Gateway to Gateway Connections: Transport

This chapter describes three options for connecting LMR gateways:

Connection Trunk (Unicast)

This section describes these aspects of unicast connection trunk connections:

Overview

In general, a trunk, or tie line, is a permanent point-to-point communication link between two voice endpoints. From a Cisco IOS software perspective, the unicast **connection trunk** command creates a permanent VoIP call between two VoIP gateways. It simulates a trunk connection by creating virtual trunk tie lines between two telephony endpoints. To the connected systems, it appears as if a T1 trunk is directly connecting them.

The key features of unicast connection trunk connections are:

- Permanent connection—always up
- Transports signaling end-to-end
- Can be used only with other Cisco IOS software-based devices supporting unicast connection trunk

From an LMR perspective, a typical application for the unicast connection trunk would be as a replacement for leased line connections. Instead of dedicating an entire circuit for LMR traffic, the circuit could be configured for IP transmission with the LMR traffic and other data traffic sharing the same line.

Notice that we continually qualify the connection trunk as either unicast or multicast. To the voice endpoints, both types present the same permanent tie-line style connection. The difference lies in how that connection is implemented across the IP network. The unicast version employs an H.323 connection setup mechanism to establish a channel to another voice gateway and directs the Real-Time Transport Protocol (RTP)/RTP Control Protocol (RTP/RTCP) datastream to that single endpoint. The multicast version directs the RTP/RTCP datastream to an IP multicast address so it can be received by any number of endpoints. Aside from the protocols used for multicast routing, no other telephony connection setup mechanism is used.

Configuration

The relevant format for the **connection trunk** command is as follows:

```
connection trunk digits [answer-mode]
```

The *digits* argument corresponds to the E.164 address used to identify the voice endpoint on the far side of the connection. The optional **answer-mode** keyword identifies this side of the connection as a "slave-side." As a slave-side, the gateway does not attempt to initiate a trunk connection, but instead waits for an incoming call before establishing the trunk. It is recommended that one side of the connection be configured with the **answer-mode** keyword. This configuration scheme minimizes the time routers take to bring up trunks and ensures that trunks go down when connections are lost between two gateways. Otherwise, the gateways might not attempt to reestablish the trunk when the underlying circuit is up again.

Although Cisco IOS software permits a range of voice endpoint options for connection trunk, in an LMR environment, the unicast connection trunk will be set up between two analog E&M ports, between two digital E&M ports, or from an analog E&M port to a digital E&M port. In addition to the physical interfaces, the configuration requires at least two voice dial peers. The VoIP voice dial peer is used to associate the E.164 address specified in the **connection trunk** command to the IP address for the gateway on the other side of the connection. The POTS voice dial peer is used to associate an incoming E.164 address to the physical interface on the gateway that will terminate the connection.

Example 4-1 shows a sample unicast trunk configuration for two routers, lmr-3745c and lmr-3725e. In this configuration, both routers connect through their FastEthernet0/1 interfaces because they are on the same LAN segment. The parts of the configurations in Example 4-1 that are referred to in the following sections are in bold.

Example 4-1 Sample Unicast Connection Trunk Configuration

Master Gateway lmr-3745c	Slave Gateway lmr-3725e
!	!
hostname lmr-3745c	hostname lmr-3725e
!	!
controller T1 1/0	!
framing esf	!
crc-threshold 320	!
linecode b8zs	!
cablelength short 133	!
! This line creates a digital LMR voice	!
! port. We use one time slot per voice	!
! port to ensure a 1:1 mapping between	!
! voice ports on the trunk connection.	!
ds0-group 0 timeslots 1 type e&m-lmr	!
!	!
interface FastEthernet0/1	interface FastEthernet0/1
ip address 10.40.0.2 255.255.255.0	ip address 10.40.0.1 255.255.255.0
duplex auto	duplex auto
speed auto	speed auto
!	!
voice-port 1/0:0	voice-port 1/0/0
! With this line, we control the flow of	lmr m-lead audio-gate-in
! voice frames onto the network based on	lmr e-lead seize
! the state of the M-lead from the LMR	operation 4-wire
! device. If the M-lead is raised, then	type 3
! we send voice frames on the connection.	signal lmr
lmr m-lead audio-gate-in	no echo-cancel enable
lmr e-lead seize	**connection trunk 40002 answer-mode**
no echo-cancel enable	!
connection trunk 40001	!
!	!
!	!
!	!
!	!
dial-peer voice 10400001 voip	dial-peer voice 10400002 voip
destination-pattern 4...1	destination-pattern 4...2
session target ipv4:10.40.0.1	session target ipv4:10.40.0.2
codec g711ulaw	codec g711ulaw
vad aggressive	vad aggressive
!	!
dial-peer voice 374540002 pots	**dial-peer voice 372540001 pots**
destination-pattern 40002	destination-pattern 40001
port 1/0:0	port 1/0/1
!	!
end	end

Operation

The stages in the life of a unicast trunk connection between two voice interfaces are as follows:

Connection Initialization

In this stage, we actually build the permanent connection between the endpoints. The gateway routers:

- Establish an H.323 connection.

- Perform H.225.0 session negotiation.
- Synchronize the endpoints.

Data transfer

Once the connection is established, the endpoints sit in an inactive state sending connection maintenance frames, until either end decides to send data, which is the active state.

- Inactive state:
  - RTCP sender and receiver reports are sent.
  - Signaling keepalives are sent.
- Active state:
  - RTCP sender and receiver reports are sent.
  - Signaling transitions are sent.
  - RTP voice frames are sent.

Disconnect

Because this is a permanent connection, it will stay up as long as the gateways, their respective voice interfaces, and the IP connection between them stay up.

Connection Initialization

From a Cisco IOS software perspective, when the voice interface on the master side of the connection, router lmr-3745c, enters the up state, both administratively and operationally, the gateway places a call to the E.164 address specified in the **connection trunk** command. In this case, the number dialed is 40001. This address matches the destination pattern address configured on the VoIP dial peer with tag 10400001. Thus, the target IP address for the H.323 connection for the device on the slave side of the connection, router lmr-3725e, is 10.40.0.1.

Looking at the frames on the wire, an H.323 connection is established from the master to the slave gateways as seen in Table 4-1.

