Grundlegende Informationen über Class-Based Weighted Fair Queuing auf ATM

Inhalt

Einführung

Dieses Dokument bietet eine Einführung in die Warteschlangenverwaltung von Datenverkehr mithilfe der CBWFQ-Technologie (Class-Based Weighted Fair Queuing).

Weighted Fair Queuing (WFQ) ermöglicht langsame Verbindungen, wie z. B. serielle Verbindungen, um eine faire Behandlung aller Arten von Datenverkehr zu gewährleisten. Er klassifiziert den Datenverkehr in verschiedene Datenflüsse (auch als Kommunikation bezeichnet), die auf Informationen auf Layer 3 und Layer 4 basieren, z. B. IP-Adressen und TCP-Ports. Dies geschieht, ohne dass Sie Zugriffslisten definieren müssen. Dies bedeutet, dass Datenverkehr mit niedriger Bandbreite gegenüber Datenverkehr mit hoher Bandbreite Priorität hat, da Datenverkehr mit hoher Bandbreite die Übertragungsmedien im Verhältnis zum zugewiesenen Gewicht teilt. WFQ weist jedoch gewisse Einschränkungen auf:

- Es ist nicht skalierbar, wenn der Flow-Betrag erheblich zunimmt.
- Natives WFQ ist für Hochgeschwindigkeitsschnittstellen wie ATM-Schnittstellen nicht verfügbar.

CBWFQ bietet eine Lösung für diese Einschränkungen. Im Gegensatz zu Standard-WFQ können Sie mit CBWFQ Datenverkehrsklassen definieren und Parameter wie Bandbreite und Warteschlangenbeschränkungen auf diese Klassen anwenden. Die Bandbreite, die Sie einer Klasse zuweisen, wird verwendet, um das "Gewicht" dieser Klasse zu berechnen. Daraus wird auch das Gewicht jedes Pakets errechnet, das den Klassenkriterien entspricht. WFQ wird auf die Klassen (die mehrere Flüsse enthalten können) statt auf die Flüsse selbst angewendet.

Weitere Informationen zur Konfiguration von CBWFQ erhalten Sie, wenn Sie auf die folgenden Links klicken:

VC Class-Based Weighted Fair Queuing (Per-VC CBWFQ) auf Cisco Routern der Serien 7200, 3600 und 2600

VC Class-Based Weighted Fair Queuing auf RSP-basierten Plattformen.

Bevor Sie beginnen

Konventionen

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

Voraussetzungen

Für dieses Dokument bestehen keine besonderen Voraussetzungen.

