Connection Trunk Configuration Guide, Cisco IOS XE Release 3S

Revised: March 7, 2016,

Connection Trunk

A trunk (tie-line) is a permanent point-to-point communication line between two voice ports. The **connection trunk** command creates a permanent Voice over IP (VoIP) call between two VoIP gateways. It simulates a trunk connection through the creation of virtual trunk tie-lines between two telephony endpoints. To the connected systems, it appears as if a T1 trunk is directly connected between them.

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 Connection Trunk

- Connection Trunk mode is supported on T1/E1 channel associated signaling (CAS) interfaces. A connection trunk is not supported on T1/E1 interfaces that are using common channel signaling (CCS); for example, QSIG and PRI Q.931. A connection trunk is not supported on Foreign Exchange Office (FXO) ports configured for ground-start.
- Connection Trunk mode is a permanent connection; the VoIP call is always connected independently of the plain old telephone service (POTS) port being on-hook or off-hook. Connection Trunk has statically configured endpoints and does not require a user to dial to connect calls. It also allows supplemental call signaling, such as hookflash or point-to-point hoot-n-holler, to be passed over the IP network between the two telephony devices.
- Connection Trunk mode is supported with these voice port combinations:
  - recEive and transMit (E & M) to E & M (same type)
  - FXO to Foreign Exchange Station (FXS)
  - FXS to FXS (with no signaling)

Note These voice port combinations are permitted between analog to analog, digital to digital, and analog to digital interfaces. Also, when you are configuring FXS to FXS, signaling can not be conveyed because it would not be a transparent path. The connected devices (FXOs) would be trying to signal each other. It is possible to get this design to work if you set the voice path to always be open. Configure **signal-type ext-signal** to the VoIP dial peer, and the router will no longer wait for signaling before it opens the voice path.

- A connection trunk T1 CAS to E1 CAS mapping does not work by default. Bit-order manipulation on the gateways must be performed and may not always work, based on the PBX support of various ABCD bit signaling.

- A connection trunk allows private line, automatic ringdown-Off-Premise-Extension (PLAR-OPX) type of functionality between FXO and FXS ports. This allows remote stations (connected to FXS ports) to appear to the PBX as physically connected stations. If this remote station does not answer a call, it can be rolled-over to centralized voicemail (if it is configured on the PBX).
- A connection trunk, such as PLAR, does not require the router to collect digits from the telephony device. The permanent VoIP call is created when the router is booted and IP connectivity is established. Because of this, the existing customer dial plan does not have to be altered.
- A connection trunk can pass some telephony signaling, such as hookflash, but it does not pass proprietary PBX signaling. It is not a Transparent CCS (T-CSS) feature.
- A connection trunk, such as PLAR, is defined per voice-port. This means that the voice-port can not operate both in Connection Trunk mode and Collect Dialed-Digits mode. The only instance where this might not be completely desirable would be in a remote office that needs to also dial between local extensions without the use of a centralized PBX. This would require the path of the call to go over the VoIP network and back, as opposed to it being switched within the router. Normally, this should not be a concern.

Information About Connection Trunk

Configuration Guidelines

The connection trunk must be configured on both ends of the trunk. When you are configuring a connection trunk with analog interfaces, it must be defined per voice-port. When you are configuring a connection trunk with digital interfaces, there are several options:

