

# Compreendendo e Troubleshooting de Gatekeeper TTL e Processo de Envelhecimento

## Índice

[Introdução](#)

[Pré-requisitos](#)

[Requisitos](#)

[Componentes Utilizados](#)

[Convenções](#)

[Dicas sobre tempo para atividade](#)

[Depurações do Cisco Gatekeeper](#)

[Informações Relacionadas](#)

## [Introdução](#)

Este documento descreve como o gatekeeper Cisco envelhece para fora os valores-limite usando o valor do Time to Live (TTL) em casos diferentes. Os comandos de debug e show são usados mostrar como o TTL trabalha em várias maneiras.

## [Pré-requisitos](#)

### [Requisitos](#)

Leitores deste documento devem estar cientes destes tópicos:

- Aplicação de Cisco H.323, incluindo o gatekeeper Cisco. Para uma compreensão básica de H.323 gatekeepers, refira a [compreensão H.323 gatekeepers](#).

### [Componentes Utilizados](#)

As informações neste documento são baseadas nestas versões de software e hardware.

- Com a finalidade deste documento, o Software Release 12.3(9) de Cisco IOS® é usado para recolher a informação. Assegure-se de que você use o Cisco IOS com a característica da funcionalidade de gatekeeper de H.323. Isto é denotado com um **x no** nome da imagem de IOS Cisco. Por exemplo, o Cisco IOS válido para que o Cisco 3640 atue como um porteiro é c3640-ix-mz.123-9.bin.
- Gatekeeper Cisco (todas as Plataformas). **Nota:** O NetMeeting foi usado como um valor-limite de H.323 no exemplo neste documento porque não fornece um valor TTL. Para obter informações sobre de como configurar o NetMeeting com Cisco IOS gateway, refira [Como configuram o Microsoft NetMeeting com Cisco IOS gateway](#).

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se você estiver trabalhando em uma rede ativa, certifique-se de que entende o impacto potencial de qualquer comando antes de utilizá-lo.

## Convenções

Para obter mais informações sobre convenções de documento, consulte as [Convenções de dicas técnicas Cisco](#).

## Dicas sobre tempo para atividade

Estas são algumas pontas em como o TTL trabalha em um gatekeeper Cisco e em como os trabalhos de processo de envelhecimento para os valores-limite em casos diferentes.

- O gatekeeper Cisco espera ouvir-se periodicamente do valor-limite através das mensagens da solicitação de registro (RRQ) (de pouco peso ou completo).
- Para timeouts de ponto final, o gatekeeper Cisco paga a atenção ao valor TTL fornecido pelo valor-limite no mensagem de RRQ. Há um padrão duramente codificado de 1800 segundos (trinta minutos) para o timeout de ponto final. Este valor pode ser mudado com o `<time_value> ttl do ponto final de comando CLI do gatekeeper Cisco`. Isto muda o comportamento para todos os valores-limite de H.323 v1, ou os valores-limite de H.323v2 e mais atrasados que não incluem o valor TTL no mensagem de RRQ.
- O gatekeeper Cisco executa do “um processo de envelhecimento valor-limite” periodicamente. Este processo varia baseado na carga de CPU atual de cada um minuto a cada cinco minutos. Para cada vinte por cento da utilização CPU, o intervalo do envelhecimento aumenta em um minuto, até um máximo de cinco minutos. Para não sobrecarregar o CPU quando há muitos valores-limite, o processo de envelhecimento é executado somente com cinqüenta valores-limite pela passagem. Se há mais, a seguir aqueles estão adiados até o PNF seguinte do temporizador. Este pode ser um a cinco minutos.
- Se o campo `timetolive` está incluído na mensagem do Registro de RRQ, da admissão e do estado (RAS), os usos do porteiro que o valor do campo para cancelar o padrão de sistema ou o valor configuraram usando o comando `gatekeeper cli do <time_value> ttl do valor-limite`. Uma vez que esse período de tempo decorre sem se ouvir do valor-limite, o PNF seguinte do temporizador atravessa o processo da limpeza para esse valor-limite. O pior caso é o TTL enviado pelo valor-limite mais cinco minutos (se o gatekeeper Cisco está consistentemente sob a carga de CPU pesada). Um cenário do pior caso mais provável é o TTL timeout mais um minuto.
- Quando o valor-limite não inclui o campo `timetolive` no mensagem de RRQ, o gatekeeper Cisco trata o valor-limite como se não apoia o TTL. Neste caso, uma vez que o porteiro já não recebe RRQ desse valor-limite, faz o TTL timeout (o padrão de 1800 segundos, ou o valor especificado no [comando endpoint ttl](#)). Então manda três pedidos da informação (IRQ) com intervalos em qualquer lugar de um a cinco minuto cada um (baseado na carga de CPU do porteiro). Depois que envia três IRQ, e não recebe nenhuma resposta, o gatekeeper Cisco remove finalmente o valor-limite.
- Se o valor-limite está em uma chamada ativa, não está envelhecido para fora até que o atendimento esteja terminado.