Table 4-1 Unicast Connection Trunk Connection Initialization Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
1	0	10.40.0.2	Broadcast	ARP[1]	—	—	60	Who has 10.40.0.1? Tell 10.40.0.2
2	0.000298	10.40.0.1	10.40.0.2	ARP	—	—	60	10.40.0.1 is at 00:07:b3:5d:0a:91
3	1.99729	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [SYN] Seq=2509514642 Ack=0 Win=4128 Len=0
4	1.997989	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [SYN, ACK] Seq=3049137647 Ack=2509514643 Win=4128 Len=0
5	1.998175	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [ACK] Seq=2509514643 Ack=3049137648 Win=4128 Len=0

Table 4-1 Unicast Connection Trunk Connection Initialization Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
6	1.999282	10.40.0.2	10.40.0.1	H.255.0	11000	1720	361	CS: setup OpenLogicalChannel [Unreassembled Packet]
7	2.009371	10.40.0.1	10.40.0.2	H.255.0	1720	11000	177	CS: callProceeding OpenLogicalChannel
8	2.011518	10.40.0.1	10.40.0.2	H.255.0	1720	11000	124	CS: alerting
9	2.013301	10.40.0.1	10.40.0.2	H.255.0	1720	11000	165	CS: connect

1. Address Resolution Protocol

After establishing the MAC-layer address for the slave gateway in frames 1 and 2, the master gateway opens a TCP connection to the H.323 port (1720) on the slave in frames 3 through 5. The master issues an H.225 call setup message containing the calling number (40002), the called number (40001), the media channel (RTP/18654), and the media control channel (RTCP/18655). The slave responds with an H.225 call proceeding message that contains its media channel (RTP/16664) and media control channel (RTCP/16665).

The called number in the call setup message matches the destination pattern configured on the POTS dial peer with tag 372541001 on the slave gateway in Example 4-1. Under that dial peer, the destination pattern is associated with voice port 1/0/0. Voice port 1/0/0 also has a connection trunk configured. The E.164 address on that connection trunk command (40002) matches the calling number contained in the call setup. The slave gateway responds again to the master with an alerting message. If the calling and called addresses on the slave gateway did not correspond to the called and calling addresses on the master, the slave would have responded with a release complete message after the call proceeding message and the trunk would not be created.

Immediately following the alerting message, the slave gateway sends out a connect message indicating everything is set up on its end. So, at this point, both gateways have presumably set up all the structures on their respective ends to handle the data stream. They also know each other's RTP and RTCP ports to use in sending the data. However, before the real voice traffic commences, the gateways send synchronization frames as shown in Table 4-2.

Table 4-2 Unicast Connection Trunk Synchronization Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
14	2.07382	10.40.0.2	10.40.0.1	RTP	18654	16664	214	Payload type=ITU-T G.711 PCMU, SSRC=199491586, Seq=8120, Time=40347
15	2.078833	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0, Mark
16	2.0897	10.40.0.1	10.40.0.2	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
17	2.093819	10.40.0.2	10.40.0.1	RTP	18654	16664	214	Payload type=ITU-T G.711 PCMU, SSRC=199491586, Seq=8121, Time=40507

Table 4-2 Unicast Connection Trunk Synchronization Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
18	2.098826	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0, Mark
19	2.109381	10.40.0.1	10.40.0.2	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0

The master side of the connection sends an RTP voice frame containing 20 ms of voice in frames 14 and 17, and a channel-associated signaling (CAS) signaling frame in frames 15 and 18 every 20 ms. The slave side responds with a CAS signaling frame in frames 16 and 19. This three-frame handshake occurs 24 times over the next half-second as shown in Table 4-3.

Table 4-3 Three-Frame Handshake Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
89	7.520661	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
90	7.571345	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
91	10.98381	10.40.0.2	10.40.0.1	RTCP	18655	16665	146	Sender Report
92	11.25979	10.40.0.1	10.40.0.2	RTCP	16665	18655	146	Sender Report
93	12.52268	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
94	12.57333	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
95	13.76831	10.40.0.1	10.40.0.2	RTCP	16665	18655	146	Sender Report
96	14.15189	10.40.0.2	10.40.0.1	RTCP	18655	16665	146	Sender Report
97	17.52471	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
98	17.57529	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
99	18.73985	10.40.0.1	10.40.0.2	RTCP	16665	18655	146	Sender Report
100	20.94287	10.40.0.2	10.40.0.1	RTCP	18655	16665	146	Sender Report
101	22.52673	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
102	22.57727	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0

After the synchronization period, in the absence of any voice or signaling frames traversing the connection, each side of the connection issues a keepalive frame. The default period of these keepalive messages is five seconds. The rate can be adjusted using the following configuration command:

signal keepalive {*seconds* | **disabled**}

Although it is possible to turn off the keepalive messages altogether, it is not advisable because their presence helps the gateways know when there is a problem with the connection.

These keepalive frames use the CAS signaling payload type (123). They reflect the current state of the signaling across the connection. For instance, if the LMR device connected to the slave side has its E-lead high, then the keepalive frames will carry the 0xF (1111) signaling bits across to the master side of the connection to be played out.

Included in the mix of frames for the connection are the RTCP media control channel frames. Each side of the connection will issue a sender report or receiver report about every five seconds. See *RFC 1889 - RTP: A Transport Protocol for Real-Time Applications* for more information on the contents of these frames. Sender reports are sent when the sender has sent any RTP frames since the last report. In the quiescent state, because the default is to send keepalive CAS signaling frames every five seconds, sender reports are usually seen from each side as shown in Table 4-4.

Table 4-4 Sender Report Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
121	47.53689	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
122	47.58715	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
123	47.8469	10.40.0.2	10.40.0.1	RTCP	18655	16665	146	Sender Report
124	50.42424	10.40.0.1	10.40.0.2	RTCP	16665	18655	146	Sender Report
125	50.84008	10.40.0.2	10.40.0.1	RTCP	18655	16665	126	Receiver Report
126	52.53889	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
127	52.5891	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0
128	55.1975	10.40.0.2	10.40.0.1	RTCP	18655	16665	146	Sender Report
129	56.31227	10.40.0.1	10.40.0.2	RTCP	16665	18655	146	Sender Report
130	57.54097	10.40.0.2	10.40.0.1	RTP	18654	16664	70	Payload type=Unknown (123), SSRC=186122242, Seq=0, Time=0
131	57.59106	10.40.0.1	10.40.0.2	RTP	16664	18654	70	Payload type=Unknown (123), SSRC=482017281, Seq=0, Time=0

However, RFC 1889 provides for a jitter factor on the frames of between 0.5 and 1.5 times the nominal arrival rate, so it is possible for one side of the connection to send two RTCP frames within the keepalive time. In this case, the first RTCP frame after the keepalive will be a sender report, and the next RTCP frame will be a receiver report as shown in frames 123 through 125 of Table 4-4.