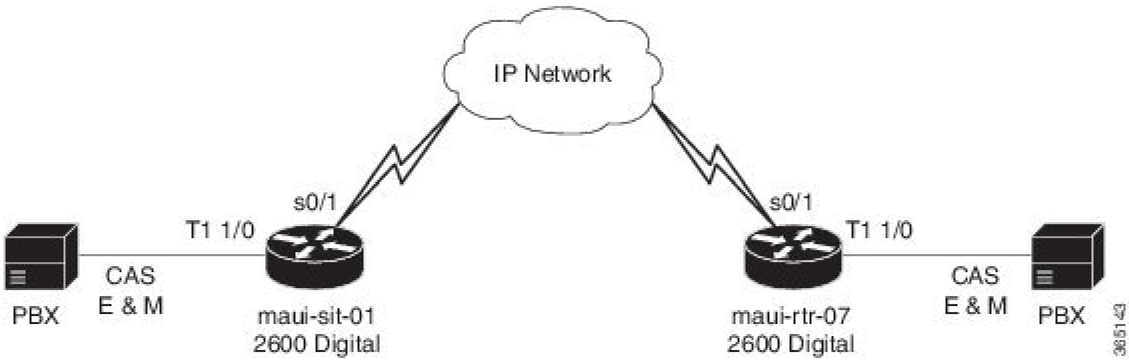
- You can define a separate **ds0-group** command for each DS0 (each timeslot), and you can use the **connection trunk** command to define each voice-port that is created. This ensures that DS0 to DS0 mapping is retained on digital trunks.
- You can define a single **ds0-group** command to handle all of the DS0s, and you can define a single **connection trunk** command on the voice-port. This reduces the amount of manual configuration that is required, but there is no guarantee of one-to-one mapping of DS0s on either end of the trunk. In addition, each time that the router reloads, the mapping can be different from the last time. Furthermore, this configuration complicates troubleshooting, because you are not able to isolate the problem to a single (or even a few) timeslots without taking down the entire trunk group. This configuration is also not recommended for T-CCS with proprietary signaling on either end of PBXs, because it would not deliver the signaling channel reliably without one-on-one mapping.
- It is recommended that one side of the connection be configured with the **answer-mode** keyword specified after the **connection trunk** string command. This makes one side of the trunk the â€œmaster side.â€ The gateway (router) with the **answer-mode** keyword is then the â€œslave side.â€ The **answer-mode** command specifies that the gateway will not attempt to initiate a trunk connection, but instead it will wait for an incoming call before it establishes the trunk. This configuration scheme minimizes the time that routers take to bring up trunks and to ensures that trunks go down when connections are lost between two gateways. Otherwise, the gateways might not attempt to re-establish the trunk when the connection is up again.

Note When you issue the **connection trunk** command, you must perform a **shutdown/no shutdown** command sequence on the voice port.

Network Diagram

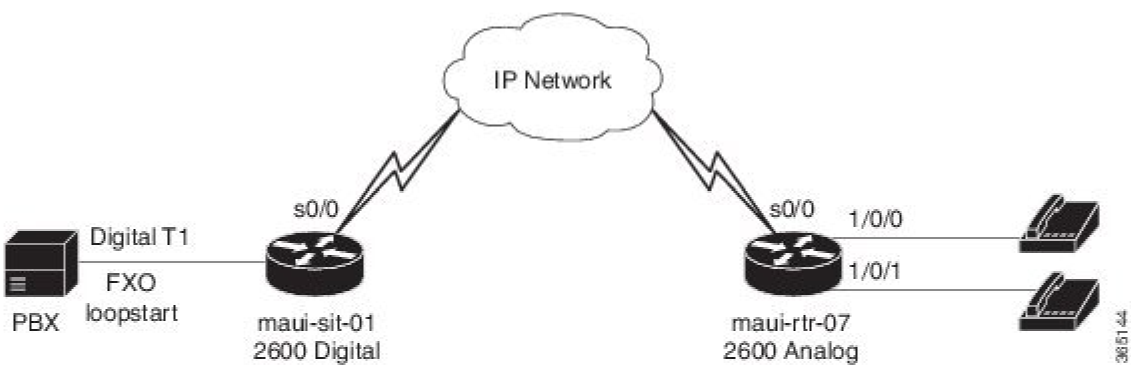
This document uses these two network setups:

Figure 1: Digital-to-digital



The previous diagram illustrates the digital-to-digital scenario, where both router sides have digital links.

Figure 2: Digital-to-analog



The previous diagram illustrates the digital-to-analog scenario, with digital on one end and analog on the other end.

Configuring Connection Trunk

Different trunk-conditioning signaling attributes may be required to match the characteristics of the different PBXs to which the router connects. For this reason, trunk-conditioning attributes are configured by creating a voice class for each set of attributes required. The trunk-conditioning attributes are configured for the voice class and the voice class is assigned to one or more dial peers.

