

Troubleshooting GUP, Alternate Endpoint and Load Balancing

Document ID: 18730

Introduction

Prerequisites

Requirements

Components Used

Conventions

Definitions

Lab Topology and Configurations

Lab Topology and Configurations

Alternate Gatekeepers

Gatekeeper Update Protocol

debug and show Commands for Gatekeeper Clusters

Debugs from "gkb-1" when it was First to Join the Cluster

Debugs from "gkb-2" when it Joined the Cluster after "gkb-1"

Debugs when an Endpoint Registers with One of the Gatekeepers in the Cluster

Debugs for when an Endpoint with Active Call Moves to Alternate Gatekeeper

Cisco Gateway Fail Over to Alternate Gatekeeper

Troubleshoot Issues with Alternate Endpoints

Verify the Gatekeeper has the Correct Alternate Endpoints

Verify whether Gatekeeper Includes Alternate Endpoints in its LCF or ACF RAS

Messages

Verify whether OGW Tries to Contact Alternates in Case Main Destination Endpoint

Fails

Troubleshoot Load Balance

Related Information

Introduction

This document is designed to help you troubleshoot and understand these Cisco Gatekeeper features:

- Gatekeeper clusters and Gatekeeper Update Protocol (GUP)
- Alternate endpoints
- Load balancing

Refer to Cisco High-Performance Gatekeeper for all necessary information about these features including feature overview, supported platforms, Cisco IOS® Software releases needed and how to configure, monitor, and maintain them.

Prerequisites

Requirements

Readers of this document should be knowledgeable of the following:

- Basic knowledge of gatekeeper functionality.
- Basic knowledge of VoIP, H.323 and Registration, Admission, and Status (RAS) signaling.

Components Used

The information in this document is based on the software and hardware versions below.

- Cisco IOS Software Release 12.3(4)T1
- Cisco Gateways: Cisco AS5300, Cisco AS5400, and Cisco 3725
- Cisco Gatekeepers: Cisco 3725 and Cisco 2611

The information presented in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If you are working in a live network, ensure that you understand the potential impact of any command before using it.

Conventions

For more information on document conventions, see the Cisco Technical Tips Conventions.

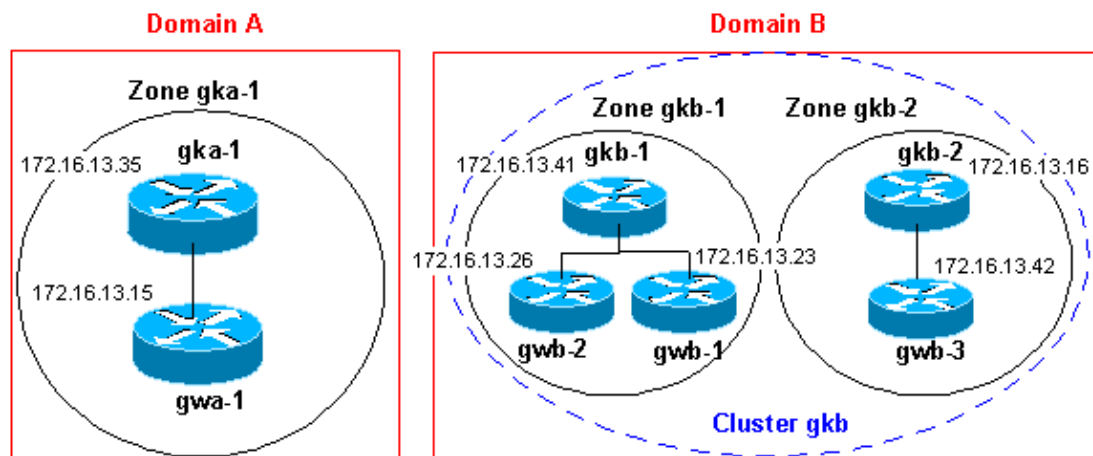
Definitions

Term	Definition
ARQ	Admission Request (ARQ) is a RAS message sent from a Cisco H.323 endpoint to a gatekeeper that requests an admission to establish a call.
ACF	Admission Confirm (ACF) is a RAS message sent from the gatekeeper to the endpoint that confirms the acceptance of a call.
ARJ	Admission Rejection (ARJ) is a RAS message from the gatekeeper to the endpoint that rejects the admission request.
GCF	Gatekeeper Confirm (GCF) is a RAS message sent from a gatekeeper to the Cisco H.323 endpoint that confirms the discovery of the gatekeeper.
GRQ	Gatekeeper Request (GRQ) is a RAS message sent from a Cisco H.323 endpoint to discover the gatekeeper.
GUP	Gatekeeper Update Protocol is used to share the information between gatekeepers in a cluster about their endpoints and active calls.
LCF	Location Confirm (LCF) is a RAS message sent from one gatekeeper to another that confirms the Location Request (LRQ) and includes the IP address of terminating endpoint.
LRJ	Location Reject (LRJ) is a RAS message sent from one gatekeeper to another that rejects the LRQ.
LRQ	Location Request is a RAS message sent from one gatekeeper to another that requests the IP address of a remote terminating endpoint.
RAS	RAS protocol allows a gatekeeper to perform Registration, Admission, and Status check of the endpoint.

RCF	Registration Confirm (RCF) is a RAS message sent from the gatekeeper to the endpoint that confirms the registration.
RRJ	Registration Reject (RRJ) is a RAS message sent from the gatekeeper that rejects the registration request.
RRQ	Registration Request (RRQ) is a RAS message sent from the endpoint to the gatekeeper that requests to register with it.
URQ	Unregistration Request (URQ) is RAS message sent from the endpoint to the gatekeeper that requests to un-register with it.

Lab Topology and Configurations

To explain how the features work and how to troubleshoot, a lab setup was built with this topology:



Lab Topology and Configurations

The basic configurations of all the gateways and gatekeepers are in the table below. With different cases, a certain change in configuration was needed. The change is indicated when this happens. The configurations below only contain the parts that are essential to the gateway or gatekeeper functionality for this lab.

The configurations of "gwb-1" and "gwb-2" are almost similar (except for IP address and H.323 ID). Therefore, only gwb-1 is shown below.

gwa-1
<pre>! controller E1 3/0 pri-group timeslots 1-2,16 ! interface Ethernet0/0 ip address 172.16.13.15 255.255.255.224 half-duplex h323-gateway voip interface h323-gateway voip id gka-1 ipaddr 172.16.13.35 1718 h323-gateway voip h323-id gwa-1 h323-gateway voip tech-prefix 1# !</pre>

```

voice-port 3/0:15
!
!
dial-peer voice 5336 pots
  incoming called-number
  destination-pattern 5336
  direct-inward-dial
  port 3/0:15
  prefix 21
!
dial-peer voice 3653 voip
  incoming called-number
  destination-pattern 3653
  session target ras
  dtmf-relay h245-alphanumeric
  codec g711ulaw
!
gateway
!
ntp clock-period 17178794
ntp server 172.16.13.35
end

```

gka-1

```

!
gatekeeper
  zone local gka-1 domainA.com 172.16.13.35
  zone remote gkb domainB.com 172.16.13.41 1719
  zone prefix gkb 36*
  zone prefix gka-1 53*
  gw-type-prefix 1#* default-technology
  no shutdown
!
no scheduler
max-task-timentp master
!
end

```

gwb-1

```

!
controller E1 0
  clock source line primary
  ds0-group 0 timeslots 1-2 type r2-digital r2-compelled
!
interface Ethernet0
  ip address 172.16.13.23 255.255.255.224
  h323-gateway voip interface
  h323-gateway voip id gkb-1 ipaddr 172.16.13.41 1718
  h323-gateway voip h323-id gwb-1
  h323-gateway voip tech-prefix 2#
!
dial-peer voice 3653 pots incoming called-number
  destination-pattern 3653
  port 0:0
  prefix 21
!
dial-peer voice 5336 voip
  incoming called-number
  destination-pattern 5336
  session target ras
  dtmf-relay h245-alphanumeric
  codec g711ulaw
!

```

```
gateway
!  
ntp clock-period 17179389  
ntp server 172.16.13.35  
end
```

gwb-3

```
!  
interface Ethernet0/0  
 ip address 172.16.13.42 255.255.255.224  
 half-duplex h323-gateway voip interface  
 h323-gateway voip id gkb-1 ipaddr 172.16.13.41 1718  
 h323-gateway voip  
 h323-id gwb-3  
 h323-gateway voip tech-prefix 1#  
!  
voice-port 3/0/0  
!  
voice-port 3/0/1  
!  
dial-peer voice 3653 pots  
 destination-pattern 3653  
 port 3/0/0  
 prefix 21  
!  
dial-peer voice 5336 voip  
 incoming called-number  
 destination-pattern 5336  
 session target ras  
 dtmf-relay h245-alphanumeric  
  
 codec g711ulaw  
  
gateway ! ntp clock-period 17179181 ntp server 172.16.13.35  
!  
end
```

gkb-1

```
!  
gatekeeper  
 zone local gkb-1 domainB.com 172.16.13.41  
 zone remote gka-1 domainA.com 172.16.13.35 1719  
 zone cluster local gkb gkb-1  
 element gkb-2 172.16.13.16 1719  
 gw-type-prefix 2#* default-technology  
 no shutdown  
!  
ntp clock-period 17179580  
ntp server 172.16.13.35  
!  
end
```

gkb-2

```
!  
gatekeeper  
 zone local gkb-2 domainB.com 172.16.13.16  
 zone cluster local gkb gkb-2  
 element gkb-1 172.16.13.41 1719  
!  
no shutdown  
!
```

```
ntp clock-period 17179199
ntp server 172.16.13.35
!
end
```

Alternate Gatekeepers

Prior to Cisco H.323 version 2, each zone was only controlled by a single gatekeeper. Cisco H.323 version 2 introduces the "alternate gatekeeper" idea to provide gatekeeper redundancy. Implementing the alternate gatekeeper feature allows multiple gatekeepers to control one zone. When an endpoint registers with a gatekeeper, it is provided with a list of alternate gatekeepers for the zone in which the endpoint registers, and for which alternates were specified using the CLI. If the gatekeeper fails, the endpoint may use the alternate gatekeepers in order to continue operation.