Finally, the TCP stack sends periodic keepalives for the H.323 connection during the life of the connection trunk as shown in Table 4-5. The master gateway initiates this messaging every 60 seconds.

Table 4-5 Periodic TCP Keepalive Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
134	62.00001	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [ACK] Seq=2509514949 Ack=3049137951 Win=3824 Len=0
135	62.00027	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [ACK] Seq=3049137952 Ack=2509514950 Win=3821 Len=0
186	121.9868	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [ACK] Seq=2509514949 Ack=3049137951 Win=3824 Len=0
187	121.9871	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [ACK] Seq=3049137952 Ack=2509514950 Win=3821 Len=0
234	181.9736	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [ACK] Seq=2509514949 Ack=3049137951 Win=3824 Len=0
235	181.9738	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [ACK] Seq=3049137952 Ack=2509514950 Win=3821 Len=0

Caveats

- A single **ds0-group** command can be defined to handle all the DS0s on a digital interface, and a single **connection trunk** command can be defined on the associated voice port. This would reduce the amount of manual configuration required to 1 voice port and 1 POTS dial peer, instead of 24 voice ports and 24 POTS dial peers for a fully loaded T1. However, there is no guarantee of one-to-one mapping of DS0s on either end of the trunk. In addition, each time the router reloads, the mapping can be different from last time. This configuration complicates troubleshooting because you are not able to isolate the problem to a single, or even a few, time slots without taking down the entire trunk group. This configuration may also not be practical given the configuration of the LMR units that will attach to the digital interfaces.
- The only way to guarantee that a particular DS0 on one router maps through connection trunk to a particular DS0 on another router is to create a DS0 group for each DS0 on the T1.

Connection PLAR

This section describes these aspects of PLAR connections:

Overview

Private line automatic ringdown (PLAR) circuits are switched connections between statically configured voice endpoints. The endpoints have the destination address of the connection preconfigured and so do not require user dialing to connect calls. PLAR connections are those in which a phone goes off-hook and a remote phone rings without digits being dialed. As switched calls, the connections will be torn down on certain events; for example, the calling party goes on-hook, and the resources made available for other connections.

The following are the main similarities and differences between connection PLAR mode and connection trunk (unicast) mode:

- Connection trunk mode is a permanent connection; the VoIP call is always connected independently of the associated voice port being on-hook or off-hook.
- Connection PLAR mode is a switched VoIP call; the call is set up on an as-needed basis. With connection PLAR, no bandwidth is consumed while the device connected to the associated voice port is on-hook. When a device is taken off-hook, the call is automatically connected, and the remote device is signaled to go off-hook.
- Both connection trunk (unicast) and connection PLAR modes have statically configured endpoints and do not require user dialing to connect calls.
- Connection trunk (unicast) mode allows supplemental call signaling to be passed over the IP network between the two telephony devices. Connection PLAR transports only limited types of signaling.
- Connection PLAR does not use a proprietary keepalive mechanism and is thus can establish connections to other H.323 capable endpoints.

From an LMR standpoint, PLAR connections would be employed in those leased line replacement scenarios in which the overhead of a connection trunk (unicast) connection was not needed or desired. PLAR connections would also be useful for connecting to non-Cisco IOS software-based devices that support H.323 call setup mechanisms.

Configuration

The relevant format for the **connection trunk** command is as follows:

```
connection plar digits
```

The *digits* argument corresponds to the E.164 address used to identify the voice endpoint on the far side of the connection. Although Cisco IOS software permits a range of voice endpoint options for connection PLAR, in an LMR environment a PLAR connection may be established only between two analog E&M ports, between two digital E&M ports, or between an analog E&M port and a digital E&M port. In addition to the physical interfaces, the configuration requires at least two voice dial peers. The VoIP voice dial peer is used to associate the E.164 address specified in the **connection plar** command to the IP address for the gateway on the other side of the connection. The POTS voice dial peer is used to associate an incoming E.164 address to the physical interface on the gateway that will terminate the connection.

Table 4-5 shows a sample connection PLAR configuration for two routers, lmr-3745c and lmr-3725e. In this configuration, both routers connect via their FastEthernet0/1 interfaces because they are on the same LAN segment. The parts of the configurations in Table 4-5 that are referred to in the following sections are in bold.

Example 4-2 Sample Connection PLAR Configuration

lmr-3745c	lmr-3725e
!	!
hostname lmr-3745c	hostname lmr-3725e
!	!
controller T1 1/0	!
framing esf	!
crc-threshold 320	!
linecode b8zs	!
cablelength short 133	!
!	!
interface FastEthernet0/1	interface FastEthernet0/1
ip address 10.40.0.2 255.255.255.0	**ip address 10.40.0.1 255.255.255.0**
duplex auto	duplex auto
speed auto	speed auto
!	!
voice-port 1/0:0	voice-port 1/0/0
lmr m-lead dialin	lmr m-lead dialin
lmr e-lead voice	lmr e-lead voice
no echo-cancel enable	operation 4-wire
timeouts call-disconnect 3	type 3
timeouts teardown lmr 30	signal lmr
timing hookflash-in 0	no echo-cancel enable
connection plar 41001	timeouts call-disconnect 3
!	**timeouts teardown lmr 25**
!	timing hookflash-in 0
!	connection plar
!	!
!	!
dial-peer voice 10400001 voip	dial-peer voice 10400002 voip
destination-pattern 4...1	destination-pattern 4...2
session target ipv4:10.40.0.1	session target ipv4:10.40.0.2
codec g711ulaw	codec g711ulaw
vad aggressive	vad aggressive
!	!
!	!
dial-peer voice 374541002 pots	**dial-peer voice 372541001 pots**
destination-pattern 41002	destination-pattern 41001
port 1/0:0	port 1/0/0
!	!
!	!
end	end

Operation

The stages in the life of a PLAR connection between two voice interfaces are as follows:

Connection Initialization

In this stage, we actually build the permanent connection between the endpoints. The gateway routers:

- Establish an H.323 connection.
- Perform H.225.0 session negotiation.
- Synchronize the endpoints.

Data Transfer

Once the connection is established, the endpoints sit in an inactive state sending connection maintenance frames, until either end decides to send data, which is the active state.

- Inactive state. RTCP sender and receiver reports are sent.
- Active state:
  - RTCP sender and receiver reports are sent.
  - RTP voice frames are sent.

