Compreendendo e Troubleshooting de Gatekeeper TTL e Processo de Envelhecimento

Contents

Introduction

Este documento descreve como o Cisco Gatekeeper separa os endpoints usando o valor Time to Live (TTL) em diferentes casos. Depurações e comandos **show** são usados para mostrar como o TTL funciona de várias maneiras.

Prerequisites

Requirements

Leitores deste documento devem estar cientes destes tópicos:

- Implementação do Cisco H.323, incluindo o Cisco Gatekeeper. Para obter uma compreensão básica dos Gatekeepers H.323, consulte Entendendo Gatekeepers H.323.

Componentes Utilizados

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

- Para a finalidade deste documento, o software Cisco IOS® versão 12.3(9) é usado para coletar as informações.Certifique-se de usar o Cisco IOS com o recurso de funcionalidade do gatekeeper H.323. Isso é indicado com um **x** no nome da imagem do Cisco IOS. Por exemplo, um Cisco IOS válido para o Cisco 3640 para atuar como um gatekeeper é c3640-**ix**-mz.123-9.bin.
- Cisco Gatekeeper (todas as plataformas).**Observação:** o NetMeeting foi usado como um endpoint H.323 no exemplo deste documento porque não fornece um valor TTL.Para obter informações sobre como configurar o NetMeeting com os gateways do Cisco IOS, consulte Como configurar o Microsoft NetMeeting com os gateways do Cisco IOS.

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. All of the devices used in this document started with a cleared (default) configuration. Se você estiver trabalhando em uma rede ativa, certifique-se de que entende o impacto potencial de qualquer comando antes de utilizá-lo.

Conventions

Para obter mais informações sobre convenções de documento, consulte as Convenções de dicas técnicas Cisco.

Dicas sobre tempo para atividade

Estas são algumas dicas sobre como o TTL funciona em um Cisco Gatekeeper e como o processo de envelhecimento funciona para os endpoints em diferentes casos.

- O Cisco Gatekeeper espera ouvir o ponto final periodicamente através das mensagens de Solicitação de Registro (RRQ) (leve ou cheio).
- Para timeouts de ponto de extremidade, o Cisco Gatekeeper presta atenção ao valor TTL fornecido pelo ponto de extremidade na mensagem RRQ. Há um padrão codificado de 1800 segundos (trinta minutos) para o tempo limite do ponto final. Esse valor pode ser alterado com o comando Cisco Gatekeeper CLI **endpoint ttl** ***<time_value>*** . Isso altera o comportamento de todos os endpoints H.323 v1 ou H.323 v2 e posteriores que não incluem o valor TTL na mensagem RRQ.
- O Cisco Gatekeeper executa periodicamente um "processo de envelhecimento de endpoint". Esse processo varia de acordo com a carga atual da CPU de cada minuto a cada cinco minutos. Para cada 20% da utilização da CPU, o intervalo de envelhecimento aumenta em um minuto, até um máximo de cinco minutos. Para não sobrecarregar a CPU quando há muitos endpoints, o processo de envelhecimento é executado somente por cinquenta endpoints por passagem. Se houver mais, eles serão adiados até o próximo pop-up do temporizador. Pode ser de um a cinco minutos.
- Se o campo timeToLive estiver incluído na mensagem RAS (Registro, Admissão e Status) do RRQ, o gatekeeper usa o valor desse campo para substituir o padrão do sistema ou o valor configurado usando o comando CLI do gatekeeper **end point ttl** ***<time_value>.*** Quando esse período expira sem ouvir o endpoint, o próximo pop-up do temporizador passa pelo processo de limpeza desse endpoint. O pior caso é o TTL enviado pelo endpoint mais cinco minutos (se o Cisco Gatekeeper estiver consistentemente sob carga pesada de CPU). Um cenário mais provável é o tempo limite TTL mais um minuto.
- Quando o ponto final não inclui o campo timeToLive na mensagem RRQ, o Cisco Gatekeeper trata o ponto final como se ele não suportasse TTL. Nesse caso, quando o gatekeeper não receber mais RRQs desse endpoint, ele faz o tempo limite de TTL (o padrão de 1800 segundos ou o valor especificado no comando **endpoint ttl**). Em seguida, ele envia três Solicitações de Informações (IRQs) com intervalos de um a cinco minutos cada (com base na carga da CPU do gatekeeper). Depois de enviar três IRQs e não receber nenhuma resposta, o Cisco Gatekeeper finalmente remove o ponto final.
- Se o endpoint estiver em uma chamada ativa, ele não ficará ultrapassado até que a chamada seja encerrada.
- O Cisco Gatekeeper espera ouvir do(s) endpoint(s) envolvido(s) em uma chamada com a mensagem de Resposta de Informações (IRR). Se o Cisco Gatekeeper não receber

mensagens de IRR periódicas que contenham referências ao "guid" de uma chamada, o gatekeeper espera por quatro minutos e, em seguida, envia um IRQ para o(s) endpoint(s) para o qual a chamada é destinada. Se, após mais oito minutos, o Cisco Gatekeeper ainda não tiver ouvido nada sobre essa chamada, a chamada será limpa e o gatekeeper enviará solicitações de desconexão (DRQs) aos pontos finais. Um total de aproximadamente doze minutos decorrido antes que uma chamada "em espera" seja limpa (e a largura de banda seja liberada). Este temporizador de chamada não é configurável.

- Os endpoints que são de propriedade de um Cisco Gatekeeper alternativo não são obsoletos diretamente (já que esse gatekeeper não é "proprietário" do endpoint).
- Os endpoints estáticos (criados com o comando de configuração alias static **xxxxx**) não estão obsoletos.

Depurações do Cisco Gatekeeper

Estes são alguns dos comandos **show** e **debug** que você pode usar para verificar como o TTL funciona de várias maneiras:

- show gatekeeper endpoints
- debug ras
- debug h225 asn1

Essas duas seções discutem dois casos em que o Cisco Gatekeeper separa os endpoints usando valores TTL diferentes.

Caso 1

Essa saída é dos comandos debug ras e debug h225 asn1 e é obtida de um Cisco Gatekeeper. Na depuração, o gateway tem um valor TTL de 60 segundos. O Cisco Gatekeeper confirma e aceita isso em sua mensagem de confirmação de registro (RCF), independentemente do valor TTL padrão ou configurado. Isso porque o endpoint 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 outro exemplo em que um ponto final que não enviou um valor TTL em sua mensagem RRQ foi notificado para enviar um RRQ leve antes de 120 segundos, que é o valor configurado no gatekeeper. Você pode ver nesta saída como o Cisco Gatekeeper não exclui o ponto final até que três mensagens IRQ não atendidas, mesmo que uma mensagem de solicitação de cancelamento de registro (URQ) tenha sido recebida. Os tempos entre o IRQ estão 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  RASSignalAddr   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  RASSignalAddr   Port  Zone Name         Type    Flags
--------------- ----- --------------- ----- ---------         ----    -----
Total number of active registrations = 0
```

Informações Relacionadas

- Gatekeeper de alto desempenho Cisco
- Suporte para H.323, Versão 2
- Troubleshooting de Problemas com Registro de Gatekeeper
- Suporte à Tecnologia de Voz
- Suporte aos produtos de Voz e Comunicações Unificadas
- Troubleshooting da Telefonia IP Cisco
- Suporte Técnico - Cisco Systems