



Troubleshooting Intra-Cluster Phone Calls

The case study in this appendix discusses in detail the call flow between two Cisco IP Phones within a cluster, called an intra-cluster call. This case study also focuses on Cisco CallManager and Cisco IP Phone initialization, registration, and KeepAlive processes. That is followed by a detailed explanation of an intra-cluster call flow. These processes are explained using the trace utilities and tools discussed in [Chapter 2, “Troubleshooting and Monitoring Tools and Utilities.”](#)

This appendix contains the following topics:

- [Sample Topology, page A-2](#)
- [Cisco IP Phone Initialization Process, page A-2](#)
- [Skinny Station Registration Process, page A-3](#)
- [Cisco IP Phone-to-Cisco IP Phone Call Flow within a Cluster, page A-5](#)
- [Cisco IP Phone-to-Cisco IP Phone Exchange of Skinny Station Messages During Call Flow, page A-5](#)
- [Cisco CallManager Initialization Process, page A-6](#)
- [Self-Starting Processes, page A-7](#)
- [Cisco CallManager Registration Process, page A-8](#)
- [Cisco CallManager KeepAlive Process, page A-10](#)
- [Cisco CallManager Intra-Cluster Call Flow Traces, page A-11](#)

Sample Topology

Given that you have two clusters named Cluster 1 and Cluster 2, the two Cisco CallManagers in Cluster 1 are called CCM3 and CCM4, while the two Cisco CallManagers in Cluster 2 are called CCM1 and CCM2.

The traces collected for this case study are from CCM1, which is located in Cluster 2. The call flow is based on the two Cisco IP Phones in Cluster 2. The IP addresses of these two Cisco IP Phones are 172.16.70.230 (directory number 1000), and 172.16.70.231 (directory number 1001), respectively.

Cisco IP Phone Initialization Process

The Cisco IP Phone initialization (or boot up) process is explained in detail in the following procedure.

Procedure

-
- Step 1** If you have set the appropriate options in DHCP server (such as Option 066 or Option 150), the Cisco IP Phone sends a request, at initialization, to the DHCP server to get an IP address, Domain Name System (DNS) server address, and TFTP server name or address. It also gets a default gateway address if you have set these options in the DHCP server (Option 003).
 - Step 2** If a DNS name of the TFTP sever is sent by DHCP, then a DNS sever IP address is required to map the name to an IP address. This step is bypassed if the DHCP server sends the IP address of the TFTP server. In this case study, the DHCP server sent the IP address of TFTP because DNS was not configured.
 - Step 3** If a TFTP server name is not included in the DHCP reply, then the Cisco IP Phone uses the default server name.
 - Step 4** The configuration file (.cnf) file is retrieved from the TFTP server. All .cnf files have the name SEP<mac_address>.cnf. If this is the first time the phone is registering with the Cisco CallManager, then a default file, SEPdefault.cnf, is downloaded to the Cisco IP Phone. In this case study, the first Cisco IP Phone uses the IP address 172.16.70.230 (its MAC address is SEP0010EB001720), and the second Cisco IP Phone uses the IP address 172.16.70.231 (its MAC address is SEP003094C26105).

- Step 5** All .cnf files include the IP address(es) for the primary and secondary Cisco CallManager(s). The Cisco IP Phone uses this IP address to contact the primary Cisco CallManager and to register.
- Step 6** Once the Cisco IP Phone has connected and registered with Cisco CallManager, the Cisco CallManager tells the Cisco IP Phone which executable version (called a load ID) to run. If the specified version does not match the executing version on the Cisco IP Phone, the Cisco IP Phone will request the new executable from the TFTP server and reset automatically
-

Skinny Station Registration Process

Cisco IP Phones communicate with the Cisco CallManager using the Cisco Skinny Station Protocol. The registration process allows a Skinny Station, such as the Cisco IP Phone, to inform the Cisco CallManager of its existence and to make calling possible.

[Table A-1](#) describes the primary messages in the Skinny Station registration process.