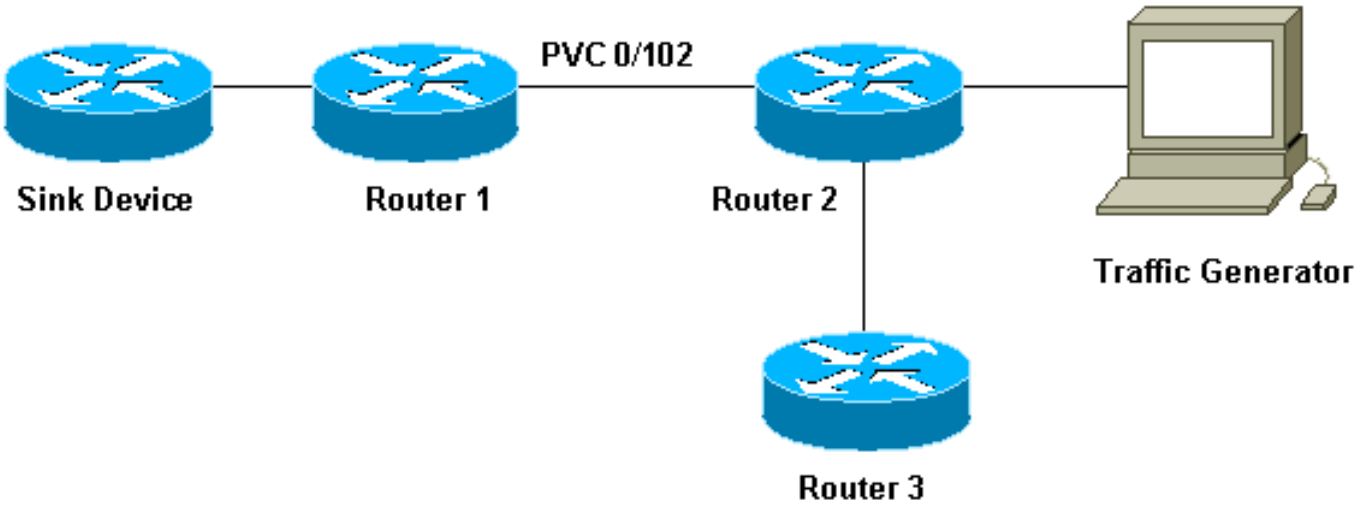
Verwendete Komponenten

Dieses Dokument ist nicht auf bestimmte Software- und Hardwareversionen beschränkt.

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.

Netzwerkdiagramm

Um zu veranschaulichen, wie WFQ funktioniert, verwenden wir die folgende Konfiguration:



In der hier verwendeten Konfiguration können Pakete in einer der beiden folgenden Warteschlangen gespeichert werden:

- Die Hardware First In First Out (FIFO)-Warteschlange am Port-Adapter und Netzwerkmodul.
- Die Warteschlange in der Cisco IOS® Software (im E/A-Speicher des Routers), in der Quality of Service (QoS)-Funktionen wie CBWFQ angewendet werden können.

Die FIFO-Warteschlange auf dem Port-Adapter speichert die Pakete, bevor sie zur Übertragung in Zellen segmentiert werden. Wenn diese Warteschlange voll ist, signalisiert der Port-Adapter oder das Netzwerkmodul der IOS-Software, dass die Warteschlange überlastet ist. Dieser Mechanismus wird als Rückdruck bezeichnet. Beim Empfang dieses Signals beendet der Router das Senden von Paketen an die FIFO-Warteschlange für die Schnittstelle und speichert die Pakete in der IOS-Software, bis die Warteschlange wieder nicht überlastet wird. Wenn die Pakete in IOS gespeichert werden, kann das System QoS-Funktionen wie CBWFQ anwenden.

Festlegen der Klingelgrenze für die Übertragung

Ein Problem bei diesem Warteschlangenmechanismus ist, dass die Verzögerung, bevor Pakete am Ende dieser Warteschlange übertragen werden können, umso größer ist, je größer die FIFO-Warteschlange an der Schnittstelle ist. Dies kann bei verzögerungsempfindlichem Datenverkehr, z. B. beim Sprachverkehr, zu schwerwiegenden Leistungsproblemen führen.

Mit dem Befehl Permanent Virtual Circuit (PVC) **tx-ring-limit** können Sie die Größe der FIFO-Warteschlange reduzieren.

```
interface ATMx/y.z point-to-point
      ip address a.b.c.d M.M.M.M
      PVC A/B
       TX-ring-limit
       service-policy output test
```

Das Limit (x), das Sie hier angeben können, ist entweder eine Anzahl von Paketen (für Cisco 2600- und 3600-Router) oder eine Menge von Partikeln (für Cisco 7200- und 7500-Router).

Die Verringerung der Größe des Übertragungsringes bietet zwei Vorteile:

- Dadurch wird die Wartezeit von Paketen in der FIFO-Warteschlange reduziert, bevor sie

segmentiert werden.
- Sie beschleunigt die Nutzung der QoS in der IOS-Software.

Auswirkungen des Grenzwerts für den Übertragungsring

Betrachten wir nun die Auswirkungen der Beschränkung des Übertragungsringes anhand der Konfiguration, die in unserem Netzwerkdiagramm oben dargestellt ist. Dabei kann von folgenden Faktoren ausgegangen werden:

- Der Datenverkehrsgenerator sendet Datenverkehr (1500-Byte-Pakete) an das Empfängergerät, und dieser Datenverkehr überlastet die PVC 0/102 zwischen Router1 und Router2.
- Router3 versucht, Router1 zu pingen.
- CBWFQ ist auf Router 2 aktiviert.

Sehen wir uns nun zwei Konfigurationen an, bei denen unterschiedliche Übertragungsringgrenzen verwendet werden, um die Auswirkungen zu erkennen.

