"Understanding and Troubleshooting Gatekeeper TTL and Aging Out Process"

Inhalt

Einführung

In diesem Dokument wird beschrieben, wie der Cisco Gatekeeper die Endpunkte in verschiedenen Fällen mithilfe des TTL-Werts (Time to Live) abruft. Debugs- und **Show**-Befehle werden verwendet, um zu zeigen, wie die TTL auf verschiedene Weise funktioniert.

Voraussetzungen

Anforderungen

Die Leser dieses Dokuments sollten über folgende Themen Bescheid wissen:

- Cisco H.323-Implementierung, einschließlich Cisco Gatekeeper. Ein grundlegendes Verständnis von H.323-Gatekeepers finden Sie in Understanding H.323 Gatekeepers.

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf diesen Software- und Hardwareversionen.

- Für die Zwecke dieses Dokuments werden die Informationen in der Cisco IOS® Software Release 12.3(9) erfasst.Stellen Sie sicher, dass Sie Cisco IOS mit der H.323-Gatekeeper-Funktion verwenden. Dies wird mit einem **x** im Cisco IOS-Image-Namen gekennzeichnet. Ein gültiges Cisco IOS für den Cisco 3640, das als Gatekeeper fungieren soll, ist z. B. c3640-**ix**-mz.123-9.bin.
- Cisco Gatekeeper (alle Plattformen).**Hinweis:** NetMeeting wurde im Beispiel in diesem Dokument als H.323-Endpunkt verwendet, da kein TTL-Wert angegeben wird.Weitere Informationen zum Konfigurieren von NetMeeting mit Cisco IOS-Gateways finden Sie unter Anleitung zum Konfigurieren von Microsoft NetMeeting mit Cisco IOS-Gateways.

Die in diesem Dokument enthaltenen Informationen wurden aus Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Sie in einem Live-Netzwerk arbeiten, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen, bevor Sie es verwenden.

Konventionen

Weitere Informationen zu Dokumentkonventionen finden Sie in den Cisco Technical Tips Conventions.

Tipps zur Live-Übertragung

Dies sind einige Tipps, wie TTL auf einem Cisco Gatekeeper funktioniert und wie der Alterungsprozess für die Endgeräte in verschiedenen Fällen funktioniert.

- Der Cisco Gatekeeper erwartet, vom Endgerät regelmäßig über die Registrierungsanfrage (RRQ)-Nachrichten (leichtgewichtig oder voll) zu hören.
- Bei Timeouts für Endpunkte achtet der Cisco Gatekeeper auf den TTL-Wert, der vom Endpunkt in der RRQ-Nachricht angegeben wird. Es gibt einen fest codierten Standardwert von 1800 Sekunden (dreißig Minuten) für das Endpunkt-Timeout. Dieser Wert kann mithilfe der CLI des Cisco Gatekeeper-Befehls **Endpoint ttl<time_value>** geändert werden. Dadurch wird das Verhalten für alle H.323 v1-Endpunkte oder H.323 v2- und spätere Endpunkte geändert, die den TTL-Wert nicht in der RQ-Nachricht enthalten.
- Cisco Gatekeeper führt regelmäßig einen "Endpunkt-Alterungsprozess" durch. Dieser Prozess variiert je nach aktueller CPU-Last von einer Minute bis alle fünf Minuten. Bei jeweils zwanzig Prozent der CPU-Auslastung nimmt das Alterungsintervall um eine Minute bis zu maximal fünf Minuten zu. Um die CPU bei vielen Endpunkten nicht zu überlasten, wird der Alterungsprozess pro Durchgang nur auf 50 Endpunkte durchgeführt. Wenn es mehr gibt, werden diese bis zum nächsten Timer-Popup zurückgestellt. Dies kann zwischen einer und fünf Minuten dauern.
- Wenn das Feld "timeToLive" in der RAS-Meldung (RRQ Registration, Admission and Status) enthalten ist, verwendet der Gatekeeper diesen Feldwert, um den Systemstandard oder den Wert zu überschreiben, der mit dem Befehl **endpoint ttl *<time_value>*gatekeeper CLI** konfiguriert wurde. Wenn dieser Zeitraum abgelaufen ist, ohne vom Endpunkt abgehört zu werden, durchläuft das nächste Timer-Popup den Bereinigungsprozess für diesen Endpunkt. Der schlimmste Fall ist die vom Endpunkt gesendete TTL plus fünf Minuten (wenn der Cisco Gatekeeper durchgehend unter hoher CPU-Last liegt). Ein wahrscheinlicheres Szenario ist das TTL-Timeout plus eine Minute.
- Wenn der Endpunkt das Feld "timeToLive" nicht in die RRQ-Nachricht einbindet, behandelt der Cisco Gatekeeper den Endpunkt so, als ob er TTL nicht unterstützt. Wenn der Gatekeeper in diesem Fall RRQs von diesem Endpunkt nicht mehr empfängt, führt er das TTL-Timeout aus (entweder der Standardwert von 1800 Sekunden oder der Wert, der im Befehl endpoint ttl angegeben ist). Anschließend sendet er drei Information Requests (IRQs) mit Intervallen von jeweils ein bis fünf Minuten (abhängig von der CPU-Last des Gatekeeper). Nachdem drei IRQs gesendet wurden und keine Antwort empfangen wurde, entfernt der Cisco Gatekeeper das Endgerät.
- Befindet sich das Endgerät in einem aktiven Anruf, wird es erst ausgeschaltet, wenn der Anruf beendet wurde.