Table A-1 *Skinny Station Registration Process Descriptions*

Message	Description
Station Register	The station sends this message to announce its existence to the controlling Cisco CallManager.
Station Reset	Cisco CallManager sends this message to command the station to reset its processes.
Station IP Port	The station sends this message to provide Cisco CallManager with the User Datagram Protocol (UDP) port to be used with the Real Time Protocol (RTP) stream.
Station Register Acknowledge	Cisco CallManager sends this message to acknowledge the registration of a station.

Table A-1 Skinny Station Registration Process Descriptions (continued)

Message	Description
Station Register Reject	Cisco CallManager sends this message to reject a registration attempt from the indicated phone. char text [StationMaxDisplayTextSize]; }; Where: text is a character string, maximum length of 33 bytes, containing a textual description of the reason that registration is rejected.
Station Capabilities Request	Cisco CallManager sends this message to request the current capabilities of the station. Station capabilities may include compression standard and other H.323 capabilities.
Station Version Request	The station sends this message to request the version number of the software load for the station.
Station Version Response	Cisco CallManager sends this message to inform the station of the appropriate software version number.
Station Capabilities Response	The station sends this message to the Cisco CallManager in response to a Station Capabilities Request. The station's capabilities are cached in the Cisco CallManager and used to negotiate terminal capabilities with an H.323 compliant terminal.
Station Button Template Request	The station sends this message to request the button template definition for that specific terminal or Cisco IP Phone.
Station Button Template Response	Cisco CallManager sends this message to update the button template information contained in the station.
Station Time Date Request	The station sends this message to request the current date and time for internal usage and for displaying as a text string.
Station Define Time and Date	Cisco CallManager uses this message to provide the date and time information to the station. It provides time synchronization for the stations.

Cisco IP Phone-to-Cisco IP Phone Call Flow within a Cluster

This section describes a Cisco IP Phone (directory number 1000) calling another Cisco IP Phone (directory number 1001) within the same cluster. The cluster is a group of Cisco CallManagers having one common Publisher Structured Query Language (SQL) database and many Subscriber SQL databases.

In this cluster scenario, CCM1 is the publisher and CCM2 is a subscriber. The two Cisco IP Phones (1000 and 1001) are registered to CCM1 and CCM2, respectively. The call flow is shown in the diagram below. The two Cisco CallManagers within a cluster communicate with each other using Intra-Cluster Control Protocol (ICCP).

When a Cisco IP Phone goes off-hook, it opens a control Skinny Station session (with TCP as the underlying protocol) with the Cisco CallManager. After call control signaling is established between the two Cisco IP Phones and their respective Cisco CallManagers, the RTP stream starts flowing directly between the two phones, as shown in the diagram below. The Skinny Station call flow messages for this intra-cluster call are explained in next section.

Cisco IP Phone-to-Cisco IP Phone Exchange of Skinny Station Messages During Call Flow

The Skinny Station, in this case a Cisco IP Phone, initiates a connection to the Cisco CallManager, and then Cisco CallManager performs digit analysis before opening a control session with the destination Skinny Station.

**Note**

The Skinny Station messages are written in English so that they can be understood by end-users. Therefore, these messages are not explained in this section. However, these call flow Skinny Station messages are explained in more detail in later sections when traces are being examined.

Cisco CallManager Initialization Process

In this section the initialization process of Cisco CallManager will be explained with the help of traces that are captured from CCM1 (identified by the IP address 172.16.70.228). As described previously, SDI traces are a very effective troubleshooting tool because they detail every packet sent between endpoints.

This section describes the events that occur when Cisco CallManager is initialized. Understanding how to read traces will help you to properly troubleshoot the various Cisco CallManager processes, and the effect of those processes on services such as conferencing and call forwarding.

The following messages from the Cisco CallManager SDI trace utility show the initialization process on one of the Cisco CallManagers, in this case, CCM1.