Disconnect

For Cisco IOS software-based LMR gateways, when the party initiating the call goes on-hook, which is the M-lead idle state, it starts the call teardown timer on that gateway. For the LMR gateway receiving the call, as soon as it stops receiving voice frames, it starts its call teardown timer. The gateway whose timer expires first initiates the actual call teardown.

Connection Initialization

In this section, we examine the PLAR connection initialization process in detail. The examples will use two Cisco IOS software-based gateways running software containing the LMR feature. Although technically either side can initiate the call that brings up the connection, we will refer to the gateway making the call as the "calling" gateway and the one on the receiving end of the call as the "called" gateway.

From a Cisco IOS software perspective, the voice interfaces associated with the PLAR connection on both the calling and called gateways must be in an up state, both administratively and operationally. In order to initiate the call, the calling gateway (lmr-3745c) must have its voice interface configured with the **lmr m-lead** command with the **dialin** option as shown in Table 4-5. The other two M-lead options, **inactive** and **audio-gate-in**, will not trigger the H.323 connection process. With the **lmr m-lead dialin** command configured, when the M-lead enters a seize state, the gateway places a call to the E.164 address specified in the **connection plar** command. In this case, the number dialed is 41001. This address matches the destination pattern address configured on the VoIP dial peer with tag 10400001. Thus, the target IP address for the H.323 connection to the called gateway (lmr-3725e) is 10.40.0.1.

Looking at the frames on the wire, an H.323 connection is established from the calling to called gateways as seen in Table 4-6.

Table 4-6 Connection PLAR Connection Initialization Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
4	10.75153	10.40.0.2	Broadcast	ARP	—	—	60	Who has 10.40.0.1? Tell 10.40.0.2
5	10.75184	10.40.0.1	10.40.0.2	ARP	—	—	60	10.40.0.1 is at 00:07:b3:5d:0a:91
6	12.74956	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [SYN] Seq=4291627303 Ack=0 Win=4128 Len=0
7	12.75026	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [SYN, ACK] Seq=1500783035 Ack=4291627304 Win=4128 Len=0

Table 4-6 Connection PLAR Connection Initialization Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
8	12.75047	10.40.0.2	10.40.0.1	TCP	11000	1720	60	11000 > 1720 [ACK] Seq=4291627304 Ack=1500783036 Win=4128 Len=0
9	12.75153	10.40.0.2	10.40.0.1	H.225.0	11000	1720	361	CS: setup OpenLogicalChannel [Unreassembled Packet]
10	12.76163	10.40.0.1	10.40.0.2	H.225.0	1720	11000	177	CS: callProceeding OpenLogicalChannel
11	12.76367	10.40.0.1	10.40.0.2	H.225.0	1720	11000	142	CS: progress
12	12.82407	10.40.0.1	10.40.0.2	H.225.0	1720	11000	165	CS: connect

After establishing the MAC-layer address for the called gateway in frames 4 and 5, the calling gateway opens a TCP connection to the H.323 port (1720) on the called gateway in frames 6 through 8. The calling gateway issues an H.225 call setup message (frame 9) containing the calling number (40002), the called number (40001), the media channel (RTP 19508), and the media control channel (RTCP 19509).

The called number in the call setup message matches the destination pattern configured on the POTS dial peer with tag 372541001 on the called gateway. Thus, the called gateway responds with an H.225 call proceeding message in frame 10 that contains its media channel (RTP/16454) and media control channel (RTCP/16455). If the called gateway had been unable to match the called number to any voice port, it would have responded with a release complete message (unassigned number) and that would have been the end of the call. But, because the called number is assigned to a voice port, the called gateway responds again to the calling gateway with a progress message in frame 11.

Note that if the E.164 address configured on the **connection plar** command on the called gateway voice port did not match the calling number, the connection would still be established. The called gateway need only find a match on its POTS dial peer destination patterns for the called number to accept the call. Of course, because the calling gateway obtained the calling number from its associated POTS dial peer, if the called gateway tried to make a return call, it would not match the voice port on the calling router.

Immediately following the progress message, the called gateway sends out a connect message in frame 12 indicating everything is set up on its end. So, at this point, both gateways have presumably set up all the structures on their respective ends to handle the data stream. They also know each other's RTP and RTCP ports to use in sending the data. However, before the real voice traffic commences, the calling gateway sends about 2 seconds-worth of voice frames to the called gateway as shown in Table 4-7.

Table 4-7 Connection PLAR Synchronization Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
13	12.82995	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6698, Time=40347
14	12.84971	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6699, Time=40507
15	12.8697	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6700, Time=40667

Table 4-7 Connection PLAR Synchronization Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
110	14.72861	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6793, Time=55547
111	14.74859	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6794, Time=55707
112	14.76859	10.40.0.2	10.40.0.1	RTP	19508	16454	134	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6795, Time=55867
113	14.76861	10.40.0.2	10.40.0.1	RTP	19508	16454	60	Payload type=Unknown (19), SSRC=287375362, Seq=6796, Time=55947
114	14.98875	10.40.0.2	10.40.0.1	RTCP	19509	16455	146	Sender Report
115	16.30351	10.40.0.2	10.40.0.2	LOOP			60	Loopback
116	16.51785	10.40.0.2	10.40.0.1	RTCP	19509	16455	126	Receiver Report
117	16.93295	10.40.0.1	10.40.0.2	RTCP	16455	19509	150	Receiver Report
118	20.00043	10.40.0.1	10.40.0.1	LOOP			60	Loopback

The called side of the connection sends a series of RTP voice frame containing 20 ms of voice in frames 13 through 112. When the voice transmission ceases, the calling party sends an RTP type 19 (comfort noise) frame in frame 113. At that point each side of the connection will issue a sender report or receiver report about every 5 seconds. See *RFC 1889 - RTP: A Transport Protocol for Real-Time Applications* for more information on the contents of these frames. Sender reports are sent when a gateway has sent any RTP frames since it last issued a report. In the quiescent state, there will not be any keepalive or other RTP frames, so we should see only receiver reports.

