



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, page 1](#)
- [Codecs configured using Preference Lists, page 2](#)
- [Prerequisites for Codec Preference Lists, page 3](#)
- [Restrictions for Codecs Preference Lists, page 3](#)
- [How to Configure Codec Preference Lists, page 4](#)
- [Troubleshooting Negotiation of an Audio Codec from a List of Codecs, page 7](#)
- [Verifying Negotiation of an Audio Codec from a List of Codecs, page 7](#)

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: 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 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. <ul style="list-style-type: none"> • 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 <i>tag</i> 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: <ul style="list-style-type: none"> • codec preference <i>value codec-type</i>[bytes payload-size fixed-bytes] • codec preference <i>value isac</i> [mode {adaptive independent}] [bit-rate value framesize { 30 60 }] [fixed]] • codec preference <i>value ilbc</i> [mode frame-size [bytes <i>payload-size</i>]] • codec preference <i>value mp4-latm</i> [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. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 6	dial-peer voice <i>number voip</i> Example: Device(config)# dial-peer voice 1 voip	Enters dial peer configuration mode for the specified VoIP dial peer.
Step 7	voice-class codec <i>tag offer-all</i> Example: Device(config-dial-peer)# voice-class codec 10	Applies the previously configured voice class and associated codecs to a dial peer. <ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • 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 cntrl 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 cntrl 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 cntrl 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 cntrl 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
```