A voice class must be configured and assigned to at least one dial peer before the trunk conditioning signaling attributes take effect.

Note This configuration supports the North America CAS Protocol and applies only to Cisco private-line or FRF.11 trunk calls. It does not apply to digital T1/E1 trunks using CCS.

To create a voice class and define the trunk-conditioning attributes, use the following commands beginning in global configuration mode:

Procedure

	Command or Action	Purpose
Step 1	Device (config)# **voice class permanent** *tag*	Creates a voice class. The tag number range is from 1 to 10000, and it must be unique on the router. **Note** The **voice-class** command in dial-peer configuration mode is entered with a hyphen. The **voice class** command in global configuration mode is entered without the hyphen.
Step 2	Device (config-voice-class)# **signal keepalive** *seconds*	(Optional) Defines the keepalive signaling packet interval. The seconds range is from 1 to 65535; the default is 5.
Step 3	Device (config-voice-class)# {**no-action** | **idle-only** | **oos-only** | **both**	(Optional) Sets the signaling pattern (when the far-end keepalive message is lost or when AIS is received from the far end). The keywords are as follows: • no-action—Sends no signaling pattern. • idle-only or oos-only—Sends only one signaling pattern. • both—Restores the default (both signaling patterns are sent). **Note** The no form of the command restores the default also.
Step 4	Device (config-voice-class)# **signal pattern** {**idle receive** | **idle transmit** | **oos receive** | **oos transmit**} *bit-pattern*	(Optional) Overrides the default values for the idle and receive OOS patterns or configures OOS transmit signaling patterns. The keywords and argument are as follows: • idle receive—Defines the signaling pattern for an idle message from the network and the signaling pattern to be sent to the PBX if the network trunk is OOS and signal sequence oos idle-only or signal sequence oos are configured. The defaults are: ◦ For near-end E&M—0000 (for T1) or 0001 (for E1) ◦ For near-end FXO loop start—0101 ◦ For near-end FXO ground start—1111 ◦ For near-end FXS—0101 ◦ For near-end MELCAS—1101 • idle transmit—Defines the signaling pattern for an idle message from the PBX. The defaults are: ◦ For near-end E&M—0000 ◦ For near-end FXO—0101 ◦ For near-end FXS loop start—0101 ◦ For near-end FXS ground start—1111 ◦ For near-end MELCAS—1101

	Command or Action	Purpose
		• oos receive—Defines the OOS signaling pattern to be sent to the PBX if the network trunk is OOS and signal sequence oos oos-only or signal sequence oos are configured. The defaults are: ◦ For near-end E&M—1111 ◦ For near-end FXO loop start—1111 ◦ For near-end FXO ground start—0000 ◦ For near-end FXS loop start—1111 ◦ For near-end FXS ground start—0101 ◦ For near-end MELCAS—1111 The receive signal pattern comes from the data network side to the PBX. The transmit signal pattern comes from the PBX to the data network side. The range for all options is from 0000 to 1111. Repeat the command entry for each signal pattern required. • oos transmit—Defines the signaling pattern for an OOS message from the PBX. There are no default signaling patterns defined. • bit-pattern—Defines the ABCD bit pattern. Valid values are from 0000 to 1111. The receive signal pattern comes from the data network side to the PBX. The transmit signal pattern comes from the PBX to the data network side. The range for all options is from 0000 to 1111. Repeat the command entry for each signal pattern required.
Step 5	Device (config-voice-class)# **signal timing oos timeout** { *seconds* | **disabled}**	(Optional) Changes the timeout period for asserting a receive OOS pattern to the PBX when signaling packets are lost. This action changes the delay time before a busyout is sent to the PBX. The keyword and argument are as follows: • s*econds*—Defines the delay duration between the loss of signaling packets and the beginning of the OOS state. The range is from 1 to 65535. The default is 30. • **disabled**—Deactivates the detection of packet loss. If no signaling packets are received from the network, the router does not send an OOS pattern to the PBX and it continues sending voice packets. Use this option to disable busyout to the PBX.
Step 6	Device (config-voice-class)# **signal timing oos restart** *seconds*	(Optional) Configures permanent voice connections to be restarted after the trunk has been OOS for a specified time. The default is no signal timing OOS pattern parameters are configured. **Note** This command has no effect if **signal timing oos timeout** is set to disabled.
Step 7	Device (config-voice-class)# **signal timing oos slave-standby** *seconds*	(Optional) Configures a slave port to return to its initial standby state after the trunk has been OOS for a specified time. The default is no signal timing OOS pattern parameters are configured.

