized as defined in RFC 3952.

The iLBC has built-in error correction functionality to provide better performance in networks with higher packet loss.

How to Configure an iLBC Codec

Configuring an iLBC Codec on a Dial Peer

The iLBC is intended for packet-based communication. Perform the following steps to configure the iLBC codec on a dial peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **rtp payload-type cisco-codec-ilbc** [*number*
5. **codec ilbc** [**mode** *frame_size* [**bytes** *payload_size*]]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 10 voip`	Enters dial-peer configuration mode for the VoIP dial peer designated by *tag*.

Step 4	**rtp payload-type cisco-codec-ilbc** [*number*
	
	Example:
	
	`Device(config-dial-peer)# rtp payload-type cisco-codec-ilbc 100`
		Identifies the payload type of a Real-Time Transport Protocol (RTP) packet. Keyword and argument are as follows:
		
		• **cisco-codec-ilbc** [*number*]--Payload type is for internet Low Bit Rate Codec (iLBC). Range: 96 to 127. Default: 116.
		
		Note Do not use the following numbers because they have preassigned values: 96, 97, 100, 117, 121 to 123, and 125 to 127. If you use these values, the command will fail. You must first reassign the value in use to a different unassigned number, for example:
		
		`rtp payload-type nse 105`
		`rtp payload-type cisco-codec-ilbc 100`

Step 5	**codec ilbc** [**mode** *frame_size* [**bytes** *payload_size*]] **Example:** `Device(config-dial-peer)# codec ilbc mode 30 bytes 200`	Specifies the voice coder rate of speech for a dial peer. Keywords and arguments are as follows: • **mode** *frame_size* --The iLBC operating frame mode that will be encapsulated in each packet. Valid entries are 20 (20ms frames for 15.2kbps bit rate) or 30 (30ms frames for 13.33 kbps bit rate). Default is 20. • **bytes** *payload_size* --Number of bytes in an RTP packet. For mode 20, valid values are 38 (default), 76, 114, 152, 190, and 228. For mode 30, valid values are 50(default), 100, 150, and 200.
Step 6	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits the current mode.

Configuring an iLBC Codec in the Voice Class

When using multiple codecs, you must create a voice class in which you define a selection order for codecs; then, you can apply the voice class to VoIP dial peers. The **voice class codec** global configuration command allows you to define the voice class that contains the codec selection order. Then, use the **voice-class codec** dial-peer configuration command to apply the class to individual dial peers.

To configure an iLBC in the voice class for multiple-codec selection order, perform the following steps.

You can configure more than one voice class codec list for your network. Configure the codec lists and apply them to one or more dial peers based on which codecs (and the order) you want supported for the dial peers. Define a selection order if you want more than one codec supported for a given dial peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class codec** *tag*
4. **codec preference** *value* **ilbc** [**mode** *frame_size*] [**bytes** *payload_size*]
5. **exit**
6. **dial-peer voice** *tag* **voip**
7. **voice-class codec** *tag*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice class codec** *tag* **Example:** `Device(config)# voice class codec 99`	Enters voice-class configuration mode and assigns an identification tag number for a codec voice class. The argument is as follows: • *tag* --Unique identifier on the router. Range is 1 to 10000.

	Command or Action	Purpose
Step 4	**codec preference** *value* **ilbc** [**mode** *frame_size*] [**bytes** *payload_size*] **Example:** `Device(config-voice-class)# codec preference 1 ilbc 30 200`	Specifies a list of preferred codecs to use on a dial peer. Keywords and arguments are as follows: • *value* --Order of preference, with 1 being the most preferred and 14 being the least preferred. • **mode** *frame_size* --The iLBC operating frame mode that will be encapsulated in each packet. Valid entries are 20 (20ms frames for 15.2kbps bit rate) or 30 (30ms frames for 13.33 kbps bit rate). Default is 20. • **bytes** *payload_size* --Number of bytes in an RTP packet. For mode 20, valid values are 38 (default), 76, 114, 152, 190, and 228. For mode 30, valid values are 50(default), 100, 150, and 200.
Step 5	**exit** **Example:** `Device(config-voice-class)# exit`	Exits the current mode.
Step 6	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 16 voip`	Enters dial-peer configuration mode for the specified VoIP dial peer.
Step 7	**voice-class codec** *tag* **Example:** `Device(config-dial-peer)# voice-class codec 99`	Assigns a previously configured codec selection preference list (the codec voice class that you defined in step 3) to the specified VoIP dial peer. **Note** The **voice-class codec**command in dial-peer configuration mode contains a hyphen. The **voice class** command in global configuration mode does not contain a hyphen.
Step 8	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits the current mode.

Verifying iLBC Support for SIP and H.323

You can use the following commands to check iLBC status:

- **show voice call summary**
- **show voice call status**

- **show voice dsmp stream**
- **show call active voice**
- **show call history voice**
- **show voice dsp and its extensions**
- **show dial-peer voice**
- **show voice dsp channel operational-status**

Feature Information for iLBC Support for SIP and H.323

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 17: Feature Information for iLBC Support for SIP and H.323

Feature Name	Releases	Feature Information
iLBC Support for SIP and H.323	12.2(11)T 12.2(15)T	The iLBC is a standard, high-complexity speech codec suitable for robust voice communication over IP. The iLBC has built-in error correction functionality that helps the codec perform in networks with high-packet loss. This codec is supported on both Session Initiation Protocol (SIP) and H.323. The following commands were introduced or modified: **codec ilbc**, **codec preference**, and **rtp payload-type**.

Feature Name	Releases	Feature Information
iLBC Support for SIP and H.323	Cisco IOS XE Release 2.5	The iLBC is a standard, high-complexity speech codec suitable for robust voice communication over IP. The iLBC has built-in error correction functionality that helps the codec perform in networks with high-packet loss. This codec is supported on both Session Initiation Protocol (SIP) and H.323. The following commands were introduced or modified: **codec ilbc**, **codec preference**, and **rtp payload-type**.

CHAPTER 18

Configuring RTP Media Loopback for SIP Calls

RTP packets are looped back toward the source device when the RTP Media Loopback for SIP Calls feature is configured on a dial peer. The SIP RTP media loopback can be used during Cisco UBE deployments to make test calls to verify the media path between the endpoints and Cisco UBE. In a voice loopback call, an echo is heard at the device originating the call. In a video loopback call, the locally captured video and the audio echo must be rendered at the source device.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites

- Media packets must be enabled to pass through the gateway.

- Use the media flow-through command in dial peer voice or voice service configuration mode to enable the media packets.

Cisco Unified Border Element

- Cisco IOS Release 15.1(4)M or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.3S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions

- SRTP, DTLS, and STUN are not supported in loopback mode.
- Fax (midcall transmit function change) is not supported.
- RSVP is not supported.
- Call transfer is not supported.

Information About RTP Media Loopback for SIP Calls

Digital Signal Processors (DSP) generate and transmit Real-time Transport Protocol (RTP) media packets from a source to a destination transport address during a SIP call session. However, when a SIP call is put on hold the DSP stops generating the RTP media packets and resumes generating and transmitting these media packets after the SIP call is resumed. This ensures that the RTP sequence number is continuous from the time of the origin to the end of a SIP call.

How to Configure RTP Media Loopback for SIP Calls

RTP packets are looped back toward the source device when the RTP Media Loopback for SIP Calls feature is configured on a dial peer. Perform this task to enable the RTP Media Loopback for SIP Calls feature on a dial peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **destination-pattern** *string*
5. **session protocol sipv2**
6. **session target loopback:rtp**
7. **incoming called-number** *string*
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 77 voip`	Specifies that the dial peer is a VoIP peer and enters dial peer voice configuration mode.
Step 4	**destination-pattern** *string* **Example:** `Router(config-dial-peer)# destination-pattern 77`	Specifies the prefix or the full E.164 number for the dial peer.
Step 5	**session protocol sipv2** **Example:** `Router(config-dial-peer)# session protocol sipv2`	Specifies the session protocol for calls with the SIP option.
Step 6	**session target loopback:rtp** **Example:** `Router(config-dial-peer)# session target loopback:rtp`	Designates a network-specific address to receive calls from a VoIP dial peer and configures all voice data to loop back to the source.
Step 7	**incoming called-number** *string* **Example:** `Router(config-dial-peer)# incoming called-number 77`	Specifies a digit string that can be matched by an incoming call to associate the call with the dial peer.
Step 8	**exit** **Example:** `Router(config-dial-peer)# exit`	Exits dial peer voice configuration mode and enters global configuration mode.

Configuration Examples for RTP Media Loopback

Example: Configuring Video Loopback with Cisco Telepresence System

The following sample output shows Media Loopback for SIP Calls configured on a Cisco Telepresence System (CTS).

```
!
codec profile 1 aacld
 fmtp "fmtp:96
profile-level-id=16;streamtype=5;mode=AAChbr;config=B98C00;sizeLength=13;indexLength=3;indexDeltaLength=3;constantDura
tion=480"
!
codec profile 2 h264
 fmtp "fmtp:112 profile-level-id=4D0028;sprop-parametersets=
R00AKAmWUgDwBDyA,SGE7jyA=;packetization-mode=1"
!
voice class codec 4
 codec preference 1 aacld profile 1
 video codec h264 profile 2
!
dial-peer voice 2000 voip
 destination-pattern 2000
 rtp payload-type cisco-codec-fax-ind 110
 rtp payload-type cisco-codec-aacld 96
 rtp payload-type cisco-codec-video-h264 112
 session protocol sipv2
 session target loopback:rtp
 incoming called-number 2000
 voice-class codec 4
 voice-class sip bandwidth audio tias-modifier 64000
 voice-class sip bandwidth video tias-modifier 4500000
!
```

Example: Configuring Video Loopback with Cisco Unified Video Advantage

The following sample output shows Media Loopback for SIP Calls configured on a Cisco Unified Video Advantage (CUVA).

```
!
codec profile 3 h264
 fmtp "fmtp:98 profile-level-id=420015"
 !
voice class codec 6
 codec preference 1 g711ulaw
 video codec h264 profile 3
 !
dial-peer voice 5000 voip
 description CUVA
 destination-pattern 5000
 rtp payload-type cisco-codec-video-h264 98
 session protocol sipv2
 session target loopback:rtp
 incoming called-number 5000
 voice-class codec 6
 voice-class sip bandwidth video tias-modifier 384000
```

Feature Information for RTP Media Loopback for SIP Calls

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Feature History Table entry for the Cisco Unified Border Element.

Table 18: Feature Information for RTP Media Loopback for SIP Calls

Feature Name	Releases	Feature Information
RTP Media Loopback for SIP Calls	15.1(4)M 15.2(1)T	RTP packets are looped back toward the source when the RTP Media Loopback for SIP Calls feature is configured on a dial peer. SIP RTP media loopback helps in verifying the media path between the device originating the call and the intermediate device. The following commands were introduced or modified: None.

CHAPTER 19

SIP Ability to Send a SIP Registration Message on a Border Element

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for SIP Ability to Send a SIP Registration Message on a Border Element

- Configure a registrar in sip UA configuration mode.

Cisco Unified Border Element

- Cisco IOS Release 12.4(24)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Configuring SIP Ability to Send a SIP Registration Message on a Border Element

The SIP: Ability to Send a SIP Registration Message on a Border Element feature allows users to register e164 numbers from the Cisco UBE without POTS dial-peers in the UP state. Registration messages can include numbers, number ranges (such as E.164-numbers), or text information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sip-ua**
4. **credentials username** *username* **password** *password* **realm** *domain-name*
5. **exit**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**sip-ua** **Example:** `Device(config)# sip-ua`	Enters sip user-agent configuration mode.
Step 4	**credentials username** *username* **password** *password* **realm** *domain-name* **Example:** `Device(config-sip-ua)# credentials username alex password test realm cisco.com`	Enters SIP digest credentials in sip-ua configuration mode.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Device(config-sip-ua)# exit`	Exits the current mode.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Feature Information for Sending a SIP Registration Message from a Cisco Unified Border Element

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 19: Feature Information for Sending a SIP Registration Message from a Cisco Unified Border Element

Feature Name	Releases	Feature Information
SIP: Ability to Send a SIP Registration Message on a Border Element	12.4(24)T	Provides the ability to send a SIP Registration Message from Cisco Unified Border Element. The following command was modified: **credentials** (SIP UA)
SIP: Ability to Send a SIP Registration Message on a Border Element	Cisco IOS XE Release 2.5	Provides the ability to send a SIP Registration Message from Cisco Unified Border Element. The following command was modified: **credentials** (SIP UA)

CHAPTER 20

Session Refresh with Reinvites

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Session Refresh with Reinvites

The **allow-connections sip to sip** command must be configured before you configure the Session refresh with Reinvites feature. For more information and configuration steps see the "Configuring SIP-to-SIP Connections in a Cisco Unified Border Element" section.

Cisco Unified Border Element

- Cisco IOS Release 12.4(20)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Information about Session Refresh with Reinvites

Configuring support for session refresh with reinvites expands the ability of the Cisco Unified Border Element to receive a REINVITE message that contains either a session refresh parameter or a change in media via a new SDP and ensure the session does not time out. The **midcall-signaling** command distinguishes between the way a Cisco Unified Communications Express and Cisco Unified Border Element releases signaling messages. Most SIP-to-SIP video and SIP-to-SIP ReInvite-based supplementary services features require the Configuring Session Refresh with Reinvites feature to be configured.

Cisco IOS Release 12.4(15)XZ and Earlier Releases

Session refresh support via OPTIONS method. For configuration information, see the "Enabling In-Dialog OPTIONS to Monitor Active SIP Sessions" section.

Cisco IOS Release 12.4(15)XZ and Later Releases

Cisco Unified BE transparently passes other session refresh messages and parameters so that UAs and proxies can establish keepalives on a call.

How to Configure Session Refresh with Reinvites

Configuring Session refresh with Reinvites

Before You Begin

Note SIP-to-SIP video calls and SIP-to-SIP ReInvite-based supplementary services fail if the **midcall-signaling**command is not configured.

Note The following features function if the **midcall-signaling** command is not configured: session refresh, fax, and refer-based supplementary services.

- Configuring Session Refresh with Reinvites is for SIP-to-SIP calls only. All other calls (H323-to-SIP, and H323-to-H323) do not require the **midcall-signaling**command be configured
- Configuring the Session Refresh with Reinvites feature on a dial-peer basis is not supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **midcall-signaling passthru**
6. **exit**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Router(config)# voice service voip`	Enters VoIP voice-service configuration mode.
Step 4	**sip** **Example:** `Router(conf-voi-serv)# sip`	Enters SIP configuration mode.
Step 5	**midcall-signaling passthru** **Example:** `Router(conf-serv-sip)# midcall-signaling passthru`	Passes SIP messages from one IP leg to another IP leg.
Step 6	**exit** **Example:** `Router(conf-serv-sip)# exit`	Exits the current mode.

	Command or Action	Purpose
Step 7	**end** **Example:** `Router(conf-serv-sip) end`	Returns to privileged EXEC mode.

Feature Information for Session Refresh with Reinvites

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Feature Name	Releases	Feature Information
Session Refresh with Reinvites	12.4(20)T	Expands the ability of the Cisco Unified BE to control the session refresh parameters and ensure the session does not time out. In Cisco IOS Release 12.4(20)T, this feature was implemented on the Cisco Unified Border Element. **midcall-signaling**
Session Refresh with Reinvites	Cisco IOS XE Release 2.5	Expands the ability of the Cisco Unified BE to control the session refresh parameters and ensure the session does not time out. In Cisco IOS XE Release 2.5, this feature was implemented on the Cisco Unified Border Element (Enterprise). **midcall-signaling**

CHAPTER 21

SIP Stack Portability

Implements capabilities to the SIP gateway Cisco IOS stack involving user-agent handling of messages, handling of unsolicited messages, support for outbound delayed media, and SIP headers and content in requests and responses.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for SIP Stack Portability

Cisco Unified Border Element

- Cisco IOS Release 12.4(2)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.5 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Information About SIP Stack Portability

The SIP Stack Portability feature implements the following capabilities to the Cisco IOS SIP gateway stack:

- It receives inbound Refer message requests both within a dialog and outside of an existing dialog from the user agents (UAs).
- It sends and receives SUBSCRIBE or NOTIFY message requests via UAs.
- It receives unsolicited NOTIFY message requests without having to subscribe to the event that was generated by the NOTIFY message request.
- It supports outbound delayed media.

It sends an INVITE message request without Session Description Protocol (SDP) and provides SDP information in either the PRACK or ACK message request for both initial call establishment and mid-call re-INVITE message requests.

- It sets SIP headers and content body in requests and responses.

The stack applies certain rules and restrictions for a subset of headers and for some content types (such as SDP) to protect the integrity of the stack's functionality and to maintain backward compatibility. When receiving SIP message requests, it reads the SIP header and any attached body without any restrictions.

To make the best use of SIP call-transfer features, you should understand the following concepts:

SIP Call-Transfer Basics

Basic Terminology of SIP Call Transfer

Call transfer allows a wide variety of decentralized multiparty call operations. These decentralized call operations form the basis for third-party call control, and thus are important features for VoIP and SIP. Call transfer is also critical for conference calling, where calls can transition smoothly between multiple point-to-point links and IP-level multicasting.

Refer Message Request

The SIP Refer message request provides call-transfer capabilities to supplement the SIP BYE and ALSO message requests already implemented on Cisco IOS SIP gateways. The Refer message request has three main roles:

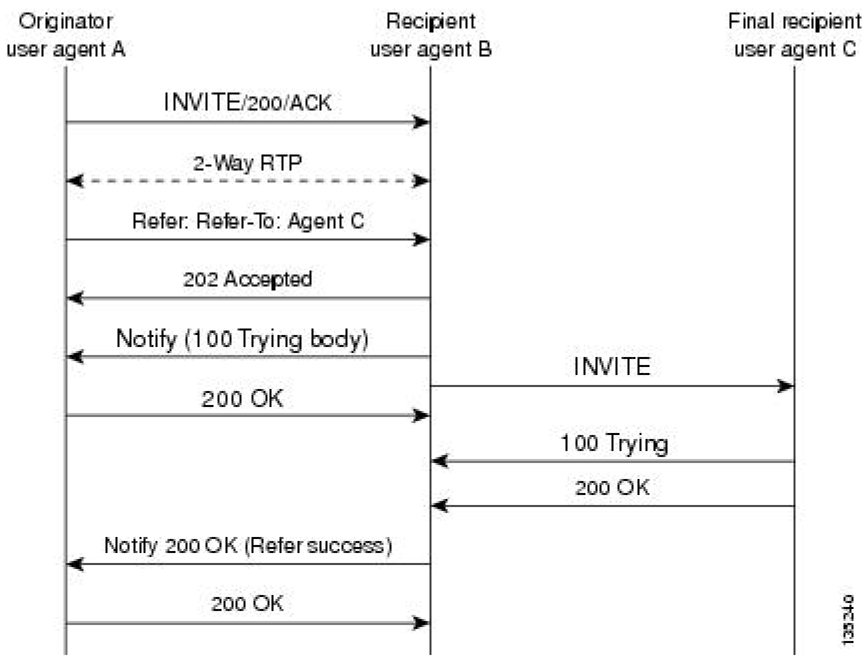
- Originator--User agent that initiates the transfer or Refer request.
- Recipient--User agent that receives the Refer request and is transferred to the final-recipient.
- Final-Recipient--User agent introduced into a call with the recipient.

Note A gateway can be a recipient or final recipient, but not an originator.

The Refer message request always begins within the context of an existing call and starts with the *originator* . The originator sends a Refer request to the *recipient* (user agent receiving the Refer request) to initiate a triggered INVITE request. The triggered INVITE request uses the SIP URL contained in the Refer-To header as the destination of the INVITE request. The recipient then contacts the resource in the Refer-To header (*final recipient*), and returns a SIP 202 (Accepted) response to the originator. The recipient also must notify the originator of the outcome of the Refer transaction--whether the final recipient was successfully contacted or not. The notification is accomplished using the SIP NOTIFY message request, SIP's event notification mechanism. A NOTIFY message with a message body of SIP 200 OK indicates a successful transfer, and a message body of SIP 503 Service Unavailable indicates an unsuccessful transfer. If the call was successful, a call between the recipient and the final recipient results.

The figure below represents the call flow of a successful Refer transaction initiated within the context of an existing call.

Figure 6: Successful Refer transaction



Refer-To Header

The recipient receives from the originator a Refer request that always contains a single Refer-To header. The Refer-To header includes a SIP URL that indicates the party to be invited and must be in SIP URL format.

Note The TEL URL format cannot be used in a Refer-To header, because it does not provide a host portion, and without one, the triggered INVITE request cannot be routed.

The Refer-To header may contain three additional overloaded headers to form the triggered INVITE request. If any of these three headers are present, they are included in the triggered INVITE request. The three headers are:

- Accept-Contact--Optional in a Refer request. A SIP Cisco IOS gateway that receives an INVITE request with an Accept-Contact does not act upon this header. This header is defined in draft-ietf-sip-callerprefs-03.txt and may be used by user agents that support caller preferences.
- Proxy-Authorization--Nonstandard header that SIP gateways do not act on. It is echoed in the triggered INVITE request because proxies occasionally require it for billing purposes.
- Replaces--Header used by SIP gateways to indicate whether the originator of the Refer request is requesting a blind or attended transfer. It is required if the originator is performing an attended transfer, and not required for a blind transfer.

All other headers present in the Refer-To are ignored, and are not sent in the triggered INVITE.

Note The Refer-To and Contact headers are required in the Refer request. The absence of these headers results in a 4*xx* class response to the Refer request. Also, the Refer request must contain exactly one Refer-To header. Multiple Refer-To headers result in a 4*xx* class response.

Referred-By Header

The Referred-By header is required in a Refer request. It identifies the originator and may also contain a signature (included for security purposes). SIP gateways echo the contents of the Referred-By header in the triggered INVITE request, but on receiving an INVITE request with this header, gateways do not act on it.

Note The Referred-By header is required in a Refer request. The absence of this header results in a 4*xx* class response to the Refer request. Also, the Refer request must contain exactly one Referred-By header. Multiple Referred-By headers result in a 4*xx* class response.

NOTIFY Message Request

Once the outcome of the Refer transaction is known, the recipient of the Refer request must notify the originator of the outcome of the Refer transaction--whether the final-recipient was successfully contacted or not. The notification is accomplished using the NOTIFY message request, SIP's event notification mechanism. The notification contains a message body with a SIP response status line and the response class in the status line indicates the success or failure of the Refer transaction.

The NOTIFY message must do the following:

- Reflect the same To, From, and Call-ID headers that were received in the Refer request.
- Contain an Event header refer.
- Contain a message body with a SIP response line. For example: SIP/2.0 200 OK to report a successful Refer transaction, or SIP/2.0 503 Service Unavailable to report a failure. To report that the recipient disconnected before the transfer finished, it must use SIP/2.0 487 Request Canceled.

Two Cisco IOS commands pertain to the NOTIFY message request:

- The **timers notify** command sets the amount of time that the recipient should wait before retransmitting a NOTIFY message to the originator.

- The **retry notify** command configures the number of times a NOTIFY message is retransmitted to the originator.

Note For information on these commands, see the *Cisco IOS Voice Command Reference* .

Types of SIP Call Transfer Using the Refer Message Request

This section discusses how the Refer message request facilitates call transfer.

There are two types of call transfer: blind and attended. The primary difference between the two is that the Replaces header is used in attended call transfers. The Replaces header is interpreted by the final recipient and contains a Call-ID header, indicating that the initial call leg is to be replaced with the incoming INVITE request.

As outlined in the Refer message request, there are three main roles:

- Originator--User agent that initiates the transfer or Refer request.
- Recipient--User agent that receives the Refer request and is transferred to the final recipient.
- Final-Recipient--User agent introduced into a call with the recipient.

A gateway can be a recipient or final recipient, but not an originator.

Blind Call-Transfer Process

A blind, or unattended, transfer is one in which the transferring phone connects the caller to a destination line before ringback begins. This is different from a consultative, or attended, transfer in which one of the transferring parties either connects the caller to a ringing phone (ringback heard) or speaks with the third party before connecting the caller to the third party. Blind transfers are often preferred by automated devices that do not have the capability to make consultation calls.