Beispiel A

In diesem Beispiel haben wir den Übertragungsring auf drei eingestellt (**TX-Ring-limit=3**). Hier sehen wir den Ping-Befehl für Router1 von Router 3.

```
POUND#ping ip
     Target IP address: 6.6.6.6
     Repeat count [5]: 100000
     Datagram size [100]:
     Timeout in seconds [2]:
     Extended commands [n]:
     Sweep range of sizes [n]:
     Type escape sequence to abort.
     Sending 100000, 100-byte ICMP Echos to 6.6.6.6, timeout is 2 seconds:
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!![snip]
     Success rate is 98 percent (604/613), round-trip min/avg/max = 164/190/232 ms
```

Beispiel B

In diesem Beispiel haben wir den Übertragungsring auf 40 eingestellt (**TX-Ring-limit=40**). Folgendes wird angezeigt, wenn der gleiche Ping verwendet wird wie in Beispiel A:

```
POUND#ping ip
     Target IP address: 6.6.6.6
     Repeat count [5]: 10000
     Datagram size [100]: 36
     Timeout in seconds [2]: 10
     Extended commands [n]:
     Sweep range of sizes [n]:
     Type escape sequence to abort.
```

```
Sending 10000, 36-byte ICMP Echos to 6.6.6.6, timeout is 10 seconds:
!!!!!!!!!!!!.
Success rate is 92 percent (12/13), round-trip min/avg/max = 6028/6350/6488
```

Wie Sie hier sehen können, je größer der Grenzwert für den Übertragungsring ist, desto größer ist die Ping-Round-Trip-Zeit (RTT). Daraus können wir schließen, dass ein großer Übertragungsring zu erheblichen Verzögerungen bei der Übertragung führen kann.

Funktionsweise von CBWFQ

Nachdem wir nun die Auswirkungen der Größe der Hardware-FIFO-Warteschlange gesehen haben, wollen wir uns genauer ansehen, wie CBWFQ funktioniert.

Native WFQ weist jedem Gespräch ein Gewicht zu und plant dann die Übertragungszeit für jedes Paket der verschiedenen Datenflüsse. Das Gewicht ist eine Funktion der IP-Rangfolge jedes Datenflusses, und die Planungszeit hängt von der Paketgröße ab. Klicken Sie hier, um weitere Informationen zu WFQ anzuzeigen.

CBWFQ weist jeder konfigurierten Klasse anstatt jedem Datenfluss ein Gewicht zu. Dieses Gewicht ist proportional zur für jede Klasse konfigurierten Bandbreite. Genauer gesagt ist das Gewicht eine Funktion der Schnittstellenbandbreite geteilt durch die Klassenbandbreite. Daher gilt: Je größer der Parameter "bandwidth" ist, desto geringer das Gewicht.

Die Paketplanungszeit kann mit der folgenden Formel berechnet werden:

```
scheduling tail_time= queue_tail_time + pktsize * weight
```

Gesamtbandbreite der Schnittstelle

Sehen wir uns an, wie der Router die gesamte Schnittstellenbandbreite zwischen den verschiedenen Klassen aufteilt. Um die Klassen zu bedienen, verwendet der Router Kalenderwarteschlangen. Jede dieser Kalenderwarteschlangen speichert Pakete, die mit derselben Scheduling_tail_time übertragen werden müssen. Der Router verarbeitet diese Kalenderwarteschlangen nacheinander. Sehen wir uns diesen Prozess an:

1. Wenn eine Überlastung des Port-Adapters auftritt, wenn ein Paket an der Ausgabeschnittstelle eingeht, führt dies zu Warteschlangen in IOS (in diesem Fall CBWFQ).
2. Der Router berechnet eine Planungszeit für dieses ankommende Paket und speichert es in der Kalenderwarteschlange, die dieser Planungszeit entspricht. Pro Klasse kann nur ein Paket in einer bestimmten Kalenderwarteschlange gespeichert werden.
3. Wenn es an der Zeit ist, die Kalenderwarteschlange zu warten, in der das Paket gespeichert wurde, löscht das IOS diese Warteschlange und sendet die Pakete an die FIFO-Warteschlange am Port-Adapter selbst. Die Größe dieser FIFO-Warteschlange wird durch den oben beschriebenen Grenzwert für den Übertragungsring bestimmt.
4. Wenn die FIFO-Warteschlange zu klein ist, um alle in der Warteschlange des gewarteten Kalenders gespeicherten Pakete zu übernehmen, plant der Router die Pakete, die für die nächste Planungszeit nicht gespeichert werden können (entsprechend ihres Gewichts), neu und ordnet sie in die entsprechende Kalenderwarteschlange ein.
5. Wenn all dies geschieht, verarbeitet der Port-Adapter die Pakete in seiner FIFO-