- The first message indicates Cisco CallManager started its initialization process.
- The second message indicates Cisco CallManager read the default database values (for this case it is the primary or publisher database).
- The third message indicates Cisco CallManager listened to the various messages on TCP port 8002.
- The fourth message shows that after listening to these messages, Cisco CallManager added a second Cisco CallManager to its list: CCM2 (172.16.70.229).
- The fifth message indicates that Cisco CallManager has started and is running Cisco CallManager version 3.1(1).

```
16:02:47.765 CCM|CMPProcMon - CallManagerState Changed - Initialization
Started.
16:02:47.796 CCM|NodeId: 0, EventId: 107 EventClass: 3 EventInfo:
Cisco CM Database Defaults Read
16:02:49.937 CCM| SDL Info - NodeId: [1], Listen IP/Hostname:
[172.16.70.228], Listen Port: [8002]
16:02:49.984 CCM|dbProcs - Adding SdlLink to NodeId: [2], IP/Hostname:
[172.16.70.229]
16:02:51.031 CCM|NodeId: 1, EventId: 1 EventClass: 3 EventInfo:
Cisco CallManager Version=<3.1(1)> started
```

Self-Starting Processes

Once Cisco CallManager is up and running, it starts several other processes within itself. Some of these processes are shown below, including MulticastPoint Manager, UnicastBridge Manager, digit analysis, and route list. The messages described during these processes can be very useful when troubleshooting a problem related to the features in Cisco CallManager.

For example, assume that the route lists are not functioning and are unusable. To troubleshoot this problem, you would monitor these traces to determine whether the Cisco CallManager has started RoutePlanManager and if it is trying to load the RouteLists. In the sample configuration below, RouteListName="ipwan" and RouteGroupName="ipwan" are loading and starting.

```
16:02:51.031 CCM|MulicastPointManager - Started
16:02:51.031 CCM|UnicastBridgeManager - Started
16:02:51.031 CCM|MediaTerminationPointManager - Started
16:02:51.125 CCM|MediaCoordinator(1) - started
16:02:51.125 CCM|NodeId: 1, EventId: 1543 EventClass: 2 EventInfo:
Database manager started
16:02:51.234 CCM|NodeId: 1, EventId: 1542 EventClass: 2 EventInfo:
Link manager started
16:02:51.390 CCM|NodeId: 1, EventId: 1541 EventClass: 2 EventInfo:
Digit analysis started
16:02:51.406 CCM|RoutePlanManager - Started, loading RouteLists
16:02:51.562 CCM|RoutePlanManager - finished loading RouteLists
16:02:51.671 CCM|RoutePlanManager - finished loading RouteGroups
16:02:51.671 CCM|RoutePlanManager - Displaying Resulting RoutePlan
16:02:51.671 CCM|RoutePlanServer - RouteList Info, by RouteList and
RouteGroup Selection Order
16:02:51.671 CCM|RouteList - RouteListName='ipwan'
16:02:51.671 CCM|RouteList - RouteGroupName='ipwan'
16:02:51.671 CCM|RoutePlanServer - RouteGroup Info, by RouteGroup and
Device Selection Order
16:02:51.671 CCM|RouteGroup - RouteGroupName='ipwan'
```

The following trace shows the RouteGroup adding the device 172.16.70.245, which is CCM3 located in Cluster 1 and is considered an H.323 device. In this case, the RouteGroup is created to route calls to CCM3 in Cluster 1 with Cisco IOS Gatekeeper permission. If there is a problem routing the call to a Cisco IP Phone located in Cluster 1, then the following messages would help you find the cause of the problem.

```
16:02:51.671 CCM|RouteGroup - DeviceName='172.16.70.245'
16:02:51.671 CCM|RouteGroup -AllPorts
```

