Compreendendo a classe baseada em Weighted Fair Queuing em ATM

Contents

Introduction

Este documento fornece uma introdução ao enfileiramento de tráfego, usando a tecnologia Class-Based Weighted Fair Queuing (CBWFQ).

O Enfileiramento moderado ponderado (WFQ) habilita enlaces de baixa velocidade, como enlaces seriais, para fornecer um tratamento moderado a todos os tipos de tráfego. Ele classifica o tráfego em fluxos diferentes (também conhecidos como conversações) com base nas informações da camada três e da camada quatro, como endereços IP e portas TCP. Ele faz isto sem que seja necessário definir listas de acesso. Isso significa que o tráfego de baixa largura de banda tem efetivamente prioridade sobre o tráfego de alta largura de banda porque o tráfego de alta largura de banda compartilha a mídia de transmissão proporcionalmente ao peso atribuído. Entretanto, o WFQ tem determinadas limitações:

- Não é escalável se a quantidade de fluxo aumentar consideravelmente.
- O WFQ nativo não está disponível em interfaces de alta velocidade como as interfaces ATM.

O CBWFQ fornece uma solução para essas limitações. Ao contrário do padrão WFQ, o CBWFQ permite definir as classes de tráfego e aplicar parâmetros, como largura de banda e limites de fila a estas classes. A largura de banda atribuída a uma classe é utilizada para calcular o “peso” daquela classe. O peso de cada pacote que atende aos critérios de classe também é calculado a partir disso. O WFQ é aplicado preferencialmente às classes (que podem incluir vários fluxos) do que aos próprios fluxos.

Para obter mais informações sobre como configurar CBWFQ, clique nos links a seguir:

Class-Based Weighted Fair Queuing por VC (CBWFQ por VC) em Roteadores Cisco 7200, 3600 e 2600.

Per-VC Class-Based Weighted Fair Queuing on RSP-based Platforms.

Antes de Começar

Conventions

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

Prerequisites

Não existem requisitos específicos para este documento.

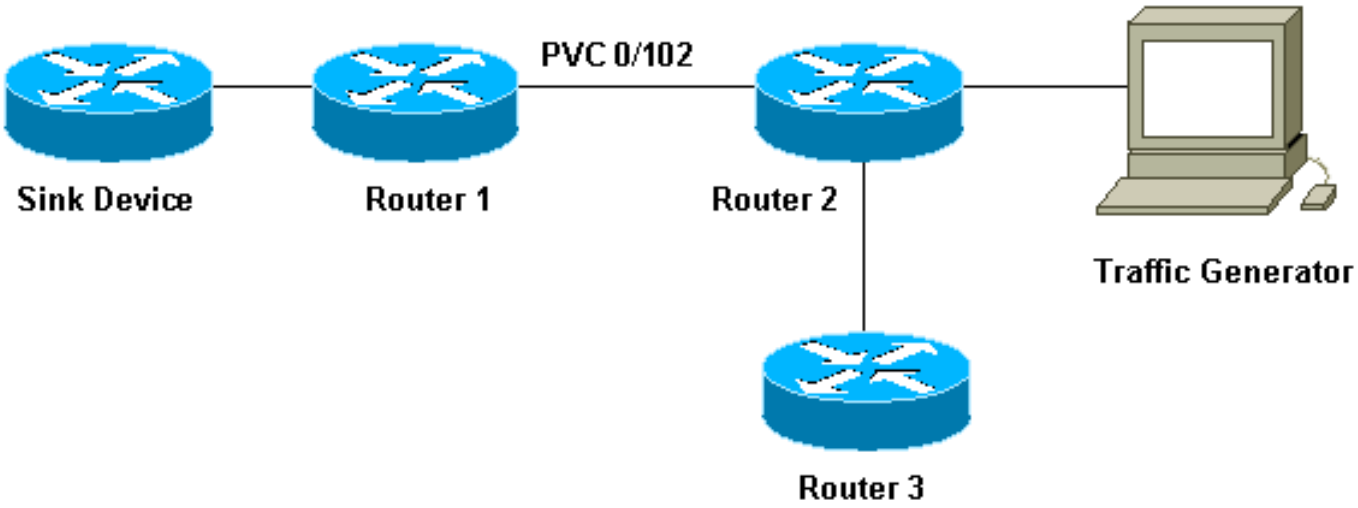
Componentes Utilizados

Este documento não se restringe a versões de software e hardware específicas.