The alternate gatekeeper list is provided to the Cisco Gatekeeper through the CLI for each zone and is transmitted to endpoints through the RCF (including lightweight) and GRQ messages. This list may also be transmitted in other messages, such as ARJ or URQ, to facilitate a controlled gatekeeper shutdown.

Alternate gatekeepers learn about existing calls through an Interrupt Request (IRQ)/Information Request Response (IRR) exchange between the gateways and the gatekeepers and keep track of these calls.

An endpoint that detects the failure of its gatekeeper can safely recover from that failure by utilizing an alternate gatekeeper for future requests, including requests for existing calls. Alternate gatekeepers have to be configured in a cluster. They share the information about the endpoints and active calls using the GUP that runs on TCP.

Gatekeeper Update Protocol

Here are some major steps and caveats of the GUP. This should also help you troubleshoot.

- Once a gatekeeper that is configured to be part of a cluster comes on line, it opens a TCP port for listening for incoming connections for the GUP protocol.
- Then it announces its presence by sending a GRQ message on a periodic basis. The default period is 30 seconds and is configurable using the gatekeeper CLI **timer cluster-element announce** command. This GRQ message contains non-standard data to each alternate gatekeeper. This non-standard data is an indicator to the alternates that the GRQ is really not a GRQ at all, but rather is just an "announcement" message. Inside the GRQ message, the gatekeeper indicates the port number that it has open to listen for the GUP protocol.
- When you receive a GRQ from the new gatekeeper, other gatekeepers in the cluster open TCP channels to that port.
- GUP GRQ messages can be one of the following messages: announcementIndication, announcementReject, registrationIndication, unregistrationIndication, and resourceIndication.
- The announcement indication also carries information about the bandwidth utilization for the zone. This allows the alternate gatekeepers to properly manage the bandwidth for a single zone, even though the gatekeepers are in separate physical devices.
- To verify whether the alternate gatekeepers are properly communicating or not, use the command **show gatekeeper zone cluster**. This command also reports the bandwidth information for the alternate gatekeepers.
- The gatekeeper assumes that the alternate gatekeeper has failed (and assumes any previously allocated bandwidth is now available), if the gatekeeper does not receive an announcement message within six announcement periods, or if the TCP connection with the gatekeeper is detected to be broken. With six announcement periods every 30 seconds, the time is three minutes, which equates to what is assumed to be the average length of a call. It should then be fairly safe to assume that bandwidth has been freed. After three minutes, this gatekeeper declares its alternate as down and sends out an update

to notify all of its registered endpoints that there is no alternate gatekeeper.

- When an endpoint registers/unregisters with a gatekeeper in a cluster, that gatekeeper uses the registrationIndication/ unregistrationIndication message to update all other gatekeepers in that cluster about this change.
- If an endpoint reported a resource change using resource availability indicator (RAI) to a gatekeeper in a cluster, that gatekeeper reports the change to all alternate gatekeepers in that cluster by using the GUP message resourceIndication.
- The GUP messages are needed for the gatekeeper in a cluster to have sufficient knowledge about every endpoint in the zone (registration, bandwidth, active calls, resources) to be able to resolve all queries locally.
- When an endpoint is switched from one gatekeeper to an alternate, the alternate needs to learn about the calls that are active on the endpoint. When a gatekeeper sends an RCF for a new registration, it also sends an IRQ to get a list of all calls on the endpoint. It is important to ensure that the IRQ does not reach the endpoint before the RCF.
- Gatekeepers in a cluster permit a shutdown, even though there are active calls, as long as there is an alternate gatekeeper defined for all zones for which there are active calls. If any zone has an active call and no alternate gatekeeper defined, the gatekeeper refuses the shutdown.
- Alternate gatekeepers accept any Disconnect Requests (DRQs) for calls they were not aware of and pass appropriate information to the Authentication, Authorization, and Accounting (AAA) and Cisco Gatekeeper Transaction Message Protocol (GKTMP) servers. This happens when that endpoint moves to the alternate gatekeeper while there are active calls. In addition, IRR messages may be sent that contain call information for calls that were not previously known. For those IRRs, call records are constructed and bandwidth is allocated accordingly.
- The gatekeeper creates a unique announcement Indication message for each alternate gatekeeper. If an alternate gatekeeper receives a message that contains a gatekeeper identifier it does not recognize (which may happen if the alternate gatekeeper is an alternate for one zone), but not another, that information is ignored. However, the alternate gatekeeper detects errors in the configuration of the alternates by examining those messages and it reports those errors to the user.
- The true power of the GUP is realized when addresses are resolved for a remote zone. Instead of the need for the remote zone to send LRQs (either in sequence or blast) to all the gatekeepers, thus increasing the messaging overhead on wide-area links, it now needs to send this query to just one of the gatekeepers in the cluster. Coupled with the new **zone cluster remote** CLI, it can round-robin between the gatekeepers in the cluster and not attempt to send LRQ to another gatekeeper in the cluster if it receives a reject from any one.
- In case a gateway was moved to an alternate gatekeeper, it always tries to register to that gatekeeper unless you issue a **no gateway** and then a **gateway** command. When the endpoint's primary gatekeeper is back online, the endpoint does not re-register to it unless the endpoint lost communication with the alternate gatekeeper. It continues to use the alternate gatekeeper for its call routing information.

debug and show Commands for Gatekeeper Clusters

The debugs below demonstrate how gatekeepers can join a cluster and how they share the information about their endpoints. **show** commands are used to show how to monitor the cluster. The debugs used are **debug gatekeeper gup asn1** and **debug h225 asn1**. This is based on the topology and configuration mentioned above.

Debugs from "gkb-1" when it was First to Join the Cluster

```
Mar  1 08:15:08.348: gk_gup_listen(): listening port = 11007

!--- Opens a TCP port (here it is 11007) to listen to GUP messages.

Mar  1 08:15:08.348: gk_gup_listen(): listening fd = 0
```

Mar 1 08:15:38.351: H225 NONSTD OUTGOING PDU ::=

value GRQnonStandardInfo ::=

!--- The non-standard data that is in the GRQ.

```
{
  gupAddress
  {
    ip 'AC100D29'H
```

!--- Listening IP address 172.16.13.41.

```
    port 11007
```

!--- Listening TCP port 11007.

```
  }
}
```

Mar 1 08:15:38.351: H225 NONSTD OUTGOING ENCODE BUFFER::= 40
AC100D29 2AFF

Mar 1 08:15:38.351:

Mar 1 08:15:38.351: RAS OUTGOING PDU ::=

value RasMessage ::= **gatekeeperRequest** :

!--- GRQ with the non-standard is sent out.

```
{
  requestSeqNum 59
  protocolIdentifier { 0 0 8 2250 0 3 }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '40AC100D292AFF'H
  }
  rasAddress ipAddress :
  {
    ip 'AC100D29'H
    port 1719
  }
  endpointType
  {
    vendor
    {
      vendor
      {
        t35CountryCode 181
        t35Extension 0
        manufacturerCode 18
      }
    }
  }
  mc FALSE
  undefinedNode FALSE
}
```

```
Mar 1 08:15:38.359: RAS OUTGOING ENCODE BUFFER ::= 01 00003A06 0008914A
000340B5 00001207 40AC100D 292AFF00 AC100D29 06B72000 B5000012 00
Mar 1 08:15:38.359:
```

Debugs from "gkb-2" when it Joined the Cluster after "gkb-1"

```
Mar 1 08:16:38.878: gk_gup_listen(): listening port = 11006
```

```
!--- Opens a TCP port (here it is 11006) to listen to GUP messages.
```

```
Mar 1 08:16:38.878: gk_gup_listen(): listening fd = 0
```

```
Mar 1 08:17:08.385: RAS INCOMING ENCODE BUFFER ::= 01 00003D06 0008914A
000340B5 00001207 40AC100D 292AFF00 AC100D29 06B72000 B5000012 00
```

```
Mar 1 08:17:08.385:
```

```
Mar 1 08:17:08.385: RAS INCOMING PDU ::=
```

```
value RasMessage ::= gatekeeperRequest :
```

```
!--- GRQ message is received from gkb-1 gatekeeper with non-standard information.
```

```
{
  requestSeqNum 62
  protocolIdentifier { 0 0 8 2250 0 3 }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '40AC100D292AFF'H
  }
  rasAddress ipAddress :
  {
    ip 'AC100D29'H
```

```
!--- RAS IP address 172.16.13.41 used gkb-1.
```

```
port 1719
```

```
!--- RAS TCP port used by gkb-1.
```

```

}
endpointType
{
  vendor
  {
    vendor
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
  }
  mc FALSE
  undefinedNode FALSE
}
}
```

```
Mar 1 08:17:08.393: H225 NONSTD INCOMING ENCODE BUFFER ::= 40
```

```
AC100D29 2AFF
Mar  1 08:17:08.393:
Mar  1 08:17:08.393: H225 NONSTD INCOMING PDU ::=
```

```
value GRQnonStandardInfo ::=
```

```
!--- gkb-2 extracts the non-standard data from the GRQ.
```

```
{
  gupAddress
  {
    ip 'AC100D29'H
```

```
!--- GUP IP address 172.16.13.41 used by gkb-1.
```

```
    port 11007
```

```
!--- GUP TCP port 11007 used by gkb-1.
```

```
  }
}
```

```
Mar  1 08:17:08.393: check_connection: checking connection to 172.16.13.41:11007
Mar  1 08:17:08.393: gk_gup_connect(): initiating connection
Mar  1 08:17:08.393: gup_connect: connecting to 172.16.13.41:11007
```

```
!--- A GUP connection is established, and updates follow.
```

```
Mar  1 08:17:08.393: gup_connect, fd = 1
Mar  1 08:17:08.401: GUP OUTGOING PDU ::=
```

```
value GUP_Information ::=
```

```
!--- GUP announcement is sent to alternate GK gkb-1.
```

```
{
  protocolIdentifier { 1 2 840 113548 10 0 0 2 }
  message announcementIndication :
  {
    announcementInterval 30
    endpointCapacity 100000
    callCapacity 100000
    hostName '676B622D32'H
    percentMemory 8
```

```
!--- Below is information about the status of gkb-2.
```

```
    percentCPU 0
    currentCalls 0
    currentEndpoints 0
    zoneInformation
    {
      {
        gatekeeperIdentifier {"gkb-2"}
        altGKIdentifier {"gkb-1"}
        totalBandwidth 0
        interzoneBandwidth 0
        remoteBandwidth 0
      }
    }
  }
}
```

```

Mar  1 08:17:08.405: GUP OUTGOING ENCODE BUFFER ::= 00 0A2A8648
86F70C0A 00000220 001E8001 86A08001 86A00467 6B622D32 10000000 00014200
0067006B 0062002D 00320800 67006B00 62002D00 31000000 000000
Mar  1 08:17:08.409:
Mar  1 08:17:08.409: Sending GUP ANNOUNCEMENT INDICATION to
172.16.13.41
Mar  1 08:17:08.413: GUP INCOMING ENCODE BUFFER ::= 00 0A2A8648
86F70C0A 00000220 001E8001 86A08001 86A00467 6B622D31 32000000 00014200
0067006B 0062002D 00310800 67006B00 62002D00 32000000 000000
Mar  1 08:17:08.413:
Mar  1 08:17:08.413: GUP INCOMING PDU ::=