Connection Teardown

Unlike the unicast connection trunk mode in which the connection stays up as long as the voice ports and underlying IP circuit are up, with connection PLAR, the voice call can be torn down from either side. Each gateway participating in the connection maintains a teardown timer. If there is an active connection, the teardown timer begins running whenever the interface M-lead is in an idle state and whenever no voice frames are received from the network. The length of this timer (in seconds) is set using the following command on the voice port:

```
timeouts teardown lmr {<5-60000> | infinity}
```

We will illustrate the operation of the teardown timer using the output of the **debug vpm signal** command. First, we examine the behavior on the calling router. When the M-lead is raised, we see the following debug messages:

```
lmr-3745c#
Jan 29 16:42:34.566 EST: htsp_process_event: [1/0:0(1), LMR_ONHOOK, E_DSP_SIG_1100]
lmr_onhook_offhook
Jan 29 16:42:34.566 EST: htsp_timer_stop htsp_setup_ind
```

```
Jan 29 16:42:34.566 EST: [1/0:0(1)] get_local_station_id calling num= calling name=
calling time=01/29 16:42  orig called=
Jan 29 16:42:34.566 EST: htsp_timer - 3000 msec
Jan 29 16:42:34.570 EST: htsp_process_event: [1/0:0(1), LMR_WAIT_SETUP_ACK,
E_HTSP_SETUP_ACK] lmr_wait_setup_ack_get_ack
Jan 29 16:42:34.570 EST: htsp_timer_stop
Jan 29 16:42:34.570 EST: htsp_process_event: [1/0:0(1), LMR_OFFHOOK, E_HTSP_PROCEEDING]
htsp_progress_notifyhtsp_call_bridged
Jan 29 16:42:34.582 EST: htsp_process_event: [1/0:0(1), LMR_OFFHOOK,
E_HTSP_VOICE_CUT_THROUGH] lmr_offhook_voice_cut
Jan 29 16:42:34.582 EST: htsp_timer_stop
Jan 29 16:42:34.582 EST: htsp_process_event: [1/0:0(1), LMR_OFFHOOK, E_HTSP_CONNECT]
lmr_offhook_connect
Jan 29 16:42:34.582 EST: htsp_timer_stop
Jan 29 16:42:34.582 EST: htsp_timer_stop2
```

The last line in bold indicates that the teardown timer has stopped running. As long as the calling gateway keeps the M-lead raised, the timer will not start. On the called side, the debug output from the same call appears as follows:

```
Jan 29 16:42:34.576 EST: htsp_timer_stop3 htsp_setup_req
Jan 29 16:42:34.576 EST: htsp_process_event: [1/0/0, LMR_ONHOOK, E_HTSP_SETUP_REQ]
lmr_onhook_setup
Jan 29 16:42:34.576 EST: htsp_timer_stop htsp_progress
Jan 29 16:42:34.576 EST: lmr_start_timer: 2000 ms
Jan 29 16:42:34.576 EST: htsp_timer - 2000 msechtsp_call_bridged
Jan 29 16:42:34.580 EST: htsp_process_event: [1/0/0, LMR_WAIT_CUT_THRU,
E_HTSP_VOICE_CUT_THROUGH] lmr_cut_thru
Jan 29 16:42:34.580 EST: htsp_timer_stop
Jan 29 16:42:34.580 EST: lmr_pak_suppress_enable FALSE
Jan 29 16:42:34.580 EST: lmr_start_timer2: 25 second
Jan 29 16:42:34.580 EST: htsp_timer2 - 25000 msec
Jan 29 16:42:34.580 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_DSP_SIG_0000]
lmr_conn_onhook
Jan 29 16:42:34.580 EST: htsp_timer_stop
Jan 29 16:42:34.580 EST: lmr_pak_suppress_enable TRUE
Jan 29 16:42:34.580 EST: lmr_start_timer2: 25 second
Jan 29 16:42:34.580 EST: htsp_timer2 - 25000 msec
Jan 29 16:42:34.604 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_V_PAK_RCVD]
lmr_conn_pkt_rcvd
Jan 29 16:42:34.604 EST: htsp_timer_stop2 lmr_offhook (0)
Jan 29 16:42:34.604 EST: [1/0/0] set signal state = 0x8 timestamp = 0
Jan 29 16:42:36.888 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_V_PAK_STOP]
lmr_conn_pkt_stoplmr_onhook (0)
Jan 29 16:42:36.888 EST: [1/0/0] set signal state = 0x0 timestamp = 0
Jan 29 16:42:36.888 EST: lmr_start_timer2: 25 second
Jan 29 16:42:36.888 EST: htsp_timer2 - 25000 msec
```

In the previous lines in bold, we actually see the timer start three times. The timeout value for this voice port is set at 25 seconds. Because we have not raised the M-lead on this side of the connection, we wholly depend on the presence of voice frames to stop the timer from running. We see this timer transition from start to stop and to start again during the two-second voice spurt sent when the call is made. When the called gateway gets another talk spurt a few seconds later, the following debug messages appear:

```
Jan 29 16:42:56.421 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_V_PAK_RCVD]
lmr_conn_pkt_rcvd
Jan 29 16:42:56.421 EST: htsp_timer_stop2 lmr_offhook (0)
Jan 29 16:42:56.421 EST: [1/0/0] set signal state = 0x8 timestamp = 0
```

When the called gateway gets the voice frames and goes off-hook, the teardown timer stops counting as shown in the bold line of the previous debug messages. When the talk spurt ends and the called gateway goes on-hook, we see the following:

```
Jan 29 16:43:30.890 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_V_PAK_STOP]
lmr_conn_pkt_stoplmr_onhook (0)
Jan 29 16:43:30.890 EST: [1/0/0] set signal state = 0x0 timestamp = 0
Jan 29 16:43:30.890 EST: lmr_start_timer2: 25 second
Jan 29 16:43:30.890 EST: htsp_timer2 - 25000 msec
```

If 25 seconds elapse since the called gateway went on-hook, the timer expires, and the call is torn down. On the called gateway, the following debug messages appear:

```
Jan 29 16:44:18.888 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_V_PAK_STOP]
lmr_conn_pkt_stoplmr_onhook (0)
Jan 29 16:44:18.888 EST: [1/0/0] set signal state = 0x0 timestamp = 0
Jan 29 16:44:18.888 EST: lmr_start_timer2: 25 second
Jan 29 16:44:18.888 EST: htsp_timer2 - 25000 msec
lmr-3725e#
lmr-3725e#
lmr-3725e#
Jan 29 16:44:43.889 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_EVENT_TIMER2]
lmr_offhook_timer
Jan 29 16:44:43.889 EST: htsp_timer_stop2
Jan 29 16:44:43.889 EST: htsp_timer_stop3
Jan 29 16:44:43.889 EST: htsp_process_event: [1/0/0, LMR_CONNECT, E_HTSP_RELEASE_REQ]
lmr_conn_release
Jan 29 16:44:43.889 EST: htsp_timer_stop
Jan 29 16:44:43.889 EST: htsp_timer_stop2 lmr_onhook (0)
Jan 29 16:44:43.889 EST: [1/0/0] set signal state = 0x0 timestamp = 0
```

On the calling gateway, the following debug messages appears:

```
lmr-3745c#
Jan 29 16:44:43.888 EST: htsp_timer_stop3
Jan 29 16:44:43.892 EST: htsp_process_event: [1/0:0(1), LMR_CONNECT, E_HTSP_RELEASE_REQ]
lmr_conn_release
Jan 29 16:44:43.892 EST: htsp_timer_stop
Jan 29 16:44:43.892 EST: htsp_timer_stop2 lmr_onhook (0)vnm_dsp_set_sig_state:[recEive and
transMit1/0:0(1)] set signal state = 0x0
```

If the called side had raised its M-lead at any time during the call, we would have seen messages similar to the bold lines in the previous debug messages indicating that timer had stopped.