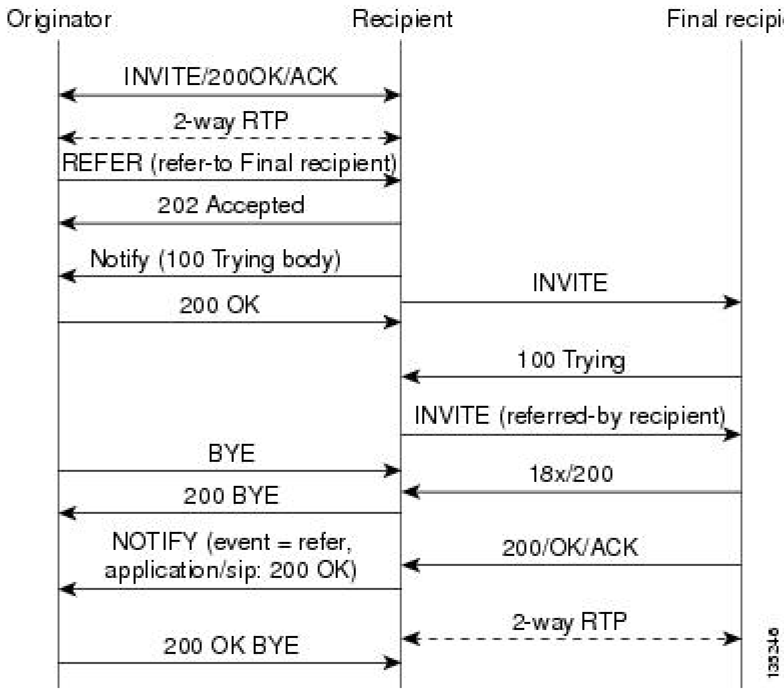
Blind transfer works as described in the Types of SIP Call Transfer Using the Refer Message Request, on page 197. The process is as follows:

1. Originator (user agent that initiates the transfer or Refer request) does the following:
    1. Sets up a call with recipient (user agent that receives the Refer request)
    2. Issues a Refer request to recipient
2. Recipient does the following:
    1. Sends an INVITE request to final recipient (user agent introduced into a call with the recipient)
    2. Returns a SIP 202 (Accepted) response to originator
    3. Notifies originator of the outcome of the Refer transaction--whether final recipient was successfully (SIP 200 OK) contacted or not (SIP 503 Service Unavailable)
3. If successful, a call is established between recipient and final recipient.
4. The original signaling relationship between originator and recipient terminates when either of the following occurs:
5. One of the parties sends a Bye request.

6 Recipient sends a Bye request after successful transfer (if originator does not first send a Bye request after receiving an acknowledgment for the NOTIFY message).

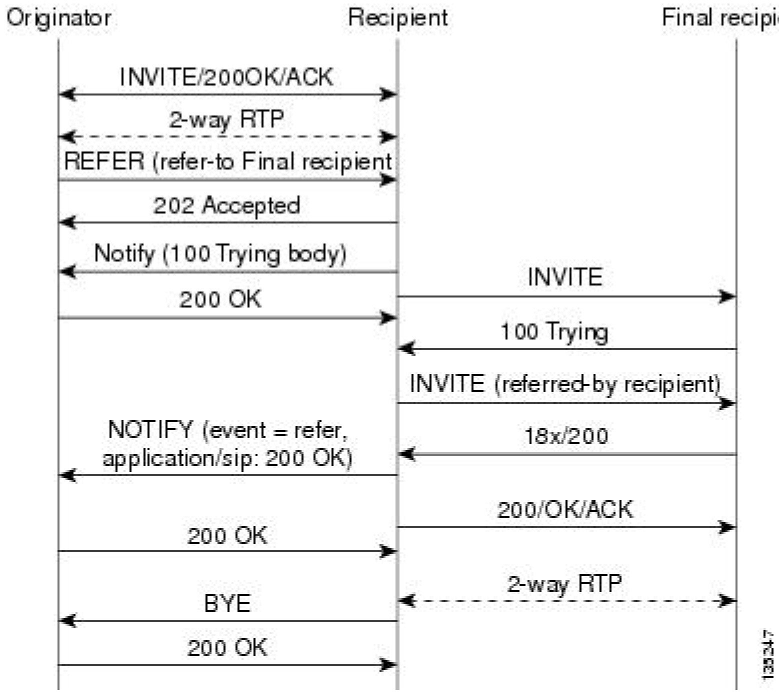
The figure below shows a successful blind or unattended call transfer in which the originator initiates a Bye request to terminate signaling with the recipient.

Figure 7: Successful Blind or Unattended Transfer--Originator Initiating a Bye Request



The figure below shows a successful blind or unattended call transfer in which the recipient initiates a Bye request to terminate signaling with the originator. A NOTIFY message is always sent by the recipient to the originator after the final outcome of the call is known.

Figure 8: Successful Blind or Unattended Transfer--Recipient Initiating a Bye Request



If a failure occurs with the triggered INVITE to the final recipient, the call between originator and recipient is not disconnected. Rather, with blind transfer the process is as follows:

1 Originator sends a re-INVITE that takes the call off hold and returns to the original call with recipient.

2 Final recipient sends an 18x informational response to recipient.

3 The call fails; the originator cannot recover the call with recipient. Failure can be caused by an error condition or timeout.

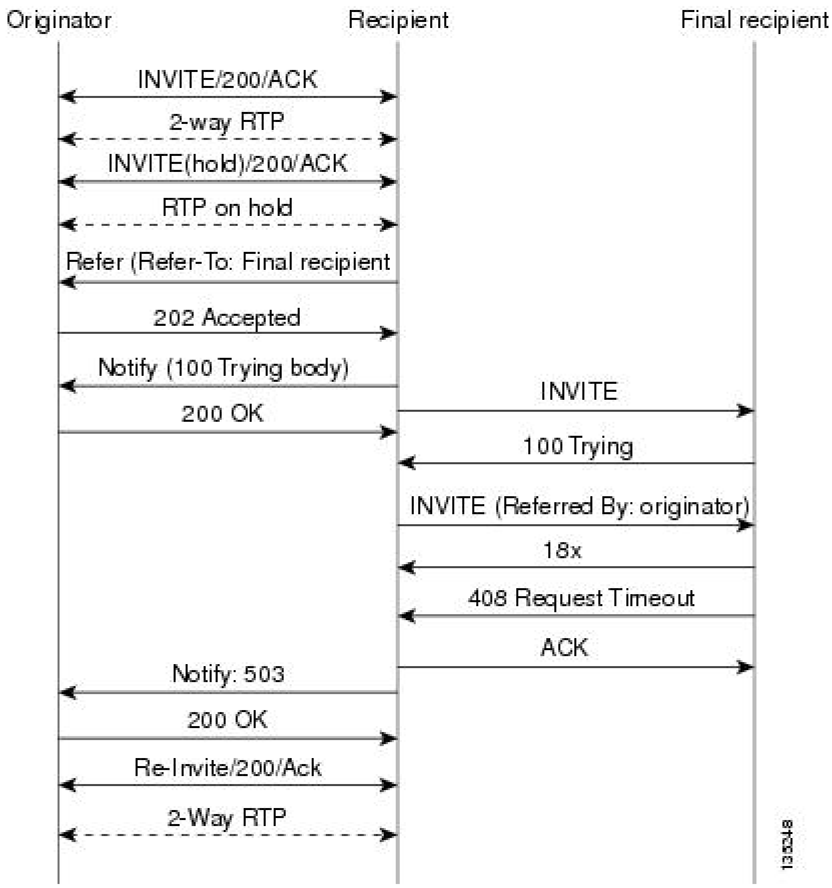
4 The call leg between originator and recipient remains active (see the figure below).

5 If the INVITE to final recipient fails (408 Request Timeout), the following occurs:

    1 Recipient notifies originator of the failure with a NOTIFY message.

2 Originator sends a re-INVITE and returns to the original call with the recipient.

Figure 9: Failed Blind Transfer--Originator Returns to Original Call with Recipient



Attended Transfer

In attended transfers, the Replaces header is inserted by the initiator of the Refer message request as an overloaded header in the Refer-To and is copied into the triggered INVITE request sent to the final recipient. The header has no effect on the recipient, but is interpreted by the final recipient as a way to distinguish between blind transfer and attended transfer. The attended transfer process is as follows:

1 Originator does the following:

   1 Sets up a call with recipient.
   2 Places recipient on hold.
   3 Establishes a call to final recipient.
   4 Sends recipient a Refer message request with an overloaded Replaces header in the Refer-To header.

2 Recipient does the following:

   1 Sends a triggered INVITE request to final recipient. (Request includes the Replaces header, identifying the call leg between the originator and the final recipient.)

2. Recipient returns a SIP 202 (Accepted) response to originator. (Response acknowledges that the INVITE has been sent.)

3. Final recipient establishes a direct signaling relationship with recipient. (Replaces header indicates that the initial call leg is to be shut down and replaced by the incoming INVITE request.)
4. Recipient notifies originator of the outcome of the Refer transaction. (Outcome indicates whether or not the final recipient was successfully contacted.)
5. Recipient terminates the session with originator by sending a Bye request.

Replaces Header

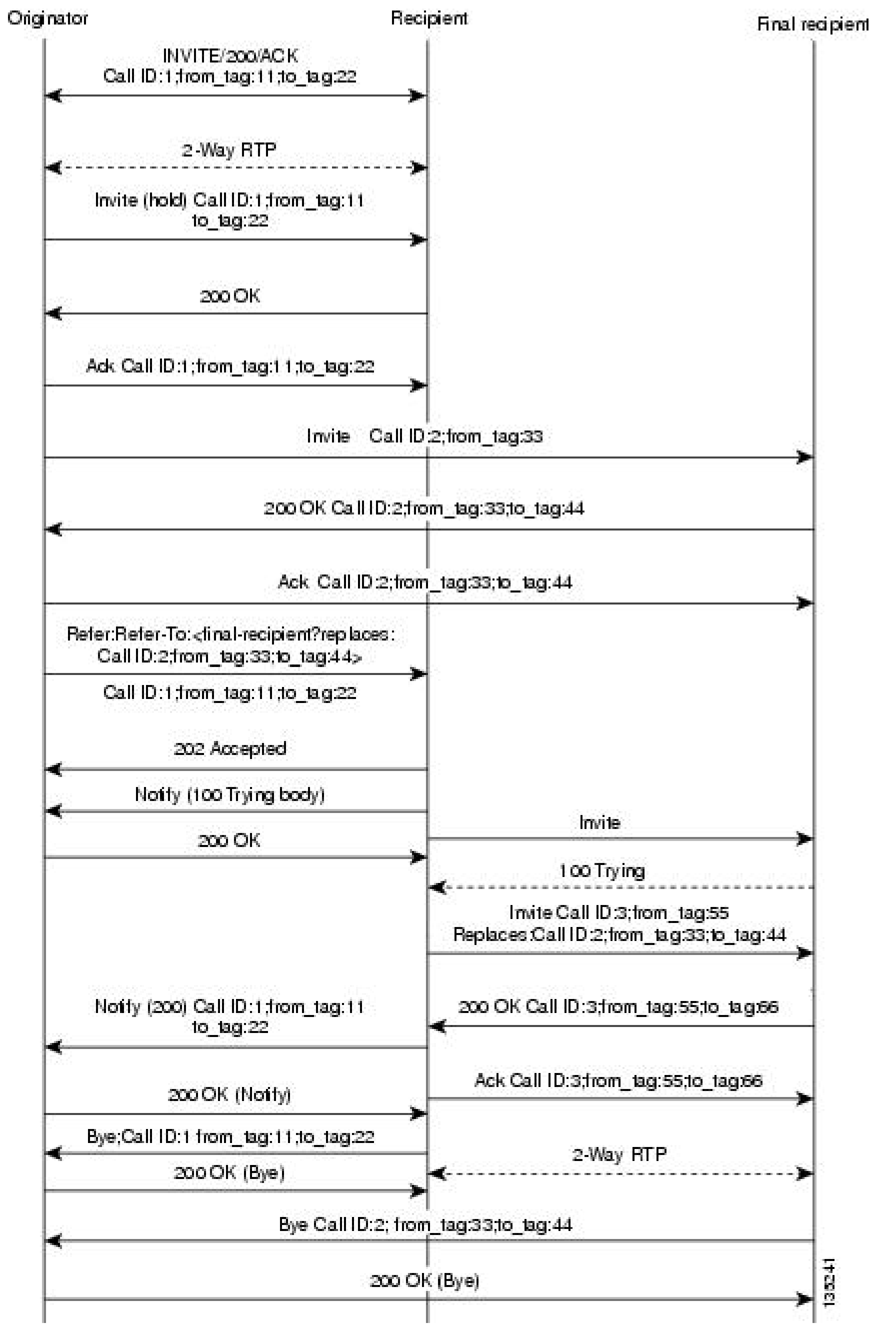
The Replaces header is required in attended transfers. It indicates to the final recipient that the initial call leg (identified by the Call-ID header and tags) is to be shut down and replaced by the incoming INVITE request. The final recipient sends a Bye request to the originator to terminate its session.

If the information provided by the Replaces header does not match an existing call leg, or if the information provided by the Replaces header matches a call leg but the call leg is not active (a Connect, 200 OK to the INVITE request has not been sent by the final-recipient), the triggered INVITE does not replace the initial call leg and the triggered INVITE request is processed normally.

Any failure resulting from the triggered INVITE request from the recipient to the final recipient does not drop the call between the originator and the final recipient. In these scenarios, all calls that are active (originator to recipient and originator to final recipient) remain active after the failed attended transfer attempt

The figure below shows a call flow for a successful attended transfer.

Figure 10: Successful Attended Transfer



Attended Transfer with Early Completion

Attended transfers allow the originator to have a call established between both the recipient and the final recipient. With attended transfer with early completion, the call between the originator and the final recipient does not have to be active, or in the talking state, before the originator can transfer it to the recipient. The originator establishes a call with the recipient and only needs to be setting up a call with the final recipient. The final recipient may be ringing, but has not answered the call from the originator when it receives a re-INVITE to replace the call with the originator and the recipient.

The process for attended transfer with early completion is as follows (see the figure below):

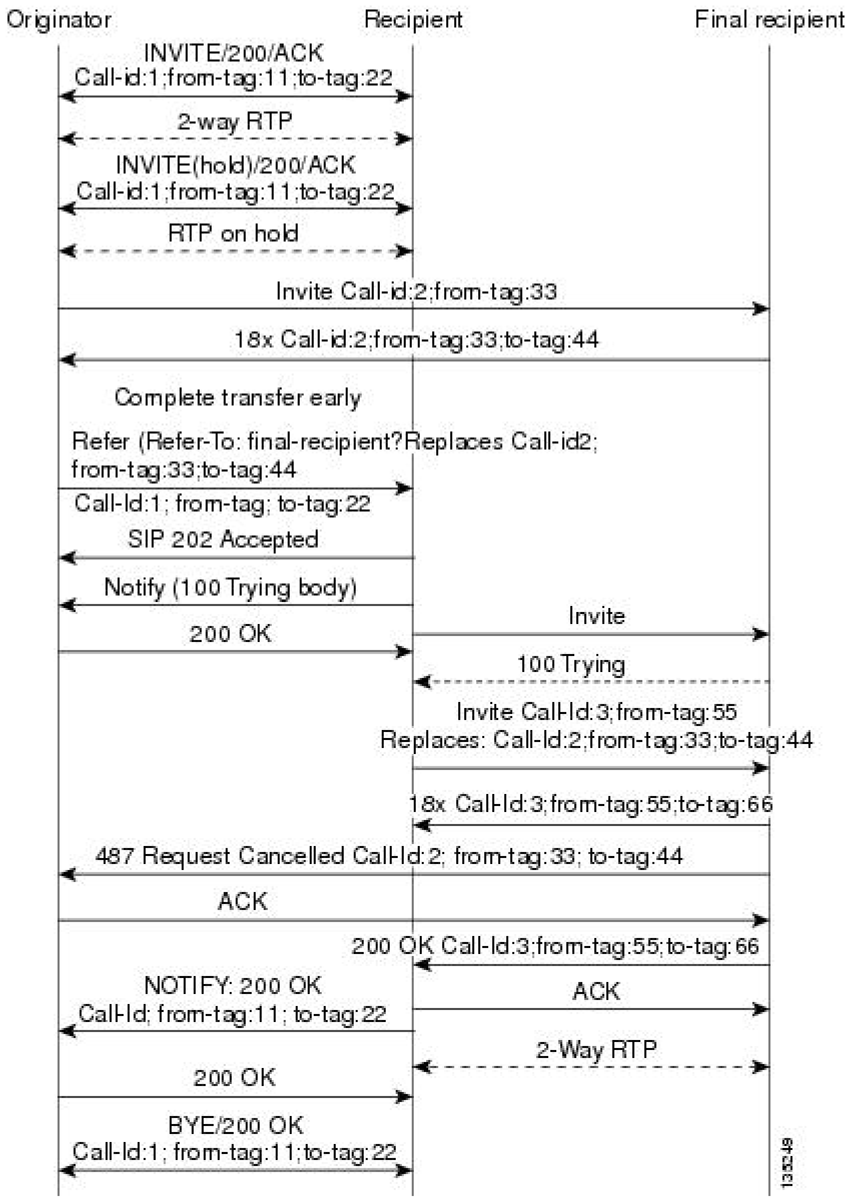
1. Originator does the following:
    1. Sets up a call with recipient.
    2. Places the recipient on hold.
    3. Contacts the final recipient.
    4. After receiving an indication that the final recipient is ringing, sends recipient a Refer message request with an overloaded Replaces header in the Refer-To header. (The Replaces header is required in attended transfers and distinguishes between blind transfer and attended transfers.)
2. Recipient does the following:
    1. Returns a SIP 202 (Accepted) response to the originator. (to acknowledge that the INVITE has been sent.)
    2. Upon receipt of the Refer message request, sends a triggered INVITE request to final recipient. (The request includes the Replaces header, which indicates that the initial call leg, as identified by the Call-ID header and tags, is to be shut down and replaced by the incoming INVITE request.)
3. Final recipient establishes a direct signaling relationship with recipient.
4. Final recipient tries to match the Call-ID header and the To or From tag in the Replaces header of the incoming INVITE with an active call leg in its call control block. If a matching active call leg is found, final recipient replies with the same status as the found call leg. However, it then terminates the found call leg with a 487 Request Cancelled response.

Note If early transfer is attempted and the call involves quality of service (QoS) or Resource Reservation Protocol (RSVP), the triggered INVITE from the recipient with the Replaces header is not processed and the transfer fails. The session between originator and final recipient remains unchanged.

1. Recipient notifies originator of the outcome of the Refer transaction--that is, whether final recipient was successfully contacted or not.

2 Recipient or originator terminates the session by sending a Bye request.

Figure 11: Attended Transfer with Early Completion



VSA for Call Transfer

You can use a vendor-specific attribute (VSA) for SIP call transfer.

Referred-By Header

For consistency with existing billing models, Referred-By and Requested-By headers are populated in call history tables as a VSA. Cisco VSAs are used for VoIP call authorization. The new VSA tag

supp-svc-xfer-byhelps to associate the call legs for call-detail-record (CDR) generation. The call legs can be originator-to-recipient or recipient-to-final-recipient.

The VSA tag **supp-svc-xfer-by** contains the user@host portion of the SIP URL of the Referred-By header for transfers performed with the Refer message request. For transfers performed with the Bye/Also message request, the tag contains user@host portion of the SIP URL of the Requested-By header. For each call on the gateway, two RADIUS records are generated: start and stop. The **supp-svc-xfer-by**VSA is generated only for stop records and is generated only on the recipient gateway--the gateway receiving the Refer or Bye/Also message.

The VSA is generated when a gateway that acts as a recipient receives a Refer or Bye/Also message with the Referred-By or Requested-By headers. There are usually two pairs of start and stop records. There is a start and stop record between the recipient and the originator and also between the recipient to final recipient. In the latter case, the VSA is generated between the recipient to the final recipient only.

Business Group Field

A new business group VSA field has been added that assists service providers with billing. The field allows service providers to add a proprietary header to call records. The VSA tag for business group ID is **cust-biz-grp-id** and is generated only for stop records. It is generated when the gateway receives an initial INVITE with a vendor dial-plan header to be used in call records. In cases when the gateway acts as a recipient, the VSA is populated in the stop records between the recipient and originator and the final recipient.

Note For information on VSAs, see the RADIUS VSA Voice Implementation Guide .

Feature Information for SIP Stack Portability

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 20: Feature Information for SIP Stack Portability

Feature Name	Releases	Feature Information
SIP Stack Portability	Cisco IOS XE Release 2.5	Implements capabilities to the SIP gateway Cisco IOS stack involving user-agent handling of messages, handling of unsolicited messages, support for outbound delayed media, and SIP headers and content in requests and responses The following commands were introduced or modified: **None**

Feature Name	Releases	Feature Information
SIP Stack Portability	12.4(2)T	Implements capabilities to the SIP gateway Cisco IOS stack involving user-agent handling of messages, handling of unsolicited messages, support for outbound delayed media, and SIP headers and content in requests and responses The following commands were introduced or modified: **None**

CHAPTER 22

Interworking of Secure RTP calls for SIP and H.323

The Session Initiation Protocol (SIP) support for the Secure Real-time Transport Protocol (SRTP) is an extension of the Real-time Transport Protocol (RTP) Audio/Video Profile (AVP) and ensures the integrity of RTP and Real-Time Control Protocol (RTCP) packets that provide authentication, encryption, and the integrity of media packets between SIP endpoints.

SIP support for SRTP was introduced in Cisco IOS Release 12.4(15)T. In this and later releases, you can configure the handling of secure RTP calls on both a global level and on an individual dial peer basis on Cisco IOS voice gateways. You can also configure the gateway (or dial peer) either to fall back to (nonsecure) RTP or to reject (fail) the call for cases where an endpoint does not support SRTP.

The option to allow negotiation between SRTP and RTP endpoints was added for Cisco IOS Release 12.4(20)T and later releases, as was interoperability of SIP support for SRTP on Cisco IOS voice gateways with Cisco Unified Communications Manager. In Cisco IOS Release 12.4(22)T and later releases, you can also configure SIP support for SRTP on Cisco Unified Border Elements (Cisco UBEs).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Interworking of Secure RTP calls for SIP and H.323

The following are prerequisites for the Interworking of Secure RTP calls for SIP and H.323 feature:

- Establish a working IP network and configure VoIP.

Note For information about configuring VoIP, see Enhancements to the Session Initiation Protocol for VoIP on Cisco Access Platforms at the following URL:
http://www.cisco.com/en/US/docs/ios/12_2t/12_2t11/feature/guide/ftsipgv1.html

- Ensure that the gateway has voice functionality configured for SIP.
- Ensure that your Cisco router has adequate memory.
- As necessary, configure the router to use Greenwich Mean Time (GMT). SIP requires that all times be sent in GMT. SIP INVITE messages are sent in GMT. However, the default for routers is to use Coordinated Universal Time (UTC). To configure the router to use GMT, issue the clock timezone command in global configuration mode and specify GMT.

Cisco Unified Border Element

- Cisco IOS Release 12.2(20)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.1S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for Interworking of Secure RTP calls for SIP and H.323

- The SIP gateway does not support codecs other than those listed in the table titled "SIP Codec Support by Platform and Cisco IOS Release" in the "Enhanced Codec Support for SIP Using Dynamic Payloads" section of the Configuring SIP QoS Features module at the following URL:
http://www.cisco.com/en/US/docs/ios/voice/sip/configuration/guide/sip_cg-qos.html
- SIP requires that all times be sent in GMT.

Feature Information for Configuring Interworking of Secure RTP Calls for SIP and H.323

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 21: Feature Information for Configuring Support for Expires Timer Reset on Receiving or Sending SIP 183 Message

Feature Name	Releases	Feature Information
Interworking of Secure RTP calls for SIP and H.323	12.4(20)T	This feature provides an option for a Secure RTP (SRTP) call to be connected from H.323 to SIP and from SIP to SIP. Additionally, this feature extends SRTP fallback support from the Cisco IOS voice gateway to the Cisco Unified Border Element. This feature uses no new or modified commands.
Interworking of Secure RTP calls for SIP and H.323	Cisco IOS XE Release 3.1S	This feature provides an option for a Secure RTP (SRTP) call to be connected from H.323 to SIP and from SIP to SIP. Additionally, this feature extends SRTP fallback support from the Cisco IOS voice gateway to the Cisco Unified Border Element. This feature uses no new or modified commands.

CHAPTER 23

Cisco UBE Support for SRTP-RTP Internetworking

The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature allows secure enterprise-to-enterprise calls and provides operational enhancements for Session Initiation Protocol (SIP) trunks from Cisco Unified Call Manager and Cisco Unified Call Manager Express. Support for Secure Real-Time Transport Protocol (SRTP)-Real-Time Transport Protocol (RTP) internetworking between one or multiple Cisco Unified Border Elements (Cisco UBEs) is enabled for SIP-SIP audio calls.

In Cisco IOS Release 15.2(1) and Cisco IOS XE Release 3.7S, the SRTP-RTP Interworking feature was extended to support supplementary services on Cisco UBEs.

Prerequisites for CUBE Support for SRTP-RTP Internetworking

- The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature is supported in Cisco Unified CallManager 7.0 and later releases.

Cisco Unified Border Element

- Cisco IOS Release 12.4(22)YB or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.7S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions for CUBE Support for SRTP-RTP Internetworking

The following features are not supported by the Cisco Unified Border Element Support for SRTP-RTP Internetworking feature:

- Asymmetric SRTP fallback configurations
- Call admission control (CAC) support
- Rotary SIP-SIP
- SRTCP-RTCP interworking
- Transcoding for SRTP-SRTP audio calls

Note Effective from Cisco IOS XE release 3.9S, SRTP-RTP interworking is supported (on ASR platforms) for video calls with no secondary video streams.