```

```
value GUP_Information ::=
```

```
!--- GUP announcement is received from alternate GK gkb-1.
```

```

{
  protocolIdentifier { 1 2 840 113548 10 0 0 2 }
  message announcementIndication :
  {
    announcementInterval 30
    endpointCapacity 100000
    callCapacity 100000
    hostName '676B622D31'H
    percentMemory 25
  }
}

```

```
!--- Below is information about the status of gkb-1.
```

```

percentCPU 0
currentCalls 0
currentEndpoints 0
zoneInformation
{
  {
    gatekeeperIdentifier {"gkb-1"}
    altGKIdentifier {"gkb-2"}
    totalBandwidth 0
    interzoneBandwidth 0
    remoteBandwidth 0
  }
}
}

```

```
Mar  1 08:17:08.421: Received GUP ANNOUNCEMENT INDICATION from 172.16.13.41
```

With the **show gatekeeper endpoint** command, there are no registered endpoints. The output is as follows:

```

gkb-1#show gatekeeper endpoints
          GATEKEEPER ENDPOINT REGISTRATION
          =====
CallSignalAddr  Port  RASSignalAddr  Port  Zone Name          Type  Flags
-----
Total number of active registrations = 0

gkb-2# show gatekeeper endpoints
          GATEKEEPER ENDPOINT REGISTRATION
          =====
CallSignalAddr  Port  RASSignalAddr  Port  Zone Name          Type  Flags
-----

```

Total number of active registrations = 0

gkb-2#

Debugs when an Endpoint Registers with One of the Gatekeepers in the Cluster

This debug is taken from "gkb-1" gatekeeper. The endpoint is "gwb-1" registering to the "gkb-1" gatekeeper with **debug h225 ans1** and **debug ras** turned ON.

```
Mar  1 08:22:47.396: RAS INCOMING ENCODE BUFFER ::= 00 A00AAD06
0008914A 000300AC 100D17E0 7D088001 3C050401 00205002 00006700 6B006200
2D003101 40040067 00770062 002D0031
Mar  1 08:22:47.396:
Mar  1 08:22:47.396: RAS INCOMING PDU ::=
```

```
value RasMessage ::= gatekeeperRequest :
```

```
!--- GRQ is received from "gwb-1" gateway.
```

```
{
  requestSeqNum 2734
  protocolIdentifier { 0 0 8 2250 0 3 }
  rasAddress ipAddress :
  {
    ip 'AC100D17'H
```

```
!--- gwb-1 IP address (172.16.13.23).
```

```
port 57469
```

```
!--- gwb-1 TCP port 57469.
```

```

}
endpointType
{
  gateway
  {
    protocol
    {
      voice :
      {
        supportedPrefixes
        {
          {
            prefix e164 : "2#"
          }
        }
      }
    }
  }
}
mc FALSE
undefinedNode FALSE
}
gatekeeperIdentifier {"gkb-1"}
endpointAlias
{
  h323-ID : {"gwb-1"}
}
}
```

Mar 1 08:22:47.404: RAS OUTGOING PDU ::=

value RasMessage ::= **gatekeeperConfirm** :

!--- GCF is sent back with alternate gatekeepers included.

```
{
  requestSeqNum 2734
  protocolIdentifier { 0 0 8 2250 0 3 }
  gatekeeperIdentifier {"gkb-1"}
  rasAddress ipAddress :
  {
    ip 'AC100D29'H
```

!--- Gatekeeper gkb-1 IP address (172.16.13.41).

```
    port 1719
  }
  alternateGatekeeper
```

!--- List of alternate gatekeepers, here is "gkb-2" only.

```
{
  {
    rasAddress ipAddress :
    {
      ip 'AC100D10'H
```

!--- Alternate gatekeeper gkb-2 IP address (172.16.13.16)

```
    port 1719
  }
  gatekeeperIdentifier {"gkb-2"}
  needToRegister TRUE
  priority 0
}
}
```

Mar 1 08:22:47.412: RAS OUTGOING ENCODE BUFFER ::= 06 800AAD06
0008914A 00030800 67006B00 62002D00 3100AC10 0D2906B7 0D001401 40AC100D
1006B708 0067006B 0062002D 003280

Mar 1 08:22:47.412:

Mar 1 08:22:47.432: RAS INCOMING ENCODE BUFFER ::= 0E C00AAE06
0008914A 00038001 00AC100D 1706B801 00AC100D 17E07D08 80013C05 04010020
50000140 04006700 77006200 2D003108 0067006B 0062002D 003100B5 00001212
8B000200 3B010001 000180

Mar 1 08:22:47.432:

Mar 1 08:22:47.436: RAS INCOMING PDU ::=

value RasMessage ::= **registrationRequest** :

!--- RRQ is received from "gwb-1" gateway.

```
{
  requestSeqNum 2735
  protocolIdentifier { 0 0 8 2250 0 3 }
  discoveryComplete TRUE
  callSignalAddress
  {
    ipAddress :
```

```
ip 'AC100D17'H
```

```
!--- Gateway gwb-1 IP address (172.16.13.23).
```

```
    port 1720
  }
}
rasAddress
{
  ipAddress :
  {
    ip 'AC100D17'H
    port 57469
  }
}
terminalType
{
  gateway
  {
    protocol
    {
      voice :
      {
        supportedPrefixes
        {
          {
            prefix e164 : "2#"
          }
        }
      }
    }
  }
  mc FALSE
  undefinedNode FALSE
}
terminalAlias
{
  h323-ID : {"gwb-1"}
}
gatekeeperIdentifier {"gkb-1"}
endpointVendor
{
  vendor
  {
    t35CountryCode 181
    t35Extension 0
    manufacturerCode 18
  }
}
timeToLive 60
keepAlive FALSE
willSupplyUUIEs FALSE
maintainConnection TRUE
}
```

```
Mar  1 08:22:47.448: GUP OUTGOING PDU ::=
```

```
value GUP_Information ::=
```

```
!--- A GUP registration indicates a message is sent to "gkb-2" to inform it
!--- about the new registered endpoint.
```

```
{
  protocolIdentifier { 1 2 840 113548 10 0 0 2 }
  message registrationIndication :
  {
    version 3
    callSignalAddress
    {
      ipAddress :
      {
        ip 'AC100D17'H
```

!--- Gateway gwb-1 IP address (172.16.13.23).

```
      port 1720
    }
  }
  rasAddress
  {
    ipAddress :
    {
      ip 'AC100D17'H
```

!--- Gateway gwb-1 IP address (172.16.13.23).

```
      port 57469
    }
  }
  terminalType
  {
    vendor
    {
      vendor
      {
        t35CountryCode 181
        t35Extension 0
        manufacturerCode 18
      }
    }
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            {
              prefix e164 : "2#*"
            }
          }
        }
      }
    }
    mc FALSE
    undefinedNode FALSE
  }
  terminalAlias
  {
    h323-ID : {"gwb-1"}
```

!--- Name/ID of the new endpoint which has just registered.

```
  }
  gatekeeperIdentifier {"gkb-1"}
```

!--- Name/ID of the gatekeeper which the new endpoint(gwb-1) has registered to.

```
resourceIndicator
{
  almostOutOfResources FALSE
}
}
```

```
Mar  1 08:22:47.460: GUP OUTGOING ENCODE BUFFER ::= 00 0A2A8648
86F70COA 00000232 020100AC 100D1706 B80100AC 100D17E0 7D2800B5
00001240 013C0505 01004050 10000140 04006700 77006200 2D003108 0067006B
0062002D 003100
Mar  1 08:22:47.464:
Mar  1 08:22:47.464: Sending GUP REGISTRATION INDICATION to 172.16.13.16
Mar  1 08:22:47.464: RAS OUTGOING PDU ::=
```

```
value RasMessage ::= registrationConfirm :
```

!--- RCF is sent back to "gwb-1" gateway.

```
{
  requestSeqNum 2735
  protocolIdentifier { 0 0 8 2250 0 3 }
  callSignalAddress
  {
  }
  terminalAlias
  {
    h323-ID : {"gwb-1"}
  }
  gatekeeperIdentifier {"gkb-1"}
  endpointIdentifier {"61809DB800000001"}
  alternateGatekeeper
  {
    {
      rasAddress ipAddress :
      {
        ip 'AC100D10'H
        port 1719
      }
      gatekeeperIdentifier {"gkb-2"}
      needToRegister TRUE
      priority 0
    }
  }
  timeToLive 60
  willRespondToIRR FALSE
  maintainConnection TRUE
}
```

```
Mar  1 08:22:47.472: RAS OUTGOING ENCODE BUFFER ::= 12 C00AAE06
0008914A 00030001 40040067 00770062 002D0031 08006700 6B006200 2D00311E
00360031 00380030 00390044 00420038 00300030 00300030 00300030 00300031
0F8A1401 40AC100D 1006B708 0067006B 0062002D 00328002 003B0100 0180
Mar  1 08:22:47.472:
```