Table 4-8 shows the disconnect mechanism from a frame trace perspective.

Table 4-8 Connection PLAR Disconnect Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
119	21.02002	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6797, Time=105867, Mark
120	21.03996	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6798, Time=106027
121	21.0599	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6799, Time=106187
122	21.07991	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6800, Time=106347

Table 4-8 Connection PLAR Disconnect Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
600	30.51938	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=7272, Time=181867
601	30.53936	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=7273, Time=182027
602	30.55936	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=7274, Time=182187
603	30.57934	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=7275, Time=182347
604	30.59933	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=7276, Time=182507
605	30.6093	10.40.0.2	10.40.0.1	RTP	19508	16454	60	Payload type=Unknown (19), SSRC=287375362, Seq=7277, Time=182667
607	34.88555	10.40.0.1	10.40.0.2	RTCP	16455	19509	150	Receiver Report
608	34.95209	10.40.0.2	10.40.0.1	RTCP	19509	16455	146	Sender Report
611	38.12519	10.40.0.2	10.40.0.1	RTCP	19509	16455	126	Receiver Report
613	40.49343	10.40.0.1	10.40.0.2	RTCP	16455	19509	126	Receiver Report
614	43.53205	10.40.0.2	10.40.0.1	RTCP	19509	16455	126	Receiver Report
615	43.90542	10.40.0.1	10.40.0.2	RTCP	16455	19509	126	Receiver Report
617	46.89014	10.40.0.1	10.40.0.2	RTCP	16455	19509	126	Receiver Report
618	49.03379	10.40.0.2	10.40.0.1	RTCP	19509	16455	126	Receiver Report
619	49.85	10.40.0.1	10.40.0.2	RTCP	16455	19509	126	Receiver Report
621	54.71046	10.40.0.2	10.40.0.1	RTCP	19509	16455	126	Receiver Report
622	55.89928	10.40.0.1	10.40.0.2	H.225.0	1720	11000	104	CS: releaseComplete
623	55.90063	10.40.0.2	10.40.0.1	H.225.0	11000	1720	104	CS: releaseComplete
624	55.90117	10.40.0.2	10.40.0.1	RTCP	19509	16455	86	Receiver Report
625	55.90234	10.40.0.1	10.40.0.2	RTCP	16455	19509	86	Receiver Report
626	55.90255	10.40.0.2	10.40.0.1	ICMP			70	Destination unreachable
627	56.09722	10.40.0.1	10.40.0.2	TCP	1720	11000	60	1720 > 11000 [ACK] Seq=1500783408 Ack=4291627661 Win=3771 Len=0

Either side may send voice traffic over the connection. In Table 4-8, note that there are no signaling frames sent advising the other side of lead status changes. In frames 119 through 604, the calling gateway sends out a stream of voice frames. Again, when the stream is finished, we see the RTP type 19 (comfort noise) frame in frame 605. For the next 25 seconds we see the expected handshake of RTCP sender and receiver report frames in frames 607 through 621.

Recall from the configuration in Example 4-2 that the called gateway (lmr-3725e) has the **timeouts teardown lmr 25** command configured on its voice port. The teardown timer starts after frame 605 in Table 4-8, and thus we see at frame 622 the called party sending a release complete message on the H.225 part of the connection. The calling party acknowledges the release by sending its own release complete message in frame 623.

Both sides send final RTCP receiver report messages in frames 624 and 625. It is interesting to note that the calling gateway has closed out its connection by the time it gets the receiver report from the called gateway. Thus, the calling gateway sends an Internet Control Message Protocol (ICMP) destination/port unreachable message to the called gateway. Finally, the TCP stack on both gateways send periodic acknowledgments on the H.323 connection every minute for the next 10 minutes until the TCP connection itself is finally torn down.

Connection Trunk (Multicast)

This section describes these aspects of multicast connection trunk connections:

Overview

The operation of a unicast connection trunk is described in the "Connection Trunk (Unicast)" section. As mentioned in that section, the other alternative to unicast operation was to send the audio traffic using a multicast connection trunk. The multicast connection trunk differs from the unicast version in the following key aspects:

- One-to-many transmission for the multicast trunk instead of one-to-one.
- No H.323 call setup between endpoints with the multicast trunk.
- No lead state signaling frames passed in the RTP stream with the multicast trunk.

With a multicast configuration, an endpoint joins a multicast group based on its configured destination multicast IP address. By joining this group, the endpoint sends its audio stream to all other members of the group. The endpoint also receives audio streams from all other members of the multicast group. So, by extension, the one-to-many transmission capability ends up serving as a many-to-many communications vehicle. It is worth noting that the endpoints need not be standard radio or telephony devices. Any application capable of decoding RTP audio streams can join the multicast group to receive and send audio.

Because the endpoints have no knowledge of how many, or if any, endpoints have joined the group, they have no way to perform a standard H.323 call setup with other endpoints. Instead, the endpoints perform the equivalent of a call setup by using the IP multicast join mechanism to make their multicast-enabled

neighbors aware that they want to communicate on a particular multicast address. The multicast-enabled neighbors connecting the various endpoints will perform the routing necessary to deliver RTP/RTCP frames transmitted by one endpoint to all the other subscribed endpoints. When an endpoint no longer wants to participate in that multicast group, it is pruned from the distribution tree and no longer factors into distribution decisions made by other multicast-enabled devices.