Information About CUBE for SRTP-RTP Internetworking

To configure support for SRTP-RTP internetworking, you should understand the following concepts:

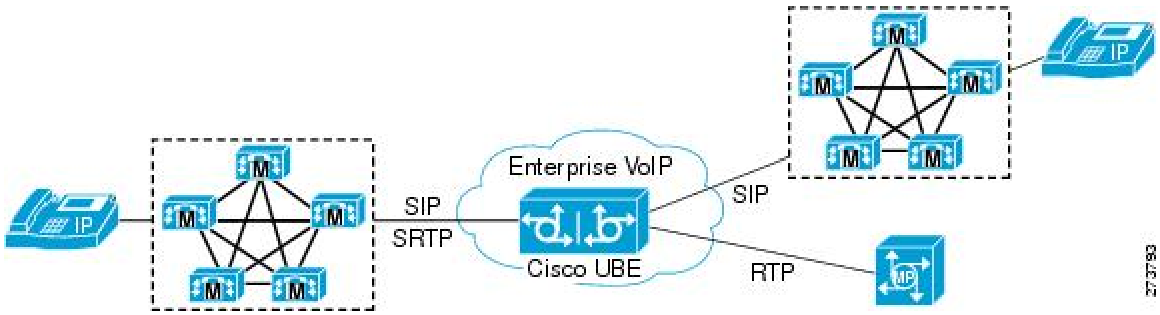
CUBE Support for SRTP-RTP Internetworking

The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature connects SRTP Cisco Unified CallManager domains with the following:

- RTP Cisco Unified CallManager domains. Domains that do not support SRTP or have not been configured for SRTP, as shown in the figure below.
- RTP Cisco applications or servers. For example, Cisco Unified MeetingPlace, Cisco WebEx, or Cisco Unity, which do not support SRTP, or have not been configured for SRTP, or are resident in a secure data center, as shown in the figure below.

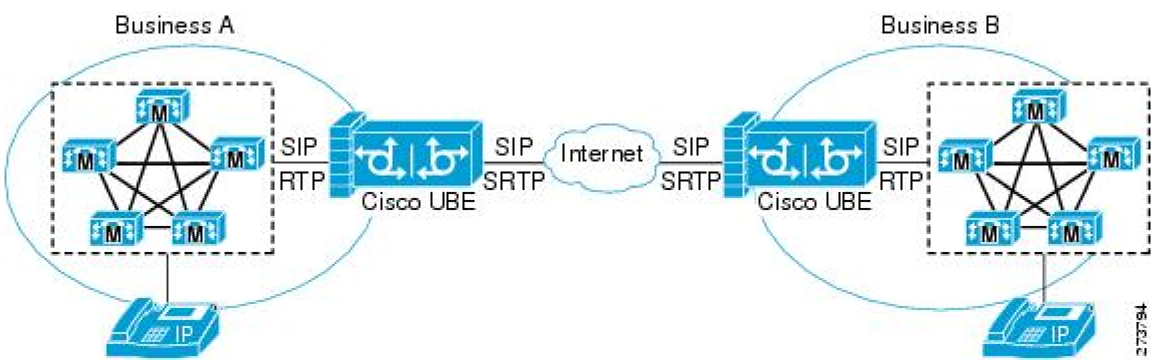
- RTP to third-party equipment. For example, IP trunks to PBXs or virtual machines, which do not support SRTP.

Figure 12: SRTP Domain Connections



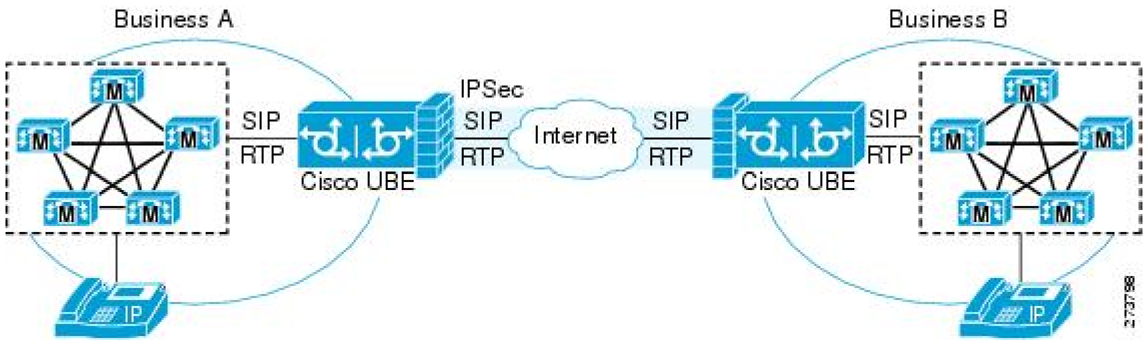
The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature connects SRTP enterprise domains to RTP SIP provider SIP trunks. SRTP-RTP internetworking connects RTP enterprise networks with SRTP over an external network between businesses. This provides flexible secure business-to-business communications without the need for static IPsec tunnels or the need to deploy SRTP within the enterprise, as shown in the figure below.

Figure 13: Secure Business-to-Business Communications



SRTP-RTP internetworking also connects SRTP enterprise networks with static IPsec over external networks, as shown inthe figure below.

Figure 14: SRTP Enterprise Network Connections



SRTP-RTP internetworking on the Cisco UBE in a network topology uses single-pair key generation. Existing audio and dual-tone multifrequency (DTMF) transcoding is used to support voice calls. SRTP-RTP internetworking support is provided in both flow-through and high-density mode. SRTP-SRTP pass-through is not impacted.

SRTP is configured on one dial peer and RTP is configured on the other dial peer using the **srtp** and **srtp fallback** commands. The dial-peer configuration takes precedence over the global configuration on the Cisco UBE.

Fallback handling occurs if one of the call endpoints does not support SRTP. The call can fall back to RTP-RTP, or the call can fail, depending on the configuration. Fallback takes place only if the **srtp fallback** command is configured on the respective dial peer. RTP-RTP fallback occurs when no transcoding resources are available for SRTP-RTP internetworking.

TLS on the Cisco Unified Border Element

The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature allows Transport Layer Security (TLS) to be enabled or disabled between the Skinny Call Control Protocol (SCCP) server and the SCCP client. By default, TLS is enabled, which provides added protection at the transport level and ensures that SRTP keys are not easily accessible. Once TLS is disabled, the SRTP keys are not protected.

SRTP-RTP internetworking is available with normal and universal transcoders. The transcoder on the Cisco Unified Border Element is invoked using SCCP messaging between the SCCP server and the SCCP client. SCCP messages carry the SRTP keys to the digital signal processor (DSP) farm at the SCCP client. The transcoder can be within the same router or can be located in a separate router. TLS should be disabled only when the transcoder is located in the same router. To disable TLS, configure the **no** form of the **tls** command in dsp farm profile configuration mode. Disabling TLS improves CPU performance.

Supplementary Services Support on the Cisco UBE for RTP-SRTP Calls

The Supplementary Services Support on Cisco UBE for RTP-SRTP Calls feature supports the following supplementary services on the Cisco UBE:

- Midcall codec change with voice class codec configuration for SRTP-RTP and SRTP pass-through calls.

- Reinvite-based call hold.
- Reinvite-based call resume.
- Music on hold (MoH) invoked from the Cisco Unified Communications Manager (Cisco UCM), where the call leg changes between SRTP and RTP for an MoH source.

  Reinvite-based call forward.
- Reinvite-based call transfer.
- Call transfer based on a REFER message, with local consumption or pass-through of the REFER message on the Cisco UBE.
- Call forward based on a 302 message, with local consumption or pass-through of the 302 message on the Cisco UBE.
- T.38 fax switchover.
- Fax pass-through switchover.
- DO-EO for SRTP-RTP calls.
- DO-EO for SRTP pass-through calls.

When the initial SRTP-RTP or SRTP pass-through call is established on the Cisco UBE, a call can switch between SRTP and RTP for various supplementary services that can be invoked on the end points. Transcoder resources are used to perform SRTP-RTP conversion on Cisco UBE. When the call switches between SRTP and RTP, the transcoder is dynamically inserted, deleted, or modified. Both normal transcoding and high-density (optimized) transcoding are supported.

For call transfers involving REFER and 302 messages (messages that are locally consumed on Cisco UBE), end-to-end media renegotiation is initiated from Cisco UBE only when you configure the supplementary-service media-renegotiate command in voice service voip configuration mode.

When supplementary services are invoked from the end points, the call can switch between SRTP and RTP during the call duration. Hence, Cisco recommends that you configure such SIP trunks for SRTP fallback.

How to Configure Cisco UBE Support for SRTP-RTP Internetworking

Configuring Cisco UBE Support for SRTP-RTP Internetworking

Configuring the Certificate Authority

Perform the steps described in this section to configure the certificate authority.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **crypto pki server** *cs-label*
5. **database level complete**
6. **grant auto**
7. **no shutdown**
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip http server** **Example:** `Device(config)# ip http server`	Enables the HTTP server on your IPv4 or IPv6 system, including the Cisco web browser user interface.
Step 4	**crypto pki server** *cs-label* **Example:** `Device(config)# crypto pki server 3854-cube`	Enables a Cisco IOS certificate server and enters certificate server configuration mode. • In the example, 3854-cube is specified as the name of the certificate server.
Step 5	**database level complete** **Example:** `Device(cs-server)# database level complete`	Controls what type of data is stored in the certificate enrollment database. • In the example, each issued certificate is written to the database.

	Command or Action	Purpose
Step 6	**grant auto** **Example:** `Device(cs-server)# grant auto`	Specifies automatic certificate enrollment.
Step 7	**no shutdown** **Example:** `Device(cs-server)# no shutdown`	Reenables the certificate server. • Create and enter a new password when prompted.
Step 8	**exit** **Example:** `Device(cs-server)# exit`	Exits certificate server configuration mode.

Configuring a Trustpoint for the Secure Universal Transcoder

Perform the task in this section to configure, authenticate, and enroll a trustpoint for the secure universal transcoder.

Before You Begin

Before you configure a trustpoint for the secure universal transcoder, you should configure the certificate authority, as described in the Configuring the Certificate Authority, on page 215.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **crypto pki trustpoint** *name*
4. **enrollment url** *url*
5. **serial-number**
6. **revocation-check** *method*
7. **rsakeypair** *key-label*
8. **end**
9. **crypto pki authenticate** *name*
10. **crypto pki enroll** *name*
11. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**crypto pki trustpoint** *name* **Example:** `Device(config)# crypto pki trustpoint secdsp`	Declares the trustpoint that the router uses and enters ca-trustpoint configuration mode. • In the example, the trustpoint is named secdsp.
Step 4	**enrollment url** *url* **Example:** `Device(ca-trustpoint)# enrollment url http://10.13.2.52:80`	Specifies the enrollment parameters of a certification authority (CA). • In the example, the URL is defined as http://10.13.2.52:80.
Step 5	**serial-number** **Example:** `Device(ca-trustpoint)# serial-number`	Specifies whether the router serial number should be included in the certificate request.
Step 6	**revocation-check** *method* **Example:** `Device(ca-trustpoint)# revocation-check crl`	Checks the revocation status of a certificate. • In the example, the certificate revocation list checks the revocation status.
Step 7	**rsakeypair** *key-label* **Example:** `Device(ca-trustpoint)# rsakeypair 3845-cube`	Specifies which key pair to associate with the certificate. • In the example, the key pair 3845-cube generated during enrollment is associated with the certificate.
Step 8	**end** **Example:** `Device(ca-trustpoint)# end`	Exits ca-trustpoint configuration mode.

	Command or Action	Purpose
Step 9	**crypto pki authenticate** *name* **Example:** `Device(config)# crypto pki authenticate secdsp`	Authenticates the CA. • Accept the trustpoint CA certificate if prompted.
Step 10	**crypto pki enroll** *name* **Example:** `Device(config)# crypto pki enroll secdsp`	Obtains the certificate for the router from the CA. • Create and enter a new password if prompted. • Request a certificate from the CA if prompted.
Step 11	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode.

Configuring DSP Farm Services

Perform the task in this section to configure DSP farm services.

Before You Begin

Before you configure DSP farm services, you should configure the trustpoint for the secure universal transcoder, as described in the Configuring a Trustpoint for the Secure Universal Transcoder, on page 217.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice-card** *slot*
4. **dspfarm**
5. **dsp services dspfarm**
6. Repeat Steps 3, 4, and 5 to configure a second voice card.
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice-card** *slot* **Example:** `Device(config)# voice-card 0`	Configures a voice card and enters voice-card configuration mode. • In the example, voice card 0 is configured.
Step 4	**dspfarm** **Example:** `Device(config-voicecard)# dspfarm`	Adds a specified voice card to those participating in a DSP resource pool.
Step 5	**dsp services dspfarm** **Example:** `Device(config-voicecard)# dsp services dspfarm`	Enables DSP farm services for a particular voice network module.
Step 6	Repeat Steps 3, 4, and 5 to configure a second voice card.	--
Step 7	**exit** **Example:** `Device(config-voicecard)# exit`	Exits voice-card configuration mode.

Associating SCCP to the Secure DSP Farm Profile

Perform the task in this section to associate SCCP to the secure DSP farm profile.

Before You Begin

Before you associate SCCP to the secure DSP farm profile, you should configure DSP farm services, as described in the Configuring DSP Farm Services, on page 219.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sccp local** *interface-type interface-number*
4. **sccp ccm** *ip-address* **identifier** *identifier-number* **version** *version-number*
5. **sccp**
6. **associate ccm** *identifier-number* **priority** *priority-number*
7. **associate profile** *profile-identifier* **register** *device-name*
8. **dspfarm profile** *profile-identifier* **transcode universal security**
9. **trustpoint** *trustpoint-label*
10. **codec** *codec-type*
11. Repeat Step 10 to configure reuired codecs.
12. **maximum sessions** *number*
13. **associate application sccp**
14. **no shutdown**
15. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**sccp local** *interface-type interface-number*
	
	Example:
	
	`Device(config)# sccp local GigabitEthernet 0/0`
		Selects the local interface that SCCP applications (transcoding and conferencing) use to register with Cisco CallManager.
		
		• In the example, the following parameters are set:
		  • GigabitEthernet is defined as the interface type that the SCCP application uses to register with Cisco CallManager.
		  • The interface number that the SCCP application uses to register with Cisco CallManager is specified as 0/0.

Step 4	**sccp ccm** *ip-address* **identifier** *identifier-number* **version** *version-number*	Adds a Cisco Unified Communications Manager server to the list of available servers.

	Command or Action	Purpose
	Example: `Device(config)# sccp ccm 10.13.2.52 identifier 1 version 5.0.1`	• In the example, the following parameters are set: • 10.13.2.52 is configured as the IP address of the Cisco Unified Communications Manager server. • The number 1 identifies the Cisco Unified Communications Manager server. • The Cisco Unified Communications Manager version is identified as 5.0.1.
Step 5	**sccp** **Example:** `Device(config)# sccp`	Enables SCCP and related applications (transcoding and conferencing) and enters SCCP Cisco CallManager configuration mode.
Step 6	**associate ccm** *identifier-number* **priority** *priority-number* **Example:** `Device(config-sccp-ccm)# associate ccm 1 priority 1`	Associates a Cisco Unified CallManager with a Cisco CallManager group and establishes its priority within the group. • In the example, the following parameters are set: • The number 1 identifies the Cisco Unified CallManager. • The Cisco Unified CallManager is configured with the highest priority within the Cisco CallManager group.
Step 7	**associate profile** *profile-identifier* **register** *device-name* **Example:** `Device(config-sccp-ccm)# associate profile 1 register sxcoder`	Associates a DSP farm profile with a Cisco CallManager group. • In the example, the following parameters are set: • The number 1 identifies the DSP farm profile. • Sxcoder is configured as the user-specified device name in Cisco Unified CallManager.
Step 8	**dspfarm profile** *profile-identifier* **transcode universal security** **Example:** `Device(config-sccp-ccm)# dspfarm profile 1 transcode universal security`	Defines a profile for DSP farm services and enters DSP farm profile configuration mode. • In the example, the following parameters are set: • Profile 1 is enabled for transcoding. • Profile 1 is enabled for secure DSP farm services.
Step 9	**trustpoint** *trustpoint-label* **Example:** `Device(config-dspfarm-profile)# trustpoint secdsp`	Associates a trustpoint with a DSP farm profile. • In the example, the trustpoint to be associated with the DSP farm profile is labeled secdsp.

	Command or Action	Purpose
Step 10	**codec** *codec-type* **Example:** `Device(config-dspfarm-profile)# codec g711ulaw`	Specifies the codecs that are supported by a DSP farm profile. • In the example, the g711ulaw codec is specified.
Step 11	Repeat Step 10 to configure reuired codecs.	--
Step 12	**maximum sessions** *number* **Example:** `Device(config-dspfarm-profile)# maximum sessions 84`	Specifies the maximum number of sessions that are supported by the profile. • In the example, a maximum of 84 sessions are supported by the profile. The maximum number of sessions depends on the number of DSPs available for transcoding.
Step 13	**associate application sccp** **Example:** `Device(config-dspfarm-profile)# associate application sccp`	Associates SCCP to the DSP farm profile.
Step 14	**no shutdown** **Example:** `Device(config-dspfarm-profile)# no shutdown`	Allocates DSP farm resources and associates them with the application.
Step 15	**exit** **Example:** `Device(config-dspfarm-profile)# exit`	Exits DSP farm profile configuration mode.

Registering the Secure Universal Transcoder to the CUBE

Perform the task in this section to register the secure universal transcoder to the Cisco Unified Border Element. The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature supports both secure transcoders and secure universal transcoders.

Before You Begin

Before you register the secure universal transcoder to the Cisco Unified Border Element, you should associated SCCP to the secure DSP farm profile, as described in the Associating SCCP to the Secure DSP Farm Profile, on page 220.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **telephony-service**
4. **sdspfarm transcode sessions** *number*
5. **sdspfarm tag** *number* *device-name*
6. **em logout** *time1* *time2* *time3*
7. **max-ephones** *max-ephones*
8. **max-dn** *max-directory-numbers*
9. **ip source-address** *ip-address*
10. **secure-signaling trustpoint** *label*
11. **tftp-server-credentials trustpoint** *label*
12. **create cnf-files**
13. **no sccp**
14. **sccp**
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device> configure terminal`	Enters global configuration mode.
Step 3	**telephony-service** **Example:** `Device(config)# telephony-service`	Enters telephony-service configuration mode.
Step 4	**sdspfarm transcode sessions** *number* **Example:** `Device(config-telephony)# sdspfarm transcode sessions 84`	Specifies the maximum number of transcoding sessions allowed per Cisco CallManager Express router. • In the example, a maximum of 84 DSP farm sessions are specified.

	Command or Action	Purpose
Step 5	**sdspfarm tag** *number* *device-name* **Example:** `Device(config-telephony)# sdspfarm tag 1 sxcoder`	Permits a DSP farm to be to registered to Cisco Unified CallManager Express and associates it with an SCCP client interface's MAC address. • In the example, DSP farm 1 is associated with the sxcoder device.
Step 6	**em logout** *time1* *time2* *time3* **Example:** `Device(config-telephony)# em logout 0:0 0:0 0:0`	Configures three time-of-day-based timers for automatically logging out all Extension Mobility feature users. • In the example, all users are logged out from Extension Mobility after 00:00.
Step 7	**max-ephones** *max-ephones* **Example:** `Device(config-telephony)# max-ephones 4`	Sets the maximum number of Cisco IP phones to be supported by a Cisco CallManager Express router. • In the example, a maximum of four phones are supported by the Cisco CallManager Express router.
Step 8	**max-dn** *max-directory-numbers* **Example:** `Device(config-telephony)# max-dn 4`	Sets the maximum number of extensions (ephone-dns) to be supported by a Cisco Unified CallManager Express router. • In the example, a maximum of four extensions is allowed.
Step 9	**ip source-address** *ip-address* **Example:** `Device(config-telephony)# ip source-address 10.13.2.52`	Identifies the IP address and port through which IP phones communicate with a Cisco Unified CallManager Express router. • In the example, 10.13.2.52 is configured as the router IP address.
Step 10	**secure-signaling trustpoint** *label* **Example:** `Device(config-telephony)# secure-signaling trustpoint secdsp`	Specifies the name of the Public Key Infrastructure (PKI) trustpoint with the certificate to be used for TLS handshakes with IP phones on TCP port 2443. • In the example, PKI trustpoint secdsp is configured.
Step 11	**tftp-server-credentials trustpoint** *label* **Example:** `Device(config-telephony)# tftp-server-credentials trustpoint scme`	Specifies the PKI trustpoint that signs the phone configuration files. • In the example, PKI trustpoint scme is configured.

	Command or Action	Purpose
Step 12	**create cnf-files** **Example:** `Device(config-telephony)# ` **`create cnf-files`**	Builds the XML configuration files that are required for IP phones in Cisco Unified CallManager Express.
Step 13	**no sccp** **Example:** `Device(config-telephony)# ` **`no sccp`**	Disables SCCP and its related applications (transcoding and conferencing) and exits telephony-service configuration mode.
Step 14	**sccp** **Example:** `Device(config)# ` **`sccp`**	Enables SCCP and related applications (transcoding and conferencing).
Step 15	**end** **Example:** `Device(config)# ` **`end`**	Exits global configuration mode.

Configuring SRTP-RTP Internetworking Support

Perform the task in this section to enable SRTP-RTP internetworking support between one or multiple Cisco Unified Border Elements for SIP-SIP audio calls. In this task, RTP is configured on the incoming call leg and SRTP is configured on the outgoing call leg.

Before You Begin

Before you configure the Cisco Unified Border Element Support for SRTP-RTP Internetworking feature, you should register the secure universal transcoder to the Cisco Unified Border Element, as described in the Registering the Secure Universal Transcoder to the CUBE, on page 223.

Note The Cisco Unified Border Element Support for SRTP-RTP Internetworking feature is available only on platforms that support transcoding on the Cisco Unified Border Element. The feature is also available only on secure Cisco IOS images on the Cisco Unified Border Element.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **destination-pattern** *string*
5. **session protocol sipv2**
6. **session target ipv4:** *destination-address*
7. **incoming called-number** *string*
8. **codec** *codec*
9. **end**
10. **dial-peer voice** *tag* **voip**
11. Repeat Steps 4, 5, 6, and 7 to configure a second dial peer.
12. **srtp**
13. **codec** *codec*
14. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**dial-peer voice** *tag* **voip**
	
	Example:
	
	`Device(config)# dial-peer voice 201 voip`
		Defines a particular dial peer, to specify the method of voice encapsulation, and enters dial peer voice configuration mode.
		
		• In the example, the following parameters are set:
		  • Dial peer 201 is defined.
		  • VoIP is shown as the method of encapsulation.

Step 4	**destination-pattern** *string* **Example:** `Device(config-dial-peer)# destination-pattern 5550111`	Specifies either the prefix or the full E.164 telephone number to be used for a dial peer string. • In the example, 5550111 is specified as the pattern for the telephone number.

	Command or Action	Purpose
Step 5	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Specifies a session protocol for calls between local and remote routers using the packet network. • In the example, the **sipv2** keyword is configured so that the dial peer uses the IEFTF SIP.
Step 6	**session target ipv4:** *destination-address* **Example:** `Device(config-dial-peer)# session target ipv4:10.13.25.102`	Designates a network-specific address to receive calls from a VoIP or VoIPv6 dial peer. • In the example, the IP address of the dial peer to receive calls is configured as 10.13.25.102.
Step 7	**incoming called-number** *string* **Example:** `Device(config-dial-peer)# incoming called-number 5550111`	Specifies a digit string that can be matched by an incoming call to associate the call with a dial peer. • In the example, 5550111 is specified as the pattern for the E.164 or private dialing plan telephone number.
Step 8	**codec** *codec* **Example:** `Device(config-dial-peer)# codec g711ulaw`	Specifies the voice coder rate of speech for the dial peer. • In the example, G.711 mu-law at 64,000 bps, is specified as the voice coder rate for speech.
Step 9	**end** **Example:** `Device(config-dial-peer)#end`	Exits dial peer voice configuration mode.

Step 10	**dial-peer voice** *tag* **voip**
	
	Example:
	
	`Device(config)# dial-peer voice 200 voip`
		Defines a particular dial peer, to specify the method of voice encapsulation, and enters dial peer voice configuration mode.
		
		• In the example, the following parameters are set:
		  • Dial peer 200 is defined.
		  • VoIP is shown as the method of encapsulation.

Step 11	Repeat Steps 4, 5, 6, and 7 to configure a second dial peer.	--
Step 12	**srtp** **Example:** `Device(config-dial-peer)# srtp`	Specifies that SRTP is used to enable secure calls for the dial peer.
Step 13	**codec** *codec*	Specifies the voice coder rate of speech for the dial peer.