Warteschlange und sendet die Zellen in der Leitung, und das IOS wechselt zur nächsten Kalenderwarteschlange.Dank dieses Mechanismus erhält jede Klasse statistisch einen Teil der Schnittstellenbandbreite, der den für sie konfigurierten Parametern entspricht.

Calendar Queue Mechanism und Transmit Ring Size (Ringgröße übertragen)

Betrachten wir nun die Beziehung zwischen dem Kalender-Warteschlangenmechanismus und der Größe des Übertragungs-Rings. Mit einem kleinen Übertragungsring kann die QoS schneller gestartet werden, und die Latenz der Pakete, die auf die Übertragung warten, wird reduziert (was für verzögerungsempfindlichen Datenverkehr wie Sprache wichtig ist). Wenn sie jedoch zu klein ist, kann sie für bestimmte Klassen einen niedrigeren Durchsatz verursachen. Dies liegt daran, dass viele Pakete neu geplant werden müssen, wenn der Übertragungsring sie nicht aufnehmen kann.

Es gibt leider keinen idealen Wert für die Größe des Senderringes, und der einzige Weg, den besten Wert zu finden, ist das Experimentieren.

Bandbreitenfreigabe

Wir können das Konzept der Bandbreitenfreigabe mit der Konfiguration betrachten, die in unserem Netzwerkdiagramm oben dargestellt ist. Der Paketgenerator erzeugt verschiedene Flows und sendet sie an das Empfängergerät. Die Gesamtverkehrsmenge, die durch diese Datenflüsse erzeugt wird, reicht aus, um die PVC zu überlasten. Wir haben CBWFQ auf Router 2 implementiert. So sieht unsere Konfiguration aus:

```
access-list 101 permit ip host 7.0.0.200 any
   access-list 101 permit ip host 7.0.0.201 any
   access-list 102 permit ip host 7.0.0.1 any
   !
   class-map small
     match access-group 101
   class-map big
     match access-group 102
   !
   policy-map test
   policy-map test
     small class
      bandwidth <x>
     big class
      bandwidth <y>
    interface atm 4/0.102
     pvc 0/102
       TX-ring-limit 3
       service-policy output test
       vbr-nrt 64000 64000
```

In unserem Beispiel ist Router2 ein Cisco 7200-Router. Dies ist wichtig, da der Grenzwert für den Übertragungsring in Partikeln ausgedrückt wird, nicht in Paketen. Pakete werden in der FIFO-Warteschlange des Port-Adapters in die Warteschlange gestellt, sobald ein freies Teilchen verfügbar ist, selbst wenn zum Speichern des Pakets mehr als ein Teilchen erforderlich ist.

Was ist ein Artikel?

Anstatt einen zusammenhängenden Speicher für einen Puffer zu reservieren, weist die

Teilchenpufferung unzusammenhängende (verstreute) Speicherstücke, so genannte Partikel, zu und verbindet sie dann zu einem logischen Paketpuffer. Dies wird als Teilchenpuffer bezeichnet. In einem solchen Schema kann ein Paket dann auf mehrere Partikel verteilt werden.

Im 7200-Router, den wir hier verwenden, beträgt die Partikelgröße 512 Byte.