Similarly, because each endpoint can receive multicast frames from multiple endpoints simultaneously, lead state signaling information is not sent between the endpoints because of the potential confusion it would cause the receiver. The multicast endpoints terminate lead states from the connected devices and just present the audio RTP frames, and associated RTCP frames, to the multicast group. Endpoints that rely on physical, as opposed to in-band, signaling need to recognize the presence of audio frames on that multicast group and initiate the lead controls necessary to signal the end device that audio is present. This can be accomplished through the use of the various voice port LMR commands. However, for those endpoints that rely on in-band or tone signaling, either all endpoints on the network will have to participate in the same tone scheme, or other equipment will be necessary to allow different units to interoperate.

Configuration

Note Designing networks for multicast routing is a subject unto itself. The myriad of configuration and design options for a multicast network are dealt with extensively by other documents and guides. In this document we will present the information necessary to configure an LMR gateway to participate in a multicast network. We will also present traffic flow, addressing, and other information useful in integrating multicast LMR ports into a multicast network. However, the multicast routing scheme chosen in the following examples may or may not work effectively with the scheme employed in your network.

A successful LMR multicast trunk connection involves four elements:

- A voice-port configured for LMR and connection trunk.
- A VoIP dial peer configured for multicast operation.
- A voice class to control keepalive signaling.
- A virtual interface and multicast enabled network interfaces.

From a voice port perspective, the Cisco IOS software configuration necessary for a connection trunk is nearly the same for both the unicast and multicast versions. The only difference is that the **answer-mode** option is not used because technically there will never be a call set up message to answer. Thus, the command to configure the multicast connection trunk is as follows:

```
connection trunk digits
```

The *digits* argument corresponds to the E.164 address used to identify the connection. The address has local significance only because no call setup message is sent to communicate that address to any other party. This E.164 address needs to match a destination pattern in a VoIP dial peer elsewhere in the router's configuration. The VoIP dial peer is configured to use the multicast session protocol and a destination address of the multicast IP address and port number for that specific multicast group.

As we learned in the "Connection Trunk (Unicast)" section, the unicast trunks send periodic keepalives to maintain the status of the connection. If a device stops receiving those keepalives, it tears down the connection and, depending on whether it is the master or the slave, attempts to restart the connection. In a point-to-point environment, this mechanism works well to maintain the integrity of the trunk. However, in a point-to-multipoint environment where each endpoint is responsible only for building its leg of the connection, this mechanism provides no useful service, thus no keepalives are sent. However, the dial

peers still look for the presence of the keepalives to determine the status of the connection. Therefore, to prevent the trunks from going down due to lack of keepalives, we need to create a voice class with the following options:

```
voice class permanent 1
 signal timing oos timeout disabled
 signal keepalive disabled
```

This voice class should be assigned to each multicast voice port.

Finally, we need to enable the LMR gateway to participate in the multicast routing set up for the network. The first step is to create a virtual host interface with the **interface vif1** command. This is a virtual interface that is similar to a loopback interface, a logical IP interface that is always up when the router is active. The virtual interface requires its own subnet with at least two addresses (that is, a 30-bit mask for IPv4 addressing). The LMR gateway uses an address on the virtual interface subnet other than the one configured to source the multicast frames generated by the voice port. For example, if the virtual interface is configured with the IP address 10.52.0.5/30, multicast frames will use 10.52.0.6 as their source IP address. In addition, the virtual interface subnet needs to be included in routing update. The subnet needs to be reachable by other multicast participants so they can trace their reverse path back to this source.

The codec configured in the dial peer with the **codec** command must be the same as the codec configured on other applications that use these LMR gateway ports. Codecs must also be the same between all parties to a connection in order for speech to be recognizable. The default code for VoIP is G.729r8.

The **ip qos dscp** command assigns all of the RTP packets with a Differentiated Services Code Point (DSCP) value of 5, giving priority to VoIP packets. If this statement is not present, speech quality may suffer. The **ip qos dscp** command implements part of Quality of Service (QoS) for the entire VoIP network by identifying the packets that require special processing. Policing and traffic shaping must be implemented also by defining the special processing for the identified packets.

Example 4-3 shows a sample multicast trunk configuration for two routers, lmr-3725e and lmr-3745c. In this configuration, both routers connect through their FastEthernet0/1 interfaces because they are on the same LAN segment. The parts of the configurations in Example 4-3 that are referred to in the following sections are in bold.

Example 4-3 Sample Multicast Connection Trunk Configuration

lmr-3725e

```
!
hostname lmr-3725e
!
ip multicast-routing
!
voice class permanent 1
 signal timing oos timeout disabled
 signal keepalive disabled
!
!
!
!
!
!
!
interface Vif1
 ip address 10.52.0.5 255.255.255.252
 ip pim sparse-dense-mode
!
interface FastEthernet0/1
 ip address 10.50.0.2 255.255.255.0
 ip pim sparse-dense-mode
 load-interval 30
 duplex auto
 speed auto
!
router ospf 50
 log-adjacency-changes
 network 10.50.0.0 0.0.0.255 area 0.0.0.0
 network 10.52.0.4 0.0.0.3 area 0.52.0.4
!
ip pim bidir-enable
!
voice-port 2/0/0
 voice-class permanent 1
 operation 4-wire
 type 3
 signal lmr
 lmr m-lead audio-gate-in
 lmr e-lead voice
 no echo-cancel enable
 no comfort-noise
 timeouts call-disconnect 3
 timing hookflash-in 0
 connection trunk 500001
!
dial-peer voice 239500001 voip
 destination-pattern 500001
 session protocol multicast
 session target ipv4:239.50.00.01:20000
 codec g711ulaw
 ip qos dscp 5 media
 vad aggressive
!
end
```

lmr-3745c

```
!
hostname lmr-3745c
!
ip multicast-routing
!
voice class permanent 1
 signal timing oos timeout disabled
 signal keepalive disabled
!
controller T1 2/0
 framing esf
 linecode b8zs
 cablelength short 133
 ds0-group 0 timeslots 1 type e&m-lmr
!
interface Vif1
 ip address 10.52.0.1 255.255.255.252
 ip pim sparse-dense-mode
!
interface FastEthernet0/1
 ip address 10.50.0.1 255.255.255.0
 ip pim sparse-dense-mode
 load-interval 30
 duplex auto
 speed auto
!
router ospf 50
 log-adjacency-changes
 network 10.50.0.0 0.0.0.255 area 0.0.0.0
 network 10.52.0.0 0.0.0.3 area 0.52.0.0
!
ip pim bidir-enable
!
voice-port 2/0:0
 voice-class permanent 1
 lmr m-lead audio-gate-in
 lmr e-lead voice
 no echo-cancel enable
 no comfort-noise
 timeouts call-disconnect 3
 timing hookflash-in 0
 connection trunk 500001
!
!
!
!
dial-peer voice 239500001 voip
 destination-pattern 500001
 session protocol multicast
 session target ipv4:239.50.00.01:20000
 codec g711ulaw
 ip qos dscp 5 media
 vad aggressive
!
end
```

Operation

The stages in the life of a multicast trunk connection are as follows:

Connection Initialization

The connection initialization process for a multicast connection trunk consists of the LMR gateway joining the multicast group identified by the multicast address configured on the associated dial peer. There is no H.323 or similar call set up mechanism.