	Command or Action	Purpose
	Example: `Device(config-dial-peer)# codec g711ulaw`	• In the example, G.711 mu-law at 64,000 bps, is specified as the voice coder rate for speech.
Step 14	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits dial peer voice configuration mode.

Troubleshooting Tips

The following commands can help troubleshoot Cisco Unified Border Element support for SRTP-RTP internetworking:

- **show crypto pki certificates**
- **show sccp**
- **show sdspfarm**

Enabling SRTP on the Cisco UBE

You can configure SRTP with the fallback option so that a call can fall back to RTP if SRTP is not supported by the other call end. Enabling SRTP is required for supporting nonsecure supplementary services such as MoH, call forward, and call transfer.

Enabling SRTP Globally

Perform this task to enable SRTP globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **srtp fallback**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters voice-service configuration mode and specifies VoIP encapsulation as the voice-encapsulation type.
Step 4	**srtp fallback** **Example:** `RoDeviceuter(conf-voi-serv)# srtp fallback`	Enables call fallback to nonsecure mode. **Note** If the secure SIP trunk is towards the Cisco UCM, you must configure the **srtp negotiate cisco** command in voice-service configuration mode for a non-Cisco fallback to work.
Step 5	**exit** **Example:** `Device(conf-voi-serv)# exit`	Exits voice service configuration mode.

Example: Enabling SRTP Globally

```
Device(config)# voice service voip
Device(conf-voi-serv)# srtp fallback
Device(conf-voi-serv)# exit
```

Enabling SRTP on a Dial Peer

Perform this task to enable SRTP on a dial peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **srtp fallback**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 10 voip`	Defines a particular dial peer to specify VoIP as the method of voice encapsulation and enters dial peer voice configuration mode.
Step 4	**srtp fallback** **Example:** `Device(config-dial-peer)# srtp fallback`	Enables specific dial-peer calls to fall back to nonsecure mode. **Note** If the secure SIP trunk is towards the Cisco UCM, you must configure the **srtp negotiate cisco** command in dial peer voice configuration mode for a non-Cisco fallback to work.
Step 5	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits dial peer voice configuration mode.

Example: Enabling SRTP on a Dial Peer

```
Device(config)# dial-peer voice 10 voip
Device(config-dial-peer)# srtp fallback
Device(config-dial-peer)# exit
```

Troubleshooting Tips

The following commands can help troubleshoot SRTP-RTP supplementary services support on Cisco UBE:

- **debug ccsip all**
- **debug sccp all**
- **debug voip ccapi inout**

Verifying SRTP-RTP Supplementary Services Support on the Cisco UBE

Perform this task to verify the configuration for SRTP-RTP supplementary services support on the Cisco UBE. The **show** commands need not be entered in any specific order.

SUMMARY STEPS

1. **enable**
2. **show call active voice brief**
3. **show sccp connection**
4. **show dspfarm dsp active**

DETAILED STEPS

Step 1 **enable**
Enables privileged EXEC mode.

Example:

```
Device> enable
```

Step 2 **show call active voice brief**
Displays call information for voice calls in progress.

Example:

```
Device# show call active voice brief
Telephony call-legs: 0
SIP call-legs: 2
H323 call-legs: 0
Call agent controlled call-legs: 0
SCCP call-legs: 2
ulticast call-legs: 0
Total call-legs: 4
0    : 1 12:49:45.256 IST Fri Jun 3 2011.1 +29060 pid:1 Answer 10008001 connected
 dur 00:01:19 tx:1653/271092 rx:2831/464284 dscp:0 media:0
 IP 10.45.40.40:7892 SRTP: on rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a

0    : 2 12:49:45.256 IST Fri Jun 3 2011.2 +29060 pid:22 Originate 20009001 connected
 dur 00:01:19 tx:2831/452960 rx:1653/264480 dscp:0 media:0
 IP 10.45.40.40:7893 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
```

```
 long duration call detected:n long duration call duration:n/a timestamp:n/a

0    : 3 12:50:14.326 IST Fri Jun 3 2011.1 +0 pid:0 Originate  connecting
 dur 00:01:19 tx:2831/452960 rx:1653/264480 dscp:0 media:0
 IP 10.45.34.252:2000 SRTP: off rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a

0    : 5 12:50:14.326 IST Fri Jun 3 2011.2 +0 pid:0 Originate  connecting
 dur 00:01:19 tx:1653/271092 rx:2831/464284 dscp:0 media:0
 IP 10.45.34.252:2000 SRTP: on rtt:0ms pl:0/0ms lost:0/0/0 delay:0/0/0ms g711ulaw TextRelay: off
 media inactive detected:n media contrl rcvd:n/a timestamp:n/a
 long duration call detected:n long duration call duration:n/a timestamp:n/a
```

Step 3 **show sccp connection**

Displays SCCP connection details.

Example:

```
Device# show sccp connection
sess_id    conn_id      stype mode     codec   sport rport ripaddr conn_id_tx

65537      4          s-xcode sendrecv g711u   17124 2000  10.45.34.252
65537      8            xcode sendrecv g711u   30052 2000  10.45.34.252

Total number of active session(s) 1, and connection(s) 2
```

Step 4 **show dspfarm dsp active**

Displays active DSP information about the DSP farm service.

Example:

```
Device# show dspfarm dsp active
SLOT DSP VERSION  STATUS CHNL USE   TYPE    RSC_ID BRIDGE_ID PKTS_TXED PKTS_RXED

0    1   30.0.209 UP     1    USED  xcode   1      4         2876      1706
0    1   30.0.209 UP     1    USED  xcode   1      5         1698      2876

Total number of DSPFARM DSP channel(s) 1
```

Configuration Examples for CUBE Support for SRTP-RTP Internetworking

SRTP-RTP Internetworking Example

The following example shows how to configure Cisco Unified Border Element support for SRTP-RTP internetworking. In this example, the incoming call leg is RTP and the outgoing call leg is SRTP.

```
enable
 configure terminal
 ip http server
 crypto pki server 3845-cube
  database level complete
```

```
  grant auto
  no shutdown
%PKI-6-CS_GRANT_AUTO: All enrollment requests will be automatically granted.
% Some server settings cannot be changed after CA certificate generation.
% Please enter a passphrase to protect the private key or type Return to exit
Password:
Re-enter password:
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]
% SSH-5-ENABLED: SSH 1.99 has been enabled
% Exporting Certificate Server signing certificate and keys...
% Certificate Server enabled.
%PKI-6-CS_ENABLED: Certificate server now enabled.
!
crypto pki trustpoint secdsp
 enrollment url http://10.13.2.52:80
 serial-number
 revocation-check crl
 rsakeypair 3845-cube
 exit
!
crypto pki authenticate secdsp
Certificate has the following attributes:
 Fingerprint MD5: CCC82E9E 4382CCFE ADA0EB8C 524E2FC1
 Fingerprint SHA1: 34B9C4BF 4841AB31 7B0810AD 80084475 3965F140
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
crypto pki enroll secdsp
% Start certificate enrollment ..
% Create a challenge password. You will need to verbally provide this password to the CA
Administrator in order to revoke your certificate. For security reasons your password will
 not be saved in the configuration. Please make a note of it.
Password:
Re-enter password:
% The subject name in the certificate will include: 3845-CUBE
% The serial number in the certificate will be: FHK1212F4MU
% Include an IP address in the subject name? [no]:
Request certificate from CA? [yes/no]: yes
% Certificate request sent to Certificate Authority
% The 'show crypto pki certificate secdsp verbose' command will show the fingerprint.
CRYPTO_PKI:  Certificate Request Fingerprint MD5: 56CE5FC3 B8411CF3 93A343DA 785C2360
CRYPTO_PKI:  Certificate Request Fingerprint SHA1: EE029629 55F5CA10 21E50F08 F56440A2
DDC7469D
%PKI-6-CERTRET: Certificate received from Certificate Authority
!
voice-card 0
 dspfarm
 dsp services dspfarm
 voice-card 1
 dspfarm
 dsp services dspfarm
 exit
!
sccp local GigabitEthernet 0/0
sccp ccm 10.13.2.52 identifier 1 version 5.0.1
sccp
SCCP operational state bring up is successful.sccp ccm group 1
 associate ccm 1 priority 1
 associate profile 1 register sxcoder
 dspfarm profile 1 transcode universal security
  trustpoint secdsp
  codec g711ulaw
  codec g711alaw
  codec g729ar8
  codec g729abr8
  codec g729r8
  codec ilbc
  codec g729br8
  maximum sessions 84
  associate application sccp
  no shutdown
  exit
!
telephony-service
```

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface EDSP0, changed state to upsdspfarm units 1
 sdspfarm transcode sessions 84
 sdspfarm tag 1 sxcoder
 em logout 0:0 0:0 0:0
 max-ephones 4
 max-dn 4
 ip source-address 10.13.2.52
Updating CNF files
CNF-FILES: Clock is not set or synchronized, retaining old versionStamps
CNF files updating complete
 secure-signaling trustpoint secdsp
 tftp-server-credentials trustpoint scme
CNF-FILES: Clock is not set or synchronized, retaining old versionStamps
CNF files update complete (post init)
 create cnf-files
CNF-FILES: Clock is not set or synchronized, retaining old versionStamps
 no sccp
!
sccp
SCCP operational state bring up is successful.
end
%SDSPFARM-6-REGISTER: mtp-1:sxcoder IP:10.13.2.52 Socket:1 DeviceType:MTP has registered.
%SYS-5-CONFIG_I: Configured from console by console
dial-peer voice 201 voip
 destination-pattern 5550111
 session protocol sipv2
 session target ipv4:10.13.25.102
 incoming called-number 5550112
 codec g711ulaw
!
dial-peer voice 200 voip
 destination-pattern 5550112
 session protocol sipv2
 session target ipv4:10.13.2.51
 incoming called-number 5550111
 srtp
 codec g711ulaw
```

Feature Information for CUBE Support for SRTP-RTP Internetworking

Table 22: Feature Information for Cisco Unified Border Element Support for SRTP-RTP Internetworking

Feature Name	Releases	Feature Information
Cisco Unified Border Element Support for SRTP-RTP Internetworking	12.4(22)YB , 15.0(1)M	This feature allows secure enterprise-to-enterprise calls. Support for SRTP-RTP internetworking between one or multiple Cisco Unified Border Elements is enabled for SIP-SIP audio calls. The following sections provide information about this feature: The following command was introduced: **tls**.

Feature Name	Releases	Feature Information
Supplementary Services Support on Cisco UBE for RTP-SRTP Calls	15.2(1)T	The SRTP-RTP Internetworking feature was enhanced to support supplementary services for SRTP-RTP calls on Cisco UBE.
Supplementary Services Support on Cisco UBE for RTP-SRTP Calls	Cisco IOS XE Release 3.7S	The SRTP-RTP Internetworking feature was enhanced to support supplementary services for SRTP-RTP calls on Cisco UBE.

CHAPTER 24

Support for SRTP Termination

This Support for SRTP Termination feature enables Cisco Unified Border Element (Cisco UBE) support for Secure Real-time Transport Protocol (SRTP) on the Session Initiation Protocol (SIP) Trunk interface.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Support for SRTP Termination

The Support for SRTP Termination feature configures Cisco Unified Border Element (Cisco UBE) support for an Secure Real-time Transport Protocol (SRTP) connection using the AES_CM_128_HMAC_SHA1_80 crypto suite. This feature implements crypto-suite negotiation and appropriately sets up the call on the following two sides:

- The Cisco Unified Call Manager (CUCM) or IP phones side—Connection between the end devices and CUBE

- SIP Trunk side—Connection between CUBE and Service Provider

Prior to the Support for SRTP Termination feature, Cisco UBE could support an SRTP connection using the AES_CM_128_HMAC_SHA1_32 crypto suite. This crypto suite is still used by default, unless Cisco UBE is configured to use AES_CM_128_HMAC_SHA1_80 crypto suite.

Cisco UBE SRTP termination can be implemented in the following ways:

- SRTP-RTP interworking—This method is used with devices (CUCM or IP Phone devices) that still support AES_CM_128_HMAC_SHA1_32 crypto suite only.
- SRTP-SRTP pass-through—This method is used with devices that support AES_CM_128_HMAC_SHA1_80 crypto suite.

Note This method of implementation is currently supported by non-CUCM end devices like Microsoft Link. This method can also be used when CUCM or IP phone devices support AES_CM_128_HMAC_SHA1_80 crypto suite.

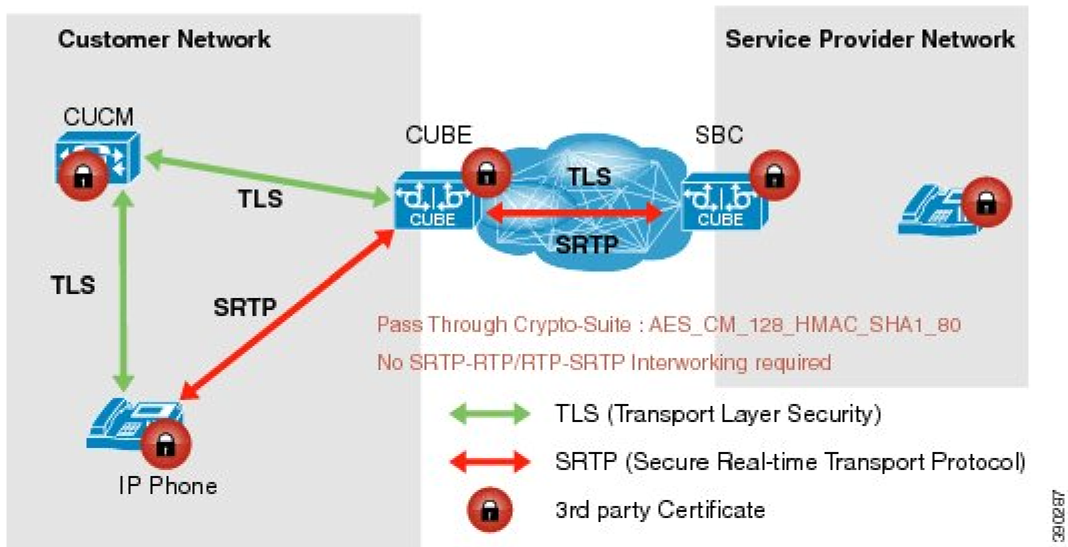
For End Devices Supporting AES_CM_128_HMAC_SHA1_80 Crypto Suite

This method is used between Cisco Unified Border Element (Cisco UBE), IP Phones, and other Cisco Unified Call Manager (CUCM) devices that support AES_CM_128_HMAC_SHA1_80 crypto suite.

- CUCM or IP Phones side—A Secure Real-time Transport Protocol (SRTP) connection using the AES_CM_128_HMAC_SHA1_80 crypto suite exists here. In the figure below, IP Phone and CUBE within the customer network connect with an SRTP connection using AES_CM_128_HMAC_SHA1_80 crypto suite.
- Session Initiation Protocol (SIP) Trunk side—An SRTP connection using the AES_CM_128_HMAC_SHA1_80 crypto suite. In the figure below, CUBE on the Customer Network

and SBC on the Service Provider Network connect with an SRTP connection using the AES_CM_128_HMAC_SHA1_80 crypto suite.

Figure 15: SRTP Connection Supporting AES_CM_128_HMAC_SHA1_80 crypto suite



For End Devices Supporting AES_CM_128_HMAC_SHA1_32 Crypto Suite

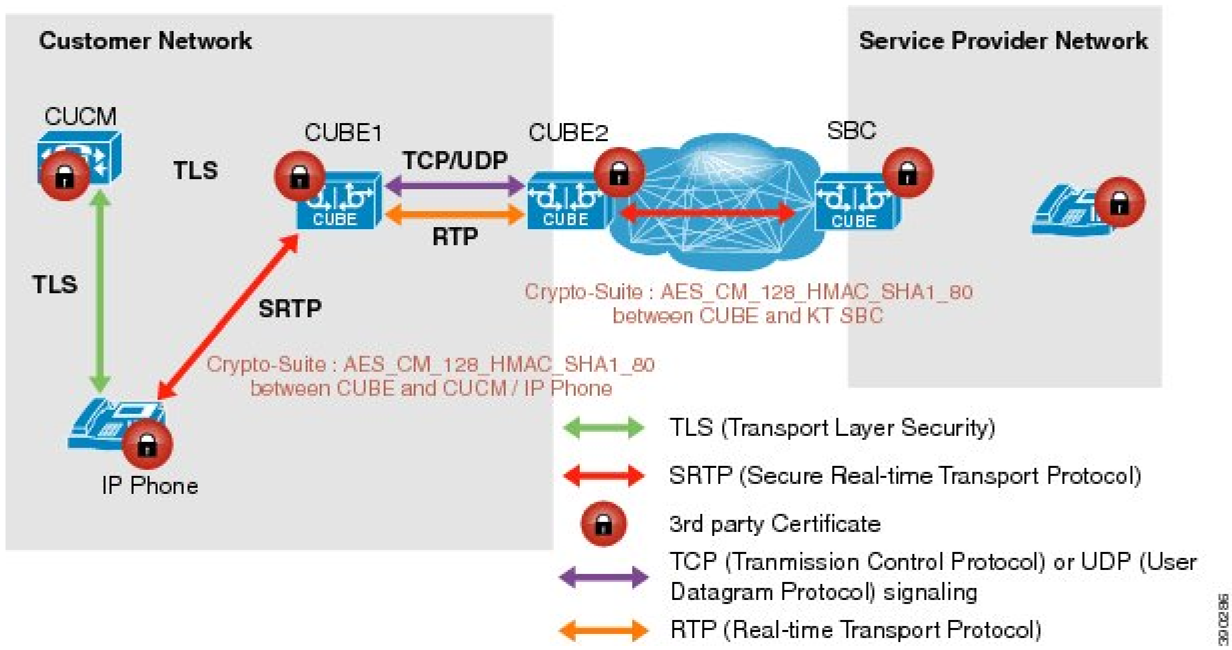
A single Cisco Unified Call Manager (Cisco UBE) device cannot terminate a Secure Real-time Transport Protocol (SRTP) connection with an IP Phone using the AES_CM_128_HMAC_SHA1_32 crypto suite and initiate an SRTP connection with an external Cisco UBE device with the AES_CM_128_HMAC_SHA1_80 crypto suite at the same time.

For Cisco Unified Call Manager (CUCM) and IP Phone devices that support only AES_CM_128_HMAC_SHA1_32 crypto suite, the interim SRTP-RTP interworking solution that is described below can be implemented.

- CUCM or IP Phone side:
    - An SRTP connection using the AES_CM_128_HMAC_SHA1_32 crypto suite exists between the IP Phone and CUBE1.
    - An RTP connection exists between CUBE1 and CUBE2.

- SIP trunk side—An SRTP connection using the AES_CM_128_HMAC_SHA1_80 crypto suite is initiated by CUBE2 here. In the image below, CUBE2 is the border element on the Customer Network and SBC is the border element on the Service Provider Network.

Figure 16: SRTP-RTP Interworking Supporting AES_CM_128_HMAC_SHA1_32 crypto suite



How to Configure Support for SRTP Termination

Configuring Crypto Authentication

Configuring Crypto Authentication (Global Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **srtp-auth** {**sha1-32** | **sha1-80**}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Specifies VoIP encapsulation and enters voice-service configuration mode.
Step 4	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters the Session Initiation Protocol (SIP) configuration mode.
Step 5	**srtp-auth** {**sha1-32** | **sha1-80**} **Example:** `Device(conf-serv-sip)# srtp-auth sha1-80`	Configures an SRTP connection on CUBE using the preferred crypto suite. • The default value is **sha1-32**.
Step 6	**end** **Example:** `Router(conf-serv-sip)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuring Crypto Authentication (Dial Peer Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **voice-class sip srtp-auth** {**sha1-32** | **sha1-80** | **system**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 15 voip`	Defines a VoIP dial peer and enters dial peer voice configuration mode.
Step 4	**voice-class sip srtp-auth** {**sha1-32** | **sha1-80** | **system**} **Example:** `Device(config-dial-peer)# voice-class sip srtp-auth sha1-80`	Configures an SRTP connection on CUBE using the preferred crypto suite. • The default value is **sha1-32**.
Step 5	**end** **Example:** `Router(conf-serv-sip)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Verifying Support for SRTP Termination

Perform this task to verify the configuration of an SRTP connection on Cisco Unified Border Element using the AES_CM_128_HMAC_SHA1_80 crypto suite. The **show** commands can be entered in any order.

SUMMARY STEPS

1. **show sip-ua calls**
2. **show sip-ua srtp**

DETAILED STEPS

Step 1 **show sip-ua calls**

Example:

The following example displays sample output for active user agent client (UAC) and user agent server (UAS) information on Session Initiation Protocol (SIP) calls:

```
Device# show sip-ua calls
Call 1
SIP Call ID                : 20894
   Media Stream 1
     Local Crypto Suite    : AES_CM_128_HMAC_SHA1_80
     Remote Crypto Suite: AES_CM_128_HMAC_SHA1_80 (AES_CM_128_HMAC_SHA1_80 AES_CM_128_HMAC_SHA1_32
)
```

Step 2 **show sip-ua srtp**

Example:

The following example displays sample output for Session Initiation Protocol (SIP) user-agent (UA) SRTP information:

```
Device# show sip-ua srtp
SIP UA SRTP
Crypto-suite Negotiation
 AES_CM_128_HMAC_SHA1_80:  3
 AES_CM_128_HMAC_SHA1_32:  2
```

Configuration Examples for Support for SRTP Termination

Example: Configuring Crypto Authentication

Example: Configuring Crypto Authentication (Global Level)

The following example shows how to configure Cisco UBE to support an SRTP connection using the AES_CM_128_HMAC_SHA1_80 crypto suite at the global level:

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# srtp-auth sha1-80
Device(conf-serv-sip)# end
```

Example: Configuring Crypto Authentication (Dial Peer Level)

The following example shows how to configure Cisco UBE to support an SRTP connection using the AES_CM_128_HMAC_SHA1_80 crypto suite at the dial peer level:

```
Device> enable
Device# configure terminal
Device(config)# dial-peer voice 15 voip
Device(config-dial-peer)# voice-class sip srtp-auth sha1-80
Device(config-dial-peer)# end
```

Additional References for Support for SRTP Termination

Related Documents

Related Topic	Document Title
Voice commands	Cisco IOS Voice Command Reference
Cisco IOS commands	Cisco IOS Master Command List, All Releases
SIP configuration tasks	SIP Configuration Guide, Cisco IOS Release 15M&T

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 Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Support for SRTP Termination

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 23: Feature Information for Support for SRTP Termination

Feature Name	Releases	Feature Information
Support for SRTP Termination	15.4(1)T	The Support for SRTP Termination feature describes how to configure Cisco Unified Border Element to support AES_CM_128_HMAC_SHA1_80 crypto suite on the Session Initiation Protocol (SIP) Trunk interface. The following commands were introduced or modified: **show sip-ua srtp**, **srtp-auth** and **voice-class sip srtp-auth**.

CHAPTER 25

Configuring RTCP Report Generation

The assisted Real-time Transport Control Protocol (RTCP) feature adds the ability for Cisco Unified Border Element (Cisco UBE) to generate standard RTCP keepalive reports on behalf of endpoints. RTCP reports determine the liveliness of a media session during prolonged periods of silence, such as call hold or mute. Therefore, it is important for the Cisco UBE to generate RTCP reports irrespective of whether the endpoints send or receive media.

Cisco UBE generates RTCP report only when inbound and outbound call legs are SIP, or SIP to H.323, or H.323 to SIP.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites

Cisco Unified Border Element

- Cisco IOS Release 15.1(2)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release <TBD> or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions

- RTCP report generation over IPv6 is not supported.
- RTCP report generation is not supported for Secure Real-time Transport Protocol (SRTP) or SRT Control Protocol (SRTCP) pass-through as Cisco UBE is not aware of the media encryption or decryption keys.
- RTCP report generation is not supported for loopback calls, T.38 fax, and modem relay calls.
- RTCP or SRTCP report generation is not supported when Cisco UBE inserts a Digital Signal Processor (DSP) for RTP-SRTP interworking on RTP and SRTP call legs.
- RTCP report generation is not supported when there is a call hold with an invalid media address such as 0.0.0.0 in Session Description Protocol (SDP) or Open Logical Channel (OLC).
- RTCP report generation is not supported for RTCP multiplexed with RTP on the same address and port.
- RTCP report generation is not supported on enterprise aggregation services routers (ASR) Cisco UBE.
- RTCP packet generation is not supported on the SIP leg when the H.323 leg puts the SIP leg on hold in a Slow Start to Delayed-Offer call.

Configuring RTCP Report Generation on Cisco UBE

RTCP keepalive packets indicate session liveliness. When configured on Cisco UBE, RTCP keepalive packets are sent on both inbound and outbound SIP or H.323 call legs.