Mit dem Befehl **show buffers** können wir überprüfen, ob die Cisco 7200 Router Partikel verwenden:

```
router2#show buffers
     [snip]
     Private particle pools:
     FastEthernet0/0 buffers, 512 bytes (total 400, permanent 400):
          0 in free list (0 min, 400 max allowed)
          400 hits, 0 fallbacks
          400 max cache size, 271 in cache
     ATM4/0 buffers, 512 bytes (total 400, permanent 400):
          0 in free list (0 min, 400 max allowed)
          400 hits, 0 fallbacks
          400 max cache size, 0 in cache
```

Test A

Die für diesen Test verwendeten Klassen "Klein" und "Groß" werden wie folgt aufgefüllt:

- Kleine Klasse - wir haben die Bandbreitenparameter auf 32 Kbit/s konfiguriert. Diese Klasse speichert zehn Pakete mit 1.500 Byte von 7.0.0.200, gefolgt von zehn Paketen mit 1.500 Byte ab 7.0.0.201.
- Große Klasse - wir haben den Bandbreitenparameter auf 16 Kbit/s konfiguriert. Diese Klasse speichert einen Fluss von zehn 1500-Byte-Paketen von 7.0.0.1.

Der Datenverkehrsgenerator sendet eine Datenverkehrslast, die für das Empfängergerät bestimmt ist, in der folgenden Reihenfolge mit 100 Mbit/s an Router2:

1. Zehn Pakete von 7.0.0.1.
2. Zehn Pakete ab 7.0.0.200.
3. Zehn Pakete ab 7.0.0.201.

Überprüfen des Flussgewichts

Sehen wir uns das Gewicht an, das auf die verschiedenen Ströme angewendet wird. Dazu können wir den Befehl **show queue ATM x/y.z** verwenden.

```
alcazaba#show queue ATM 4/0.102
     Interface ATM4/0.102 VC 0/102
     Queueing strategy: weighted fair
     Total output drops per VC: 0
     Output queue: 9/512/64/0 (size/max total/threshold/drops)
        Conversations  2/3/16 (active/max active/max total)
        Reserved Conversations 2/2 (allocated/max allocated)

  (depth/weight/total drops/no-buffer drops/interleaves) 7/128/0/0/0
     Conversation 25, linktype: ip, length: 1494
     source: 7.0.0.201, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
```

```
  (depth/weight/total drops/no-buffer drops/interleaves) 2/256/0/0/0
     Conversation 26, linktype: ip, length: 1494
     source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
```

Wenn alle Pakete von 7.0.0.200 aus dem Router in die Warteschlange gestellt wurden, wird Folgendes angezeigt:

```
alcazaba#show queue ATM 4/0.102
     Interface ATM4/0.102 VC 0/102
     Queueing strategy: weighted fair
     Total output drops per VC: 0
     Output queue: 9/512/64/0 (size/max total/threshold/drops)
        Conversations  2/3/16 (active/max active/max total)
        Reserved Conversations 2/2 (allocated/max allocated)

  (depth/weight/total drops/no-buffer drops/interleaves) 7/128/0/0/0
     Conversation 25, linktype: ip, length: 1494
     source: 7.0.0.201, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255

  (depth/weight/total drops/no-buffer drops/interleaves) 2/256/0/0/0
     Conversation 26, linktype: ip, length: 1494
     source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
```

Wie Sie hier sehen können, haben die Ströme von 7.0.0.200 und 7.0.0.201 das gleiche Gewicht (128). Dieses Gewicht entspricht der Hälfte des Gewichts, das dem Fluss von 7.0.0.1 (256) zugewiesen wurde. Dies entspricht der Tatsache, dass unsere Bandbreite für kleine Klassen doppelt so groß ist wie unsere große Klasse.

Überprüfen der Bandbreitenverteilung

Wie können wir die Bandbreitenverteilung zwischen den verschiedenen Datenflüssen überprüfen? Die FIFO-Warteschlangenmethode wird in jeder Klasse verwendet. Unsere kleine Klasse ist mit zehn Paketen aus dem ersten und zehn Paketen aus dem zweiten Datenfluss gefüllt. Der erste Datenfluss wird aus der kleinen Klasse mit 32 Kbit/s entfernt. Sobald sie gesendet wurden, werden auch die zehn Pakete aus dem anderen Datenfluss gesendet. In der Zwischenzeit werden Pakete aus unserer großen Klasse mit 16 Kbit/s entfernt.