The output above contains the **show gatekeeper endpoint** command output after all gateways register in the cluster. The above debugs happen for each endpoint registration. After all three gateways in the cluster register, the **show gatekeeper endpoint** command on both gatekeepers is as follows:

```
gkb-1#show gatekeeper endpoints
```

```
      GATEKEEPER ENDPOINT REGISTRATION
      =====
CallSignalAddr  Port  RASignalAddr  Port  Zone Name      Type      Flags
-----
172.16.13.23    1720  172.16.13.23  57469  gkb-1          VOIP-GW
      H323-ID: gwb-1
172.16.13.26    1720  172.16.13.26  49801  gkb-1          VOIP-GW
      H323-ID: gwb-2
172.16.13.42    1720  172.16.13.42  57216  gkb-1          VOIP-GW          A
```

```
!--- A flag set.
```

```
      H323-ID: gwb-3
Total number of active registrations = 3
```

```
gkb-2# show gatekeeper endpoints
```

```
      GATEKEEPER ENDPOINT REGISTRATION
      =====
CallSignalAddr  Port  RASignalAddr  Port  Zone Name      Type      Flags
-----
172.16.13.23    1720  172.16.13.23  57469  gkb-2          VOIP-GW          A
```

```
!--- A flag set.
```

```
      H323-ID: gwb-1
172.16.13.26    1720  172.16.13.26  49801  gkb-2          VOIP-GW          A
```

```
!--- A flag set.
```

```
      H323-ID: gwb-2
172.16.13.42    1720  172.16.13.42  57216  gkb-2          VOIP-GW
      H323-ID: gwb-3
```

```
Total number of active registrations = 3
```

```
!--- The "A" under the flag field means that the gatekeeper is an alternate one
!--- for this endpoint.
```

Debugs for when an Endpoint with Active Call Moves to Alternate Gatekeeper

These are the debugs from a gatekeeper that starts when the call is requested and goes until it is disconnected. Some of the unnecessary debug messages have been omitted. These debugs are from the "gkb-1" gatekeeper. The call was placed by gwa-1 registered to "gka-1" to another gateway (gwb-1) in the remote zone cluster. The debugs demonstrate how an active call flow is tracked from the primary gatekeeper to the alternate gatekeeper as the primary goes down.

```
Mar  2 23:59:26.714: RecvUDP_IPSockData  successfully rcvd message of length 84
from 172.16.13.35:1719
Mar  2 23:59:26.714: RAS INCOMING ENCODE BUFFER ::= 4A 80080801 01806986
40B50000 122C8286 B01100C8 C66C7D1
6 8011CC80 0D882828 5B8DF601 80140204 8073B85A 5C564004 00670077
0061002D 003100AC 100D2306 B70B800D 01400
400 67006B00 61002D00 310180
Mar  2 23:59:26.714:
Mar  2 23:59:26.714: RAS INCOMING PDU ::=

value RasMessage ::= locationRequest :
```

```
!--- LRQ is received from "gka-1" gatekeeper from domain A.
```

```

{
  requestSeqNum 2057
  destinationInfo
  {
    e164 : "3653"
  }
}

!--- E164 number to be resolved by the this gatekeeper.

}
nonStandardData
{
  nonStandardIdentifier h221NonStandard :
  {
    t35CountryCode 181
    t35Extension 0
    manufacturerCode 18
  }
  data '8286B01100C8C66C7D168011CC800D8828285B8D...'H
}
replyAddress ipAddress :
{
  ip 'AC100D23'H
  port 1719
}
sourceInfo
{
  h323-ID : {"gka-1"}
}
canMapAlias TRUE
}

```

```

Mar  2 23:59:26.722: LRQ (seq# 2057) rcvd
Mar  2 23:59:26.722: H225 NONSTD INCOMING ENCODE BUFFER::= 82 86B01100
C8C66C7D 168011CC 800D8828 285B8DF6
01801402 048073B8 5A5C5640 04006700 77006100 2D0031
Mar  2 23:59:26.722:
Mar  2 23:59:26.722: H225 NONSTD INCOMING PDU ::=

```

!--- LRQ nonStandardInfo decoded output.

```

value LRQnonStandardInfo ::=
{
  ttl 6
  nonstd-callIdentifier
  {
    guid 'C8C66C7D168011CC800D8828285B8DF6'H
  }
  callingOctet3a 128
  gatewaySrcInfo
  {
    e164 : "4085272923",
    h323-ID : {"gwa-1"}
  }
}
}

```

```

parse_lrq_nonstd: LRQ Nonstd decode succeeded, remlen = 84
Mar  2 23:59:26.726: H225 NONSTD OUTGOING PDU ::=

```

!--- LCF nonStandardInfo reply back to the LRQ nonStandardInfor.

```

value LCFnonStandardInfo ::=
{
  termAlias

```



```

    port 51874
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '00014004006700770062002D0031080067006B00...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            }
          }
        }
      }
    }
    mc FALSE
    undefinedNode FALSE
  }
}

```

```

Mar  2 23:59:26.742: RAS OUTGOING ENCODE BUFFER ::= 4F 080800AC
100D1706 B800AC10 0D17CAA2 40B50000 1239000
1 40040067 00770062 002D0031 08006700 6B006200 2D003101 10014004 00670077
0062002D 003100AC 100D1706 B8000
000 00000000 00000010 40080880 013C0501 0000
Mar  2 23:59:26.746:
Mar  2 23:59:26.746:  IPSOCK_RAS_sendto:   msg length 91 from 172.16.13.41:1719 to
172.16.13.35: 1719
Mar  2 23:59:26.746:           RASLib::RASsendLCF: LCF (seq# 2057) sent to 172.16.13.35
Mar  2 23:59:26.798:  RecvUDP_IPSockData  successfully rcvd message of length 129
from 172.16.13.23:51874
Mar  2 23:59:26.798: RAS INCOMING ENCODE BUFFER ::= 27 98172700 F0003600
31003900 36003200 39003600 38003000
0 30003000 30003000 30003000 31010180 69860204 8073B85A 5C564004 00670077
0061002D 003100AC 100D0F2A FA400
500 000E40B5 00001207 80000008 800180C8 C66C7D16 8011CC80 0C882828
5B8DF645 60200180 1100C8C6 6C7D1680 11C
C800D 8828285B 8DF60100
Mar  2 23:59:26.802:
Mar  2 23:59:26.802: RAS INCOMING PDU ::=

```

value RasMessage ::= **admissionRequest** :

!--- "gwb-1" sent answerCall ARQ.

```

{
  requestSeqNum 5928
  callType pointToPoint : NULL
  callModel direct : NULL
  endpointIdentifier {"6196296800000001"}
  destinationInfo
  {

```

```
e164 : "3653"
```

```
!--- E164 number the caller is trying to reach.
```

```
}  
srcInfo  
{  
  e164 : "4085272923",
```

```
!--- Caller information.
```

```
  h323-ID : {"gwa-1"}  
}  
srcCallSignalAddress ipAddress :  
{  
  ip 'AC10D0F'H
```

```
!--- Originating gateway (gwa-1) IP address and port.
```

```
  port 11002  
}  
bandwidth 1280  
callReferenceValue 14
```

```
!--- Remember call reference, since it is used when the call  
!--- is disconnected when sending the DRQ.
```

```
nonStandardData  
{  
  nonStandardIdentifier h221NonStandard :  
  {  
    t35CountryCode 181  
    t35Extension 0  
    manufacturerCode 18  
  }  
  data '80000008800180'H  
}  
conferenceID 'C8C66C7D168011CC800C8828285B8DF6'H  
activeMC FALSE  
answerCall TRUE  
canMapAlias TRUE  
callIdentifier  
{  
  guid 'C8C66C7D168011CC800D8828285B8DF6'H  
}  
willSupplyUUIEs FALSE  
}
```

```
Mar 2 23:59:26.810: ARQ (seq# 5928) rcvd  
Mar 2 23:59:26.810: H225 NONSTD INCOMING ENCODE BUFFER ::= 80 00000880 0180  
Mar 2 23:59:26.810:  
Mar 2 23:59:26.810: H225 NONSTD INCOMING PDU ::=
```

```
value ARQnonStandardInfo ::=  
{  
  sourceAlias  
  {  
  }  
  sourceExtAlias  
  {  
  }  
  callingOctet3a 128  
}
```

parse_arq_nonstd: ARQ Nonstd decode succeeded, remlen = 129
Mar 2 23:59:26.814: RAS OUTGOING PDU ::=

value RasMessage ::= **admissionConfirm** :

!--- ACF is sent back to "gwb-1".

```
{
  requestSeqNum 5928
  bandwidth 1280
  callModel direct : NULL
  destCallSignalAddress ipAddress :
  {
    ip 'AC100D17'H
```

!--- gwb-1 IP address (172.16.13.23).

```
    port 1720
  }
  irrFrequency 240
  willRespondToIRR FALSE
  uuiesRequested
  {
    setup FALSE
    callProceeding FALSE
    connect FALSE
    alerting FALSE
    information FALSE
    releaseComplete FALSE
    facility FALSE
    progress FALSE
    empty FALSE
  }
}
```

Mar 2 23:59:26.818: RAS OUTGOING ENCODE BUFFER ::= 2B 00172740 050000AC
100D1706 B800EF1A 00C00100 020000

Mar 2 23:59:26.818:

Mar 2 23:59:26.818: IPSOCK_RAS_sendto: msg length 24 from 172.16.13.41:1719 to
172.16.13.23: 51874

Mar 2 23:59:26.822: RASLib::RASSendACF: **ACF (seq# 5928) sent to
172.16.13.23**

Mar 2 23:59:36.046: GUP OUTGOING PDU ::=

value GUP_Information ::=

*!--- GUP update is sent out and it contains the information
!--- about the last call that is still active.*

```
{
  protocolIdentifier { 1 2 840 113548 10 0 0 2 }
  message announcementIndication :
  {
    announcementInterval 30
    endpointCapacity 46142
    callCapacity 68793
    hostName '676B622D31'H
    percentMemory 25
    percentCPU 0
    currentCalls 1
    currentEndpoints 2
    zoneInformation
```

```

    {
      {
        gatekeeperIdentifier {"gkb-1"}
        altGKIdentifier {"gkb-2"}
        totalBandwidth 1280

        !--- 1280 is 128 Kbps of total bandwidth used for the zone.

        interzoneBandwidth 1280
        remoteBandwidth 1280
      }
    }
  }
}

```

```

Mar  2 23:59:36.050: GUP OUTGOING ENCODE BUFFER ::= 00 0A2A8648
86F70C0A 00000220 001E40B4 3E80010C B904676
B 622D3132 00010002 01420000 67006B00 62002D00 31080067 006B0062 002D0032
40050040 05004005 00
Mar  2 23:59:36.054:
Mar  2 23:59:36.054: Sending GUP ANNOUNCEMENT INDICATION to 172.16.13.16

```

Note: At this point "gkb-1" is shutdown. This is allowed (even if it has an active call) because there is an alternate gatekeeper for that zone.