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.

Diagrama de Rede

Para ilustrar como o WFQ funciona, utilizaremos a seguinte configuração:



Na configuração usada aqui, os pacotes podem ser armazenados em uma das duas filas a seguir:

- A fila de primeiro a entrar primeiro a sair (FIFO) de hardware no adaptador de porta e no módulo de rede.
- A fila no software Cisco IOS® (na memória de entrada/saída [E/S] do roteador) onde os recursos de Qualidade de Serviço (QoS), como CBWFQ, podem ser aplicados.

A fila FIFO no adaptador de porta armazena os pacotes antes que eles sejam segmentados em células para transmissão. Quando a fila estiver cheia, o adaptador de porta ou o módulo de rede sinalizará para o software IOS que a fila está congestionada. Esse mecanismo é chamado de contrapressão. Ao receber esse sinal, o roteador pára de enviar pacotes para a fila FIFO da interface e armazena os pacotes no software IOS até que a fila esteja descongestionada novamente. Quando os pacotes são armazenados no IOS, o sistema pode aplicar recursos QoS; por exemplo, CBWFQ.

Definindo o limite de anel de transmissão

Um problema com esse mecanismo de enfileiramento é que, quanto maior a fila de FIFO na interface, maior o atraso antes que os pacotes do final dessa fila possam ser transmitidos. Isso pode causar problemas sérios de desempenho no tráfego sensível a retardos, como tráfego de voz.

O comando tx-ring-limit do Circuito Virtual Permanente (PVC) possibilita reduzir o tamanho da fila de FIFO.

```
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
```

O limite (x) que pode ser especificado aqui é um número de pacotes (para Cisco 2600 e 3600 routers) ou uma quantidade de partículas (para Cisco 7200 e 7500 routers).

A redução do tamanho do anel de transmissão tem duas vantagens:

- Reduz o tempo total que os pacotes têm de esperar na fila FIFO antes de serem

segmentados.
- Acelera o uso do QoS no software IOS.

Impacto da transmissão de limite de anel

Vejamos o impacto do limite do anel de transmissão usando a configuração mostrada no diagrama de rede acima. Podemos assumir o seguinte:

- O gerador de tráfego está enviando tráfego (pacotes de 1.500 bytes) para o dispositivo coletor e esse tráfego está sobrecarregando o PVC 0/102 entre o roteador 1 e o roteador 2.
- O Roteador 3 tenta o ping para o Roteador 1.
- O CBWFQ está ativado no roteador 2.

Agora vamos observar duas configurações usando limites de anéis de transmissão diferentes para ver o impacto que isso tem.

Exemplo A

Nesse exemplo, configuramos o anel de transmissão para três (tx-ring-limit=3). Aqui está o que vemos quando fazemos ping no roteador 1 a partir do roteador 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
```

Exemplo B

Neste exemplo, definimos o anel de transmissão como 40 (TX-ring-limit=40). Isso é o que vemos quando utilizamos o mesmo ping usado no Exemplo 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
```

Como você pode observar aqui, quanto maior o limite do anel de transmissão, maior será o tempo de RTT (ping round trip). Podemos deduzir a partir disto de que um limite grande do anel de transmissão pode levar a atrasos significativos na transmissão.

Como funciona o CBWFQ

Agora que vimos o impacto do tamanho da fila FIFO de hardware, vamos ver exatamente como o CBWFQ funciona.