Da der Datenverkehrsgenerator einen Burst von 100 Mbit/s sendet, wird die PVC überladen. Da jedoch beim Start des Tests kein Datenverkehr auf dem PVC vorhanden ist und da die Pakete von 7.0.0.1 die ersten sind, die den Router erreichen, werden einige Pakete von 7.0.0.1 vor dem CBWFQ-Start aufgrund von Überlastung (d. h. bevor der Übertragungs-Ring voll ist) gesendet.

Da die Partikelgröße 512 Byte beträgt und der Übertragungsring drei Partikel enthält, werden zwei Pakete von 7.0.0.1 gesendet, bevor eine Überlastung auftritt. Eine wird sofort an das Kabel gesendet, die andere in den drei Teilchen, die die FIFO-Warteschlange für den Port-Adapter bilden.

Die folgenden DebuggingInnen werden auf dem Empfängergerät (also lediglich einem Router) angezeigt:

```
Nov 13 12:19:34.216: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, len 1482, rcvd 4
   Nov 13 12:19:34.428: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4

   !--- congestion occurs here. Nov 13 12:19:34.640: IP: s=7.0.0.200 (FastEthernet0/1),
```

```
d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:34.856: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6,
Len 1482, rcvd 4 Nov 13 12:19:35.068: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd
4 Nov 13 12:19:35.280: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13
12:19:35.496: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13
12:19:35.708: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:35.920:
IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:36.136: IP:
s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:36.348: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:36.560: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:36.776: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:36.988: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:37.200: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:37.416: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:37.628: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:37.840: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:38.056: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:38.268: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:38.480: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:38.696: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:38.908: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:39.136: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:39.348: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:39.560: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:39.776: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:39.988: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:40.200: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 13 12:19:40.416: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
```

Da die Paketgrößen für beide Datenflüsse identisch sind, sollten wir basierend auf der Zeitformel für die Zeitplanung sehen, dass zwei Pakete aus unserer kleinen Klasse für jedes Paket aus unserer großen Klasse gesendet werden. Genau das sehen wir in den obigen Debuggen.

Test B

Beim zweiten Test werden die Klassen wie folgt aufgefüllt:

- Kleine Klasse - wir haben den Bandbreitenparameter auf 32 Kbit/s konfiguriert. Es werden zehn Pakete von 500 Byte von 7.0.0.200 generiert, gefolgt von zehn Paketen von 1.500 Byte ab 7.0.0.201.
- Große Klasse - wir haben den Bandbreitenparameter auf 16 Kbit/s konfiguriert. Die Klasse speichert einen Fluss von 1.500 Byte Paketen aus 7.0.0.1.

Der Datenverkehrsgenerator sendet eine Datenverkehrslast von 100 Mbit/s in der folgenden Reihenfolge an Router2:

1. 10 1500-Byte-Pakete von 7.0.0.1.
2. Zehn 500-Byte-Pakete von 7.0.0.200.
3. Zehn 1500-Byte-Pakete von 7.0.0.201.

FIFO wird in jeder Klasse konfiguriert.

Überprüfen des Flussgewichts

Im nächsten Schritt wird das Gewicht der klassifizierten Datenflüsse überprüft:

```
alcazaba#show queue ATM 4/0.102
   Interface ATM4/0.102 VC 0/102
```

```
   Queueing strategy: weighted fair
   Total output drops per VC: 0
   Output queue: 23/512/64/0 (size/max total/threshold/drops)
      Conversations  2/3/16 (active/max active/max total)
      Reserved Conversations 2/2 (allocated/max allocated)

  (depth/weight/total drops/no-buffer drops/interleaves) 15/128/0/0
     Conversation 25, linktype: ip, length: 494
     source: 7.0.0.200, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255

  (depth/weight/total drops/no-buffer drops/interleaves) 8/256/0/0
     Conversation 26, linktype: ip, length: 1494
     source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255

   alcazaba#show queue ATM 4/0.102
   Interface ATM4/0.102 VC 0/102
   Queueing strategy: weighted fair
   Total output drops per VC: 0
   Output queue: 13/512/64/0 (size/max total/threshold/drops)
        Conversations  2/3/16 (active/max active/max total)
        Reserved Conversations 2/2 (allocated/max allocated)

  (depth/weight/total drops/no-buffer drops/interleaves) 8/128/0/0
     Conversation 25, linktype: ip, length: 1494
     source: 7.0.0.201, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255

  (depth/weight/total drops/no-buffer drops/interleaves) 5/256/0/0
     Conversation 26, linktype: ip, length: 1494
     source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63,
```