	Command or Action	Purpose
		Note This command has no effect if **signal timing oos timeout** is set to disabled.
Step 8	Device (config-voice-class)# **signal timing oos {suppress-all | suppress-voice}** *seconds*	(Optional) Configures the router or concentrator to stop sending voice packets or voice and signaling packets to the network if it detects a transmit OOS signaling pattern from the PBX for a specified time. The default is no signal timing OOS pattern parameters are configured. **Note** An OOS transmit signaling pattern must be configured with the **signal pattern oos transmit** command (see Step 4).
Step 9	Device (config-voice-class)# **signal timing idle suppress-voice** *seconds*	(Optional) Configures the router or concentrator to stop sending voice packets after the trunk has been idle for a specified time. The default is no signal timing OOS pattern parameters are configured.
Step 10	Device (config-dial-peer)# **voice-class permanent** *tag*	Assigns the voice class to the dial peer. • *tag*—Specifies the unique number. The valid range is from 1 to 10000. **Note** The **voice-class** command in dial-peer configuration mode is entered with a hyphen. The **voice class** command in global configuration mode is entered without the hyphen.

Verify

This section provides information that you can use to confirm that your configuration is working properly.

Certain show commands are supported by the https://sso.cisco.com/autho/forms/CDClogin.html (registered customers only), which allows you to view an analysis of show command output.

- **show voice call summary** Used to verify that all trunks are up and in the S_CONNECT state.

When the trunks comes up, the console will display the message %HTSP-5-UPDOWN: Trunk port(channel) [1/0:1(1)] is up.

This is sample output from the **show voice call summary** command:

```
PORT         CODEC    VAD VTSP STATE            VPM STATE
============ ======== === ==================== ======================
0/1/0:0.1       g729r8    n  S_CONNECT             S_TRUNKED
0/1/0:0.2       g729r8    n  S_CONNECT             S_TRUNKED
```

A trunk that is not up will show up as S_TRUNK_PEND:

```
PORT         CODEC    VAD VTSP STATE            VPM STATE
============ ======== === ==================== ======================
0/1/0:0.1       -              -      -            S_TRUNK_PEND
0/1/0:0.2       g729r8    n  S_CONNECT             S_TRUNKED
```

Troubleshoot

This section provides information that you can use to troubleshoot your configuration.

Troubleshooting Commands

Certain show commands are supported by the https://sso.cisco.com/autho/forms/CDClogin.html (registered customers only), which allows you to view an analysis of show command output.

Note Before issuing debug commands, please see http://www.cisco.com/c/en/us/support/docs/dial-access/integrated-services-digital-networks-isdn-channel-associated-signaling-cas/10374-debug.html

- **show call history voice | Include DisconnectText** - Shows the disconnect reason for the last few failed calls.
- **show voice call summary** - Shows the call active on both call legs.
- **show voice dsp** - Shows that the Digital Signal Processors (DSPs) are in use and are processing packets.

The associated voice-ports on both routers must be **shutdown/no shutdown** after you configure the connection trunk. This also clears the voice ports if you see user busy as a disconnect cause.

This is sample command output from the **show voice dsp** command:

```
BOOT                        PAK
TYPE DSP CH CODEC    VERS STATE STATE  RST AI PORT     TS ABORT   TX/RX-PAK-CNT
==== === == ======= ==== ===== ====== === == ======= == ===== ===============
C549 000 01 g729r8   3.4 busy  idle     0  0 3/0:12  13     0 3522765/3578769
         00 g729r8   .41 busy  idle     0  0 3/0:0    1     0 3505023/3560759
C549 001 01 g729r8   3.4 busy  idle     0  0 3/0:13  14     0 3522761/3578601
         00 g729r8   .41 busy  idle     0  0 3/0:1    2     0 3522794/3578579
```