URQ messages are sent to all endpoints registered with the "gkb-1". These endpoints are "gwb-1" and "gwb-2" gateways. These gateways confirm the URQ by sending back UCFs. Also, gkb-1 sends a GUP unregistration indication message to the cluster alternate gatekeeper and then it closes the GUP connection.

```

Mar  2 23:59:55.914: RAS OUTGOING PDU ::=

value RasMessage ::= unregistrationRequest :
  {
    requestSeqNum 79
    callSignalAddress
    {
      ipAddress :
      {
        ip 'AC100D17'H

        !--- UnregistrationRequest (URQ) sent to gwb-1 (172.16.13.23).

        port 1720
      }
    }
  }
}

```

```

Mar  2 23:59:55.914: RAS OUTGOING ENCODE BUFFER ::= 18 00004E01
00AC100D 1706B8
Mar  2 23:59:55.914:
Mar  2 23:59:55.914: IPSOCK_RAS_sendto: msg length 12 from 172.16.13.41:1719 to
172.16.13.23: 51874
Mar  2 23:59:55.914: RASLib::RASSendURQ: URQ (seq# 79) sent to
172.16.13.23
Mar  2 23:59:55.918: RAS OUTGOING PDU ::=

value RasMessage ::= unregistrationRequest :
  {
    requestSeqNum 80
    callSignalAddress

```

```

    {
      ipAddress :
      {
        ip 'AC100D1A'H
      }
    }
  }
}

```

!--- URQ sent to gwb-2 (172.16.13.26).

```

Mar  2 23:59:55.918: RAS OUTGOING ENCODE BUFFER ::= 18 00004F01
00AC100D 1A06B8
Mar  2 23:59:55.918:
Mar  2 23:59:55.918: IPSOCK_RAS_sendto:  msg length 12 from 172.16.13.41:1719 to
172.16.13.26: 50041
Mar  2 23:59:55.918:          RASLib::RASsendURQ: URQ (seq# 80) sent to
172.16.13.26
Mar  2 23:59:55.922:  RecvUDP_IPSockData  successfully rcvd message of length 3
from 172.16.13.23:51874
Mar  2 23:59:55.922: RAS INCOMING ENCODE BUFFER ::= 1C 004E
Mar  2 23:59:55.922:
Mar  2 23:59:55.922: RAS INCOMING PDU ::=

```

```

value RasMessage ::= unregistrationConfirm :
{
  requestSeqNum 79
}

```

```

Mar  2 23:59:55.922: UCF (seq# 79) rcvd
Mar  2 23:59:55.926:  RecvUDP_IPSockData  successfully rcvd message of length 3
from 172.16.13.26:50041
Mar  2 23:59:55.926: RAS INCOMING ENCODE BUFFER ::= 1C 004F
Mar  2 23:59:55.926:
Mar  2 23:59:55.926: RAS INCOMING PDU ::=

```

```

value RasMessage ::= unregistrationConfirm :
{
  requestSeqNum 80
}

```

```

Mar  2 23:59:55.926: UCF (seq# 80) rcvd
Mar  3 00:00:01.922: GUP OUTGOING PDU ::=

```

```

value GUP_Information ::=
{
  protocolIdentifier { 1 2 840 113548 10 0 0 2 }
  message unregistrationIndication :
  {
    reason explicitUnregister : NULL
    callSignalAddress
    {
      ipAddress :
      {
        ip 'AC100D17'H
      }
    }
  }
}

```

*!--- GUP UnregistrationIndication sent to alternate gatekeeper
!--- gkb-2 (172.16.13.16) in the cluster.*

```

        port 1720
    }
}
}
}

```

```

Mar  3 00:00:01.922: GUP OUTGOING ENCODE BUFFER ::= 00 0A2A8648
86F70C0A 00000238 000100AC 100D1706 B8
Mar  3 00:00:01.926:
Mar  3 00:00:01.926: Sending GUP UNREGISTRATION INDICATION to
172.16.13.16
Mar  3 00:00:01.934: gk_gup_close_connection(): closing connection to 172.16.13.16
Mar  3 00:00:01.934: gk_gup_close_listen(): closing listen

```

Here is the debug from "gkb-2". The debugs that show the registration of the moved endpoint "gwb-1" and "gwb-2" are omitted, since they look like normal registration. The purpose here is to show the acceptance of the DRQ of the active call on "gwb-1" when it is moved to "gkb-2".

```

Mar  3 00:00:24.307: RecvUDP_IPSockData  successfully rcvd message of length 77
from 172.16.13.23:51874
Mar  3 00:00:24.307: RAS INCOMING ENCODE BUFFER ::= 3E 172C1E00 36003100
38003400 44004300 34004300 30003000 30003000 3
0003000 300033C8 C66C7D16 8011CC80 0C882828 5B8DF600 0E21A100 1100C8C6
6C7D1680 11CC800D 8828285B 8DF60180
Mar  3 00:00:24.311:
Mar  3 00:00:24.311: RAS INCOMING PDU ::=

```

```

value RasMessage ::= disengageRequest :

```

```

!--- DRQ is received with call reference 14 and normal clearing
!--- disconnect cause code.
!--- This information is passed to the accounting server and the GKTMP
!--- server if configured.

```

```

{
  requestSeqNum 5933
  endpointIdentifier {"6184DC4C00000003"}
  conferenceID 'C8C66C7D168011CC800C8828285B8DF6'H
  callReferenceValue 14
  disengageReason normalDrop : NULL
  callIdentifier
  {
    guid 'C8C66C7D168011CC800D8828285B8DF6'H
  }
  answeredCall TRUE
}

```

```

Mar  3 00:00:24.311: DRQ (seq# 5933) rcvd
Mar  3 00:00:24.315: RAS OUTGOING PDU ::=

```

```

value RasMessage ::= disengageConfirm :

```

```

!--- DCF is sent to "gwb-1".

```

```

{
  requestSeqNum 5933
}

```

```

Mar  3 00:00:24.315: RAS OUTGOING ENCODE BUFFER ::= 40 172C

```

```
Mar  3 00:00:24.315:
Mar  3 00:00:24.315:  IPSOCK_RAS_sendto:   msg length 3 from 172.16.13.16:1719 to
172.16.13.23: 51874
Mar  3 00:00:24.315:          RASLib::RASSendDCF: DCF (seq# 5933) sent to
172.16.13.23
gkb-2#
```

Cisco Gateway Fail Over to Alternate Gatekeeper

By default, Cisco Gateways send a light weight RRQ every 45 seconds. In case the gatekeeper did not send any URQ to the gateway (due to a broken routing issue, for example), the gateway (upon not hearing a RCF or RRJ for its light weight RRQ) tries twice with five seconds between each. If the third attempt fails, it immediately considers the gatekeeper as dead and registers with the alternate gatekeeper that uses RRQ. In a scenario where the gateway starts the initial registration process with the gatekeeper, it sends out the GRQ to locate the gatekeeper IP address. If there is a GCF reply back, gateway sends the RRQ to the primary gatekeeper specified. If for any reason the gatekeeper rejects the registration request, the gateway does not try to contact its alternate gatekeeper. It starts this process (GRQ, GCF and RRQ) over again with the primary gatekeeper.

The gateway only contacts the alternative gatekeeper when the connectivity to the primary gatekeeper is lost and there is no reply back. If the primary gatekeeper does not reply back to the GRQ message when the gateway first sends out to discover the gatekeeper, then after three failed attempts (approximately five minutes per attempt), the gateway contacts the alternative gatekeeper. In a situation where the primary gatekeeper goes down after the gateway has registered with it, the gateway loses keepalives messages from the primary gatekeeper. After missing three consecutive keepalive messages, the gateway declares the primary gatekeeper as down, and it starts the registration process again.

Troubleshoot Issues with Alternate Endpoints

A calling endpoint can recover from a call setup failure by sending a setup message to one of the alternate endpoints. The call can fail for many reasons: gateway is down and gatekeeper is not aware of it at the time of sending the ACF or LCF, there are no resources on the gateway and did not report that to the gatekeeper, call fails because of an incorrect configuration on the main endpoint, and more.

Note: The originating endpoint only tries to contact the alternate gatekeepers if the call fails before the alert stage (alert or progress). If the calls fails due to user busy or no answer, the originating endpoint does not try any other alternates.

The gatekeeper learns about that alternate for a certain endpoint either by manual configuration using the gatekeeper CLI command **endpoint alt-ep** or from any received RAS messages. Cisco supports a maximum of 20 alternates for each endpoint, no matter how the gatekeeper learns them.

The issues that you need to take a look at include:

- If the gatekeeper has the correct alternate endpoint as desired.
- If the gatekeeper includes the alternate endpoints in its LCF or ACF RAS messages.
- If the OGW tries to contact the alternates in case the main destination endpoint fails.

To show how to troubleshoot these issues, the same topology as above is used with this change on "gkb-1" gatekeeper configuration to include two alternate gateways: "gwb-3" and "gwb-1" for the gateway "gwb-2". Here is the configuration of "gkb-1" gatekeeper:

```
!
gatekeeper
zone local gkb-1 domainB.com 172.16.13.41
```

```

zone remote gka-1 domainA.com 172.16.13.35 1719
zone cluster local gkb gkb-1
  element gkb-2 172.16.13.16 1719
!
gw-type-prefix 2#* default-technology
bandwidth total zone gkb-1 512
bandwidth session zone gkb-1 512
no shutdown
endpoint alt-ep h323id gwb-2 172.16.13.42

!--- 172.16.13.42 is gwb-3.

endpoint alt-ep h323id gwb-2 172.16.13.23

!--- 172.16.13.23 is gwb-1.

!