- O gatekeeper Cisco espera ouvir-se dos valores-limite envolvidos em um atendimento com a mensagem do Information Response (IRR). Se o gatekeeper Cisco não recebe os mensagens IRR periódico que contêm referências ao “guid” para um atendimento, o porteiro espera quatro minutos, e manda então um IRQ aos valores-limite que o atendimento está para. Se, após oito mais minutos, o gatekeeper Cisco não tem ouvido ainda qualquer coisa sobre esse atendimento, o atendimento está limpadado e o porteiro envia as requisições de desligamento (DRQ) aos valores-limite. Um total de aproximadamente doze passagens dos minutos antes de um atendimento “de oscilação” é limpadado (e a largura de banda livrada). Este temporizador do atendimento não é configurável.
- Os valores-limite que são possuídos por um gatekeeper Cisco alternativo não são envelhecidos para fora diretamente (desde que este porteiro realmente “não possui” o valor-limite).
- Os Pontos finais estáticos (criados com a [estática](#) xxxxx do comando configuration [aliás](#)) não são envelhecidos para fora.

## Depurações do Cisco Gatekeeper

Estes são alguns dos comandos **show and debug** que você pode se usar para verificar como o TTL trabalha em várias maneiras:

- [show gatekeeper endpoints](#)
- [debug ras](#)
- [debug h225 asn1](#)

Estas duas seções discutem dois casos onde o gatekeeper Cisco envelhece para fora os valores-limite usando valores diferentes TTL.

### Caso 1

Esta saída é do e é tomada de um [gatekeeper Cisco](#). Debugar, o gateway tem um valor TTL de 60 segundos. O gatekeeper Cisco confirma e aceita este em sua mensagem de confirmação de registro (RCF), não importa o que seu valor do padrão ou do ponto final configurado TTL é. Isto é porque o valor-limite inclui um valor TTL.

```
Mar  2 23:52:50.797: RAS INCOMING ENCODE BUFFER ::= 0E 400FD206 0008914A
00030000 0100AC10 0D2AE26A 00040067 006B0062 0
02D0032 00B50000 12128F00 02003B01 80211E00 36003100 36004600 32004400
43004300 30003000 30003000 30003000 30003101 00
0180
Mar  2 23:52:50.797:
Mar  2 23:52:50.797: RAS INCOMING PDU ::=
```

```
value RasMessage ::= registrationRequest : { requestSeqNum 4051 protocolIdentifier { 0 0 8 2250
0 3 } discoveryComplete FALSE callSignalAddress { } rasAddress { ipAddress : { ip 'AC100D2A'H
port 57962 } } terminalType { mc FALSE undefinedNode FALSE } gatekeeperIdentifier {"gkb-2"}
endpointVendor { vendor { t35CountryCode 181 t35Extension 0 manufacturerCode 18 } } timeToLive
60 !--- TTL value. keepAlive TRUE endpointIdentifier {"616F2DCC00000001"} willSupplyUUIEs FALSE
maintainConnection TRUE } Mar  2 23:52:50.805: RRQ (seq# 4051) rcvd Mar  2 23:52:50.805: RAS
OUTGOING PDU ::= value RasMessage ::= registrationConfirm : { requestSeqNum 4051
protocolIdentifier { 0 0 8 2250 0 3 } callSignalAddress { } gatekeeperIdentifier {"gkb-2"}
endpointIdentifier {"616F2DCC00000001"} alternateGatekeeper { { rasAddress ipAddress : { ip
'AC100D29'H port 1719 } gatekeeperIdentifier {"gkb-1"} needToRegister TRUE priority 0 } }
timeToLive 60 willRespondToIRR FALSE maintainConnection TRUE } Mar  2 23:52:50.813: RAS OUTGOING
ENCODE BUFFER ::= 12 400FD206 0008914A 00030008 0067006B 0062002D 00321E00 36003100 3 6004600
```

```
32004400 43004300 30003000 30003000 30003000 3000310F 8A140140 AC100D29 06B70800 67006B00
62002D00 31800200 3B 010001 80 Mar 2 23:52:50.813: Mar 2 23:52:50.817: IPSOCK_RAS_sendto: msg
length 86 from 172.16.13.16:1719 to 172.16.13.42: 57962 Mar 2 23:52:50.817: RASLib::RASSendRCF:
RCF (seq# 4051) sent to 172.16.13.42
```