Perform this task to configure RTCP report generation on Cisco UBE.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **allow-connections** *from-type* **to** *to-type*
5. **rtcp keepalive**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Router(config)# voice service voip`	Enters voice service configuration mode.
Step 4	**allow-connections** *from-type* **to** *to-type* **Example:** `Router(conf-voi-serv)# allow-connections sip to sip`	Allows connections between SIP endpoints in a VoIP network.
Step 5	**rtcp keepalive** **Example:** `Router(conf-voi-serv)# rtcp keepalive`	Configures RTCP keepalive report generation.
Step 6	**end** **Example:** `Router(conf-voi-serv)# end`	Exits voice service configuration mode and returns to privileged EXEC mode.

Troubleshooting Tips

Use the following debug commands for debugging related to RTCP keepalive packets:

- **debug voip rtcp packet** --Shows details related to RTCP keepalive packets such as RTCP sending and receiving paths, Call ID, Globally Unique Identifier (GUID), packet header, and so on.

```
Router# debug voip rtcp packet
01:06:27.450: //6/xxxxxxxxxxxx/RTP//Event/voip_rtp_send_rtcp_keepalive: Generate RTCP
Keepalive
*Mar 17 01:06:27.450: rtcp_send_report: Attributes
        (src ip=192.168.30.3, src port=17101, dst ip=192.168.30.4, dst port=18619
         bye=0, initial=1, ssrc=0x07111E02, keepalive=1)
*Mar 17 01:06:27.450: rtcp_construct_keepalive_report: Constructed Report
        (rtcp=0x2E5AF214, ssrc=0x07111E02, source->ssrc=0x00001E03, total_len=36)
2E5AF210:          80C90001 07111E02 81CA0006      .I.......J..
2E5AF220: 07111E02 010F302E 302E3040 392E3435  ......0.0.0@9.45
2E5AF230: 2E33302E 33000000 00                 .30.3....
```

Caution Under moderate traffic loads, the **debug voip rtp packet** command produces a high volume of output and the command should be enabled only when the call volume is very low.

- **debug voip rtp packet** --Shows details about VoIP RTP packet debugging trace.

```
Router# debug voip rtp packet
VOIP RTP All Packets debugging is on
```

- **debug voip rtp session** --Shows all RTP session debug information.

```
Router# debug voip rtp session
VOIP RTP All Events debugging is on
```

- **debug voip rtp error** --Shows details about debugging trace for RTP packet error cases.

```
Router# debug voip rtp error
VOIP RTP Errors debugging is on
```

- **debug ip rtp protocol** --Shows details about RTP protocol debugging trace.

```
Router# debug ip rtp protocol
RTP protocol debugging is on
```

- **debug voip rtcp session** --Shows all RTCP session debug information.

```
Router# debug voip rtcp session
VOIP RTCP Events debugging is on
```

- **debug voip rtcp error** -- Shows details about debugging trace for RTCP packet error cases.

```
Router# debug voip rtcp error
VOIP RTCP Errors debugging is on
```

Feature Information for Configuring RTCP Report Generation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Feature History Table entry for the Cisco Unified Border Element. .

Table 24: Feature Information for Configuring RTCP Report Generation

Feature Name	Releases	Feature Information
Assisted RTCP	15.1(2)T	This feature adds the ability for Cisco UBE to generate standard RTCP keepalive reports on behalf of endpoints and ensures the liveliness of a media session during prolonged periods of silence, such as call hold. The following commands were introduced or modified in this release: **rtcp keepalive**, **debug voip rtcp**, **debug voip rtp**, **debug ip rtp protocol**, and **ip rtcp report interval**.

Feature History Table entry for the Cisco Unified Border Element (Enterprise) .

Table 25: Feature Information for Configuring RTCP Report Generation

Feature Name	Releases	Feature Information
Assisted RTCP	TBD	This feature adds the ability for Cisco UBE to generate standard RTCP keepalive reports on behalf of endpoints and ensures the liveliness of a media session during prolonged periods of silence, such as call hold. The following commands were introduced or modified in this release: **rtcp keepalive**, **debug voip rtcp**, **debug voip rtp**, **debug ip rtp protocol**, and **ip rtcp report interval**.

CHAPTER 26

SIP SRTP Fallback to Nonsecure RTP

The SIP SRTP Fallback to Nonsecure RTP feature enables a Cisco IOS Session Initiation Protocol (SIP) gateway to fall back from Secure Real-time Transport Protocol (SRTP) to Real-time Transport Protocol (RTP) by accepting or sending an RTP/Audio-Video Profile(AVP) (RTP) profile in response to an RTP/SAVP (SRTP) profile. This feature also allows inbound and outbound SRTP calls with nonsecure SIP signaling schemes (such as SIP URL) and provides the administrator the flexibility to configure Transport Layer Security (TLS), IPsec, or any other security mechanism used in the lower layers for secure signaling of crypto attributes.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for SIP SRTP Fallback to Nonsecure RTP

Cisco Unified Border Element

- Cisco IOS Release 12.4(22)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.1S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Configuring SIP SRTP Fallback to Nonsecure RTP

To enable this feature, see the "Configuring SIP Support for SRTP" section of the Cisco IOS SIP Configuration Guide, Release 15.1 at the following URL: http://www.cisco.com/en/US/docs/ios/voice/sip/configuration/guide/sip_cg-srtp_ps10592_TSD_Products_Configuration_Guide_Chapter.html

Detailed command information for the **srtp**, **srtp negotiate**, and **voice-class sip srtp negotiate**commands is located in the Cisco IOS Voice Command Reference http://www.cisco.com/en/US/docs/ios/voice/command/reference/vr_book.html

Feature Information for SIP SRTP Fallback to Nonsecure RTP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 26: Feature Information for SIP SRTP Fallback to Nonsecure RTP

Feature Name	Releases	Feature Information
SIP SRTP Fallback to Nonsecure RTP	12.4(22)T	The SIP SRTP Fallback to Nonsecure RTP feature enables a Cisco IOS Session Initiation Protocol (SIP) gateway to fall back from SRTP to RTP by accepting or sending an RTP/AVP(RTP) profile in response to an RTP/SAVP(SRTP) profile. This feature also allows inbound and outbound SRTP calls with nonsecure SIP signaling schemes (such as SIP URL) and provides the administrator the flexibility to configure TLS, IPsec, or any other security mechanism used in the lower layers for secure signaling of crypto attributes. The following commands were introduced or modified: **srtp (voice)**, **srtp negotiate**, and **voice-class sip srtp negotiate**

Feature Name	Releases	Feature Information
SIP SRTP Fallback to Nonsecure RTP	Cisco IOS XE Release 3.1S	The SIP SRTP Fallback to Nonsecure RTP feature enables a Cisco IOS Session Initiation Protocol (SIP) gateway to fall back from SRTP to RTP by accepting or sending an RTP/AVP(RTP) profile in response to an RTP/SAVP(SRTP) profile. This feature also allows inbound and outbound SRTP calls with nonsecure SIP signaling schemes (such as SIP URL) and provides the administrator the flexibility to configure TLS, IPsec, or any other security mechanism used in the lower layers for secure signaling of crypto attributes. The following commands were introduced or modified: **srtp (voice)**, **srtp negotiate**, and **voice-class sip srtp negotiate**

CHAPTER 27

Configuring Support for Interworking Between RSVP Capable and RSVP Incapable Networks

The Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based Resource Reservation Protocol (RSVP) support for basic audio call and supplementary services on Cisco UBE. This feature improves the interoperability between RSVP and non-RSVP networks. RSVP functionality added to Cisco UBE helps you to reserve the required bandwidth before making a call.

This feature extends RSVP support to delayed-offer to delayed-offer and delayed-offer to early-offer calls, along with the early-offer to early-offer calls.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites

- RSVP policies allow you to configure separate bandwidth pools with varying limits so that any one application, such as video, can consume all the RSVP bandwidth on a specified interface at the expense of other applications, such as voice, which would be dropped.
- To limit bandwidth per application, you must configure a bandwidth limit before configuring Support for the Interworking Between RSVP Capable and RSVP Incapable Networks feature. See the Configuring RSVP on an Interface task.

Cisco Unified Border Element

- Cisco IOS Release 15.0(1)XA or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release <TBD> or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions

The Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature has the following restrictions:

- Segmented RSVP is not supported.
- Interoperability between Cisco UBE and Cisco Unified Communications Manager is not available.
- RSVP-enabled video calls are not supported.

Configuring RSVP on an Interface

You must allocate some bandwidth for the interface before enabling RSVP. Perform this task to configure RSVP on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *port*
4. **ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot* / *port* **Example:** `Router(config)# interface FastEthernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]] **Example:** `Router(config-if)# ip rsvp bandwidth 10000 100000`	Enables RSVP for IP on an interface.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Optional RSVP on the Dial Peer

Perform this task to configure optional RSVP at the dial peer level. This configuration allows you to have uninterrupted call even if there is a failure in bandwidth reservation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**]
5. **req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer 77 voip`	Enters dial peer voice configuration mode.
Step 4	**no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] **Example:** `Router(config-dial-peer)# no acc-qos controlled-load`	Removes any value configured for the **acc-qos** command. • • **controlled-load** --Indicates that RSVP guarantees a single level of preferential service, presumed to correlate to a delay boundary. The controlled load service uses admission (or capacity) control to ensure that preferential service is received even when the bandwidth is overloaded. • **guaranteed-delay** --Indicates that RSVP reserves bandwidth and guarantees a minimum bit rate and preferential queueing if the bandwidth reserved is not exceeded.
Step 5	**req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]	Configures the desired quality of service (QoS) to be used. • Calls continue even if there is a failure in bandwidth reservation.

	Command or Action	Purpose
	Example: `Router(config-dial-peer)# req-qos controlled-load`	**Note** Configure the **req-qos** commandusing the same keyword that you used to configure the **acc-qos** command, either **controlled-load** or **guaranteed-delay**. That is, if you configured **acc-qos controlled-load** command in the previous step, then use the **req-qos controlled-load** command here.
Step 6	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring EO to EO DO to DO and DO to EO at the Dial Peer

Perform this task to configure support for EO to EO, DO to DO, and DO to EO at the dial peer level.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**]
5. **req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]
6. **exit**
7. **interface** *type* **slot/port**
8. **ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]]
9. **exit**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 77 voip`	Enters dial peer voice configuration mode.
Step 4	**no acc-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] **Example:** `Router(config-dial-peer)# no acc-qos controlled-load`	Removes any value configured for the **acc-qos** command. • • **controlled-load** --Indicates that RSVP guarantees a single level of preferential service, presumed to correlate to a delay boundary. The controlled load service uses admission (or capacity) control to ensure that preferential service is received even when the bandwidth is overloaded. • **guaranteed-delay** --Indicates that RSVP reserves bandwidth and guarantees a minimum bit rate and preferential queueing if the bandwidth reserved is not exceeded.
Step 5	**req-qos** {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]] **Example:** `Router(config-dial-peer)# req-qos controlled-load`	Configures the desired quality of service (QoS) to be used. • Calls continue even if there is a failure in bandwidth reservation. **Note** Configure the **req-qos** commandusing the same keyword that you used to configure the **acc-qos** command, either **controlled-load** or **guaranteed-delay**. That is, if you configured the **acc-qos controlled-load** command in the previous step, then use the **req-qos controlled-load** command here.
Step 6	**exit** **Example:** `Router(config-dial-peer)# exit`	Exits dial peer voice configuration mode and returns to global configuration mode.
Step 7	**interface** *type* **slot/port** **Example:** `Router(config)# interface FastEthernet 0/1`	Configures an interface type and enters interface configuration mode.

	Command or Action	Purpose
Step 8	**ip rsvp bandwidth** [*reservable-bw* [*max-reservable-bw*] [**sub-pool** *reservable-bw*]] **Example:** `Router(config-if)# ip rsvp bandwidth 10000 100000`	Enables RSVP for IP on an interface.
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Mandatory RSVP on the Dial Peer

Perform this task to configure Mandatory RSVP on the dial peer. This configuration ensures that the call does not connect if sufficient bandwidth is not allocated.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **acc-qos** {**best-effort** | **controlled-load** | **guaranteed-delay**} [**audio** | **video**]
5. **req-qos** {**best-effort** [**audio** | **video**] | {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer 77 voip`	Enters dial peer voice configuration mode.
Step 4	**acc-qos** {**best-effort** | **controlled-load** | **guaranteed-delay**} [**audio** | **video**] **Example:** `Router(config-dial-peer)# acc-qos best-effort`	Configures mandatory RSVP on the dial-peer. • **best-effort** --Indicates that Resource Reservation Protocol (RSVP) makes no bandwidth reservation. This is the default. • **controlled-load** --Indicates that RSVP guarantees a single level of preferential service, presumed to correlate to a delay boundary. The controlled load service uses admission (or capacity) control to ensure that preferential service is received even when the bandwidth is overloaded. • **guaranteed-delay** --Indicates that RSVP reserves bandwidth and guarantees a minimum bit rate and preferential queueing if the bandwidth reserved is not exceeded.
Step 5	**req-qos** {**best-effort** [**audio** | **video**] | {**controlled-load** | **guaranteed-delay**} [**audio** | **video**] [**bandwidth** [**default** *bandwidth-value*] [**max** *bandwidth-value*]]} **Example:** `Router(config-dial-peer)# req-qos controlled-load`	Configures mandatory RSVP on the dial-peer. • Calls continue even if there is a drop in the bandwidth reservation.
Step 6	**end** **Example:** `Router(config-dial-peer)# end`	(Optional) Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring Midcall RSVP Failure Policies

Perform this task to enable call handling policies for a midcall RSVP failure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **voice-class sip rsvp-fail-policy** {**video** | **voice**} **post-alert** {**optional keep-alive** | **mandatory** {**keep-alive** | **disconnect retry** *retry-attempts*}} **interval** *seconds*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**voice-class sip rsvp-fail-policy** {**video** | **voice**} **post-alert** {**optional keep-alive** | **mandatory** {**keep-alive** | **disconnect retry** *retry-attempts*}} **interval** *seconds* **Example:** `Router(config-dial-peer)# voice-class sip rsvp-fail-policy voice post-alert mandatory keep-alive interval 50`	Enables call handling policies for a midcall RSVP failure. • • **optional keep-alive** --The keepalive messages are sent when RSVP fails only if RSVP negotiation is optional. • **mandatory keep-alive** --The keepalive messages are sent when RSVP fails only if RSVP negotiation is mandatory. **Note** Keepalive messages are sent at 30-second intervals when a postalert call fails to negotiate RSVP regardless of the RSVP negotiation setting (mandatory or optional).
Step 5	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring DSCP Values

Perform this task to configure different Differentiated Services Code Point (DSCP) values based on RSVP status.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos dscp** {*dscp-value* | *set-af* | *set-cs* | **default** | **ef**} {**signaling** | **media** [**rsvp-pass** | **rsvp-fail**] | **video**[**rsvp-none**| **rsvp-pass** | **rsvp-fail**]}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**ip qos dscp** {*dscp-value* | *set-af* | *set-cs* | **default** | **ef**} {**signaling** | **media** [**rsvp-pass** | **rsvp-fail**] | **video**[**rsvp-none**| **rsvp-pass** | **rsvp-fail**]} **Example:** `Router(config-dial-peer)# ip qos dscp af11 media rsvp-pass`	Configures DSCP values based on RSVP status. • • **media rsvp-pass** --Specifies that the DSCP value applies to media packets with successful RSVP reservations.

	Command or Action	Purpose
		• **media rsvp-fail** --Specifies that the DSCP value applies to packets (media or video) with failed RSVP reservations. • The default DSCP value for all media (voice and fax) packets is **ef**. **Note** You must configure the DSCP values for all cases: **media rsvp-pass** and **media rsvp-fail**.
Step 5	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring an Application ID

Perform this task to configure a specific application ID for RSVP establishment.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos policy-locator** {**video** | **voice**} [**app** *app-string*] [**guid** *guid-string*] [**sapp** *subapp-string*] [**ver** *version-string*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**ip qos policy-locator** {**video** | **voice**} [**app** *app-string*] [**guid** *guid-string*] [**sapp** *subapp-string*] [**ver** *version-string*] **Example:** `Router(config-dial-peer)# ip qos policy-locator voice`	Configures a QoS policy locator (application ID) used to deploy RSVP policies for specifying bandwidth reservations on Cisco IOS Session Initiation Protocol (SIP) devices.
Step 5	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer voice configuration mode and returns to privileged EXEC mode.

Configuring Priority

Perform this task to configure priorities for call preemption.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *tag* **voip**
4. **ip qos defending-priority** *defending-pri-value*
5. **ip qos preemption-priority** *preemption-pri-value*
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *tag* **voip** **Example:** `Router(config)# dial-peer voice 66 voip`	Enters dial peer voice configuration mode.
Step 4	**ip qos defending-priority** *defending-pri-value* **Example:** `Router(config-dial-peer)# ip qos defending-priority 66`	Configures the RSVP defending priority value for determining QoS.
Step 5	**ip qos preemption-priority** *preemption-pri-value* **Example:** `Router(config-dial-peer)# ip qos preemption-priority 75`	Configures the RSVP preemption priority value for determining QoS.
Step 6	**end** **Example:** `Router(config-dial-peer)# end`	Exits dial peer configuration mode and returns to privileged EXEC mode.

Troubleshooting the Support for Interworking Between RSVP Capable and RSVP Incapable Networks Feature

Use the following commands to debug any errors that you may encounter when you configure the Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature.

- **debug call rsvp-sync events**
- **debug call rsvp-sync func-trace**
- **debug ccsip all**

- **debug ccsip messages**
- **debug ip rsvp messages**
- **debug sccp all**

Verifying Support for Interworking Between RSVP Capable and RSVP Incapable Networks

This task explains how to display information to verify the configuration for the Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature. These commands need not be entered in any specific order.

SUMMARY STEPS

1. **enable**
2. **show sip-ua calls**
3. **show ip rsvp installed**
4. **show ip rsvp reservation**
5. **show ip rsvp interface detail** [*interface-type number*]
6. **show sccp connections details**
7. **show sccp connections rsvp**
8. **show sccp connections internal**
9. **show sccp** [**all** | **connections** | **statistics**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show sip-ua calls** **Example:** `Router# show sip-ua calls`	(Optional) Displays active user agent client (UAC) and user agent server (UAS) information on SIP calls.
Step 3	**show ip rsvp installed** **Example:** `Router# show ip rsvp installed`	(Optional) Displays RSVP-related installed filters and corresponding bandwidth information.

	Command or Action	Purpose
Step 4	**show ip rsvp reservation** **Example:** `Router# show ip rsvp reservation`	(Optional) Displays RSVP-related receiver information currently in the database.
Step 5	**show ip rsvp interface detail** [*interface-type number*] **Example:** `Router# show ip rsvp interface detail GigabitEthernet 0/0`	(Optional) Displays the interface configuration for hello.
Step 6	**show sccp connections details** **Example:** `Router# show sccp connections details`	(Optional) Displays SCCP connection details, such as call-leg details.
Step 7	**show sccp connections rsvp** **Example:** `Router# show sccp connections rsvp`	(Optional) Displays information about active SCCP connections that are using RSVP.
Step 8	**show sccp connections internal** **Example:** `Router# show sccp connections internal`	(Optional) Displays the internal SCCP details, such as time-stamp values.
Step 9	**show sccp** [**all** | **connections** | **statistics**] **Example:** `Router# show sccp statistics`	(Optional) Displays SCCP information, such as administrative and operational status.

Feature Information for Configuring Support for Interworking Between RSVP Capable and RSVP Incapable Networks

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.
ISR Feature table entry

Table 27: Feature Information for Interworking Between RSVP Capable and RSVP Incapable Networks

Feature Name	Releases	Feature Information
Interworking Between RSVP Capable and RSVP Incapable Networks	15.0(1)XA 15.1(3)T	The Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based RSVP support for basic audio call and supplementary services on the Cisco UBE. Support for Configuring EO-EO, DO-DO and DO-EO support on dial peer was introduced in 15.1(3)T. 15.1(3)T--Configuring EO-EO, DO-DO and DO-EO support on dial peer.

```
ASR Feature table entry
```

Table 28: Feature Information for Interworking Between RSVP Capable and RSVP Incapable Networks

Feature Name	Releases	Feature Information
Interworking Between RSVP Capable and RSVP Incapable Networks	TBD	The Support for Interworking Between RSVP Capable and RSVP Incapable Networks feature provides precondition-based RSVP support for basic audio call and supplementary services on the Cisco UBE.

CHAPTER 28

VoIP for IPv6

VoIPv6 adds IPv6 capability to existing VoIP features. VoIPv6 requires IPv6 and IPv4 dual-stack support on voice gateways and MTP, IPv6 support for SIP trunks, and SCCP-controlled analog voice phones. In addition, the SBC functionality of connecting SIP IPv4 or H.323 IPv4 network to SIP IPv6 network is implemented on a Cisco Unified Border Element to facilitate migration from VoIPv4 to VoIPv6.

Prerequisites

Listing the minimum SW release is required. Use the following wording:

Cisco Unified Border Element

- Cisco IOS Release 12.4(22)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.3S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Configuring VoIP for IPv6

To enable this feature, see the "Implementing VoIP for IPv6" section in the *Cisco IOS IPv6 Configuration Guide, Release 15.0*.

Detailed command information for the VoIP for IPv6 commands is located in the *Cisco IOS IPv6 Command Reference*.

Feature Information for VoIP for IPv6

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 29: Feature Information for VoIP for IPv6

Feature Name	Releases	Feature Information
VoIP for IPv6	12.4(22)T	VoIPv6 adds IPv6 capability to existing VoIP features. Additionally, the SBC functionality of connecting SIP IPv4 or H.323 IPv4 network to SIP IPv6 network is implemented on a Cisco Unified Border Element to facilitate migration from VoIPv4 to VoIPv6. The following commands were introduced or modified: None.

Table 30: Feature Information for VoIP for IPv6

Feature Name	Releases	Feature Information
VoIP for IPv6	Cisco IOS XE Release 3.3S	VoIPv6 adds IPv6 capability to existing VoIP features. Additionally, the SBC functionality of connecting SIP IPv4 or H.323 IPv4 network to SIP IPv6 network is implemented on a Cisco Unified Border Element to facilitate migration from VoIPv4 to VoIPv6. The following commands were introduced or modified: None.

CHAPTER 29

Mid-call Signaling Consumption

The Cisco Unified Border Element BE Mid-call Signaling support aims to reduce the interoperability issues that arise due to consuming mid-call RE-INVITES/UPDATES.

Mid-call Re-INVITEs/UPDATEs can be consumed in the following ways:

- Mid-call Signaling Passthrough - Media Change
- Mid-call Signaling Block
- Mid-call Signaling Codec Preservation

Note This feature should be used as a last resort only when there is no other option in CUBE. This is because configuring this feature can break video-related features. For Delay-offer Re-INVITE, the configured codec will be passed as an offer in 200 message to change the codec, the transcoder is added in the answer.

Feature Information for Mid-call Signaling

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 31: Feature Information for Mid-call Signaling

Feature Name	Releases	Feature Information
Mid-call Re-INVITE Consumption	Cisco IOS 15.2(1)T Cisco IOS XE 3.6S	The Mid-call Re-INVITE consumption feature consumes mid-call Re-INVITEs from CUBE and helps to avoid interoperability issues because of these re-invites The following commands were introduced or modified: **midcall-signaling**.
Mid-call Codec Preservation	Cisco IOS 15.3(2)T Cisco IOS XE 3.9S	The Mid-call Codec Preservation feature helps to disables codec negotiation in the middle of a call and preserves the codec negotiated before the call. The following commands were introduced or modified: **midcall-signaling preserve-codec** , **voice-class sip midcall-signaling preserve-codec** .
Mid-call Re-INVITE Consumption Enhancements	Cisco IOS 15.5(3)M Cisco IOS XE 3.16S	Mid-call signaling Re-INVITE consumption is enhanced to support: • Re-INVITE based call transfer • Call transfer with REFER Consume • Normalization of call hold in a call set-up

Prerequisites

- Enable CUBE application on a device
- Cisco IOS Release 15.2(1)T or later, or Cisco IOS-XE Release 15.2(2)S or later must be installed.
- **supplementary-service media-renegotiate** must be configured in global voice service voip mode.

Mid-call Signaling Passthrough - Media Change

Passthrough media change method optimizes or consumes mid-call, media-related signaling within the call. Mid-call signaling changes will be passed through only when bidirectional media like T.38 or video is added. The command **midcall-signaling passthru media-change** needs to be configured to enable passthrough media change.