```

Verify the Gatekeeper has the Correct Alternate Endpoints

To see if the gatekeeper has the right alternate endpoints, use the **show gatekeeper endpoints alternates** command.

```

gkb-1#show gatekeeper endpoints alternates
GATEKEEPER ENDPOINT REGISTRATION
=====
CallSignalAddr      Port      RASSignalAddr      Port      Zone Name      Type
-----
172.16.13.23        1720      172.16.13.23        54670     gkb-1          VOIP-GW
  H323-ID: gwb-1
172.16.13.26        1720      172.16.13.26        57233     gkb-1          VOIP-GW
  H323-ID: gwb-2
  ALT_EP: 172.16.13.42 <1720>
           172.16.13.23 <1720>

!--- This shows the information about all collected endpoints.

172.16.13.42        1720      172.16.13.42        58430     gkb-1          VOIP-GW
  H323-ID: gwb-3
Total number of active registrations = 3

ALL CONFIGURED ALTERNATE ENDPOINTS

!--- Only manually configured.

=====
Endpoint H323 Id      RASSignalAddr      Port
-----
gwb-2                 172.16.13.42        1720
gwb-2                 172.16.13.23        1720
gkb-1#

```

Verify whether Gatekeeper Includes Alternate Endpoints in its LCF or ACF RAS Messages

To see if the gatekeeper sends the IP address for alternate endpoints, you can turn on **debug h225 asn1** and look at the ACF message or LCF. This is an example debug taken from "gkb-1".

```

Mar  3 04:12:47.676: H225 NONSTD OUTGOING ENCODE BUFFER::= 00 01400400
67007700 62002D00 32080067 00
6B0062 002D0031 01100140 04006700 77006200 2D003200 AC100D1A 06B80000

```

```
00000000 00000000
Mar  3 04:12:47.676:
Mar  3 04:12:47.676: RAS OUTGOING PDU ::=
```

```
value RasMessage ::= locationConfirm :
{
  requestSeqNum 2070
  callSignalAddress ipAddress :
  {
    ip 'AC100D1A'H
```

```
!--- This is IP address of main destination.
```

```
    port 1720
  }
  rasAddress ipAddress :
  {
    ip 'AC100D1A'H
    port 50041
  }
  nonStandardData
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '00014004006700770062002D0032080067006B00...'H
  }
  destinationType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            }
          }
        }
      }
    }
    mc FALSE
    undefinedNode FALSE
  }
  alternateEndpoints
```

```
!--- Alternate endpoints.
```

```
{
  {
    callSignalAddress
    {
      ipAddress :
      {
        ip 'AC100D2A'H
```

```
!--- This is the first alternate IP address (172.16.13.42 gw3).
```

```
    port 1720
  },      ipAddress :
  {
    ip 'AC100D17'H
```

!--- This is the second alternate IP address (172.16.13.23 gw-1).

```
    }
  }
}
port 1720
}
```

Verify whether OGW Tries to Contact Alternates in Case Main Destination Endpoint Fails

This section shows how the OGW reacts when it receives alternate endpoints in its ACF message. In this example the call is made to fail when it attempts to contact the main terminating endpoint (gw). Debugs to turn on here are **debug voip ccapi inout** and **debug h225 asn1**.

The first thing you see in the debug is the ccapi message that shows the originating telephony leg.

```
Mar  3 04:12:47.616: cc_api_call_setup_ind (vdbPtr=0x6264A60C,
callInfo={called=3653,called_oct3=0x8
0,calling=4085272923,calling_oct3=0x21,calling_oct3a=0x80,calling_xlated=false,subsc
riber_type_str=R
egularLine,fdest=1,peer_tag=5336, prog_ind=0},callID=0x62155454)
Mar  3 04:12:47.616: cc_api_call_setup_ind type 13 , prot 0
Mar  3 04:12:47.620: cc_process_call_setup_ind (event=0x6231C454)
Mar  3 04:12:47.620: >>>CCAPI handed cid 51 with tag 5336 to app "DEFAULT"
Mar  3 04:12:47.620: sess_appl: ev(24=CC_EV_CALL_SETUP_IND), cid(51), disp(0)
Mar  3 04:12:47.620: sess_appl: ev(SSA_EV_CALL_SETUP_IND), cid(51), disp(0)
Mar  3 04:12:47.620: ssaCallSetupInd
Mar  3 04:12:47.620: ccCallSetContext (callID=0x33, context=0x626EAC9C)
Mar  3 04:12:47.620: ssaCallSetupInd cid(51), st(SSA_CS_MAPPING),oldst(0),
ev(24)ev->e.evCallSetupIn
d.nCallInfo.finalDestFlag = 1
Mar  3 04:12:47.620: ssaCallSetupInd finalDest cllng(4085272923), cllcd(3653)
Mar  3 04:12:47.620: ssaCallSetupInd cid(51), st(SSA_CS_CALL_SETTING),oldst(0),
ev(24)dpMatchPeersMo
reArg result= 0
Mar  3 04:12:47.620: ssaSetupPeer cid(51) peer list: tag(3653) called number (3653)
Mar  3 04:12:47.620: ssaSetupPeer cid(51), destPat(3653), matched(4), prefix(),
peer(62663E7C), peer
->encapType (2)
Mar  3 04:12:47.620: ccCallProceeding (callID=0x33, prog_ind=0x0)
Mar  3 04:12:47.620: ccCallSetupRequest (Inbound call = 0x33, outbound peer =3653,
dest=,
    params=0x62327730 mode=0, *callID=0x62327A98, prog_ind = 0)
Mar  3 04:12:47.624: ccCallSetupRequest numbering_type 0x80
Mar  3 04:12:47.624: ccCallSetupRequest encapType 2 clid_restrict_disable 1
null_orig_clg 0 clid_tra
nsparent 0 callingNumber 4085272923

Mar  3 04:12:47.624: dest pattern 3653, called 3653, digit_strip 0
Mar  3 04:12:47.624: callingNumber=4085272923, calledNumber=3653,
redirectNumber= display_info= call
ing_oct3a=80
Mar  3 04:12:47.624: accountNumber=, finalDestFlag=1,
guid=2d3a.ac33.16a4.11cc.8068.8828.285b.8df6
Mar  3 04:12:47.624: peer_tag=3653
Mar  3 04:12:47.624: ccIFCallSetupRequestPrivate: (vdbPtr=0x621B2360, dest=,
callParams={called=3653
,called_oct3=0x80, calling=4085272923,calling_oct3=0x21, calling_xlated=false,
subscriber_type_str=
```

RegularLine, fdest=1, voice_peer_tag=3653},mode=0x0) vdbPtr type = 1

!--- The OGW establishes the second leg.

```
Mar 3 04:12:47.624: ccIFCallSetupRequestPrivate: (vdbPtr=0x621B2360, dest=,
callParams={called=3653
, called_oct3 0x80, calling=4085272923,calling_oct3 0x21, calling_xlated=false,
fdest=1, voice_pee
r_tag=3653}, mode=0x0, xltrc=-5)
Mar 3 04:12:47.624: ccSaveDialpeerTag (callID=0x33, dialpeer_tag=0xE45)
Mar 3 04:12:47.624: ccCallSetContext (callID=0x34, context=0x626EB9A4)
Mar 3 04:12:47.624: ccCallReportDigits (callID=0x33, enable=0x0)
Mar 3 04:12:47.624: cc_api_call_report_digits_done (vdbPtr=0x6264A60C,
callID=0x33, disp=0)
Mar 3 04:12:47.624: sess_appl: ev(52=CC_EV_CALL_REPORT_DIGITS_DONE),
cid(51), disp(0)
Mar 3 04:12:47.624: cid(51)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_REPORT_DIGITS_DONE)
oldst(SSA_CS_MAPPING)cfid(-1)csz(0)in(1)fDest(1)
Mar 3 04:12:47.624: -cid2(52)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_MAPPING)
Mar 3 04:12:47.624: ssaReportDigitsDone cid(51) peer list: (empty)
Mar 3 04:12:47.624: ssaReportDigitsDone callid=51 Reporting disabled.
Mar 3 04:12:47.628: H225 NONSTD OUTGOING PDU ::=
```

```
value ARQnonStandardInfo ::=
{
  sourceAlias
  {
  }
  sourceExtAlias
  {
  }
  callingOctet3a 128
  interfaceSpecificBillingId "ISDN-VOICE"
}
```

```
Mar 3 04:12:47.628: H225 NONSTD OUTGOING ENCODE BUFFER ::= 80 000008A0
01800B12 4953444E 2D564F49 43
45
Mar 3 04:12:47.628:
Mar 3 04:12:47.628: RAS OUTGOING PDU ::=
```

```
value RasMessage ::= admissionRequest :
```

!--- ARQ is sent to the gatekeeper.

```
requestSeqNum 2210
callType pointToPoint : NULL
callModel direct : NULL
endpointIdentifier {"81206D2C00000001"}
destinationInfo
{
  e164 : "3653"
}
srcInfo
{
  e164 : "4085272923",
  h323-ID : {"gwa-1"}
}
bandwidth 640
callReferenceValue 26
nonStandardData
{
```

```

nonStandardIdentifier h221NonStandard :
{
  t35CountryCode 181
  t35Extension 0
  manufacturerCode 18
}
data '80000008A001800B124953444E2D564F494345'H
}
conferenceID '2D3AAC3316A411CC80688828285B8DF6'H
activeMC FALSE
answerCall FALSE
canMapAlias TRUE
callIdentifier
{
  guid '2D3AAC3316A411CC80698828285B8DF6'H
}
willSupplyUUIEs FALSE
}

```

```

Mar 3 04:12:47.636: RAS OUTGOING ENCODE BUFFER ::= 27 8808A100 F0003800
31003200 30003600 44003200 4
3003000 30003000 30003000 30003000 31010180 69860204 8073B85A 5C564004
00670077 0061002D 00314002 80
001A40 B5000012 13800000 08A00180 0B124953 444E2D56 4F494345 2D3AAC33
16A411CC 80688828 285B8DF6 04E
02001 8011002D 3AAC3316 A411CC80 69882828 5B8DF601 00
Mar 3 04:12:47.640:
Mar 3 04:12:47.656: RAS INCOMING ENCODE BUFFER ::= 80 050008A1 2327
Mar 3 04:12:47.656:
Mar 3 04:12:47.656: RAS INCOMING PDU ::=

```

```

value RasMessage ::= requestInProgress :
{
  requestSeqNum 2210
  delay 9000
}

```

```

Mar 3 04:12:47.704: RAS INCOMING ENCODE BUFFER ::= 2B 0008A140
028000AC 100D1A06 B800EF1A 10C01201 1
0000200 AC100D2A 06B800AC 100D1706 B8010002 0000
Mar 3 04:12:47.704:
Mar 3 04:12:47.704: RAS INCOMING PDU ::=

```

```

value RasMessage ::= admissionConfirm :

```

!--- ACF is received.

```

{
  requestSeqNum 2210
  bandwidth 640
  callModel direct : NULL
  destCallSignalAddress ipAddress :

```

!--- Primary destination endpoint.

```

{
  ip 'AC100D1A'H
  port 1720
}
irrFrequency 240
alternateEndpoints

```


h323-message-body setup :

!--- H.225 setup sent to primary endpoint.

```
{
  protocolIdentifier { 0 0 8 2250 0 2 }
  sourceAddress
  {
    h323-ID : {"gwa-1"}
  }
  sourceInfo
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            {
              prefix e164 : "1#"
            }
          }
        }
      }
    }
    mc FALSE
    undefinedNode FALSE
  }
  activeMC FALSE
  conferenceID '2D3AAC3316A411CC80688828285B8DF6'H
  conferenceGoal create : NULL
  callType pointToPoint : NULL
  sourceCallSignalAddress ipAddress :
  {
    ip 'AC100D0F'H
    port 11025
  }
  callIdentifier
  {
    guid '2D3AAC3316A411CC80698828285B8DF6'H
  }
  fastStart
  {
    '0000000C6013800A04000100AC100D0F47F1'H,
    '400000060401004C6013801114000100AC100D0F...'H
  }
  mediaWaitForConnect FALSE
  canOverlapSend FALSE
}
h245Tunneling TRUE
nonStandardControl
{
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '6001020001041F04038090A31803A983816C0C21...'H
  }
}
```

```
}  
}
```

```
Mar 3 04:12:47.740: H225.0 OUTGOING ENCODE BUFFER ::= 20 A0060008  
914A0002 01400400 67007700 61002D0  
0 31088001 3C050401 00204000 2D3AAC33 16A411CC 80688828 285B8DF6  
00451C07 00AC100D 0F2B1111 002D3AAC  
3316A411 CC806988 28285B8D F6320212 0000000C 6013800A 04000100  
AC100D0F 47F11D40 00000604 01004C60  
13801114 000100AC 100D0F47 F000AC10 0D0F47F1 01000100 06A00180  
2D0140B5 00001226 60010200 01041F04 0  
38090A3 1803A983 816C0C21 80343038 35323732 39323370 05803336 3533  
Mar 3 04:12:47.744:  
Mar 3 04:12:47.760: H225.0 INCOMING ENCODE BUFFER ::= 25 80060008  
914A0004 11001100 2D3AAC33 16A411C  
C 80698828 285B8DF6 10800180  
Mar 3 04:12:47.760:  
Mar 3 04:12:47.760: H225.0 INCOMING PDU ::=
```

```
value H323_UserInformation ::=  
{  
  h323-uu-pdu  
  {  
    h323-message-body releaseComplete :
```

```
!-- First setup message failed.
```

```
{  
  protocolIdentifier { 0 0 8 2250 0 4 }  
  callIdentifier  
  {  
    guid '2D3AAC3316A411CC80698828285B8DF6'H  
  }  
}  
h245Tunneling TRUE  
}
```

```
Mar 3 04:12:47.776: H225 NONSTD OUTGOING PDU ::=
```

```
value H323_UU_NonStdInfo ::=  
{  
  version 2  
  protoParam qsigNonStdInfo :  
  {  
    iei 4  
    rawMesg '04038090A31803A983816C0C2180343038353237...'H  
  }  
}
```

```
Mar 3 04:12:47.776: H225 NONSTD OUTGOING ENCODE BUFFER ::= 60 01020001  
041F0403 8090A318 03A98381 6C  
0C2180 34303835 32373239 32337005 80333635 33  
Mar 3 04:12:47.776:  
Mar 3 04:12:47.776: H225.0 OUTGOING PDU ::=
```

```
value H323_UserInformation ::=  
{  
  h323-uu-pdu  
  {
```

h323-message-body setup :

!--- Second setup sent to alternate endpoint.

```
{
  protocolIdentifier { 0 0 8 2250 0 2 }
  sourceAddress
  {
    h323-ID : {"gwa-1"}
  }
  sourceInfo
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            {
              prefix e164 : "1#"
            }
          }
        }
      }
    }
    mc FALSE
    undefinedNode FALSE
  }
  activeMC FALSE
  conferenceID '2D3AAC3316A411CC80688828285B8DF6'H
  conferenceGoal create : NULL
  callType pointToPoint : NULL
  sourceCallSignalAddress ipAddress :
  {
    ip 'AC100D0F'H
    port 11027
  }
  callIdentifier
  {
    guid '2D3AAC3316A411CC80698828285B8DF6'H
  }
  fastStart
  {
    '0000000C6013800A04000100AC100D0F47F1'H,
    '400000060401004C6013801114000100AC100D0F...'H
  }
  mediaWaitForConnect FALSE
  canOverlapSend FALSE
}
h245Tunneling TRUE
nonStandardControl
{
  {
    nonStandardIdentifier h221NonStandard :
    {
      t35CountryCode 181
      t35Extension 0
      manufacturerCode 18
    }
    data '6001020001041F04038090A31803A983816C0C21...'H
  }
}
```

```
}  
}
```

```
Mar 3 04:12:47.796: H225.0 OUTGOING ENCODE BUFFER ::= 20 A0060008  
914A0002 01400400 67007700 61002D0  
0 31088001 3C050401 00204000 2D3AAC33 16A411CC 80688828 285B8DF6  
00451C07 00AC100D 0F2B1311 002D3AAC  
3316A411 CC806988 28285B8D F6320212 0000000C 6013800A 04000100  
AC100D0F 47F11D40 00000604 01004C60  
13801114 000100AC 100D0F47 F000AC10 0D0F47F1 01000100 06A00180 2D0140B5  
00001226 60010200 01041F04 0  
38090A3 1803A983 816C0C21 80343038 35323732 39323370 05803336 3533  
Mar 3 04:12:47.800:  
Mar 3 04:12:47.872: H225.0 INCOMING ENCODE BUFFER ::= 21 80060008  
914A0003 00078E11 002D3AAC 3316A41  
1 CC806988 28285B8D F6390219 0000000C 60138011 14000100 AC100D17  
479E00AC 100D1747 9F1D4000 00060401  
004C6013 80111400 0100AC10 0D0F47F0 00AC100D 17479F01 00010008 800180  
Mar 3 04:12:47.872:  
Mar 3 04:12:47.876: H225.0 INCOMING PDU ::=
```

```
value H323_UserInformation ::=  
{  
  h323-uu-pdu  
  {  
    h323-message-body callProceeding :
```

```
!--- Call proceeding received.
```

```
{  
  protocolIdentifier { 0 0 8 2250 0 3 }  
  destinationInfo  
  {  
    mc FALSE  
    undefinedNode FALSE  
  }  
  callIdentifier  
  {  
    guid '2D3AAC3316A411CC80698828285B8DF6'H  
  }  
  fastStart  
  {  
    '0000000C6013801114000100AC100D17479E00AC...'H,  
    '400000060401004C6013801114000100AC100D0F...'H  
  }  
}  
h245Tunneling TRUE  
}
```

```
Mar 3 04:12:47.884: H225.0 OUTGOING PDU ::=
```

```
value H323_UserInformation ::=  
{  
  h323-uu-pdu  
  {  
    h323-message-body empty : NULL  
    h245Tunneling TRUE  
    h245Control  
    {  
      '0270010600088175000380138000140001000001...'H  
    }  
  }  
}
```

```

    }
}

Mar  3 04:12:47.884: H225.0 OUTGOING ENCODE BUFFER ::= 28 10010006
C0018063 01610270 01060008 8175000
3 80138000 14000100 00010000 0100000C C0010001 00048000 104810B5 0000120C
52747044 746D6652 656C6179
00008000 16830150 80001583 01408000 12830110 80000020 C0130080 01020000
16020015 00120010 000000
Mar  3 04:12:47.888:
Mar  3 04:12:47.888: H225.0 OUTGOING PDU ::=

```

```

value H323_UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body empty : NULL
    h245Tunneling TRUE
    h245Control
    {
      '01003C4010F3'H
    }
  }
}

```

```

Mar  3 04:12:47.892: H225.0 OUTGOING ENCODE BUFFER ::= 28 10010006
C0018008 01060100 3C4010F3
Mar  3 04:12:47.892:
Mar  3 04:12:47.892: cc_api_call_proceeding(vdbPtr=0x621B2360, callID=0x34,
prog_ind=0x0)
Mar  3 04:12:47.896: sess_appl: ev(21=CC_EV_CALL_PROCEEDING), cid(52),
disp(0)
Mar  3 04:12:47.896: cid(52)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_PROCEEDING)
oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(0)fDest(0)
Mar  3 04:12:47.896: -cid2(51)st2(SSA_CS_CALL_SETTING)oldst2(SSA_CS_CALL_SETTING)
Mar  3 04:12:47.896: ssaCallProc
Mar  3 04:12:47.896: ccGetDialpeerTag (callID=0x33)
Mar  3 04:12:47.896: ssaIgnore cid(52), st(SSA_CS_CALL_SETTING),oldst(1), ev(21)
Mar  3 04:12:47.900: H225.0 INCOMING ENCODE BUFFER ::= 28 10010008
C0018063 01610270 01060008 8175000
6 80138000 14000100 00010000 0100000C C0010001 00048000 104810B5 0000120C
52747044 746D6652 656C6179
00008000 16830150 80001583 01408000 12830110 80000020 C0130080 01020000
16020015 00120010 000000
Mar  3 04:12:47.904:
Mar  3 04:12:47.904: H225.0 INCOMING PDU ::=

```

```

value H323_UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body empty : NULL
    h245Tunneling TRUE
    h245Control
    {
      '0270010600088175000680138000140001000001...'H
    }
  }
}

```

!--- Some of the unnecessary H.225 debug messages are deleted here.

```

Mar 3 04:12:52.116: H225.0 INCOMING ENCODE BUFFER ::= 23 80060008
914A0003 000A8600 11002D3A AC3316A
4 11CC8069 8828285B 8DF60100 01000880 0180
Mar 3 04:12:52.120:
Mar 3 04:12:52.120: H225.0 INCOMING PDU ::=

```

```

value H323_UserInformation ::=
{
  h323-uu-pdu
  {
    h323-message-body alerting :

```

!--- Alerting message received.

```

{
  protocolIdentifier { 0 0 8 2250 0 3 }
  destinationInfo
  {
    mc FALSE
    undefinedNode FALSE
  }
  callIdentifier
  {
    guid '2D3AAC3316A411CC80698828285B8DF6'H
  }
  h245Tunneling TRUE
}
}

```

```

Mar 3 04:12:52.124: cc_api_call_alert(vdbPtr=0x621B2360, callID=0x34,
prog_ind=0x8, sig_ind=0x1)
Mar 3 04:12:52.124: sess_appl: ev(7=CC_EV_CALL_ALERT), cid(52), disp(0)
Mar 3 04:12:52.124: cid(52)st(SSA_CS_CALL_SETTING)ev(SSA_EV_CALL_ALERT)
oldst(SSA_CS_CALL_SETTING)cfid(-1)csz(0)in(0)fDest(0)

```

Troubleshoot Load Balance

With the load balance feature, you can set the gatekeeper with a certain threshold for the number of calls, memory, CPU, and number of registered endpoints. Once that threshold is reached, the gatekeeper moves registered Cisco H.323 endpoints to an alternate gatekeeper or rejects new calls and registrations. Load balancing is enabled using the following gatekeeper CLI command:

```

Router(config-gk)#
load-balance [endpoints
max-endpoints] [calls max-calls]
[cpu max-%cpu][memory
max-%mem-used]

```

When the threshold is met, the gatekeeper uses the RRJ RAS message to inform the endpoint about the alternate gatekeepers and the reject reason. When this message is received, the endpoint sends a new RRQ to the alternate gatekeeper. Once it is registered with the alternate gatekeeper, it uses the GUP message to inform all gatekeepers in the cluster about the new registered endpoint.

Some of the issues to watch for when you troubleshoot are to check the configuration on the gatekeeper and making sure alternate gatekeepers and load balancing are functional. The topology above is used for troubleshooting purposes. The configuration of "gkb-1" gatekeeper is changed to show the following cases:

- How the gatekeeper can reject a call when a threshold is met.
- How the gatekeeper can move the registration of an endpoint to an alternate gatekeeper when a threshold is met.

To debug the load balancing feature, use **debug gatekeeper load** and **debug h225 asn1** to see how the gatekeeper reacts when the threshold is met.

Here is the configuration of the "gkb-1" gatekeeper that is used to cover the two cases mentioned above (number of calls threshold and number of registered end points):

```
!
gatekeeper
zone local gkb-1 domainB.com 172.16.13.41
zone remote gka-1 domainA.com 172.16.13.35 1719
zone cluster local gkb gkb-1
element gkb-2 172.16.13.16 1719
!
security token required-for all
gw-type-prefix 2#* default-technology
bandwidth total zone gkb-1 512
bandwidth session zone gkb-1 512
load-balance endpoints 2 calls 1
!--- maximum of 2 endpoints and call threshold is 1
no shutdown
!
!
```

A call is made through the gatekeeper gkb-1. While that call is up, another call is made. The captured debug shows what the load balance debug looks like and how the gatekeeper rejects the second call because the threshold was met. You can use the following command to show how many active calls run using the gatekeeper:

```
gkb-1#show gatekeeper call
Total number of active calls = 1.
                        GATEKEEPER CALL INFO
                        =====
LocalCallID           Age(secs)           BW
5-29514                9                   128(Kbps)
  Endpt(s): Alias     E.164Addr
    src EP: gwa-1     4085272923
  Endpt(s): Alias     E.164Addr
    dst EP: gwb-1     3653
      CallSignalAddr  Port  RASSignalAddr  Port
      172.16.13.23   1720  172.16.13.23   54670
```

Here is the debug of H.225 asn1 and the gatekeeper load when the second call is requested:

```
Mar  3 05:04:55.354: RAS INCOMING ENCODE BUFFER ::= 4A 80080501 01806986
40B50000 12298286 B0110075 7
95BF216 AB11CC80 95882828 5B8DF601 81110201 80866940 04006700 77006100
2D003100 AC100D23 06B70B80 0D
014004 0067006B 0061002D 00310180
Mar  3 05:04:55.358:
Mar  3 05:04:55.358: RAS INCOMING PDU ::=

value RasMessage ::= locationRequest :

!--- LRQ is received.

{
  requestSeqNum 2054
  destinationInfo
```

```

    {
      e164 : "3653"
    }
    nonStandardData
    {
      nonStandardIdentifier h221NonStandard :
      {
        t35CountryCode 181
        t35Extension 0
        manufacturerCode 18
      }
      data '8286B0110075795BF216AB11CC80958828285B8D...'H
    }
    replyAddress ipAddress :
    {
      ip 'AC100D23'H
      port 1719
    }
    sourceInfo
    {
      h323-ID : {"gka-1"}
    }
    canMapAlias TRUE
  }

```

```

Mar  3 05:04:55.362: H225 NONSTD INCOMING ENCODE BUFFER ::= 82 86B01100
75795BF2 16AB11CC 80958828 28
5B8DF6 01811102 01808669 40040067 00770061 002D0031
Mar  3 05:04:55.366:
Mar  3 05:04:55.366: H225 NONSTD INCOMING PDU ::=

```

```

value LRQnonStandardInfo ::=
  {
    ttl 6
    nonstd-callIdentifier
    {
      guid '75795BF216AB11CC80958828285B8DF6'H
    }
    callingOctet3a 129
    gatewaySrcInfo
    {
      e164 : "5336",
      h323-ID : {"gwa-1"}
    }
  }

```

```

Mar  3 05:04:55.366: gk_load_overloaded: Overloaded due to reaching specified call
limits

```

```

!--- Number of calls threshold has met.

```

```

Mar  3 05:04:55.370: RAS OUTGOING PDU ::=

```

```

value RasMessage ::= locationReject :

```

```

!--- LRJ is sent.

```

```

  {
    requestSeqNum 2054
    rejectReason undefinedReason : NULL
  }

```

For the second example, "gkb-1" gatekeeper has two registered endpoints. Registration for another endpoint is attempted. The gatekeeper moved the endpoint, trying to register to the alternate gatekeeper "gkb-2", since they are in the same cluster. Here is the debug message for **debug h225 asn1** and **debug gatekeeper gup asn1** for this case:

```
Mar  3 05:21:05.682: RAS INCOMING PDU ::=
value RasMessage ::= registrationRequest :
!--- RRQ message is received.
{
  requestSeqNum 4621
  protocolIdentifier { 0 0 8 2250 0 3 }
  discoveryComplete TRUE
  callSignalAddress
  {
    ipAddress :
    {
      ip 'AC100D2A'H
      port 1720
    }
  }
  rasAddress
  {
    ipAddress :
    {
      ip 'AC100D2A'H
      port 49998
    }
  }
  terminalType
  {
    gateway
    {
      protocol
      {
        voice :
        {
          supportedPrefixes
          {
            {
              prefix e164 : "1#"
            }
          }
        }
      }
    }
  }
  mc FALSE
  undefinedNode FALSE
}
terminalAlias
{
  h323-ID : {"gwb-3"}
}
gatekeeperIdentifier {"gkb-1"}
endpointVendor
{
  vendor
  {
    t35CountryCode 181
    t35Extension 0
    manufacturerCode 18
  }
}
```

```

}
timeToLive 60
tokens
{
    {
        tokenOID { 1 2 840 113548 10 1 2 1 }
        timeStamp 731136065
        challenge '5A70CA112E6C7A3834792BD64FF7AD2F'H
        random 58
        generalID {"gwb-3"}
    }
}
cryptoTokens
{
    cryptoEPPwdHash :
    {
        alias h323-ID : {"gwb-3"}
        timeStamp 731136065
        token
        {
            algorithmOID { 1 2 840 113549 2 5 }
            paramS
            {
            }
            hash "B1C1DAD962BEE42B1E53F368238B1D8"
        }
    }
}
keepAlive FALSE
willSupplyUUIEs FALSE
maintainConnection TRUE
}

```

Mar 3 05:21:05.698: **gk_load_overloaded: Overloaded due to reaching specified endpoint limits**

!--- Endpoint threshold is met.

Mar 3 05:21:05.702: RAS OUTGONG PDU ::=

value RasMessage ::= **registrationReject** :

!--- RRJ is sent.

```

{
    requestSeqNum 4621
    protocolIdentifier { 0 0 8 2250 0 3 }
    rejectReason resourceUnavailable : NULL
}

```

!--- Reject reason.

```

gatekeeperIdentifier {"gkb-1"}
altGKInfo

```

!--- List of alternate gatekeepers.

```

{
    alternateGatekeeper
    {
        {
            rasAddress ipAddress :
            {

```

```
        ip 'AC100D10'H
        port 1719
    }
    gatekeeperIdentifier {"gkb-2"}
    needToRegister TRUE
    priority 0
}
altGKisPermanent TRUE
```

!--- Informs the endpoint that the move is permanent.

}

```
Mar  3 05:21:05.706: RAS OUTGOING ENCODE BUFFER::= 16 80120C06 0008914A
00038101 00080067 006B0062 0
02D0031 07001600 0140AC10 0D1006B7 08006700 6B006200 2D003280 80
Mar  3 05:21:05.706:
Mar  3 05:21:05.782: Received GUP REGISTRATION INDICATION from 172.16.13.16
```

!--- GUP update for the new endpoint.

gkb-1#

Related Information

- [Voice Technology Support](#)
- [Voice and IP Communications Product Support](#)
- [Recommended Reading: Troubleshooting Cisco IP Telephony](#)
- [Technical Support – Cisco Systems](#)

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2009 – 2010 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

Updated: Feb 02, 2006

Document ID: 18730