O WFQ nativo atribui um peso a cada conversação e, em seguida, agenda o tempo de transmissão para cada pacote dos diferentes fluxos. O peso é uma função da precedência IP de cada fluxo, e o tempo de programação depende do tamanho do pacote. Clique aqui para obter mais detalhes sobre o WFQ.

O CBWFQ atribui um peso a cada classe configurada em vez de cada fluxo. Esse peso é proporcional à largura de banda configurada para cada classe. O peso é, mais precisamente, uma função da largura de banda da interface dividida pela largura de banda da classe. Portanto, quanto maior o parâmetro de largura de banda, menor o peso.

Podemos calcular o tempo de programação do pacote usando a seguinte fórmula:

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

Divisão da largura de banda total da interface

Vamos ver como o roteador divide a largura de banda total da interface entre as diferentes classes. Para atender às classes, o roteador usa filas de calendário. Cada uma dessas filas de calendário armazena pacotes que devem ser transmitidos no mesmo valor de scheduling_tail_time. O roteador então serve essas filas de calendário, uma de cada vez. Vamos examinar esse processo:

1. Se ocorrer congestionamento no adaptador de porta quando um pacote chega na interface de saída, isso causa enfileiramento no IOS (CBWFQ, neste caso).
2. O roteador calcula um tempo de programação para esse pacote que está chegando e o armazena na fila de calendário correspondente a esse tempo de programação. Apenas um pacote por classe pode ser armazenado em uma fila de calendário específica.
3. Quando é o momento de prestar serviço à fila do calendário na qual o pacote foi armazenado, o IOS esvazia essa fila e envia os pacotes à fila FIFO no próprio adaptador de porta. O tamanho dessa fila FIFO é determinado pelo limite do anel de transmissão descrito acima.
4. Se a fila FIFO for muito pequena para abranger todos os pacotes mantidos na fila baseada em calendário que está sendo atendida, o roteador reagendará os pacotes que não puderem ser armazenados para o próximo agendamento de tempo (conforme sua importância) e os colocará de acordo com a fila baseada em calendário.
5. Quando tudo isso é feito, o adaptador de porta trata os pacotes em sua fila FIFO e envia as células no fio e o IOS se move para a próxima fila do calendário.Graça a esse mecanismo, cada classe recebe estatisticamente uma parte da largura de banda da interface

correspondente aos parâmetros configurados para ela.

Mecanismo de fila de calendário versus tamanho do anel de transmissão

Vamos ver o relacionamento entre o mecanismo de fila do calendário e o tamanho do anel de transmissão. Um pequeno anel de transmissão permite que o QoS inicie mais rapidamente e reduz a latência dos pacotes que estão aguardando transmissão (o que é importante para tráfego sensível a retardos, como pacotes de voz). Entretanto, se for muito pequeno pode levar a um ritmo de transferência mais lento para certas classes. Isso ocorre porque muitos pacotes podem ter que ser reagendados se o anel de transmissão não puder acomodá-los.

Infelizmente, não há um valor ideal para o tamanho do anel de transmissão e a única maneira de encontrar o melhor valor é experimentando.

Compartilhamento de largura de banda

Podemos analisar o conceito de compartilhamento de largura de banda utilizando a configuração mostrada em nosso diagrama de rede, acima. O gerador de pacotes produz diferentes fluxos e os envia para o dispositivo coletor. A quantidade total de tráfego representada por esses fluxos é suficiente para sobrecarregar o PVC. Implementamos o CBWFQ no Roteador 2. Veja abaixo um exemplo da nossa configuração:

```
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
```

Em nosso exemplo, o Roteador 2 é um roteador Cisco 7200. Isto é importante porque o limite do anel de transmissão é expresso em partículas, não em pacotes. Os pacotes são enfileirados na fila de FIFO de adaptador de porta assim que uma partícula livre estiver disponível, mesmo que mais de uma seja necessária para armazenar o pacote.

O que é uma partícula?