Part of the initialization process shows that Cisco CallManager is adding "Dns" (Directory Numbers). By reviewing these messages, you can determine whether the Cisco CallManager has read the directory number from the database.

```
16:02:51.671 CCM|NodeId: 1, EventId: 1540 EventClass: 2 EventInfo:
Call control started
16:02:51.843 CCM|ProcessDb - Dn = 2XXX, Line = 0,
Display = , RouteThisPattern, NetworkLocation = OffNet,
DigitDiscardingInstruction = 1, WhereClause =
16:02:51.859 CCM|Digit analysis: Add local pattern 2XXX , PID: 1,80,1
16:02:51.859 CCM|ForwardManager - Started
16:02:51.984 CCM|CallParkManager - Started
16:02:52.046 CCM|ConferenceManager - Started
```

In the following traces the Device Manager in Cisco CallManager is statically initializing two devices. The device with IP address 172.17.70.226 is a Gatekeeper and the device with IP address 172.17.70.245 is another Cisco CallManager in a different cluster. That Cisco CallManager is registered as an H.323 Gateway with this Cisco CallManager.

```
16:02:52.250 CCM|DeviceManager: Statically Initializing Device;
DeviceName=172.16.70.226
16:02:52.250 CCM|DeviceManager: Statically Initializing Device;
DeviceName=172.16.70.245
```

Cisco CallManager Registration Process

Another important part of the SDI trace is the registration process. When a device is powered up, it gets information via DHCP, connects to the TFTP server for its .cnf file, and then connects to the Cisco CallManager specified in the .cnf file. The device could be an MGCP Gateway, a Skinny Gateway, or a Cisco IP Phone. Therefore, it is important to be able to discover whether or not devices have successfully registered on the Cisco AVVID network.

In the following trace, Cisco CallManager has received new connections for registration. The registering devices are MTP_nsa-cm1 (MTP services on CCM1), and CFB_nsa-cm1 (Conference Bridge service on CCM1). These are software services running on Cisco CallManager but are treated internally as different external services and are therefore assigned a TCPHandle, socket number, and port number as well as a device name.

```
16:02:52.750 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fbaa00, Socket=0x594, IPAddr=172.16.70.228, Port=3279,
StationD=[0,0,0]
16:02:52.750 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fe05e8, Socket=0x59c, IPAddr=172.16.70.228, Port=3280,
StationD=[0,0,0]
16:02:52.781 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=MTP_nsa-cml, TCPHandle=0x4fbaa00, Socket=0x594,
IPAddr=172.16.70.228, Port=3279, StationD=[1,45,2]
16:02:52.781 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=CFB_nsa-cml, TCPHandle=0x4fe05e8, Socket=0x59c,
IPAddr=172.16.70.228, Port=3280, StationD=[1,96,2]
```

In the following trace, Skinny Station messages are sent between a Cisco IP Phone and Cisco CallManager. The Cisco IP Phone (172.16.70.231) is registering with Cisco CallManager. [Table A-1](#) provides the descriptions of Skinny Station messages. As soon as Cisco CallManager receives the registration request from a Cisco IP Phone, it assigns a TCPHandle number to this device. This number remains the same until the device or Cisco CallManager is restarted. Therefore, you can follow all the events related to a particular device by searching for or keeping track of the device's TCPHandle number, which appears in hexadecimal notation. Also, notice that Cisco CallManager provides the load ID to the Cisco IP Phone. Based on this load ID, the Cisco IP Phone runs the executable file (acquired from the TFTP server) that corresponds to the device.

```
16:02:57.000 CCM|StationInit - New connection accepted. DeviceName=,
TCPHandle=0x4fbbc30, Socket=0x5a4, IPAddr=172.16.70.231, Port=52095,
StationD=[0,0,0]
16:02:57.046 CCM|NodeId: 1, EventId: 1703 EventClass: 2 EventInfo:
Station Alarm, TCP Handle: 4fbbc30, Text: Name=SEP003094C26105
Load=AJ.30 Params=Status/IPAddr LastTime=A P1: 2304(900) P2:
-414838612 (e74610ac)
16:02:57.046 CCM|StationInit - ***** InboundStim - AlarmMessageID
tcpHandle=0x4fbbc30 Message="Name=SEP003094C26105 Load=AJ.30
Params=Status/IPAddr LastTime=A" Parm1=2304 (900) Parm2=-414838612
(e74610ac)
16:02:57.093 CCM|StationInit - Processing StationReg. regCount: 1
DeviceName=SEP003094C26105, TCPHandle=0x4fbbc30, Socket=0x5a4,
IPAddr=172.16.70.231, Port=52095, StationD=[1,85,1]
16:02:57.093 CCM|StationInit - InboundStim - IpPortMessageID:
32715(0x7fcb) tcpHandle=0x4fbbc30
```