Wie Sie in der obigen Ausgabe sehen können, haben die Ströme von 7.0.0.200 und 7.0.0.201 das gleiche Gewicht erhalten (128). Dieses Gewicht ist die Hälfte der Größe des Gewichts, das dem Fluss von 7.0.0.1 zugewiesen wurde. Dies entspricht der Tatsache, dass kleine Klassen eine Bandbreite haben, die doppelt so groß ist wie unsere große Klasse.

Überprüfen der Bandbreitenverteilung

Wir können die folgenden DebuggingInnen vom Empfängergerät generieren:

```
Nov 14 06:52:01.761: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
   Nov 14 06:52:01.973: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4

   !--- Congestion occurs here. Nov 14 06:52:02.049: IP: s=7.0.0.200 (FastEthernet0/1),
d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.121: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6,
Len 482, rcvd 4 Nov 14 06:52:02.193: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd
4 Nov 14 06:52:02.269: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14
06:52:02.341: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.413:
IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.629: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:02.701: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.773: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.849: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:02.921: IP: s=7.0.0.200
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 Nov 14 06:52:03.149: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:03.361: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:03.572: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:03.788: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:04.000: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:04.212: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:04.428: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:04.640: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:04.852: IP: s=7.0.0.201
```

```
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:05.068: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:05.280: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:05.492: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:05.708: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:05.920: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:06.132: IP: s=7.0.0.201
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:06.348: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4 Nov 14 06:52:06.560: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
```

In diesem Szenario haben die Flüsse in unserer kleinen Klasse nicht die gleiche Paketgröße. Daher ist die Paketverteilung nicht so trivial wie bei Test A oben.

Planungszeiten

Sehen wir uns nun die Planungszeiten für jedes Paket genauer an. Die Planungszeit für Pakete wird nach folgender Formel berechnet:

```
scheduling tail_time= sub_queue_tail_time + pktsize *
     weight
```

Für verschiedene Paketgrößen wird für die Planungszeit folgende Formel verwendet:

```
500 bytes (small class): scheduling tail_time = x + 494 * 128
     = x + 63232
     1500 bytes (small class): scheduling tail_time = x + 1494 *
     128 = x + 191232
     1500 bytes (big class): scheduling tail_time = x + 1494 *
     256 = x + 382464
```

Anhand dieser Formeln können wir sehen, dass sechs Pakete von 500 Byte aus unserer kleinen Klasse für jedes Paket von 1500 Byte aus unserer großen Klasse übertragen werden (dargestellt in der obigen Debugausgabe).

Wir können auch sehen, dass zwei Pakete von 1500 Byte aus unserer kleinen Klasse für ein Paket von 1500 Byte aus unserer großen Klasse gesendet werden (dargestellt in der Debug-Ausgabe oben).

Aus unseren Tests können wir Folgendes schließen:

- Die Größe des Übertragungsringes (TX-Ring-Limit) bestimmt, wie schnell der Warteschlangenmechanismus beginnt zu funktionieren. Die Auswirkung beim Anheben des Ping-RTT wird angezeigt, wenn der Grenzwert für den Übertragungsring ansteigt. Wenn Sie also CBWFQ oder Low Latency Queueing [LLQ] implementieren, sollten Sie die Beschränkung für den Übertragungsring reduzieren.
- CBWFQ ermöglicht die gerechte Aufteilung der Schnittstellenbandbreite zwischen verschiedenen Klassen.

Zugehörige Informationen

- VC Class-Based Weighted Fair Queuing (Per-VC CBWFQ) auf Cisco Routern der Serien

7200, 3600 und 2600

- Class-Based Weighted Fair Queuing auf RSP-basierten Plattformen
- Gewichtete Fair-Warteschlangen auf ATM
- IP to ATM Class of Service Technology Support
- ATM-Technologieunterstützung
- Technischer Support und Dokumentation - Cisco Systems