Restrictions for Mid-call Signaling Passthrough - Media Change

- SIP-H.323 calls are not supported.
- TDM Gateways are not supported.
- Session Description Protocol (SDP) -passthrough is not supported
- When **codec T** is configured, the offer from CUBE has only audio codecs, and so the video codecs are not consumed.
- Re-invites are not consumed if media flow-around is configured.
- Re-invites are not consumed if media anti-tromboning is configured.
- Video transcoding is not supported.
- Secure Real-time Protocol - Real-time Protocol (SRTP-RTP) supplementary services are not supported.
- Multicast Music On Hold (MMOH) is not supported.
- When the **midcall-signaling passthru media-change** command is configured and high-density transcoder is enabled, there might be some impact on Digital Signal Processing (DSP) resources as the transcoder might be used for all the calls.
- Session timer is handled leg by leg whenever this feature is configured.
- More than two m-lines in the SDP is not supported.
- Alternative Network Address Types (ANAT) is not supported.
- Video calls and Application streams are not supported when mid-call signaling block is configured.

Behavior of Mid-call Re-INVITE Consumption

- If mid-call signaling block is enabled on either of call-legs, video parameters and application streams are not negotiated, and are rejected in the answer.
- When flow around and offer-all is configured, CUBE performs codec renegotiation even if mid-call signaling block is configured globally.
- Below behavior is for refer consume scenario:
    - REFER consume is supported for blind, alert and consult call transfers.
    - Existing codecs or DTMF is used for local bridging of new call legs. No Re-INVITE or UPDATE is sent for media re-negotiation after REFER.

  - ◦ Call gets dropped when DSP is required but not available.
  - ◦ A call can be escalated to video only if transferee and transfer-to dial-peers do not have mid-call signaling block configured.
  - ◦ Video calls are de-escalated if mid-call signaling block configuration on transfer-to dial-peer.
  - ◦ For Re-INVITE based call-transfer involving Cisco Unified Communications Manager, all Re-INVITE are locally answered and transcoder is invoked if negotiated codecs are different than the codecs before call-transfer.

- The following table provides the details of the behavior when the initial call is establish without 'sendrecv' parameter, that means, the initial call is established with 'sendonly', 'recvonly' or 'inactive'.

Scenario	Behavior
If an Offer is received with 'sendonly' and mid-call block is configured on any or both call legs	Offer is sent with 'sendrecv'.
If an Answer is received with 'sendonly' and the peer leg supports mid-call signaling	Answer is sent with 'sendonly'. Resume transaction is end-to-end.
If an Answer is received with 'sendonly' and the peer leg does not supports mid-call signaling	Answer is sent with 'sendrecv'. Resume transaction is consumed.
If Offer as well as Answer is received with 'sendonly' and Offering leg does not support mid-call signaling	Answer is sent with 'recvonly'. Resume from Offering leg is end-to-end. Resume from answering leg is consumed.
If Offer as well as Answer is received with 'sendonly' and Answering leg does not support mid-call signaling	Answer is sent with 'inactive'. Resume from Offering leg is consumed. Resume from answering leg is end-to-end.
If Offer as well as Answer is received with 'sendonly' and both legs do not support mid-call signaling	Answer is sent with ' recvonly'. Resume transaction is consumed.

Configuring Passthrough of Mid-call Signalling

Perform this task to configure passthrough of mid-call signaling (as Re-invites) only when bidirectional media is added.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Configure passthrough of mid-call signaling changes only when bidirectional media is added.
    - In Global VoIP SIP configuration mode

      midcall-signaling passthru media-change
    - In dial-peer configuration mode

      voice-class sip mid-call signaling passthru media-change
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Configure passthrough of mid-call signaling changes only when bidirectional media is added.
	• In Global VoIP SIP configuration mode
	midcall-signaling passthru media-change
	• In dial-peer configuration mode
	voice-class sip mid-call signaling passthru media-change
	
	Example:
	In Global VoIP SIP configuration mode:
	`Device(config)# voice service voip`
	`Device(conf-voi-serv)# sip`
	`Device(conf-serv-sip)# midcall-signaling passthru media-change`
	
	Example:
	In Dial-peer configuration mode:
	`Device(config)# dial-peer voice 2 voip`
	`Device(config-dial-peer)# voice-class sip mid-call signaling passthru media-change`
		Re-Invites are passed through only when bidirectional media is added.

	Command or Action	Purpose
Step 4	**end**	Exits to privileged EXEC mode.

Example Configuring Passthrough SIP Messages at Dial Peer Level

The following example shows how to passthrough SIP messages at the dial peer Level:

```
dial-peer voice 600 voip
 destination-pattern 2222222222
 session protocol sipv2
 session target ipv4:9.45.38.39:9001
voice-class sip mid-call signaling passthru media-change
 incoming called-number 1111111111
voice-class codec  2 offer-all
dial-peer voice 400 voip
 destination-pattern 1111111111
 session protocol sipv2
 session target ipv4:9.45.38.39:9000
 incoming called-number 2222222222
voice-class codec 1 offer-all
```

Example Configuring Passthrough SIP Messages at the Global Level

The following example shows how to passthrough SIP messages at the global level:

```
Device(config)# voice service voip
Device(conf-voi-serv)# no ip address trusted authenticate
Device(conf-voi-serv)# allow-connections sip to sip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# midcall-signaling passthru media-change
```

Mid-call Signaling Block

The Block method blocks all mid-call media-related signaling to the specific SIP trunk. The command **midcall-signaling block** needs to be configured to enable this behavior. Video escalation and T.38 call flow are rejected when the **midcall-signaling block** command is configured. This command should be configured only when basic call is the focus and mid-call can be consumed.

Restrictions for Mid-Call Signaling Block

- SIP-H.323 calls are not supported.
- TDM Gateways are not supported.
- Session Description Protocol (SDP) -passthrough is not supported
- Video calls and Application streams are not supported.
- When media flow-around is configured, Mid-call INVITE is rejected with 488 error message.
- Re-invites are not consumed if media anti-tromboning is configured.

- SRTP-RTP supplementary services are not supported.
- Multicast Music On Hold (MMOH) is not supported.
- When the **midcall-signaling passthru media-change** command is configured and high-density transcoder is enabled, there might be some impact on Digital Signal Processing (DSP) resources as the transcoder might be used for all the calls.
- Session timer is handled leg by leg whenever this feature is configured.
- More than two m-lines in the SDP is not supported.
- Alternative Network Address Types (ANAT) is not supported.

Blocking Mid-Call Signaling

Perform this task to block mid-call signaling:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Configure blocking of mid-call signaling changes:
   - In Global VoIP SIP configuration mode
     midcall-signaling block
   - In dial-peer configuration mode
     voice-class sip mid-call signaling block
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	Configure blocking of mid-call signaling changes: • In Global VoIP SIP configuration mode **midcall-signaling block**	Mid-call signaling is always blocked.

	Command or Action	Purpose

	• In dial-peer configuration mode
	voice-class sip mid-call signaling block
	
	Example:
	In Global VoIP SIP configuration mode:
	`Device(config)# voice service voip`
	`Device(conf-voi-serv)# sip`
	`Device(conf-serv-sip)# midcall-signaling block`
	
	Example:
	In Dial-peer configuration mode:
	`Device(config)# dial-peer voice 2 voip`
	`Device(config-dial-peer)# voice-class sip mid-call signaling block`	

Step 4	**end**	Exits to privileged EXEC mode.

Example Blocking SIP Messages at Dial Peer Level

```
dial-peer voice 107 voip
 destination-pattern 74000
 session protocol sipv2
 session target ipv4:9.45.36.9
 incoming called-number 84000
 voice-class codec 1 offer-all
!
dial-peer voice 110 voip
 destination-pattern 84000
 session protocol sipv2
 session target ipv4:9.45.35.2
 incoming called-number 74000
 voice-class codec 1 offer-all
 voice-class sip mid-call signaling block
!
```

Example: Blocking SIP Messages at the Global Level

The following example shows how to block SIP messages at the global Level

```
Device(config)#voice service voip
Device(config-voi-serv)#no ip address trusted authenticate
Device(config-voi-serv)#allow-connections sip to sip
Device(config-voi-serv)#sip
Device(config-serv-sip)#midcall-signaling block
```

Mid Call Codec Preservation

Mid call codec preservation defines whether a codec can be negotiated after a call has been initiated. You can enable or disable codec negotiation in the middle of a call.

Configuring Mid Call Codec Preservation

This tasks disables codec negotiation in the middle of a call and preserves the codec negotiated before the call.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Enter one of the following to disable midcall codec renegotiation:
    - In Global VoIP SIP configuration mode
      midcall-signaling preserve-codec
    - In dial-peer configuration mode
      voice-class sip midcall-signaling preserve-codec
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Enter one of the following to disable midcall codec renegotiation:
	• In Global VoIP SIP configuration mode
	midcall-signaling preserve-codec
	• In dial-peer configuration mode
	voice-class sip midcall-signaling preserve-codec
	
	Example:
	`Device(config)# voice service voip`
	`Device(conf-voi-serv)# sip`
	`Device(conf-serv-sip)# midcall-signaling preserve-codec`
		Disables codec negotiation in the middle of a call and preserves the codec negotiated before the call.

	Command or Action	Purpose

	Example:
	`Device(config)# dial-peer voice 10 voip`
	`Device(conf-dial-peer)# voice-class sip midcall-signaling preserve-codec`	

Step 4	**end** **Example:** `Device(conf-serv-sip)# end`	Exits to privileged EXEC mode.

Example: Configuring Mid Call Codec Preservation at the Dial Peer Level

Example: Configuring Mid Call Codec Preservation at the Dial Peer Level

```
dial-peer voice 107 voip
 destination-pattern 74000
 session protocol sipv2
 session target ipv4:9.45.36.9
 incoming called-number 84000
 voice-class codec 1 offer-all
!
dial-peer voice 110 voip
 destination-pattern 84000
 session protocol sipv2
 session target ipv4:9.45.35.2
 incoming called-number 74000
 voice-class codec 1 offer-all
 voice-class sip midcall-signaling preserve-codec
!
```

Example: Configuring Mid Call Codec Preservation at the Global Level

Example: Configuring Mid Call Codec Preservation at the Global Level

```
Device(config)# voice service voip
Device(conf-voi-serv)# no ip address trusted authenticate
Device(conf-voi-serv)# allow-connections sip to sip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# midcall-signaling preserve-codec
```

CHAPTER 30

Support for Software Media Termination Point

The Support for Software Media Termination Point (MTP) feature bridges the media streams between two connections allowing Cisco Unified Communications Manager (Cisco UCM) to relay calls that are routed through SIP or H.323 endpoints via Skinny Call Control Protocol (SCCP) commands. These commands allow Cisco UCM to establish an MTP for call signaling.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Support for Software Media Termination Point

This feature extends the software MTP support to the Cisco Unified Border Element (Enterprise). Software MTP is an essential component of large-scale deployments of Cisco UCM. This feature enables new capabilities so that the Cisco UBE can function as an Enterprise Edge Cisco Session Border Controller for large-scale deployments that are moving to SIP trunking.

How to Configure Support for Software Media Termination Point

Prerequisites

- For the software MTP to function properly, codec and packetization must be configured the same way on both in call legs and out call legs.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 2.6 or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Restrictions

- RSVP Agent is not supported in software MTP.
- Hardware MTP for repacketization is not supported.
- Call Threshold is not supported for standalone software MTP.
- Per-call debugging is not supported.

Configuring Support for Software Media Termination Point

To enable and configure the Support for Software Media Termination Point feature, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **sccp local** *interface-type interface-number* [**port** *port-number*]
4. **sccp ccm** {*ipv4-address* | *ipv6-address* | *dns*} **identifier** *identifier-number* [**port** *port-number*] **version** *version-number*
5. **sccp**
6. **sccp ccm group** *group-number*
7. **associate ccm** *identifier-number* **priority** *number*
8. **associate profile** *profile-identifier* **register** *device-name*
9. **dspfarm profile** *profile-identifier* {**conference** | **mtp** | **transcode**} [**security**]
10. **maximum sessions** {**hardware** | **software**} *number*
11. **associate application sccp**
12. **no shutdown**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**sccp local** *interface-type interface-number* [**port** *port-number*] **Example:** `Router(config)# sccp local gigabitethernet0/0/0`	Selects the local interface that SCCP applications (transcoding and conferencing) use to register with Cisco UCM. • *interface type* --Can be an interface address or a virtual-interface address such as Ethernet. • *interface number* --Interface number that the SCCP application uses to register with Cisco UCM. • (Optional) **port** *port-number*--Port number used by the selected interface. Range is 1025 to 65535. Default is 2000.
Step 4	**sccp ccm** {*ipv4-address* | *ipv6-address* | *dns*} **identifier** *identifier-number* [**port** *port-number*] **version** *version-number*	Adds a Cisco UCM server to the list of available servers and sets the following parameters: • *ipv4-address* --IP version 4 address of the Cisco UCM server.

	Command or Action	Purpose
	Example: `Router(config)# sccp ccm 10.1.1.1 identifier 1 version 7.0+`	• *ipv6-address* --IP version 6 address of the Cisco UCM server. • *dns* --DNS name. • **identifier** --Specifies the number that identifies the Cisco UCM server. Range is 1 to 65535. • **port** *port-number* (Optional)--Specifies the TCP port number. Range is 1025 to 65535. Default is 2000. • **version** *version-number* --Cisco UCM version. Valid versions are 3.0, 3.1, 3.2, 3.3, 4.0, 4.1, 5.0.1, 6.0, and 7.0+. There is no default value.
Step 5	**sccp** **Example:** `Router(config)# sccp`	Enables the Skinny Client Control Protocol (SCCP) and its related applications (transcoding and conferencing).
Step 6	**sccp ccm group** *group-number* **Example:** `Router(config)# sccp ccm group 10`	Creates a Cisco UCM group and enters SCCP Cisco UCM configuration mode. • *group-number* --Identifies the Cisco UCM group. Range is 1 to 50.
Step 7	**associate ccm** *identifier-number* **priority** *number* **Example:** `Router(config-sccp-ccm)# associate ccm 10 priority 3`	Associates a Cisco UCM with a Cisco UCM group and establishes its priority within the group: • *identifier-number* --Identifies the Cisco UCM. Range is 1 to 65535. There is no default value. • **priority** *number* --Priority of the Cisco UCM within the Cisco UCM group. Range is 1 to 4. There is no default value. The highest priority is 1.
Step 8	**associate profile** *profile-identifier* **register** *device-name* **Example:** `Router(config-sccp-ccm)# associate profile 1 register MTP0011`	Associates a DSP farm profile with a Cisco UCM group: • *profile-identifier* --Identifies the DSP farm profile. Range is 1 to 65535. There is no default value. • **register** *device-name* --Device name in Cisco UCM. A maximum of 15 characters can be entered for the device name.
Step 9	**dspfarm profile** *profile-identifier* {**conference** | **mtp** | **transcode**} [**security**] **Example:** `Router(config-sccp-ccm)# dspfarm profile 1 mtp`	Enters DSP farm profile configuration mode and defines a profile for DSP farm services: • *profile-identifier* --Number that uniquely identifies a profile. Range is 1 to 65535. There is no default. • **conference** --Enables a profile for conferencing. • **mtp** --Enables a profile for MTP.

	Command or Action	Purpose
		• **transcode** --Enables a profile for transcoding. • **security** (Optional)-- Enables a profile for secure DSP farm services.
Step 10	**maximum sessions** {**hardware** | **software**} *number* **Example:** `Router(config-dspfarm-profile)# maximum sessions software 10`	Specifies the maximum number of sessions that are supported by the profile. • **hardware** --Number of sessions that MTP hardware resources can support. • **software** --Number of sessions that MTP software resources can support. • *number* --Number of sessions that are supported by the profile. Range is 0 to x. Default is 0. The x value is determined at run time depending on the number of resources available with the resource provider.
Step 11	**associate application sccp** **Example:** `Router(config-dspfarm-profile)# associate application sccp`	Associates SCCP to the DSP farm profile.
Step 12	**no shutdown** **Example:** `Router(config-dspfarm-profile)# no shutdown`	Changes the status of the interface to the UP state.

Examples

The following example shows a sample configuration for the Support for Software Media Termination Point feature:

```
sccp local GigabitEthernet0/0/1
sccp ccm 10.13.40.148 identifier 1 version 6.0
sccp
!
sccp ccm group 1
 bind interface GigabitEthernet0/0/1
 associate ccm 1 priority 1
 associate profile 6 register RR_RLS6
!
 dspfarm profile 6 mtp
 codec g711ulaw
 maximum sessions software 100
 associate application SCCP
!
!
```

```
gateway
media-inactivity-criteria all
timer receive-rtp 400
```

Troubleshooting Tips

To verify and troubleshoot this feature, use the following **show** commands:

- To verify information about SCCP, use the **show sccp** command:

```
Router# show sccp

SCCP Admin State: UP
Gateway IP Address: 10.13.40.157, Port Number: 2000
IP Precedence: 5
User Masked Codec list: None
Call Manager: 10.13.40.148, Port Number: 2000
                Priority: N/A, Version: 6.0, Identifier: 1
                Trustpoint: N/A
```

- To verify information about the DSPfarm profile, use the **show dspfarm profile** command:

```
Router# show dspfarm profile 6

Dspfarm Profile Configuration
 Profile ID = 6, Service = MTP, Resource ID = 1
 Profile Description :
 Profile Service Mode : Non Secure
 Profile Admin State : UP
 Profile Operation State : ACTIVE
 Application : SCCP   Status : ASSOCIATED
 Resource Provider : NONE   Status : NONE
 Number of Resource Configured : 100
 Number of Resource Available : 100
 Hardware Configured Resources : 0
 Hardware Available Resources : 0
 Software Resources : 100
 Codec Configuration
 Codec : g711ulaw, Maximum Packetization Period : 30
```

- To display statistics for the SCCP connections, use the **show sccp connections** command:

```
Router# show sccp connections

sess_id    conn_id    stype mode     codec   ripaddr         rport sport
16808048   16789079     mtp   sendrecv g711u 10.13.40.20     17510 7242
16808048   16789078     mtp   sendrecv g711u 10.13.40.157     6900 18050
```

- To display information about RTP connections, use the **show rtpspi call** command:

```
Router# show rtpspi call
RTP Service Provider info:
No. CallId dstCallId Mode       LocalRTP RmtRTP LocalIP    RemoteIP   SRTP
 22     19        Snd-Rcv   7242     17510  0x90D080F  0x90D0814    0
 19     22        Snd-Rcv   18050    6900   0x90D080F  0x90D080F    0
```

- To display information about VoIP RTP connections, use the **show voip rtp connections** command:

```
Router# show voip rtp connections
VoIP RTP Port Usage Information
Max Ports Available: 30000, Ports Reserved: 100, Ports in Use: 102
```

```
Port range not configured, Min: 5500, Max: 65499
VoIP RTP active connections :
No. CallId     dstCallId  LocalRTP  RmtRTP   LocalIP         RemoteIP
1     114        117        19822    24556  10.13.40.157     10.13.40.157
2     115        116        24556    19822  10.13.40.157     10.13.40.157
3     116        115        19176    52625  10.13.40.157     10.13.40.20
4     117        114        16526    52624  10.13.40.157     10.13.40.20
```

- Additional, more specific, **show** commands that can be used include the following:
  - **show sccp connection callid**
  - **show sccp connection connid**
  - **show sccp connection sessionid**
  - **show rtpspi call callid**
  - **show rtpspi stat callid**
  - **show voip rtp connection callid**
  - **show voip rtp connection type**
- To isolate specific problems, use the **debug sccp** command:
  - **debug sccp** [**all** | **config** | **errors** | **events** | **keepalive** | **messages** | **packets** | **parser** | **tls**]

Feature Information for Support for Software Media Termination Point

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.
Feature Historey Table for the ASR

Table 32: Feature Information for Support for Software Media Termination Point

Feature Name	Releases	Feature Information
Support for Software Media Termination Point	Cisco IOS XE Release 2.6 S	Software Media Termination Point (MTP) provides the capability for Cisco Unified Communications Manager (Cisco UCM) to interact with a voice gateway via Skinny Client Control Protocol (SCCP) commands. These commands allow the Cisco UCM to establish an MTP for call signaling.

CHAPTER 31

Cisco Unified Communication Trusted Firewall Control

Cisco Unified Communications Trusted Firewall Control pushes intelligent services onto the network through a Trusted Relay Point (TRP) firewall. Firewall traversal is accomplished using Session Traversal Utilities for NAT(STUN) on a TRP collocated with a Cisco Unified Communications Manager Express (Cisco Unified CME) or a Cisco Unified Border Element.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites

Cisco Unified Border Element

- Cisco IOS Release 12.4(22)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.3S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Configuring Cisco Unified Communication Trusted Firewall Control

To enable this feature, see the "Cisco Unified Communications Trusted Firewall Control" feature guide.

Detailed command information for the **stun**, **stun flowdata agent-id**, **stun flowdata keepalive**, **stun flowdata shared-secret**, **stun usage firewall-traversal flowdata**, **voice-class stun-usage**commands is located in the *Cisco IOS Voice Command Reference*.

Feature Information for Cisco Unified Communication Trusted Firewall Control

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 33: Feature Information for Cisco Unified Communication Trusted Firewall Control

Feature Name	Releases	Feature Information
Cisco Unified Communications Trusted Firewall Control	12.4(22)T	Cisco Unified Communications Trusted Firewall Control pushes intelligent services into the network through Trust Relay Point (TRP). The following commands were introduced or modified: **stun**, **stun flowdata agent-id**, **stun flowdata keepalive**, **stun flowdata shared-secret**, **stun usage firewall-traversal flowdata**, **voice-class stun-usage**.

Table 34: Feature Information for Cisco Unified Communication Trusted Firewall Control

Feature Name	Releases	Feature Information
Cisco Unified Communications Trusted Firewall Control	Cisco IOS XE Release 3.3S	Cisco Unified Communications Trusted Firewall Control pushes intelligent services into the network through Trust Relay Point (TRP). The following commands were introduced or modified: **stun**, **stun flowdata agent-id**, **stun flowdata keepalive**, **stun flowdata shared-secret**, **stun usage firewall-traversal flowdata**, **voice-class stun-usage**.

CHAPTER 32

Cisco Unified Communication Trusted Firewall Control-Version II

Cisco Unified Communications Trusted Firewall Control pushes intelligent services onto the network through a Trusted Relay Point (TRP) firewall. TRP is a Cisco IOS service feature, which is similar to the Resource Reservation Protocol (RSVP) agent. Firewall traversal is accomplished using Session Traversal Utilities for NAT (STUN) on a TRP colocated with a Cisco Unified Communications Manager Express (Cisco Unified CME), Cisco Unified Border Element, and Media Termination Points (MTP).

This release introduces the following features:

- Noncolocated firewall for UC SIP trunks
- Support Firewall traversal for Cisco Unified Border Element call flows in which the media flow through the Media Termination Points such as MTP, Transcoder, or Conference bridge with Trust Relay Point (TRP) enabled.
- Firewall traversal for additional Cisco Unified Border Element call flows using STUN.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Cisco Unified Communication Trusted Firewall Control-Version II

Cisco Unified Border Element

- Cisco IOS Release 15.0(1)T or a later release must be installed and running on your Cisco Unified Border Element.

Cisco Unified Border Element (Enterprise)

- Cisco IOS XE Release 3.3S or a later release must be installed and running on your Cisco ASR 1000 Series Router.

Configuring Cisco Unified Communication Trusted Firewall Control-Version II

To enable this feature, see the "Cisco Unified Communications Trusted Firewall Control-Version II" feature guide.

Detailed command information for the **stun flowdata catlife** command is located in the *Cisco IOS Voice Command Reference*.

Feature Information for Cisco Unified Communication Trusted Firewall Control-Version II

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 35: Feature Information for Cisco Unified Communication Trusted Firewall Control-Version II

Feature Name	Releases	Feature Information
Cisco Unified Communication Trusted Firewall Control-Version II	15.0(1)T	Cisco Unified Communications Trusted Firewall Control pushes intelligent services into the network through Trust Relay Point (TRP). The following command was introduced: **stun flowdata catlife**.

Table 36: Feature Information for Cisco Unified Communication Trusted Firewall Control-Version II

Feature Name	Releases	Feature Information
Cisco Unified Communication Trusted Firewall Control-Version II	Cisco IOS XE Release 3.3S	Cisco Unified Communications Trusted Firewall Control pushes intelligent services into the network through Trust Relay Point (TRP). The following command was introduced: **stun flowdata catlife**.

CHAPTER 33

Domain-Based Routing Support on the Cisco UBE

First Published: June 15, 2011

Last Updated: July 22, 2011

The Domain-based routing feature provides support for matching an outbound dial peer based on the domain name or IP address provided in the request URI of the incoming SIP message or an inbound dial peer.

Domain-based routing enables for calls to be routed on the outbound dialpeer based on the domain name or IP address provided in the request Uniform Resource Identifier (URI) of incoming Session IP message.