Cisco CallManager KeepAlive Process

Both the station, device, or service and the Cisco CallManager use the following messages to maintain a knowledge of the communications channel between them. The messages are used to begin the KeepAlive sequence that ensures that the communications link between the Cisco CallManager and the station remains active. The following messages can originate from either the Cisco CallManager or the station.

```
16:03:02.328 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=MTP_nsa-cm2,
TCPHandle=0x4fa7dc0, Socket=0x568, IPAddr=172.16.70.229, Port=1556,
StationD=[1,45,1]
16:03:02.328 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=CFB_nsa-cm2,
TCPHandle=0x4bf8a70, Socket=0x57c, IPAddr=172.16.70.229, Port=1557,
StationD=[1,96,1]
16:03:06.640 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=SEP0010EB001720,
TCPHandle=0x4fbb150, Socket=0x600, IPAddr=172.16.70.230, Port=49211,
StationD=[1,85,2]
16:03:06.703 CCM|StationInit - InboundStim - KeepAliveMessage -
Forward KeepAlive to StationD. DeviceName=SEP003094C26105,
TCPHandle=0x4fbbc30, Socket=0x5a4, IPAddr=172.16.70.231, Port=52095,
StationD=[1,85,1]
```

The messages in the following trace depict the KeepAlive sequence which indicates that the communications link between the Cisco CallManager and the station is active. Again, these messages can originate from either the Cisco CallManager or the station.

```
16:03:02.328 CCM|MediaTerminationPointControl -
stationOutputKeepAliveAck tcpHandle=4fa7dc0
16:03:02.328 CCM|UnicastBridgeControl - stationOutputKeepAliveAck
tcpHandle=4bf8a70
16:03:06.703 CCM|StationInit - InboundStim - IpPortMessageID:
32715(0x7fcb) tcpHandle=0x4fbbc30
16:03:06.703 CCM|StationD - stationOutputKeepAliveAck
tcpHandle=0x4fbbc30
```

Cisco CallManager Intra-Cluster Call Flow Traces

The following SDI traces explore the intra-cluster call flow in detail. The Cisco IP Phones in the call flow can be identified by the directory number (dn), tcpHandle, and IP address. A Cisco IP Phone (dn: 1001, tcpHandle: 0x4fbbc30, IP address: 172.16.70.231) located in Cluster 2 is calling another Cisco IP Phone in the same Cluster (dn=1000, tcpHandle= 0x4fbb150, IP address= 172.16.70.230). Remember that you can follow a device through the trace by looking at the TCP handle value, time stamp, or name of the device. The TCP handle value for the device remains the same until the device is rebooted or goes offline.

The following traces show that the Cisco IP Phone (1001) has gone off-hook. The trace below shows the unique messages, TCP handle, and the called number, which are displayed on the Cisco IP Phone. There is no calling number at this point, because the user has not tried to dial any digits. The information below is in the form of Skinny Station messages between the Cisco IP Phones and the Cisco CallManager.

```
16:05:41.625 CCM|StationInit - InboundStim - OffHookMessageID
tcpHandle=0x4fbbc30
16:05:41.625 CCM|StationD - stationOutputDisplayText
tcpHandle=0x4fbbc30, Display= 1001
```

The next trace shows Skinny Station messages going from Cisco CallManager to a Cisco IP Phone. The first message is to turn on the lamp on the calling party's Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputSetLamp stim: 9=Line
instance=1 lampMode=LampOn tcpHandle=0x4fbbc30
```

The stationOutputCallState message is used by Cisco CallManager to notify the station of certain call related information.