- Der Cisco Gatekeeper erwartet, von den Endpunkten, die an einem Anruf mit der Information Response (IRR)-Nachricht beteiligt sind, gehört zu werden. Wenn der Cisco Gatekeeper keine periodischen IRR-Nachrichten mit Verweisen auf die "GUID" für einen Anruf empfängt, wartet der Gatekeeper vier Minuten und sendet dann eine IRQ an die Endpunkte, für die der Anruf bestimmt ist. Wenn der Cisco Gatekeeper nach acht weiteren Minuten noch nichts von diesem Anruf gehört hat, wird der Anruf gelöscht, und der Gatekeeper sendet Disengagement Requests (DRQs) an die Endpunkte. Insgesamt sind etwa zwölf Minuten vergangen, bevor ein "gefährliches" Gespräch bereinigt wird (und die Bandbreite frei wird). Dieser Anruf-Timer ist nicht konfigurierbar.
- Endpunkte, die einem alternativen Cisco Gatekeeper gehören, werden nicht direkt veraltet (da dieser Gatekeeper den Endpunkt nicht "besitzt").
- Statische Endpunkte (die mit dem Konfigurationsbefehl alias static **xxxxx** erstellt wurden) werden nicht ausgereift.

Cisco Gatekeeper-Debugger

Dies sind einige der Befehle **show** und **debug**, mit denen Sie überprüfen können, wie die TTL funktioniert:

- Gatekeeper-Endpunkte anzeigen
- Debug ras
- debug h225 asn1

In diesen beiden Abschnitten werden zwei Fälle erläutert, in denen Cisco Gatekeeper die Endpunkte mithilfe verschiedener TTL-Werte abruft.

Fall 1

Diese Ausgabe stammt aus den **Befehlen** debug ras und **debug h**225 asn1 und wird von einem Cisco Gatekeeper übernommen. Beim Debuggen hat das Gateway einen TTL-Wert von 60 Sekunden. Der Cisco Gatekeeper bestätigt und akzeptiert dies in seiner RCF-Nachricht (Registration Confirmation), unabhängig von dem standardmäßigen oder konfigurierten TTL-Wert des Endpunkts. Dies liegt daran, dass der Endpunkt einen TTL-Wert enthält.

```
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
```

Fall 2

Dies ist ein weiteres Beispiel, bei dem ein Endpunkt, der in seiner RRQ-Nachricht keinen TTL-Wert gesendet hat, benachrichtigt wurde, dass er vor 120 Sekunden einen einfachen RRQ sendet. Dies ist der auf dem Gatekeeper konfigurierte Wert. In dieser Ausgabe sehen Sie, wie der Cisco Gatekeeper den Endpunkt erst dann löscht, wenn drei ungeantwortete IRQ-Nachrichten empfangen wurden, obwohl eine Nachricht für eine Unregistration Request (URQ) eingegangen ist. Die Zeiten zwischen dem IRQ liegen zwischen ein und fünf Minuten.

```
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
```

Zugehörige Informationen

- Cisco High-Performance Gatekeeper
- Unterstützung für H.323 Version 2
- Fehlerbehebung bei Gatekeeper-Registrierungsproblemen
- Unterstützung von Sprachtechnologie
- Produkt-Support für Sprach- und Unified Communications
- Fehlerbehebung bei Cisco IP-Telefonie
- Technischer Support - Cisco Systems