Feature Information for Domain-Based Routing Support on the Cisco UBE

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Table 37: Feature Information for Domain-Based Routing Support on the Cisco UBE

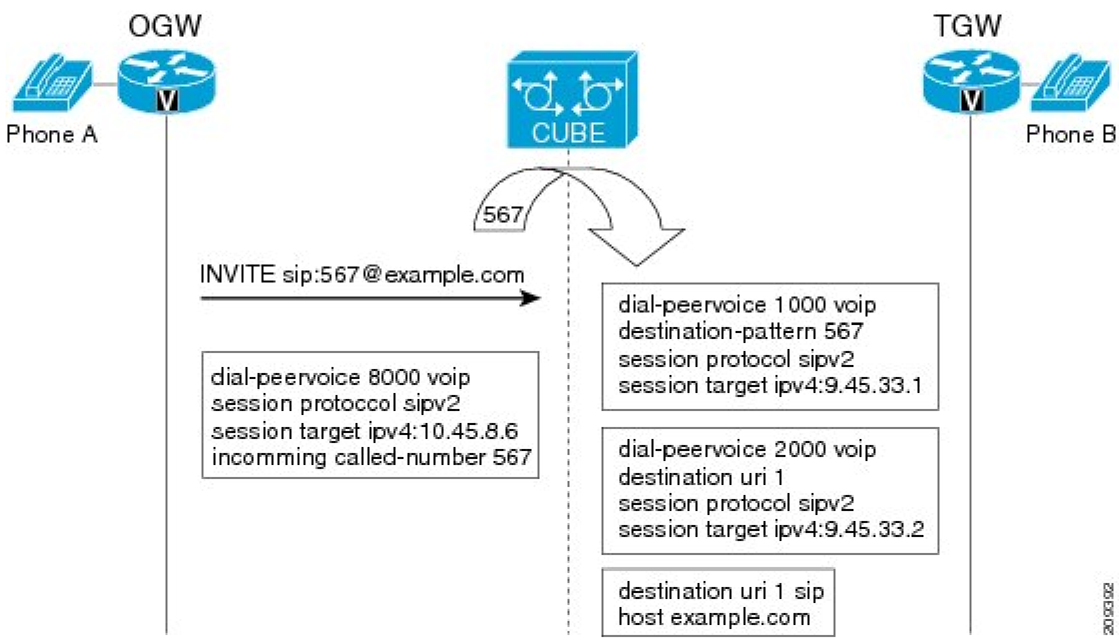
Feature Name	Releases	Feature Information
Domain Based Routing Support on the Cisco UBE	15.2(1)T	The domain-based routing enables for calls to be routed on the outbound dial peer based on the domain name or IP address provided in the request URI (Uniform Resource Identifier) of incoming SIP message. The following commands were introduced or modified: **call-route**, **voice-class sip call-route**.
Domain Based Routing Support on the Cisco UBE	Cisco IOS XE Release 3.8S	The domain-based routing enables for calls to be routed on the outbound dial peer based on the domain name or IP address provided in the request URI (Uniform Resource Identifier) of incoming SIP message. The following commands were introduced or modified: **call-route**, **voice-class sip call-route**.

Restrictions for Domain-Based Routing Support on the Cisco UBE

Domain-based routing support is available only for SIP-SIP call flows.

Information About Domain-Based Routing Support on the Cisco UBE

When a dial peer has an application configured as a session application, then only the user parameter of the request URI is used and is sent from the inbound SIP SPI to the application. The session application performs a match on an outbound dial peer based on the user parameter of the request URI sent from the inbound dial peer. In the figure below, 567 is the user portion of the request-URI that is passed from the inbound dial peer to the application and the matching outbound dial-peer found is 1000.



With the introduction of the domain-based routing feature, all parameters including the domain name of the request URI will be sent to the application and the outbound dial peer can be matched with any parameter. In Figure 1, when the domain name example.com is used to match an outbound dial peer the resulting dial peer is 2000. The **call route url** command is used for configuring domain-based routing.

How to Configure Domain-Based Routing Support on the Cisco UBE

Configuring Domain-Based Routing at Global Level

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **call-route url**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Enters voice service configuration mode.
Step 4	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters voice service SIP configuration mode.
Step 5	**call-route url** **Example:** `Device(conf-serv-sip)# call-route url` **Example:**	Routes calls based on the URL.
Step 6	**exit** **Example:** `Device(conf-serv-sip)# exit`	Exits the current mode.

Configuring Domain-Based Routing at Dial Peer Level

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dial-peer voice** *dial-peer tag* **voip**
4. **voice-class sip call-route url**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dial-peer voice** *dial-peer tag* **voip** **Example:** `Device(config)#` **dial-peer voice** *2* **voip**	Enter dial peer voice configuration mode.
Step 4	**voice-class sip call-route url** **Example:** `Device(config-dial-peer)#` **Example:** Routes calls based on the URL	
Step 5	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits the current mode.

Verifying and Troubleshooting Domain-Based Routing Support on the Cisco UBE

SUMMARY STEPS

1. **enable**
2. **debug ccsip all**
3. **debug voip dialpeer inout**

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode.

Example:
```
Device> enable
```

Step 2 **debug ccsip all**

Enables all SIP-related debugging.

Example:
```
Device# debug ccsip all
Received:
INVITE sip:5555555555@[2208:1:1:1:1:1:1:1118]:5060 SIP/2.0
Via: SIP/2.0/UDP [2208:1:1:1:1:1:1:1115]:5060;branch=z9hG4bK83AE3
Remote-Party-ID: <sip:2222222222@[2208:1:1:1:1:1:1:1115]>;party=calling;screen=no;privacy=off
From: <sip:2222222222@[2208:1:1:1:1:1:1:1115]>;tag=627460F0-1259
To: <sip:5555555555@[2208:1:1:1:1:1:1:1118]>
Date: Tue, 01 Mar 2011 08:49:48 GMT
Call-ID: B30FCDEB-431711E0-8EDECB51-E9F6B1F1@2208:1:1:1:1:1:1:1115
Supported: 100rel,timer,resource-priority,replaces
Require: sdp-anat
Min-SE:  1800
Cisco-Guid: 2948477781-1125585376-2396638033-3925258737
User-Agent: Cisco-SIPGateway/IOS-15.1(3.14.2)PIA16
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, UPDATE, REFER, SUBSCRIBE, NOTIFY, INFO, REGISTER
CSeq: 101 INVITE
Max-Forwards: 70
Timestamp: 1298969388
Contact: <sip:2222222222@[2208:1:1:1:1:1:1:1115]:5060>
Expires: 180
Allow-Events: telephone-event
Content-Type: application/sdp
Content-Disposition: session;handling=required
Content-Length: 495
v=0
o=CiscoSystemsSIP-GW-UserAgent 7880 7375 IN IP6 2208:1:1:1:1:1:1:1115
s=SIP Call
c=IN IP6 2208:1:1:1:1:1:1:1115
t=0 0
a=group:ANAT 1 2
m=audio 17836 RTP/AVP 0 101 19
c=IN IP6 2208:1:1:1:1:1:1:1115
a=mid:1
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:19 CN/8000
a=ptime:20
m=audio 18938 RTP/AVP 0 101 19
c=IN IP4 9.45.36.111
a=mid:2
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:19 CN/8000
a=ptime:20
“Received:
INVITE sip:2222222222@[2208:1:1:1:1:1:1:1117]:5060 SIP/2.0
Via: SIP/2.0/UDP [2208:1:1:1:1:1:1:1116]:5060;branch=z9hG4bK38ACE
Remote-Party-ID: <sip:5555555555@[2208:1:1:1:1:1:1:1116]>;party=calling;screen=no;privacy=off
From: <sip:5555555555@[2208:1:1:1:1:1:1:1116]>;tag=4FE8C9C-1630
To: <sip:2222222222@[2208:1:1:1:1:1:1:1117]>;tag=1001045C-992
Date: Thu, 10 Feb 2011 12:15:08 GMT
Call-ID: 5DEDB77E-ADC11208-808BE770-8FCACF34@2208:1:1:1:1:1:1:1117
Supported: 100rel,timer,resource-priority,replaces,sdp-anat
Min-SE:  1800
Cisco-Guid: 1432849350-0876876256-2424621905-3925258737
```

```
User-Agent: Cisco-SIPGateway/IOS-15.1(3.14.2)PIA16
Allow: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, UPDATE, REFER, SUBSCRIBE, NOTIFY, INFO, REGISTER
CSeq: 101 INVITE
Max-Forwards: 70
Timestamp: 1297340108
Contact: <sip:5555555555@[2208:1:1:1:1:1:1:1116]:5060>
Expires: 180
Allow-Events: telephone-event
Content-Type: application/sdp
Content-Length: 424
v=0
o=CiscoSystemsSIP-GW-UserAgent 8002 7261 IN IP6 2208:1:1:1:1:1:1:1116
s=SIP Call
c=IN IP6 2208:1:1:1:1:1:1:1116
t=0 0
m=image 17278 udptl t38
c=IN IP6 2208:1:1:1:1:1:1:1116
a=T38FaxVersion:0
a=T38MaxBitRate:14400
a=T38FaxFillBitRemoval:0
a=T38FaxTranscodingMMR:0
a=T38FaxTranscodingJBIG:0
a=T38FaxRateManagement:transferredTCF
a=T38FaxMaxBuffer:200
a=T38FaxMaxDatagram:320
a=T38FaxUdpEC:t38UDPRedundancy"
```

Step 3 **debug voip dialpeer inout**

The **debug ccsip all** and **debug voip dialpeer inout** commands can be entered in any order and any of the commands can be used for debugging depending on the requirement.

Example:

```
Displays information about the voice dial peers
Device# debug voip dialpeer inout

voip dialpeer inout debugging is on
```

The following event shows the calling and called numbers:

Example:

```
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Calling Number=4085550111, Called Number=3600, Voice-Interface=0x0,
   Timeout=TRUE, Peer Encap Type=ENCAP_VOIP, Peer Search Type=PEER_TYPE_VOICE,
   Peer Info Type=DIALPEER_INFO_SPEECH
```

The following event shows the incoming dial peer:

Example:

```
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Result=Success(0) after DP_MATCH_INCOMING_DNIS; Incoming Dial-peer=100
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Calling Number=4085550111, Called Number=3600, Voice-Interface=0x0,
   Timeout=TRUE, Peer Encap Type=ENCAP_VOIP, Peer Search Type=PEER_TYPE_VOICE,
   Peer Info Type=DIALPEER_INFO_SPEECH
*May  1 19:32:11.731: //-1/6372E2598012/DPM/dpAssociateIncomingPeerCore:
   Result=Success(0) after DP_MATCH_INCOMING_DNIS; Incoming Dial-peer=100
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Calling Number=, Called Number=3600, Peer Info Type=DIALPEER_INFO_SPEECH
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Match Rule=DP_MATCH_DEST; Called Number=3600
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersCore:
   Result=Success(0) after DP_MATCH_DEST
```

```
*May  1 19:32:11.735: //-1/6372E2598012/DPM/dpMatchPeersMoreArg:
   Result=SUCCESS(0)
```

The following event shows the matched dial peers in the order of priority:

Example:

```
List of Matched Outgoing Dial-peer(s):
     1: Dial-peer Tag=3600
     2: Dial-peer Tag=36
```

Configuration Examples for Domain-Based Routing Support on the Cisco UBE

Example Configuring Domain-Based Routing Support on the Cisco UBE

The following example shows how to enable domain-based routing support on the Cisco UBE:

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# call-route url
Device(conf-serv-sip)# exit
Device(config)# dial-peer voice 2 voip
Device(config-dial-peer)# voice-class sip call-route url
Device(config-dial-peer)# exit
```

CHAPTER 34

URI-Based Dialing Enhancements

The URI-Based Dialing Enhancements feature describes the enhancements made to Uniform Resource Identifier (URI)-based dialing on Cisco Unified Border Element (Cisco UBE) for Session Initiation Protocol (SIP) calls. The URI-Based Dialing Enhancements feature includes support for call routing on Cisco UBE when the user part of the incoming Request-URI is non-E164 (for example, INVITE sip:user@abc.com).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About URI-Based Dialing Enhancements

Cisco Unified Communications Manager (CUCM) supports dialing using directory Uniform Resource Identifiers (URIs) for call addressing. Directory URIs follow the username@host format where the host portion is an IPv4 address or a fully qualified domain name. A directory URI is a string of characters that can be used to identify a directory number. If that directory number is assigned to a phone, CUCM can route calls to that phone using the directory URI. URI dialing is available for Session Initiation Protocol (SIP) and Signaling Connection Control Part (SCCP) endpoints that support directory URIs.

The primary use of URI-based dialing is peer-to-peer calling between enterprises using complete URI addresses (that is, 'username@host'). The host part of the URI identifies the destination to which the call should be routed. In earlier Cisco Unified Border Element (Cisco UBE) URI routing, the URI was replaced in the SIP header with the destination server IP address. Then routing of calls was based on the following restrictions:

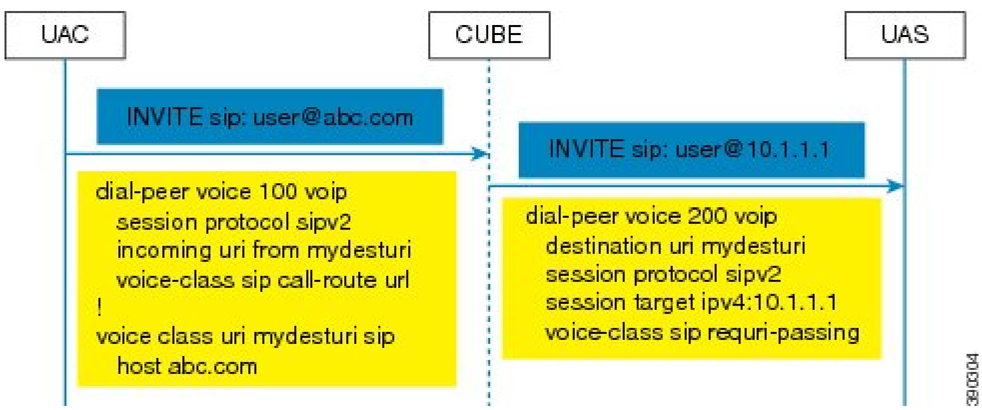
- The user part of the incoming Request-URI must be an E164 number.
- The outgoing Request-URI is always set to the session target information of the outbound dial peer.

The URI-Based Dialing Enhancements feature extends support for Cisco UBE URI-based routing of calls. With these enhancements Cisco UBE supports:

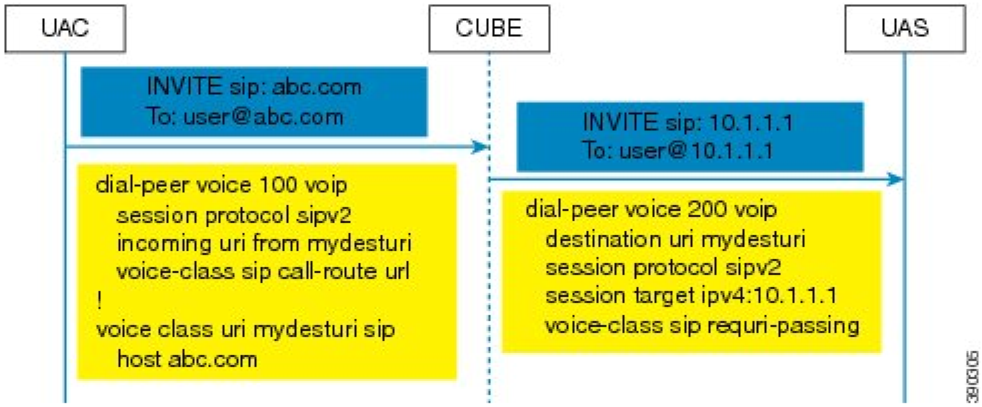
- URI-based routing when the user part of the incoming Request-URI is non-E164 (for example, INVITE sip:user@abc.com).
- URI-based routing when the user part is not present. The user part is an optional parameter in the URI (for example, INVITE sip:abc.com).
- Copying the outgoing Request-URI and To header from the inbound Request-URI and To header respectively.
- Deriving (optionally) the session target for the outbound dial peer from the host portion of the inbound URI.
- URI-based routing for 302, Refer, and Bye Also scenarios.
- Call hunting where the subsequent dial peer is selected based on URI.
- Pass through of 302, with the host part of Contact: unmodified.

Call Flows for URI-Based Dialing Enhancements

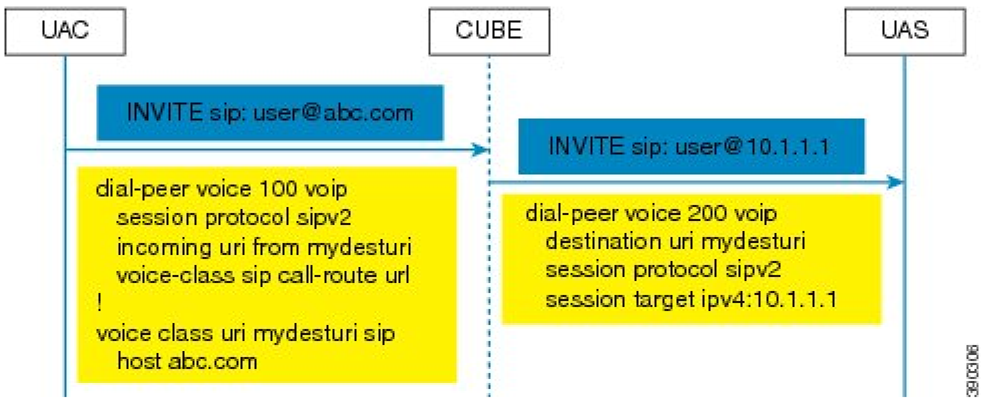
Case1: URI dialing with username being E164 or non-E164 number and Request-URI host copied from the inbound leg.



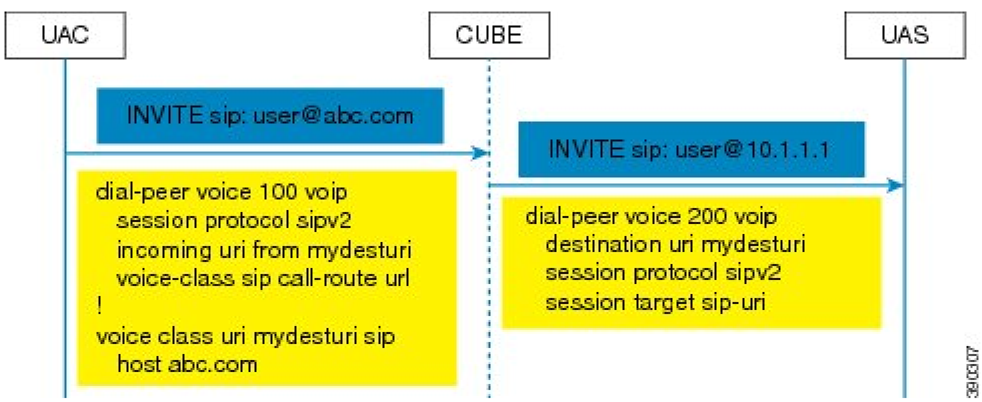
Case 2: Incoming Request-URI does not contain user part. The To: header information is also copied from the peer leg when the **requri-passing** command is enabled.



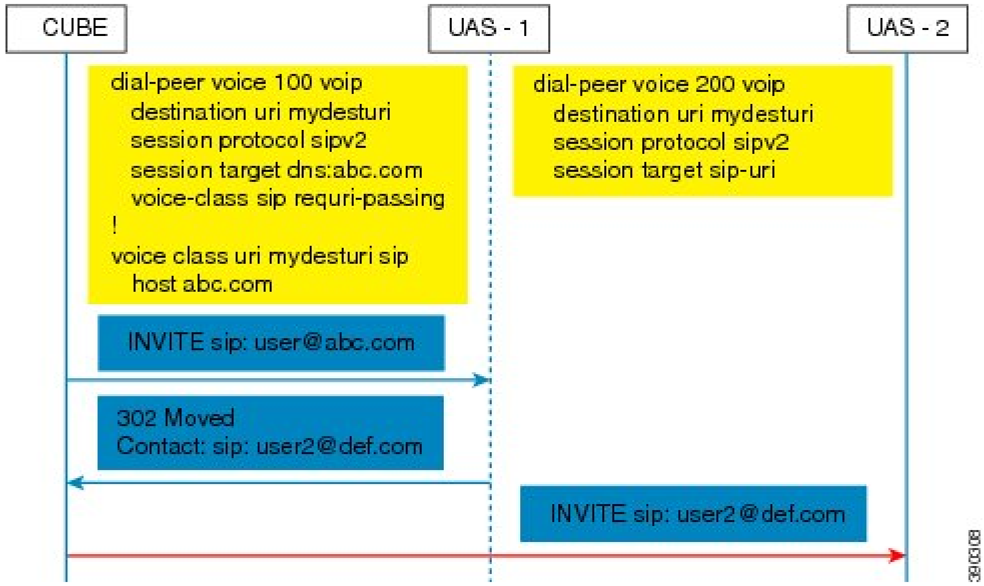
Case 3: The old behavior of setting the outbound Request-URI to session target is retained when the **requri-passing** command is not enabled.



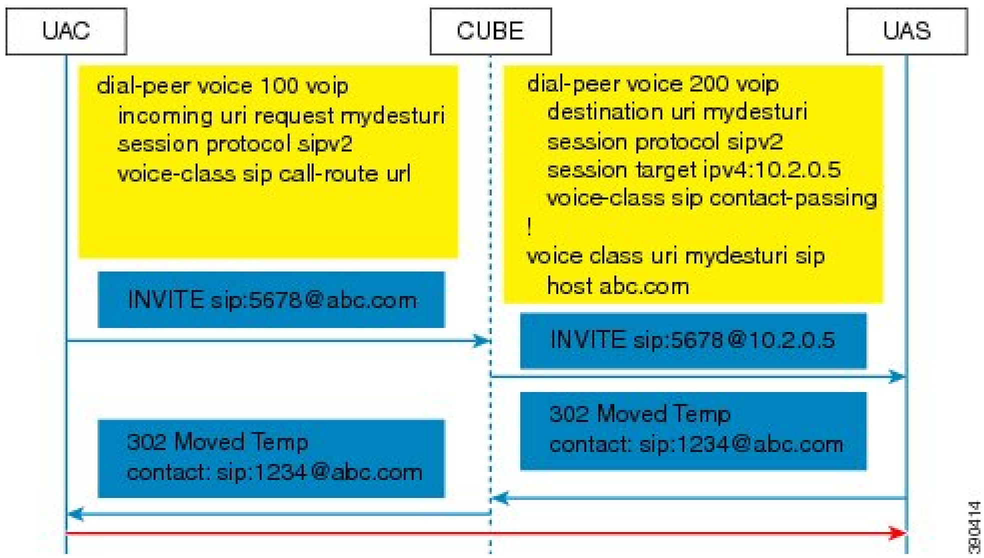
Case 4: The session target derived from the host part of the URI. The outgoing INVITE is sent to resolved IP address of the host part of the URI.



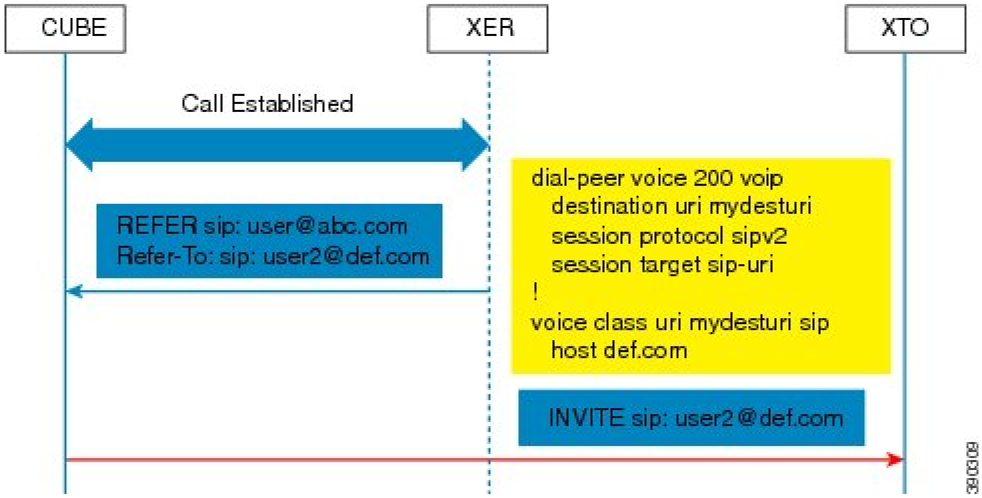
Case 5: Pass through of contact URI to request URI.



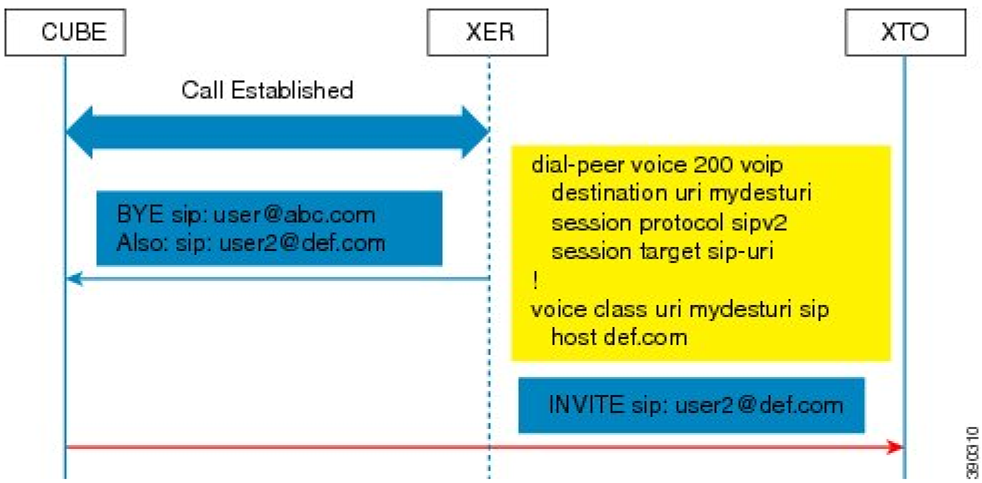
Case 6: In 302 pass-through, contact header can be passed through from one leg to another by using the **contact-passing** command.



Case 7: Pass through of refer-to URI to request URI.



Case 8: URI routing based on BYE Also header.



How to Configure URI-Based Dialing Enhancements

Configuring Pass Through of SIP URI Headers

Perform these to configure the pass through of the host part of the Request-Uniform Resource Identifier (URI) and To Session Initiation Protocol (SIP) headers. By default, Cisco Unified Border Element (Cisco UBE) sets the host part of the URI to the value configured under the session target of the outbound dial peer. For more information, see Case 1 in the "Call Flows for URI-based Dialing Enhancements" section.

Configuring Pass Though of Request URI and To Header URI (Global Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **requri-passing**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Specifies VoIP encapsulation and enters voice service configuration mode.
Step 4	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters the Session Initiation Protocol (SIP) configuration mode.
Step 5	**requri-passing** **Example:** `Router(conf-serv-sip)# requri-passing`	Enables pass through of the host part of the Request-URI and To SIP headers. By default, Cisco UBE sets the host part of the URI to the value configured under the session target of the outbound dial peer.
Step 6	**end** **Example:** `Router(conf-serv-sip)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuring Pass Though of Request URI and To Header URI (Dial Peer Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class uri** *tag* **sip**
4. **host** *hostname-pattern*
5. **exit**
6. **dial-peer voice** *tag* **voip**
7. **session protocol sipv2**
8. **destination uri** *tag*
9. **session target ipv4:***ip-address*
10. **voice-class sip requri-passing** [**system**]
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *tag* **sip** **Example:** `Device(config)# voice class uri mydesturi sip`	Creates a voice class for matching dial peers to a Session Initiation Protocol (SIP) and enters voice URI class configuration mode.
Step 4	**host** *hostname-pattern* **Example:** `Device(config-voice-uri-class)# host example.com`	Matches a call based on the host field in a SIP Uniform Resource Identifier (URI).
Step 5	**exit** **Example:** `Device(config-voice-uri-class)# exit`	Exits voice URI class configuration mode.
Step 6	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 22 voip`	Defines a VoIP dial peer and enters dial peer configuration mode.

	Command or Action	Purpose
Step 7	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Specifies a session protocol for calls between local and remote routers using the Internet Engineering Task Force (IETF) SIP.
Step 8	**destination uri** *tag* **Example:** `Device(config)# destination uri mydesturi`	Specifies the voice class used to match a dial peer to the destination URI of an outgoing call.
Step 9	**session target ipv4:***ip-address* **Example:** `Device(config-dial-peer)# session target ipv4:10.1.1.2`	Designates a network-specific address to receive calls from a VoIP.
Step 10	**voice-class sip requri-passing** [**system**] **Example:** `Device(config-dial-peer)# voice-class sip requri-passing system`	Enables the pass through of SIP URI headers.
Step 11	**end** **Example:** `Device(config-dial-peer)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuring Pass Through of 302 Contact Header

Configuring Pass Through of 302 Contact Header (Global Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice service voip**
4. **sip**
5. **contact-passing**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice service voip** **Example:** `Device(config)# voice service voip`	Specifies VoIP encapsulation and enters voice service configuration mode.
Step 4	**sip** **Example:** `Device(conf-voi-serv)# sip`	Enters voice service SIP configuration mode.
Step 5	**contact-passing** **Example:** `Router(conf-serv-sip)# contact-passing`	Enables pass through of the contact header from one leg to the other leg in 302 pass through scenario.
Step 6	**end** **Example:** `Router(conf-serv-sip)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuring Pass Through of 302 Contact Header (Dial Peer Level)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class uri** *destination-tag* **sip**
4. **user-id** *id-tag*
5. **exit**
6. **voice service voip**
7. **allow-connections sip to sip**
8. **dial-peer voice** *tag* **voip**
9. **session protocol sipv2**
10. **destination uri** *destination-tag*
11. **voice-class sip contact-passing**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *destination-tag* **sip** **Example:** `Device(config)# voice class uri mydesturi sip`	Creates a voice class for matching dial peers to a Session Initiation Protocol (SIP) and enters voice URI class configuration mode.
Step 4	**user-id** *id-tag* **Example:** `Device(config-voice-uri-class)# user-id 5678`	Matches a call based on the User ID portion of the Uniform Resource Identifier (URI).
Step 5	**exit** **Example:** `Device(config-voice-uri-class)# exit`	Exits voice URI class configuration mode.

	Command or Action	Purpose
Step 6	**voice service voip** **Example:** `Device(config)# voice service voip`	Specifies Voice over IP (VoIP) as the voice encapsulation type and enters voice service configuration mode.
Step 7	**allow-connections sip to sip** **Example:** `Device(conf-voi-serv)# allow-connections sip to sip`	Allows connections between SIP endpoints in a VoIP network.
Step 8	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 200 voip`	Defines a VoIP dial peer and enters dial peer configuration mode.
Step 9	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Specifies a session protocol for calls between local and remote routers using the Internet Engineering Task Force (IETF) SIP.
Step 10	**destination uri** *destination-tag* **Example:** `Device(config-dial-peer)# destination uri mydesturi`	Specifies the voice class used to match a dial peer to the destination URI of an outgoing call.
Step 11	**voice-class sip contact-passing** **Example:** `Device(config-dial-peer)# voice-class sip contact-passing`	Enables pass through of the contact header from one leg to the other leg in 302 pass through scenario.
Step 12	**end** **Example:** `Device(config-dial-peer)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Deriving of Session Target from URI

Perform this task to derive the session target from the host part of the Uniform Resource Identifier (URI). The outgoing INVITE is sent to the resolved IP address of the host part of the URI. For more information, see Case 4 in the "Call Flows for URI-Based Dialing Enhancements" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **voice class uri** *destination-tag* **sip**
4. **host** *hostname-pattern*
5. **exit**
6. **dial-peer voice** *tag* **voip**
7. **session protocol sipv2**
8. **destination uri** *destination-tag*
9. **session target sip-uri**
10. **exit**
11. **voice class uri** *source-tag* **sip**
12. **host** *hostname-pattern*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**voice class uri** *destination-tag* **sip** **Example:** `Device(config)# voice class uri mydesturi sip`	Creates or modifies a voice class for matching dial peers to a Session Initiation Protocol (SIP) or telephone (TEL) Uniform Resource Identifier (URI) and enters voice URI class configuration mode.
Step 4	**host** *hostname-pattern* **Example:** `Device(config-voice-uri-class)# host destination.com`	Matches a call based on the host field in a SIP URI.
Step 5	**exit** **Example:** `Device(config-voice-uri-class)# exit`	Exits voice URI class configuration mode.
Step 6	**dial-peer voice** *tag* **voip** **Example:** `Device(config)# dial-peer voice 25 voip`	Defines a VoIP dial peer and enters dial peer configuration mode.

	Command or Action	Purpose
Step 7	**session protocol sipv2** **Example:** `Device(config-dial-peer)# session protocol sipv2`	Specifies a session protocol for calls between local and remote routers using the Internet Engineering Task Force (IETF) SIP.
Step 8	**destination uri** *destination-tag* **Example:** `Device(config-dial-peer)# destination uri mydesturi`	Specifies the voice class used to match a dial peer to the destination URI of an outgoing call.
Step 9	**session target sip-uri** **Example:** `Device(config-dial-peer)# session target sip-uri`	Derives session target from incoming URI.
Step 10	**exit** **Example:** `Device(config-dial-peer)# exit`	Exits dial peer voice configuration mode.
Step 11	**voice class uri** *source-tag* **sip** **Example:** `Device(config)# voice class uri mysourceuri sip`	Creates or modifies a voice class for matching dial peers to a SIP or TEL URI and enters voice URI class configuration mode.
Step 12	**host** *hostname-pattern* **Example:** `Device(config-voice-uri-class)# host abc.com`	Matches a call based on the host field in a SIP URI.
Step 13	**end** **Example:** `Device(config-voice-uri-class)# end`	Ends the current configuration session and returns to privileged EXEC mode.

Configuration Examples for URI-Based Dialing Enhancements

Example: Configuring Pass Though of Request URI and To Header URI

Example: Configuring Pass Though of Request URI and To Header URI (Global Level)

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
```

```
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# requri-passing
Device(conf-serv-sip)# end
```