```
16:05:41.625 CCM|StationD - stationOutputCallState tcpHandle=0x4fbbc30
```

The stationOutputDisplayPromptStatus message is used by Cisco CallManager to cause a call-related prompt message to be displayed on the Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbbc30
```

The stationOutputSelectSoftKey message is used by Cisco CallManager to cause the Skinny Station to select a specific set of soft keys.

```
16:05:41.625 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
```

The next message is used by Cisco CallManager to instruct the Skinny Station as to the correct line context for the display.

```
16:05:41.625 CCM|StationD - stationOutputActivateCallPlane
tcpHandle=0x4fbbc30
```

In the following message, the digit analysis process is ready to identify incoming digits and check them for potential routing matches in the database. The entry, `cn=1001`, represents the calling party number where `dd=""` represents the dialed digit, which would show the called part number. Note that `StationInit` messages are sent by the phone, `StationD` messages are sent by Cisco CallManager, and digit analysis is performed by Cisco CallManager.

```
16:05:41.625 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="")
16:05:41.625 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
```

The following debug message shows that the Cisco CallManager is providing inside dial tone to the calling party Cisco IP Phone.

```
16:05:41.625 CCM|StationD - stationOutputStartTone: 33=InsideDialTone
tcpHandle=0x4fbbc30
```

Once Cisco CallManager detects an incoming message and recognizes that the keypad button 1 has been pressed on the Cisco IP Phone, it immediately stops the output tone.

```
16:05:42.890 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 1 tcpHandle=0x4fbbc30
16:05:42.890 CCM|StationD - stationOutputStopTone tcpHandle=0x4fbbc30
16:05:42.890 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
16:05:42.890 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="1")
16:05:42.890 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.203 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
16:05:43.203 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="10")
16:05:43.203 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.406 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
```

```
16:05:43.406 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="100")
16:05:43.406 CCM|Digit analysis:
potentialMatches=PotentialMatchesExist
16:05:43.562 CCM|StationInit - InboundStim - KeypadButtonMessageID
kpButton: 0 tcpHandle=0x4fbbc30
16:05:43.562 CCM|Digit analysis: match(fqcn="", cn="1001", pss="",
dd="1000")
```

Once the Cisco CallManager has received enough digits to match, it provides the digit analysis results in a table format. Any extra digits pressed on the phone after this point will be ignored by Cisco CallManager, since a match has already been found.

```
16:05:43.562 CCM|Digit analysis: analysis results
16:05:43.562 CCM| |PretransformCallingPartyNumber=1001
|CallingPartyNumber=1001
|DialingPattern=1000
|DialingRoutePatternRegularExpression=(1000)
|PotentialMatches=PotentialMatchesExist
|DialingSdlProcessId=(1,38,2)
|PretransformDigitString=1000
|PretransformPositionalMatchList=1000
|CollectedDigits=1000
|PositionalMatchList=1000
|RouteBlockFlag=RouteThisPattern
```

The next trace shows that Cisco CallManager is sending out this information to a called party phone (the phone is identified by the tcpHandle number).

```
16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=1000,
CalledParty=1000, tcpHandle=0x4fbb150
```

The next trace indicates that Cisco CallManager is ordering the lamp to blink for incoming call indication on the called party's Cisco IP Phone.

```
16:05:43.578 CCM|StationD - stationOutputSetLamp stim: 9=Line
instance=1 lampMode=LampBlink tcpHandle=0x4fbb150
```

In the following traces, Cisco CallManager is providing ringer, display notification, and other call-related information to the called party's Cisco IP Phone. Again, you can see that all messages are directed to the same Cisco IP Phone because the same tcpHandle is used throughout the traces.

```
16:05:43.578 CCM|StationD - stationOutputSetRinger: 2=InsideRing
tcpHandle=0x4fbb150
```

```
16:05:43.578 CCM|StationD - stationOutputDisplayNotify
tcpHandle=0x4fbb150
16:05:43.578 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbb150
16:05:43.578 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbb150
```