The next sample output is the most common debug output for the **debug voip ccapi inout** command. This debug was taken under the common mistake of a missing POTS peer on the called side. In the example, the analog side router does not have a POTS peer to terminate the trunk; the digital calling side will have these debugs in this situation:

```
maui-slt-01#

*Mar  1 00:11:19.903: cc_api_call_setup_ind (vdbPtr=0x620B2DE8,
callInfo={called=2000,called_oct3=0x81,calling=,calling_oct3=0x0,
calling_oct3a=0x0,calling_xlated=false,subscriber_type_str=RegularLine
,fdest=1,peer_tag=2, prog_ind=3},callID=0x621C45F0)
*Mar  1 00:11:19.903: cc_api_call_setup_ind type 3 , prot 0
*Mar  1 00:11:19.903: cc_process_call_setup_ind (event=0x62332908)
*Mar  1 00:11:19.903: >>>>CCAPI handed cid 3 with tag 2 to app "DEFAULT"
*Mar  1 00:11:19.907: sess_appl: ev(24=CC_EV_CALL_SETUP_IND), cid(3), disp(0)
*Mar  1 00:11:19.907: sess_appl: ev(SSA_EV_CALL_SETUP_IND), cid(3), disp(0)
*Mar  1 00:11:19.907: ssaCallSetupInd
*Mar  1 00:11:19.907: ccCallSetContext (callID=0x3, context=0x621C4E90)
*Mar  1 00:11:19.907: ssaCallSetupInd cid(3), st(SSA_CS_MAPPING),oldst(0),
ev(24)ev->e.evCallSetupInd.nCallInfo.finalDestFlag = 1
*Mar  1 00:11:19.907: ssaCallSetupInd finalDest cllng(1000), clled(2000)
*Mar  1 00:11:19.907: ssaCallSetupInd cid(3), st(SSA_CS_CALL_SETTING),
oldst(0), ev(24)dpMatchPeersMoreArg result= 0
*Mar  1 00:11:19.907: ssaSetupPeer cid(3) peer list:
tag(1) called number (2000)
*Mar  1 00:11:19.907: ssaSetupPeer cid(3), destPat(2000), matched(1),
prefix(), peer(61EE565C), peer->encapType (2)
*Mar  1 00:11:19.907: ccCallProceeding (callID=0x3, prog_ind=0x0)
*Mar  1 00:11:19.907: ccCallSetupRequest (Inbound call = 0x3, outbound
peer =1, dest=, params=0x6233BD30 mode=0, *callID=0x6233C098, prog_ind = 3)
*Mar  1 00:11:19.907: ccCallSetupRequest numbering_type 0x81
*Mar  1 00:11:19.907: ccCallSetupRequest encapType 2 clid_restrict_disable 1
null_orig_clg 1 clid_transparent 0 callingNumber 1000
*Mar 1 00:11:19.907: dest pattern 2..., called 2000, digit_strip 0
*Mar 1 00:11:19.907: callingNumber=1000, calledNumber=2000, redirectNumber=
display_info= calling_oct3a=0
*Mar 1 00:11:19.907: accountNumber=, finalDestFlag=1,
guid=1d0d.9a0f.14f0.11cc.8008.b3df.433e.6402
*Mar 1 00:11:19.911: peer_tag=1
*Mar 1 00:11:19.911: ccIFCallSetupRequestPrivate: (vdbPtr=0x621D74DC, dest=,
callParams={called=2000,called_oct3=0x81, calling=1000,calling_oct3=0x0,
```

```
calling_xlated=false, subscriber_type_str=RegularLine, fdest=1,
voice_peer_tag=1}, mode=0x0) vdbPtr type = 1
*Mar 1 00:11:19.911: ccIFCallSetupRequestPrivate: (vdbPtr=0x621D74DC, dest=,
callParams={called=2000, called_oct3 0x81, calling=1000,calling_oct3 0x0,
calling_xlated=false, fdest=1, voice_peer_tag=1}, mode=0x0, xltrc=-5)
*Mar 1 00:11:19.911: ccSaveDialpeerTag (callID=0x3, dialpeer_tag=0x1)
*Mar 1 00:11:19.911: ccCallSetContext (callID=0x4, context=0x624C3094)