Data Transfer

Once the connection is established, the endpoints will sit in an in-active state sending connection maintenance frames, until either end decides to send data (the active state).

- In-active state: RTCP sender and receiver reports are sent.
- Active state:

  RTCP sender and receiver reports are sent.

  RTP voice frames are sent.

Disconnect

The trunk will remain up until such time as the port is shut down.

Data Transfer

Table 4-9 shows the multicast connection trunk traffic during inactive and active states between the routers with the configurations listed in Example 4-3. Because the routers are configured to use Protocol Independent Multicast (PIM) as their multicast routing protocol, we see PIM hello frames from each LMR gateway every 30 seconds. We also see the RTCP receiver reports generated from each gateway on average of every 5 seconds. Note that the source address of the RTCP frames is the IP address of that gateway's virtual interface plus one. When the lmr-3725c router begins audio transmission at frame 188, it simply starts streaming frames out to the configured multicast IP address and port number. The default codec on the dial peer is G.729, so the RTP frames are only 74 bytes long, including Ethernet header.

Table 4-9 Multicast Connection Trunk Frames

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
119	21.02002	10.40.0.2	10.40.0.1	RTP	19508	16454	214	Payload type=ITU-T G.711 PCMU, SSRC=287375362, Seq=6797, Time=105867, Mark
136	141.2141	10.50.0.2	224.0.0.13	PIMv2	—	—	72	Hello
137	141.5193	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report
140	143.5913	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report
143	145.787	10.50.0.1	224.0.0.13	PIMv2	—	—	72	Hello
145	146.8313	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report

Table 4-9 Multicast Connection Trunk Frames (continued)

Frame	Time (sec)	Source	Destination	Protocol	Src Port	Dest Port	Length	Information
146	147.0996	10.52.0.6	239.50.0.1	RTCP	17005	20001	166	Receiver Report
171	167.801	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report
175	170.4909	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report
176	170.6267	10.50.0.2	224.0.0.13	PIMv2			72	Hello
178	172.3691	10.52.0.6	239.50.0.1	RTCP	17005	20001	166	Receiver Report
181	174.8909	10.52.0.2	239.50.0.1	RTCP	22969	20001	166	Receiver Report
182	175.1609	10.50.0.1	224.0.0.13	PIMv2			72	Hello
186	178.2789	10.52.0.6	239.50.0.1	RTCP	17005	20001	166	Receiver Report
188	179.4989	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5467, Time=213741899, Mark
189	179.5189	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5468, Time=213742059
190	179.5389	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5469, Time=213742219
191	179.5589	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5470, Time=213742379
192	179.5789	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5471, Time=213742539
193	179.5989	10.52.0.6	239.50.0.1	RTP	17004	20000	74	Payload type=ITU-T G.729, SSRC=268632070, Seq=5472, Time=213742699

Caveats

One of the main configuration mistakes made when configuring multicast connection trunks is forgetting to assign the appropriate voice class to the voice port in order to disable keepalive activity. In many cases, the global voice class gets created with the correct options, but it never gets assigned to the voice port. The result is that the voice port will enjoy two-way communication for about 15 seconds after it comes up, and then it will receive only audio. Using most of the display commands, the trunk will appear operational. However, if we look at the trunk signaling, it will show us that the trunk recognizes a lack of keepalives.

The following output shows the signaling and supervisory status of a multicast trunk with the voice class assigned to the voice port:

```
lmr-3725e# show voice trunk-conditioning signaling

2/0/0 :
hardware-state IDLE signal type is NorthamericanCAS
```

```
status : IDLE
forced playout pattern = 0x0
last-TX-ABCD=0000, last-RX-ABCD=0000
idle monitoring : tx
tx_idle = TRUE, rx_idle = FALSE, tx_oos = FALSE, lost_keepalive = FALSE
trunk_down_timer = 0, rx_ais_duration = 0, idle_timer = 0,tx_oos_timer = 0

lmr-3725e# show voice trunk-conditioning supervisory

SLOW SCAN, SCAN LMR
2/0/0 : state : TRUNK_SC_CONNECT, voice : off , signal : on ,master
        status: rcv IDLE, trunk connected
        sequence oos : idle-only
        pattern :rx_idle = 0000 tx_idle = 0000
        timing : idle = 0, restart = 0, standby = 0, timeout = 0
        supp_all = 0, supp_voice = 0, keep_alive = 0
        timer: oos_ais_timer = 0, timer = 0
        voice packet detection enable
```

Now look at the output for the same commands when the voice class is omitted from the voice port configuration. The first thing to notice are the words "lost keepalive" in bold in the status line for both commands. If your multicast trunk connection is not working and you see these words, then you need to check the configuration for your voice port and ensure that the voice class is properly configured for that port as in the configurations in Example 4-3.

```
lmr-3725e# show voice trunk-conditioning signaling

2/0/0 :
hardware-state IDLE signal type is NorthamericanCAS
status : lost keepalive, IDLE
forced playout pattern = 0x0
last-TX-ABCD=0000, last-RX-ABCD=0000
idle monitoring : tx
tx_idle = TRUE, rx_idle = FALSE, tx_oos = FALSE, lost_keepalive = TRUE
trunk_down_timer = 0, rx_ais_duration = 0, idle_timer = 0,tx_oos_timer = 0

lmr-3725e# show voice trunk-conditioning supervisory

SLOW SCAN, SCAN LMR
2/0/0 : state : TRUNK_SC_CONN_DEFAULT_IDLE, voice : off , signal : on ,master
        status: rcv IDLE, lost keepalive, trunk connected
        sequence oos : no-action
        pattern :tx_idle = 0000
        timing : idle = 0, restart = 0, standby = 0, timeout = 30
        supp_all = 0, supp_voice = 0, keep_alive = 5
        timer: oos_ais_timer = 93, timer = 93
        voice packet detection enable
```