Notice that Cisco CallManager is also providing similar information to the calling party's Cisco IP Phone. Again, the tcpHandle is used to differentiate between Cisco IP Phones.

```
16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=,
CalledParty=1000, tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputCallInfo
CallingPartyName=1001, CallingParty=1001, CalledPartyName=1000,
CalledParty=1000, tcpHandle=0x4fbbc30
```

In the next trace, Cisco CallManager provides an alerting or ringing tone to the calling party's Cisco IP Phone, notifying that the connection has been established.

```
16:05:43.578 CCM|StationD - stationOutputStartTone: 36=AlertingTone
tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputCallState tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputSelectSoftKeys
tcpHandle=0x4fbbc30
16:05:43.578 CCM|StationD - stationOutputDisplayPromptStatus
tcpHandle=0x4fbbc30
```

At this point, the called party's Cisco IP Phone goes off-hook. Therefore, Cisco CallManager stops generating the ringer tone to calling party.

```
16:05:45.140 CCM|StationD - stationOutputStopTone tcpHandle=0x4fbbc30
```

In the following messages, Cisco CallManager causes the Skinny Station to begin receiving a Unicast RTP stream. To do so, Cisco CallManager provides the IP address of the called party as well as codec information, and packet size in msec (milliseconds). PacketSize is an integer containing the sampling time in milliseconds used to create the RTP packets.



Note

Normally this value is set to 30msec. In this case it is set to 20msec.

```
16:05:45.140 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x4fbbc30 myIP: e74610ac (172.16.70.231)
16:05:45.140 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

Similarly, Cisco CallManager provides information to the called party (1000).

```
16:05:45.140 CCM|StationD - stationOutputOpenReceiveChannel
tcpHandle=0x4fbb150 myIP: e64610ac (172.16.70.230)
16:05:45.140 CCM|StationD - ConferenceID: 0 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

Cisco CallManager has received the acknowledgment message from called party for establishing the open channel for RTP stream, as well as the IP address of the called party. This message is to inform the Cisco CallManager of two pieces of information about the Skinny Station. First, it contains the status of the open action. Second, it contains the receive port address and number for transmission to the remote end. The IP address of the transmitter (calling part) of the RTP stream is `ipAddr`, and `PortNumber` is the IP port number of the RTP stream transmitter (calling party).

```
16:05:45.265 CCM|StationInit - InboundStim -
StationOpenReceiveChannelAckID tcpHandle=0x4fbb150, Status=0,
IpAddr=0xe64610ac, Port=17054, PartyID=2
```

The following messages are used by Cisco CallManager to order the station to begin transmitting the audio stream to the indicated remote Cisco IP Phone's IP address and port number.

```
16:05:45.265 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x4fbbc30 myIP: e74610ac (172.16.70.231)
16:05:45.265 CCM|StationD - RemoteIpAddr: e64610ac (172.16.70.230)
RemoteRtpPortNumber: 17054 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
```

In the following traces, the previously explained messages are sent to the called party. These messages are followed by the messages indicating the RTP media stream has been started between the called and calling party.

```
16:05:45.312 CCM|StationD - stationOutputStartMediaTransmission
tcpHandle=0x4fbb150 myIP: e64610ac (172.16.70.230)
16:05:45.328 CCM|StationD - RemoteIpAddr: e74610ac (172.16.70.231)
RemoteRtpPortNumber: 18448 msecPacketSize: 20
compressionType: (4)Media_Payload_G711Ulaw64k
16:05:46.203 CCM|StationInit - InboundStim - OnHookMessageID
tcpHandle=0x4fbbc30
```

The calling party's Cisco IP Phone finally goes on-hook, which terminates all the control messages between the Skinny Station and Cisco CallManager as well as the RTP stream between Skinny Stations.

```
16:05:46.203 CCM|StationInit - InboundStim - OnHookMessageID  
tcpHandle=0x4fbbc30
```