*Mar 1 00:11:19.911: ccCallReportDigits (callID=0x3, enable=0x0)
*Mar 1 00:11:19.911: cc_api_call_report_digits_done (vdbPtr=0x620B2DE8,
callID=0x3, disp=0)
*Mar 1 00:11:19.911: sess_appl: ev(52=CC_EV_CALL_REPORT_DIGITS_DONE),
cid(3), disp(0)
*Mar 1 00:11:19.911: cid(3)st(SSA_CS_CALL_SETTING)ev
(SSA_EV_CALL_REPORT_DIGITS_DONE)oldst(SSA_CS_MAPPING)
cfid(-1)csize(0)in(1)fDest(1)
*Mar 1 00:11:19.911: -cid2(4)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_MAPPING)
*Mar 1 00:11:19.911: ssaReportDigitsDone cid(3) peer list: (empty)
*Mar 1 00:11:19.911: ssaReportDigitsDone callid=3 Reporting disabled.
*Mar 1 00:11:19.947: cc_api_call_disconnected(vdbPtr=0x621D74DC,
callID=0x4, cause=0x1)
*Mar 1 00:11:19.947: sess_appl: ev(11=CC_EV_CALL_DISCONNECTED), cid(4), disp(0)
*Mar 1 00:11:19.947: cid(4)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_DISCONNECTED)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(0)fDest(0)
*Mar 1 00:11:19.947: -cid2(3)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
*Mar 1 00:11:19.951: ssaDiscSetting
*Mar 1 00:11:19.951: ssa: Disconnected cid(4) state(1) cause(0x1)
*Mar 1 00:11:19.951: ccCallDisconnect (callID=0x4, cause=0x1 tag=0x0)
*Mar 1 00:11:19.951: ccCallDisconnect (callID=0x3, cause=0x1 tag=0x0)
*Mar 1 00:11:19.951: cc_api_call_disconnect_done(vdbPtr=0x620B2DE8, callID=0x3,
disp=0, tag=0x0)
*Mar 1 00:11:19.955: sess_appl: ev(12=CC_EV_CALL_DISCONNECT_DONE), cid(3),
disp(0)
*Mar 1 00:11:19.955: cid(3)st(SSA_CS_DISCONNECTING)ev
(SSA_EV_CALL_DISCONNECT_DONE)oldst(SSA_CS_CALL_SETTING)
cfid(-1)csize(0)in(1)fDest(1)
*Mar 1 00:11:19.955: -cid2(4)st2(SSA_CS_DISCONNECTING)oldst2(SSA_CS_CALL_SETTING)
*Mar 1 00:11:19.955: ssaDisconnectDone
*Mar 1 00:11:19.963: cc_api_icpif: expect factor = 0
*Mar 1 00:11:19.963: cc_api_call_disconnect_done(vdbPtr=0x621D74DC,
callID=0x4, disp=0, tag=0x0)
*Mar 1 00:11:19.967: sess_appl: ev(12=CC_EV_CALL_DISCONNECT_DONE),
cid(4), disp(0)
*Mar 1 00:11:19.967: cid(4)st(SSA_CS_DISCONNECTING)ev
(SSA_EV_CALL_DISCONNECT_DONE)oldst(SSA_CS_CALL_SETTING)
cfid(-1)csize(1)in(0)fDest(0)
*Mar 1 00:11:19.967: ssaDisconnectDone
```

Additional References for Connection Trunk

Related Documents

Related Topic	Document Title
Cisco IOS commands	http://www.cisco.com/en/US/docs/ios/12_2/voice/configuration/guide/fvvfax_c.html

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Feature Information for Connection Trunk

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 Connection Trunk

Feature Name	Releases	Feature Information
Connection Trunk	Cisco IOS XE Release 3.17S	The Connection Trunk simulates a trunk connection through the creation of virtual trunk tie-lines between two telephony endpoints. The following commands were introduced by this feature: **show voice call summary**, **show call history voice**, **show voice dsp**, .

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA 95134-1706
USA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands