Устранение неисправностей GUP, альтернативных конечных точек и распределения загрузки

Содержание

Введение

Этот документ содержит информацию, которая может помочь в устранении неполадок и использовании следующих функций Cisco Gatekeeper:

- Кластеры привратника и протокол обновления привратника (GUP)
- Альтернативные конечные точки

- Распределение нагрузки

См. Cisco Высокоэффективное Сторожевое устройство для всей необходимой информации об этих функциях включая обзор характеристик, поддерживаемые платформы, необходимые версии программного обеспечения Cisco IOS и как настроить, контролирует и поддерживает их.

Предварительные условия

Требования

Ознакомление с этим документом требует наличия следующих знаний:

- Базовые знания о функциональных возможностях сторожевого устройства.
- Базовые знания о VoIP, H.323 и сигнализации о регистрации, доступе и статусе (RAS).

Используемые компоненты

Сведения в этом документе основаны на версиях оборудования и программного обеспечения, указанных ниже.

- Программное обеспечение Cisco IOS версии 12.3 (4) T1
- Шлюзы Cisco: Cisco AS5300, Cisco AS5400 и Cisco 3725
- Сторожевые устройства Cisco: Cisco 3725 и Cisco 2611

Сведения, содержащиеся в данном документе, были получены с устройств в специальной лабораторной среде. Все устройства, описанные в данном документе, были запущены с конфигурацией по умолчанию. При работе с реальной сетью необходимо полностью осознавать возможные результаты использования всех команд.

Условные обозначения

Дополнительные сведения об условных обозначениях см. в документе Технические рекомендации Cisco. Условные обозначения.

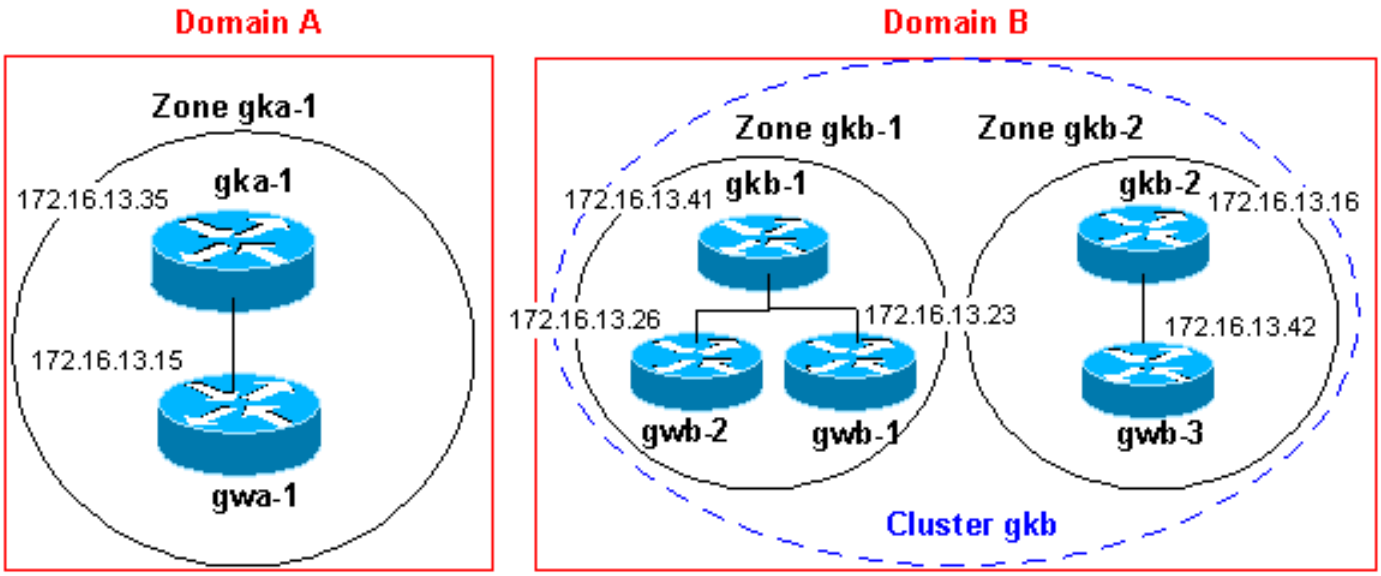
Определения

Условие	Определение
ARQ	Запрос допуска (ARQ) – это сообщение RAS, отправленное с конечной точки Cisco H.323 на привратник, который запрашивает допуск на установление вызова.
ACF	Подтверждение допуска (ACF) – это сообщение RAS, отправленное от привратника к конечной точке, которая подтверждает прием вызова.
ARJ	Отклонение входа представляет собой сообщение сервера удаленного доступа от шлюза к конечной точке, которая отклоняет

	требование входа.
GCF	Gatekeeper Confirm (GCF) является сообщением RAS, передаваемым от сторожевого устройства до оконечной точки Cisco H.323, которая подтверждает обнаружение сторожевого устройства.
GRQ	Запрос сторожевого устройства (GRQ) является сообщением RAS, передаваемым от оконечной точки Cisco H.323 для обнаружения сторожевого устройства.
GUP	Протокол обновления привратника используется для обмена информации между привратниками в кластере о конечных точках и активных вызовах.
LCF	Подтверждение размещения (LCF) - это RAS сообщение, посланное с одного привратника на другой, которое подтверждает Запрос расположения (LRQ) и включает в себя IP-адрес завершающей конечной точки.
LRJ	Отказ в местонахождении (LRJ) является сообщением RAS, передаваемым от одного сторожевого устройства до другого, который отклоняет LRQ.
LRQ	Запрос местонахождения – это сообщение RAS, отправленное с одного привратника к другому, которое запрашивает IP-адрес удаленной конечной точки.
RAS	Протокол RAS позволяет привратнику выполнять проверку регистрации, доступа и статуса конечной точки.
RCF	Регистрация Подтверждает (RCF), сообщение RAS, передаваемое от сторожевого устройства до оконечной точки, которая подтверждает регистрацию.
RRJ	Отказ в регистрации (RRJ) - это сообщение RAS, отправляемое привратником при отклонении запроса на регистрацию.
RRQ	Запрос регистрации (RRQ) представляет собой сообщение RAS, отправленное привратнику с конечной точки в целях регистрации.
URQ	Запрос на регистрацию (URQ) представляет собой сообщение RAS, передаваемое из конечной точки на шлюз, запрашивающий разрегистрацию с конечной точкой.

Топология лаборатории и настройки

Чтобы объяснить, как функции работают и как устранить неполадки, лабораторная установка была создана с этой топологией:



Топология лаборатории и настройки

Базовые конфигурации всех шлюзов и привратников приведены в таблице внизу. В других случаях требуются некоторые изменения конфигурации. Изменение обозначено, когда это происходит. Конфигурации, приведенные ниже, содержат только части, существенные для функциональных возможностей шлюза или привратника для этих лабораторных условий.

Конфигурации "gwb-1" и "gwb-2" почти подобны (за исключением IP-адреса и ID H.323). Поэтому только `gwb-1` показывают ниже.

Гва 1

```
!
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#
!
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
```

Альтернативные привратники

До выпуска H.323 версии 2 управление каждой зоной осуществлялось только одним привратником. Cisco H.323 версии 2 представляет концепцию "альтернативного привратника", обеспечивающую избыточность привратников. Реализация функции альтернативного привратника позволяет осуществлять контроль одной зоны несколькими привратниками. Когда конечная точка регистрируется в привратнике, ей предлагается список альтернативных привратников для зоны, в которой регистрируется эта точка. Для этой зоны альтернативные варианты определяются с помощью CLI. Если сторожевое устройство отказывает, оконечная точка может использовать альтернативные сторожевые устройства для продолжения операции.

Список альтернативных привратников предоставляется Cisco Gatekeeper с помощью CLI для каждой зоны и передается в конечные точки с помощью сообщений RCF (в т.ч. упрощенных) и GRQ. Этот список может также быть передан в других сообщениях, таких как ARJ или URQ, для упрощения управляемого завершения сторожевого устройства.

Альтернативные сторожевые устройства изучают о существующих вызовах через Запрос на прерывание (IRQ) / Information Request Response (IRR) обмен между шлюзами и сторожевыми устройствами и отслеживают эти вызовы.

Оконечная точка, которая обнаруживает сбой ее сторожевого устройства, может безопасно восстановиться с того сбоя путем использования альтернативного сторожевого устройства для будущих запросов, включая запросы о существующих вызовах. В кластере должны быть настроены альтернативные привратники. Они делятся информацией об оконечных точках и активных вызовах с помощью GUP, который работает на TCP.

Протокол обновления привратника

Вот некоторые главные действия и предупреждения GUP. Это также должно помочь в устранении ошибки.

- После подключения к сети привратника, настроенного в качестве участника кластера, на привратнике открывается TCP-порт для прослушивания входящих соединений по протоколу GUP.
- Затем это объявляет о своем присутствии путем передачи сообщения GRQ на периодической основе. Период по умолчанию составляет 30 секунд и является конфигурируемым использованием кластерного элемента таймера гэйткипера CLI, объявляют о команде. Это сообщение GRQ содержит нестандартные данные для каждого альтернативного привратника. Эти нестандартные данные являются индикатором к альтернативам, что GRQ является действительно не GRQ вообще, а скорее является просто сообщением "announcement". В сообщении GRQ сторожевое устройство указывает на номер порта, который это имеет открытый для прислушиваний к протоколу GUP.
- При получении GRQ от нового привратника другие привратники в кластере открывают каналы TCP к этому порту.
- Сообщения GUP GRQ могут быть одним из следующих сообщений: announcementIndication, announcementReject, registrationIndication, unregistrationIndication и resourceIndication.
- Индикация оповещения также содержит информацию об использовании пропускной способности для зоны. Это позволяет альтернативным привратникам правильно управлять пропускной способностью для одной зоны, даже если привратники физически находятся в разных устройствах.
- Чтобы проверить, связываются ли альтернативные сторожевые устройства должным образом или нет, используйте команду show gatekeeper zone cluster. Эта команда также выводит информацию о полосе пропускания для альтернативных привратников.
- Сторожевое устройство предполагает, что альтернативное сторожевое устройство отказало (и предполагает, что любая ранее выделенная полоса теперь доступна), если сторожевое устройство не получает сообщение уведомления в шести периодах уведомления, или если TCP - подключение со сторожевым устройством обнаружен, чтобы быть сломанным. Объявления рассылаются каждые 30 секунд, что в сумме дает

три минуты и предположительно равняется средней длительности вызова. Затем с большой вероятностью можно судить о том, что полоса пропускания освобождена. После трех минут это сторожевое устройство объявляет свою альтернативу как вниз и отсылает обновление, чтобы уведомить все его зарегистрированные оконечные точки, что нет никакого альтернативного сторожевого устройства.

- Если конечная точка регистрируется/регистрация отменяется привратником в кластере, привратник использует сообщение registrationIndication/ unregistrationIndication для обновления всех остальных привратников в этом кластере, чтобы оповестить их об этом изменении.
- Если на конечной точке отмечено изменение ресурса путем подачи индикатора доступности ресурса (RAI) на привратник в кластере, то этот привратник докладывает всем остальным об изменении в кластере при помощи сообщения resourceIndication протокола GUP.
- Сообщения GUP необходимы для обеспечения привратника в кластере достаточными сведениями (регистрация, полоса пропускания, активные вызовы ресурсы) о каждой конечной точке в зоне, чтобы он мог разрешить все запросы локально.
- Когда оконечная точка коммутирована от одного сторожевого устройства до альтернативы, альтернатива должна учиться о вызовах, которые активны на оконечной точке. При отправке RCF для новой регистрации привратник также отправляет IRQ, чтобы получить список всех вызовов на конечной точке. Важно убедиться, что IRQ не достигает конечной точки перед RCF.
- Привратники в кластере разрешают отключение, даже если есть активные вызовы, пока есть альтернативный привратник, определенный для всех зон, для которых имеются активные вызовы. Если какая-либо зона имеет активный вызов и никакое определенное альтернативное сторожевое устройство, сторожевое устройство отказывается от завершения.
- Альтернативные сторожевые устройства принимают любые Запросы отключения (DRQ) для вызовов, о которых они не знали и передают соответствующую информацию к Протоколу транзакции сообщений Аутентификации, авторизации и учета (AAA) и Сторожевого устройства Cisco (GKTMP) серверы. Это происходит, когда та оконечная точка перемещается к альтернативному сторожевому устройству, в то время как существуют активные вызовы. Кроме того, могут быть отправлены сообщения IRR, содержащие информацию по прежде неизвестным вызовам. Для сообщений IRR создаются записи вызовов и полоса пропускания распределяется соответствующим образом.
- Привратник создает уникальное сообщение-уведомление для каждого альтернативного привратника. Если дополнительный привратник получает сообщение с идентификатором привратника, который не удается распознать (это возможно, если дополнительный привратник используется для одной зоны), эти данные игнорируются. Однако альтернативное сторожевое устройство обнаруживает ошибки в конфигурации альтернатив путем исследования тех сообщений, и она сообщает о тех ошибках пользователю.
- Когда адреса решены для удаленной зоны, истинное питание GUP понято. Вместо потребности в удаленной зоне для передачи LRQ (или в последовательности или в уничтожении) ко всем сторожевым устройствам, таким образом увеличивая служебную информацию об обмене сообщениями на глобальных каналах, это теперь должно передать этот запрос ко всего одному из сторожевых устройств в кластере. Вместе с новым CLI zone cluster remote это может циклический алгоритм между сторожевыми

устройствами в кластере и не пытаться передать LRQ к другому сторожевому устройству в кластере, если это получает отклонение от кого-либо.

- **В случае перемещения шлюза к альтернативному привратнику он всегда будет пытаться зарегистрироваться у этого привратника, пока не будет подана команда no gateway, а затем - команда gateway.** Когда основной привратник конечной точки возвращается в интерактивный режим, конечная точка повторно не регистрирует его, если только конечная точка не потеряет связь с альтернативным привратником. Он продолжает использовать альтернативный привратник для своих сведений о маршрутизации вызова.

команды debug и show для кластеров привратника

Отладки ниже демонстрируют, как сторожевые устройства могут присоединиться к кластеру и как они делятся информацией о своих оконечных точках. **команды show используются для демонстрации того, как контролируется кластер**. Используемые отладки являются asn1 протокола GUP сторожевого устройства отладки и debug h225 asn1. Это основано на топологии и конфигурации, описанных выше.

Выполняет отладку из gkb-1 при первом присоединении к кластеру

```
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:
```

Отладка с "gkb-2" при присоединении к кластеру после "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**

С [командой show gatekeeper endpoint](#) нет никаких зарегистрированных оконечных точек. Выходные данные выглядят следующим образом:

```
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#
```

[Отладка проводится, когда конечная точка регистрируется одной из систем управления шлюза в кластере](#)

Эта отладка получена от привратника "gkb-1". [Конечная точка "gwb-1" регистрируется для привратника "gkb-1" с включенными параметрами debug h225 ans1 и debug ras.](#)

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
86F70C0A 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:
```

Выходные данные выше содержат выходные данные [команды show gatekeeper endpoint](#) после всего регистра шлюзов в кластере. Вышеупомянутые отладки происходят для каждой регистрации оконечной точки. После всех трех шлюзов в кластерном регистре **команда show gatekeeper endpoint** на обоих сторожевых устройствах следующие:

```
gkb-1#show gatekeeper endpoints GATEKEEPER ENDPOINT REGISTRATION
================================ CallSignalAddr Port RASSignalAddr 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 RASSignalAddr 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.
```

Отладка сценария перемещения конечной точки с активным вызовом к дополнительному привратнику

Это отладки от сторожевого устройства, которое запускается, когда вызов запрашивают и идет, пока он не разъединен. Некоторые ненужные сообщения отладки были опущены. Эти отладки от сторожевого устройства "gkb-1". Вызов был размещен Гва 1 зарегистрированный к "gka-1" к другому шлюзу (gwb-1) в удаленном зональном кластере. Отладки демонстрируют, как поток активного вызова отслежен от основного сторожевого устройства альтернативному сторожевому устройству как основные движения вниз.

```
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 { h323-ID : {"gwb-1"} } gkID {"gkb-1"} gateways { { gwType voip : NULL gwAlias { h323-ID : {"gwb-1"} *!--- Gateway gwb-1 is the resolved terminating gateway sent back for the request.* } sigAddress { ip 'AC100D17'H *!--- Gateway gwb-1 IP address (172.16.13.23).* port 1720 } resources { maxDSPs 0 inUseDSPs 0 maxBChannels 0 inUseBChannels 0 activeCalls 0 bandwidth 0 inuseBandwidth 0 } } } } Mar 2 23:59:26.734: H225 NONSTD OUTGOING ENCODE BUFFER::= 00 01400400 67007700 62002D00 31080067 006B0062 002D0031 01100140 04006700 77006200 2D003100 AC100D17 06B80000 00000000 00000000 Mar 2 23:59:26.734: Mar 2 23:59:26.734: RAS OUTGOING PDU ::= value RasMessage ::= **locationConfirm** : *!--- LCF is sent back with "gwb-1" as the resolved terminating gateway.* { requestSeqNum 2057 callSignalAddress ipAddress : { ip 'AC100D17'H *!--- Resolved terminating gateway gwb-1 IP address (172.16.13.23).* port 1720 } rasAddress ipAddress : { ip 'AC100D17'H *!--- Resolved terminating gateway gwb-1 IP address (172.16.13.23).* port 51874 } nonStandardData { nonStandardIdentifier h221NonStandard : { t35CountryCode 181 t35Extension 0 manufacturerCode 18 } data '0001400400670077006200 2D0031080067006B00...'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 3800300 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 'AC100D0F'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 inforomation !--- 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
```

Примечание: На этом этапе "gkb-1" является завершением. Это позволено (даже если это имеет активный вызов), потому что существует альтернативное сторожевое устройство для той зоны.

Сообщения URQ передаются всем оконечным точкам, зарегистрированным в "gkb-1". Эти оконечные точки являются "gwb-1" и "gwb-2" шлюзами. Эти шлюзы подтверждают URQ путем передачи UCF обратно. Кроме того, gkb-1 передает сообщение об отмене регистрации GUP кластерному альтернативному сторожевому устройству, и затем он закрывает GUP - подключение.

```
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). port 1720 } } } 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
```

Вот отладка от "gkb-2". Отладки, которые показывают регистрацию перемещенной оконечной точки "gwb-1" и "gwb-2", опущены, так как они похожи на стандартную регистрацию. Цель здесь состоит в том, чтобы показать принятие DRQ активного вызова на "gwb-1", когда это перемещено в "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#
```

Переключение при отказе шлюза Cisco на альтернативного привратника

По умолчанию шлюзы Cisco передают RRQ легкого веса каждые 45 секунд. В случае, если сторожевое устройство не передало URQ к шлюзу (из-за сломанной проблемы маршрутизации, например), шлюз (на не слушание RCF или RRJ для его RRQ легкого веса) попытки дважды с пятью секундами между каждым. Если третья попытка отказывает, она сразу рассматривает сторожевое устройство как мертвое и регистрируется в альтернативном сторожевом устройстве, которое использует RRQ. В сценарии, где шлюз запускает начальный процесс регистрации со сторожевого устройства, это отсылает GRQ для определения местоположения IP-адреса сторожевого устройства. Если существует ответ GCF назад, шлюз передает RRQ заданному основному сторожевому устройству. Если по какой-либо причине сторожевое устройство отклоняет запрос регистрации, шлюз не пытается связаться со своим альтернативным сторожевым устройством. Это запускает этот процесс (GRQ, GCF и RRQ) снова с основным сторожевым устройством.

Шлюз только связывается с альтернативным сторожевым устройством, когда подключение основному сторожевому устройству потеряно и назад нет никакого ответа. Если основное сторожевое устройство не отвечает назад на сообщение GRQ, когда шлюз сначала отсылает для обнаружения сторожевого устройства, то после трех неудачных попыток (приблизительно пять минут на попытку), шлюз связывается с альтернативным сторожевым устройством. В ситуации, где основное сторожевое устройство выключается после того, как шлюз зарегистрировался в нем, шлюз теряет сообщения пакетов Keepalive от основного сторожевого устройства. После пропавших без вести трех последовательных сообщений поддержки активности шлюз объявляет основное сторожевое устройство как вниз, и это запускает процесс регистрации снова.

Проблемы устранения неполадок и альтернативные конечные точки

Конечная точка вызова может устранить ошибку установки вызова, отправив сообщение установки одной из альтернативных точек. Вызов может не состояться по ряду причин: шлюз не отвечает и привратник не знает об этом в момент отправки ACF или LCF, на шлюзе нет ресурсов и привратник не был об этом уведомлен, вызов не выполняется из-за неверной настройки на главной конечной точке и так далее.

Примечание: Если вызов отказывает перед аварийным этапом (предупреждение или выполнение), исходная оконечная точка только пытается связаться с альтернативными сторожевыми устройствами. Если вызовы отказывают из-за абонента занят или никакого ответа, исходная оконечная точка не пробует никакие другие альтернативы.

Сторожевое устройство учится о той альтернативе для определенной оконечной точки или настройкой вручную с помощью alt-ep оконечной точки команды CLI гэйткипера или из любых полученных сообщений RAS. Cisco поддерживает максимум 20 альтернатив для

каждой оконечной точки, независимо от того как сторожевое устройство изучает их.

Нужно обратить внимание на проблему, включающую:

- Если сторожевое устройство имеет правильную альтернативную оконечную точку, как желаемый.
- Если сторожевое устройство включает альтернативные оконечные точки в свой LCF или сообщения ACF RAS.
- Если OGW пытается связаться с альтернативами в случае, если отказывает оконечная точка главного направления.

Чтобы показать, как решить эти проблемы, та же топология как выше используется с этим изменением на конфигурации сторожевого устройства "gkb-1" для включения двух альтернативных шлюзов: "gwb-3" и "gwb-1" для шлюза "gwb-2". Вот конфигурация сторожевого устройства "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
 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. !
```

Проверьте, что Сторожевое устройство имеет Правильные альтернативные оконечные точки

Чтобы увидеть, содержит ли привратник правильные альтернативные конечные точки, используйте команду show gatekeeper endpoints alternates.

```
gkb-1#show gatekeeper endpoints alternates GATEKEEPER ENDPOINT REGISTRATION
================================= CallSignalAddr Port RASSignalAddr Port Zone Name Type Flags -
------------------ ------ ------------------- ------- --------------- --------- --------
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 A 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#
```

Проверка наличия дополнительных конечных точек в сообщениях LCF или ACF RAS привратника

Чтобы видеть, передает ли сторожевое устройство IP-адрес за альтернативными оконечными точками, можно включить **debug h225 asn1** и посмотреть на сообщение ACF или LCF. Это пример отладки, взятый из "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
'0001400400670077006200 2D0032080067006B00...'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 gwb-3). port 1720 }, ipAddress : { ip 'AC100D17'H !--- This is the
second alternate IP address (172.16.13.23 gwb-1). port 1720 } } } } }
```

Проверьте, пытается ли OGW связаться с альтернативными точками в случае, если конечный пункт основного места назначения недоступен

Этот раздел показывает, как OGW реагирует на получение альтернативной конечной точки в его сообщении ACF. В данном примере происходит сбой выполнения вызова при попытке установления соединения с основной терминальной конечной точкой (шлюзом).
Включаемые здесь отладки - debug voip ccapi inout и debug h225 asn1.

Первое, что вы увидите при отладке – сообщение ccapi, которое указывает на исходную ветвь телефонии.

```
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), clled(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)csize(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** *!--- List of alternate endpoints.* **{ { callSignalAddress { ipAddress : { ip 'AC100D2A'H** *!--- 172.16.13.42.* **port 1720 }, ipAddress : { ip 'AC100D17'H** *!--- 172.16.13.23.* **port 1720 } } } }** willRespondToIRR FALSE uuiesRequested { setup FALSE callProceeding FALSE connect FALSE alerting FALSE information FALSE releaseComplete FALSE facility FALSE progress FALSE empty FALSE } } Mar 3 04:12:47.720: H225 NONSTD OUTGOING PDU ::= value H323_UU_NonStdInfo ::= { version 2 protoParam qsigNonStdInfo : { iei 4 rawMesg '04038090A31803A983816C0C2180343038353237...'H } } Mar 3 04:12:47.720: H225 NONSTD OUTGOING ENCODE BUFFER::= 60 01020001 041F0403 8090A318 03A98381 6C 0C2180 34303835 32373239 32337005 80333635 33 Mar 3 04:12:47.724: Mar 3 04:12:47.724: H225.0 OUTGOING PDU ::= value H323_UserInformation ::= { h323-uu-pdu { **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, '4000000604010004C6013801114000100AC100D0F...'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)csize(0)in(0)fDest(0)
```

Устранение проблем распределения нагрузки

С функцией балансировки нагрузки можно установить сторожевое устройство с определенным порогом для количества вызовов, памяти, ЦП и количества зарегистрированных оконечных точек. Как только тот порог достигнут, шаги сторожевого устройства зарегистрировали оконечные точки Cisco H.323 альтернативному сторожевому устройству или отклоняют новые требования и регистрацию. Балансировка нагрузки включается при помощи следующей команды привратника CLI:

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

Когда порог будет достигнут, привратник использует сообщение RRJ RAS для информирования конечной точки об альтернативных привратниках и причине отказа. Когда

это сообщение получено, оконечная точка передает новый RRQ альтернативному сторожевому устройству. После регистрации другим привратником он с помощью сообщения GUP сообщает всем привратникам в кластере о новой зарегистрированной конечной точке.

Среди проблем, которые могут произойти при устранении неполадок, можно назвать проверку конфигурации на привратнике и проверку того, являются ли альтернативные привратники и балансировка нагрузки функциональными. Топология, приведенная выше, используется для устранения неисправностей. Конфигурация "gkb-1" привратника изменена, чтобы показать следующие случаи:

- Когда порог встречен, как сторожевое устройство может отклонить требование.
- Как привратник может передать регистрацию конечной точки альтернативному привратнику при достижении порога.

Для отладки функции распределения нагрузки используйте debug gatekeeper load и **debug h225 asn1**, чтобы видеть, как сторожевое устройство реагирует, когда встречен порог.

Ниже приведена конфигурация привратника "gkb-1", используемая в двух приведенных выше случаях (число порога вызовов и число зарегистрированных конечных точек):

```
!
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 ! !
```

Вызов выполнен через gkb-1 сторожевого устройства. В то время как тот вызов подключен, другой вызов выполнен. Собранная отладочная информация показывает, что представляет собой отладка баланса загрузки, и как привратник отклоняет второй вызов, поскольку достигнуто пороговое значение. Следующая команда может использоваться для того, чтобы показать число активных вызовов, выполняемых с помощью привратника:

```
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
```

Здесь выполняется отладка H.225 asn1 и загрузка привратника, когда запрошен второй вызов:

```
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 }
```

Для второго примера привратник "gkb-1" сторож имеет две зарегистрированных конечных точки. Регистрация для другой оконечной точки предпринята. Сторожевое устройство переместило оконечную точку, пытаясь зарегистрировать альтернативному сторожевому устройству "gkb-2", так как они находятся в том же кластере. **Вот сообщение отладки для debug h225 asn1 и debug gatekeeper gup asn1 для данного случая:**

```
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#
```

Дополнительные сведения

- Поддержка голосовых технологий
- Поддержка продуктов голосовой и IP-связи
- Устранение неполадок в системах IP-телефонии Cisco
- Техническая поддержка - Cisco Systems