## Caso 2

Este é um outro exemplo onde um valor-limite que não enviase um valor TTL em seu mensagem de RRQ seja notificado para enviar um RRQ de pouco peso antes de 120 segundos, que seja o valor configurado no porteiro. Você pode ver nesta saída como o gatekeeper Cisco não suprime do valor-limite até três mensagens não respondida de IRQ, mesmo que uma mensagem do Unregistration Request (URQ) seja recebida. Os tempos entre o IRQ realizam-se entre um e cinco minutos.

```
gka-1#show logging Syslog logging: enabled (0 messages dropped, 0 messages rate-limited, 0
flushes, 0 overruns) Console logging: disabled Monitor logging: level debugging, 1076 messages
logged Buffer logging: level debugging, 4257 messages logged Logging Exception size (4096 bytes)
Trap logging: level informational, 60 message lines logged Log Buffer (9999999 bytes): Mar 14
06:28:31.771: RAS INCOMING ENCODE BUFFER::= 0C 80000006 0008914A 00020001 00AB4555 BF06B801
00AB4555 BF05C502 00014007 006B0065 00740070 00610074 0065006C 60B50053 4C164D69 63726F73
6F6674AE 204E6574 4D656574 696E67AE 0003332E 3000 Mar 14 06:28:31.783: Mar 14 06:28:31.787: RAS
INCOMING PDU ::= value RasMessage ::= registrationRequest : { requestSeqNum 1 protocolIdentifier
{ 0 0 8 2250 0 2 } discoveryComplete FALSE callSignalAddress { ipAddress : { ip 'AB4555BF'H port
1720 } } rasAddress { ipAddress : { ip 'AB4555BF'H port 1477 } } terminalType { terminal { } mc
FALSE undefinedNode FALSE } terminalAlias { h323-ID : {"ketpatel"} } endpointVendor { vendor {
t35CountryCode 181 t35Extension 0 manufacturerCode 21324 } productId
'4D6963726F736F6674AE204E65744D656574696E...'H versionId '332E3000'H } } Mar 14 06:28:31.811:
RAS OUTGOING PDU ::= value RasMessage ::= registrationConfirm : { requestSeqNum 1
protocolIdentifier { 0 0 8 2250 0 3 } callSignalAddress { } terminalAlias { h323-ID :
{"ketpatel"} } gatekeeperIdentifier {"gka-1"} endpointIdentifier {"81F6A89800000001"}
alternateGatekeeper { } timeToLive 120 willRespondToIRR FALSE maintainConnection FALSE } Mar 14
06:28:31.823: RAS OUTGOING ENCODE BUFFER::= 12 C0000006 0008914A 00030001 4007006B 00650074
00700061 00740065 006C0800 67006B00 61002D00 311E0038 00310046 00360041 00380039 00380030
00300030 00300030 00300030 00310F8A 01000200 77010001 00 gka-1#show gatekeeper endpoints
GATEKEEPER ENDPOINT REGISTRATION ===== CallSignalAddr Port
RASSignalAddr Port Zone Name Type Flags -----
-- ----- 171.69.85.191 1720 171.69.85.191 1477 gka-1 TERM H323-ID: ketpatel Total number of
active registrations = 1 Mar 14 06:28:31.835: Mar 14 06:28:31.835: RAS OUTGOING PDU ::= value
RasMessage ::= infoRequest : { requestSeqNum 70 callReferenceValue 0 callIdentifier { guid
'00000000000000000000000000000000'H } } Mar 14 06:28:31.839: RAS OUTGOING ENCODE BUFFER::= 56
00004500 000B0011 00000000 00000000 00000000 00000000 00 Mar 14 06:28:31.843: Mar 14
06:28:31.847: RAS INCOMING ENCODE BUFFER::= 58 80004502 03C00038 00310046 00360041 00380039
00380030 00300030 00300030 00300030 003100AB 4555BF05 C50100AB 4555BF06 B8024007 006B0065
00740070 00610074 0065006C 4007006B 00650074 00700061 00740065 006C Mar 14 06:28:31.859: Mar 14
06:28:31.859: RAS INCOMING PDU ::= value RasMessage ::= infoRequestResponse : { requestSeqNum 70
endpointType { terminal { } mc FALSE undefinedNode FALSE } endpointIdentifier
{"81F6A89800000001"} rasAddress ipAddress : { ip 'AB4555BF'H port 1477 } callSignalAddress {
ipAddress : { ip 'AB4555BF'H port 1720 } } endpointAlias { h323-ID : {"ketpatel"}, h323-ID :