Example: Configuring Pass Though of Request URI and To Header URI (Dial Peer Level)

```
! Configuring URI voice class destination
Device(config)# voice class uri mydesturi sip
Device(config-voice-uri-class)# host xyz.com
Device(config-voice-uri-class)# exit

! Configuring outbound dial peer
Device(config)# dial-peer voice 13 voip
Device(config-dial-peer)# session protocol sipv2
Device(config-dial-peer)# destination uri mydesturi
Device(config-dial-peer)# session target ipv4:10.1.1.1
Device(config-dial-peer)# voice-class sip requri-passing system
Device(config-dial-peer)# end
```

Example: Configuring Pass Through of 302 Contact Header

Example: Configuring Pass Through of 302 Contact Header (Global Level)

```
Device> enable
Device# configure terminal
Device(config)# voice service voip
Device(conf-voi-serv)# sip
Device(conf-serv-sip)# contact-passing
Device(conf-serv-sip)# end
```

Example: Configuring Pass Through of 302 Contact Header (Dial Peer Level)

```
! Configuring URI voice class destination
Device> enable
Device# configure terminal
Device(config)# voice class uri mydesturi sip
Device(config-voice-uri-class)# user-id 5678
Device(config-voice-uri-class)# exit

! Configuring outbound dial peer
Device(config)# voice service voip
Device(conf-voi-serv)# allow-connections sip to sip
Device(conf-voi-serv)# dial-peer voice 200 voip
Device(config-dial-peer)# session protocol sipv2
Device(config-dial-peer)# destination uri mydesturi
Device(config-dial-peer)# voice-class sip contact-passing
Device(config-dial-peer)# end
```

Example: Deriving Session Target from URI

```
Device> enable
Device# configure terminal
Device(config)# voice class uri mydesturi sip
Device(config-voice-uri-class)# host destination.com
Device(config-voice-uri-class)# exit
!
Device(config)# dial-peer voice 25 voip
Device(config-dial-peer)# session protocol sipv2
Device(config-dial-peer)# destination uri mydesturi
```

```
Device(config-dial-peer)# session target sip-uri
Device(config-dial-peer)# exit
!
Device(config)# voice class uri mysourceuri sip
Device(config-voice-uri-class)# host abc.com
Device(config-voice-uri-class)# end
```

Additional References for URI-Based Dialing Enhancements

Related Documents

Related Topic	Document Title
Voice commands	Cisco IOS Voice Command Reference
Cisco IOS commands	Cisco IOS Master Command List, All Releases
SIP configuration tasks	SIP Configuration Guide, Cisco IOS Release 15M&T

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 Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for URI-Based Dialing Enhancements

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 38: Feature Information for URI-Based Dialing Enhancements

Feature Name	Releases	Feature Information
URI-Based Dialing Enhancements	15.4(1)T	The URI-Based Dialing Enhancements feature includes support for call routing on Cisco UBE when the user-part of the incoming Request-URI is non-E164 (for example, INVITE sip:user@abc.com). The following commands were introduced or modified: **contact-passing**, **requri-passing**, **session target sip-uri** and **voice-class sip requri-passing**

CHAPTER 35

Fax Detection for SIP Call and Transfer

Fax detection is the capability to detect automatically whether an incoming call is voice or fax. For calls coming from an IP trunk to the CUBE, the Fax Detection for SIP Call and Transfer feature is used to detect CNG tones (calling tones) so that the fax server can handle the actual fax transmission or redirect the fax call to a configured fax number. Once the tone is detected, the same will be reported to the session application on the incoming TDM call leg, and based on the configuration, the T.38 fax relay session is setup locally.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Fax Detection for SIP Call and Transfer

- For FAX detection to work, the **cng-fax-detect** command under DSP farm and the **detect-fax** command must be configured in the inbound dial-peer.
- Only the g711ulaw codec can be used for detecting fax CNG tone.
- The **cng-fax-detect** command can be configured up to maximum length of 256 characters.

- The phone number can be of a maximum length of 32 characters.
- SCCP-based transcoding is not supported; only LTI-based transcoder is supported.

Information About Fax Detection for SIP Call and Transfer

When a call comes in from an IP trunk to the CUBE, it loops the call to a locally present Voice XML (VXML) gateway, which establishes an auto-attended call. The CUBE monitors the incoming audio stream. The incoming call could be through a fax machine's handset and then switching to transmit a fax. When the CUBE detects a CNG tone, it can be handled in two modes based on the configuration:

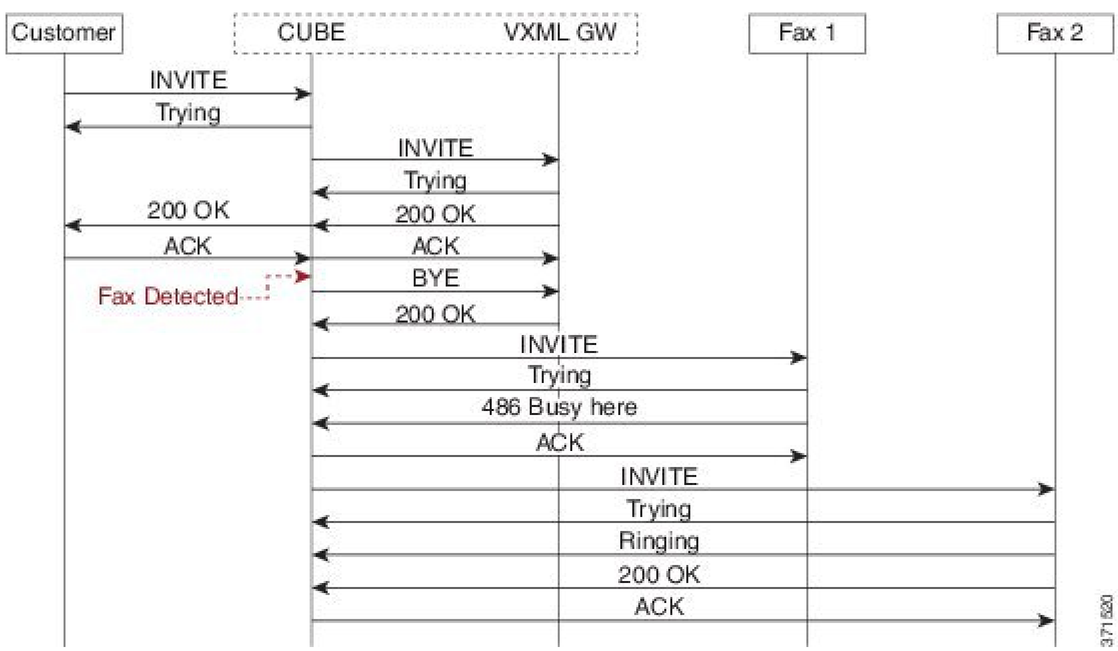
- Trigger a SIP-REFER message to a remote fax server that handles the actual fax transmission.
- Redirect the fax call to a configured fax number(s) locally.

Mode 1—Local Redirect

In the local redirect mode, the call is redirected to local fax numbers and the redirect call setup will be initiated locally.

Local redirect can be configured with multiple fax numbers. The CUBE will try to set up call to the first configured fax number till the last fax number, until the call is successfully established. After a call setup is successful, the remaining fax numbers are ignored. There is no limit to the number of fax numbers that can be configured for local redirect. The maximum length of a command can be 256 characters.

Figure 17: Local Redirect Call Flow



For each call, a digital signal processor (DSP) channel or resource is allocated to detect CNG tone. In the call flow, as the first fax machine returned an error, the CUBE tries to establish the call with the second fax machine.

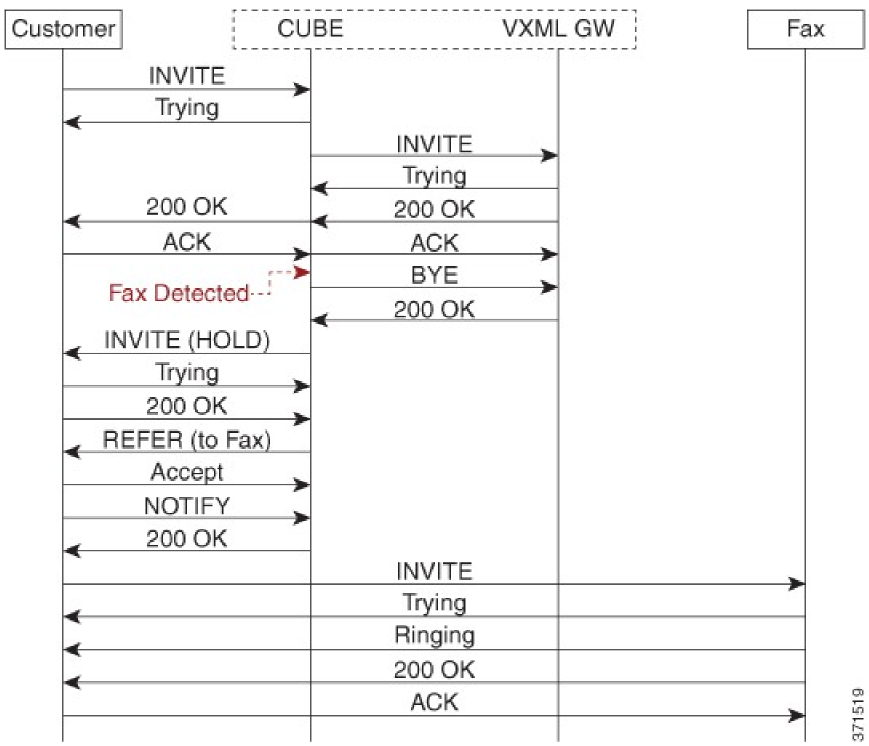
Note CUBE will send out a normal VOICE SDP INVITE to the local FAX machine after the CNG tone is detected. It does not send out a FAX negotiation SDP.

Mode 2—Refer Redirect

In this mode, redirect through SIP-REFER message is configured for remote fax numbers.

Refer redirect can be configured with only one remote FAX number. A SIP REFER message is sent back to the incoming dial-peer to redirect the call (similar to blind transfer). The refer redirect can be configured for local fax numbers also.

Figure 18: Refer Redirect Call Flow



For each call, a DSP channel or resource is allocated to detect the CNG tone. Refer will be sent with the remote FAX number in the Refer-to header.

How to Configure Fax Detection for SIP Calls

Enabling CNG Fax Detection

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **dspfarm profile** *tag* **transcode universal**
4. **cng-fax-detect**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**dspfarm profile** *tag* **transcode universal** **Example:** `Device(config)# dspfarm profile 5 transcode universal`	Enters DSP farm profile configuration mode and enables the profile for transcoding.
Step 4	**cng-fax-detect** **Example:** `Device(config-dspfarm-profile)# cng-fax-detect`	Enables CNG tone detection.
Step 5	**end** **Example:** `Device(config-dspfarm-profile)# end`	Returns to privileged EXEC mode.

Verifying Fax Detection for SIP Calls

SUMMARY STEPS

1. **enable**
2. **show call active voice compact**
3. **show dspfarm dsp active**
4. **show call active voice compact**

DETAILED STEPS

Step 1 **enable**

Example:
`Device>` **enable**

Enables privileged EXEC mode.

Step 2 **show call active voice compact**

Example:
This is a sample output of call setup when the call is connected to the VXML gateway after being looped:

`Device#` **show call active voice compact**

```
<callID>  A/O FAX T<sec> Codec     type     Peer Address       IP R<ip>:<udp>
Total call-legs: 3
    9    ANS   T4      g711ulaw    VOIP        P808808      9.42.25.145:17940
   10    ORG   T4      g711ulaw    VOIP        P309903      9.42.25.149:16396
   11    ANS   T4      g711ulaw    VOIP        P808808      9.42.25.149:16394
```

Step 3 **show dspfarm dsp active**

Example:
This is a sample output of the DSP channel reserved to detect CNG tone after the call is set up.

`Device#` **show dspfarm dsp active**

```
    SLOT   DSP VERSION  STATUS CHNL USE   TYPE    RSC_ID BRIDGE_ID PKTS_TXED PKTS_RXED
     0      2   36.1.0   UP     1   USED  xcode    1      9         228       119
     0      2   36.1.0   UP     1   USED  xcode    1      10        113       251
Total number of DSPFARM DSP channel(s) 1
```

Step 4 **show call active voice compact**

Example:
This is a sample output of FAX call setup with local redirect in the CUBE:

`Device#` **show call active voice compact**

```
<callID>  A/O FAX T<sec> Codec       type      Peer Address     IP R<ip>:<udp>
Total call-legs: 2
```

```
28     ANS       T4     g711ulaw    VOIP        P808808       9.0.0.174:14662
29     ORG       T4     g711ulaw    VOIP        P12101        9.0.0.174:14652
```

Troubleshooting Fax Detection for SIP Calls

You can enable the logs of the following **debug** or **show** commands, which are helpful in debugging fax detection for SIP calls:

- **debug ccsip verbose**
- **debug voip ccapi all**
- **debug voip dsmp all**
- **debug voip hpi all**
- **debug media resource provisioning all**
- **show call active voice compact**
- **show dspfarm dsp active**
- **show voip rtp connections**

Configuration Examples for Fax Detection for SIP Calls

Example: Configuring Local Redirect

In this example, three dial-peers are used. Each incoming dial-peer is associated with one service. If customers want to run their own VXML script, then can run the script in the initial incoming dial-peer (dial-peer 410 in the example below) The call is looped using translation profile to get one more incoming dial-peer in which VXML script is run for default IVR session (dial-peer 412 in the example below).

```
voice translation-rule 1 //Translation Rule//
 rule 1 /903309/ /309903/

voice translation-profile vxml
 translate called 1

dial-peer voice 410 voip
 description "Incoming dial-peer to GW"
 translation-profile incoming vxml
 session protocol sipv2
 incoming called-number 903309
 codec g711ulaw
 detect-fax mode local 12101 12102 12103 12104 //Local Redirect command//

dial-peer voice 411 voip
 description "Outgoing dial-peer to VXML GW"
 destination-pattern 309903
 session protocol sipv2
 session target ipv4:9.42.25.149 //CUBE IP for looping the call//
 codec g711ulaw
```

```
dial-peer voice 412 voip
 description "Incoming dial-peer to VXML GW"
 service tx_n50_1 //VXML Service//
 session protocol sipv2
 incoming called-number 309903
 codec g711ulaw
```

Example: Configuring Refer Redirect

In Refer mode, only one fax number can be configured.

```
voice translation-rule 1 //Translation Rule//
 rule 1 /903309/ /309903/

voice translation-profile vxml
 translate called 1

dial-peer voice 410 voip
 description "Incoming dial-peer to GW"
 translation-profile incoming vxml
 session protocol sipv2
 incoming called-number 903309
 codec g711ulaw
 detect-fax mode refer 332211 //Refer Redirect command//

dial-peer voice 411 voip
 description "Outgoing dial-peer to VXML GW"
 destination-pattern 309903
 session protocol sipv2
 session target ipv4:9.42.25.149 //CUBE IP for looping the call//
 codec g711ulaw

dial-peer voice 412 voip
 description "Incoming dial-peer to VXML GW"
 service tx_n50_1 //VXML Service//
 session protocol sipv2
 incoming called-number 309903
 codec g711ulaw
```

Feature Information for Fax Detection for SIP Call and Transfer

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 39: Feature Information for Fax Detection for SIP Call and Transfer

Feature Name	Releases	Feature Information
Fax Detection for SIP Call and Transfer	15.4(2)T	Fax detection is the capability to detect automatically whether an incoming call is voice or fax. For calls coming from an IP trunk to a the CUBE, the Fax Detection for SIP Call and Transfer feature is used to detect CNG tones (calling tones) so that the fax server can handle the actual fax transmission or redirect the fax call to a configured fax number. The following commands were introduced: **cng-fax-detect** and **detect-fax mode**.