{"ketpatel"} } } Mar 14 06:30:42.208: RAS OUTGOING PDU ::= value RasMessage ::= infoRequest : {
requestSeqNum 71 callReferenceValue 0 callIdentifier { guid '00000000000000000000000000000000'H
} } Mar 14 06:30:42.212: RAS OUTGOING ENCODE BUFFER::= 56 00004600 000B0011 00000000 00000000
00000000 00000000 00 Mar 14 06:30:42.216: Mar 14 06:30:42.216: RAS INCOMING ENCODE BUFFER::= 58
80004602 03C00038 00310046 00360041 00380039 00380030 00300030 00300030 00300030 003100AB
4555BF05 C50100AB 4555BF06 B8024007 006B0065 00740070 00610074 0065006C 4007006B 00650074
00700061 00740065 006C Mar 14 06:30:42.228: Mar 14 06:30:42.232: RAS INCOMING PDU ::= value
RasMessage ::= infoRequestResponse : { requestSeqNum 71 endpointType { terminal { } mc FALSE
undefinedNode FALSE } endpointIdentifier {"81F6A89800000001"} rasAddress ipAddress : { ip
'AB4555BF'H port 1477 } callSignalAddress { ipAddress : { ip 'AB4555BF'H port 1720 } }
endpointAlias { h323-ID : {"ketpatel"}, h323-ID : {"ketpatel"} } } Mar 14 06:32:05.938: RAS
INCOMING ENCODE BUFFER::= 19 40000101 00AB4555 BF06B802 4007006B 00650074 00700061 00740065
006C4007 006B0065 00740070 00610074 0065006C 1E003800 31004600 36004100 38003900 38003000
30003000 30003000 30003000 31 Mar 14 06:32:05.950: Mar 14 06:32:05.950: RAS INCOMING PDU ::=
```

```

value RasMessage ::= unregistrationRequest : { requestSeqNum 2 callSignalAddress { ipAddress : {
ip 'AB4555BF'H port 1720 } } endpointAlias { h323-ID : {"ketpatel"}, h323-ID : {"ketpatel"} }
endpointIdentifier {"81F6A89800000001"} } Mar 14 06:32:05.962: RAS OUTGOING PDU ::= value
RasMessage ::= unregistrationConfirm : { requestSeqNum 2 } Mar 14 06:32:05.962: RAS OUTGOING
ENCODE BUFFER ::= 1C 0001 Mar 14 06:32:05.966: gka-1#show gatekeeper endpoints GATEKEEPER
ENDPOINT REGISTRATION ===== CallSignalAddr Port RASignalAddr Port
Zone Name Type Flags -----
171.69.85.191 1720 171.69.85.191 1477 gka-1 TERM Total number of active registrations = 1 Mar 14
06:33:42.223: RAS OUTGOING PDU ::= value RasMessage ::= infoRequest : { requestSeqNum 72
callReferenceValue 0 callIdentifier { guid '00000000000000000000000000000000'H } } Mar 14
06:33:42.227: RAS OUTGOING ENCODE BUFFER ::= 56 00004700 000B0011 00000000 00000000 00000000
00000000 00 Mar 14 06:33:42.231: Mar 14 06:34:42.234: RAS OUTGOING PDU ::= value RasMessage ::=
infoRequest : { requestSeqNum 73 callReferenceValue 0 callIdentifier { guid
'00000000000000000000000000000000'H } } Mar 14 06:34:42.238: RAS OUTGOING ENCODE BUFFER ::= 56
00004800 000B0011 00000000 00000000 00000000 00000000 00 Mar 14 06:34:42.242: Mar 14
06:35:42.244: RAS OUTGOING PDU ::= value RasMessage ::= infoRequest : { requestSeqNum 74
callReferenceValue 0 callIdentifier { guid '00000000000000000000000000000000'H } } Mar 14
06:35:42.248: RAS OUTGOING ENCODE BUFFER ::= 56 00004900 000B0011 00000000 00000000 00000000
00000000 00 Mar 14 06:35:42.252: gka-1# gka-1#show gatekeeper endpoints GATEKEEPER ENDPOINT
REGISTRATION ===== CallSignalAddr Port RASignalAddr Port Zone Name
Type Flags ----- Total number of
active registrations = 0

```

## [Informações Relacionadas](#)

- [Gatekeeper de alto desempenho Cisco](#)
- [Suporte H.323 versão 2](#)
- [Troubleshooting de Problemas com Registro de Gatekeeper](#)
- [Suporte à Tecnologia de Voz](#)
- [Suporte ao Produto de Voz e Comunicações Unificadas](#)
- [Troubleshooting da Telefonia IP Cisco](#)
- [Suporte Técnico - Cisco Systems](#)