Em vez de alocar uma parte da memória contígua para um buffer, o buffer de partícula aloca partes não contíguas (dispersas) de memória, chamadas partículas, e, em seguida, as conecta para formar um buffer de pacotes lógicos. Isso é chamado de um buffer de partícula. Em tal esquema, um pacote pode então se espalhar em várias partículas.

No roteador 7200 que estamos usando aqui, o tamanho da partícula é de 512 bytes.

Podemos verificar se os Cisco 7200 routers fazem uso de partículas utilizando o commando show buffers:

```
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
```

Teste A

As classes "small" e "big" que estamos utilizando para esse teste são preenchidas da seguinte maneira:

- Classe pequena - configuramos os parâmetros de largura de banda como 32 kbps. Esta classe armazena dez pacotes de 1500 bytes de 7.0.0.200, seguidos por dez pacotes de 1500 bytes de 7.0.0.201
- Classe grande – configuramos o parâmetro de largura de banda para 16 kbps. Essa classe armazena um fluxo de dez pacotes de 1500 bytes de 7.0.0.1.

O gerador de tráfego envia uma intermitência de tráfego destinado ao dispositivo de pia a 100 Mbps para o Roteador 2 na seguinte ordem:

1. Dez pacotes de 7.0.0.1.
2. Dez pacotes de 7.0.0.200.
3. Dez pacotes de 7.0.0.201.

Verificando o peso do fluxo

Vamos ver o peso aplicado aos fluxos diferentes. Para fazer isto, podemos utilizar o comando show queue ATM x/y.z.

```
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
```

Quando todos os pacotes de 7.0.0.200 foram enfileirados para fora do roteador, podemos ver o seguinte:

```
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
```

Como você pode ver aqui, os fluxos de 7.0.0.200 e 7.0.0.201 têm o mesmo peso (128). Esse peso é metade do peso atribuído ao fluxo de 7.0.0.1 (256). Isto corresponde ao fato de que nossa largura de banda de classe pequena tem duas vezes o tamanho da nossa classe grande.

Verificando a distribuição de largura de banda

Como a distribuição da largura de banda é verificada entre os fluxos diferentes? O método de enfileiramento FIFO é usado em cada classe. Nossa classe pequena está cheia com dez pacotes do primeiro fluxo e dez pacotes do segundo fluxo. O primeiro fluxo é removido da classe pequena, em 32 kbps. Assim que forem enviados, os dez pacotes do outro fluxo são também enviados. Enquanto isso, os pacotes de nossa classe grande são removidos a 16 kbps.

Podemos ver que, como o gerador de tráfego está enviando uma rajada a 100 Mbps, o PVC será sobrecarregado. Entretanto, como não há tráfego no PVC quando o teste é iniciado e já que os pacotes de 7.0.0.1 são os primeiros a alcançar o roteador, alguns pacotes de 7.0.0.1 serão enviados antes de o CBWFQ ser iniciado devido a congestionamento (em outras palavras, antes que o anel de transmissão esteja cheio).

Como o tamanho da partícula é de 512 bytes e o tamanho do anel de transmissão é de três partículas, podemos notar que dois pacotes do 7.0.0.1 são enviados antes do congestionamento. Um é imediatamente enviado no fio e o segundo é armazenado em três partículas formando a fila FIFO do adaptador de porta.

Podemos ver as depurações a seguir no dispositivo de depósito (que é simplesmente um roteador):

```
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
```

Como o tamanho dos pacotes para os dois fluxos é o mesmo, com base na fórmula de tempo de programação, deveríamos ver dois pacotes de nossa classe pequena enviados para cada pacote de nossa classe grande. É exatamente isso que podemos conferir nas depurações acima.

Teste B

Para o nosso segundo teste, vamos preencher as classes da seguinte maneira:

- Classe pequena configuramos o parâmetro de largura de banda para 32 kbps. Dez pacotes de 500 bytes de 7.0.0.200 são gerados, seguidos por dez pacotes de 1500 bytes de 7.0.0.201.
- Classe grande – configuramos o parâmetro de largura de banda para 16 kbps. A classe armazena um fluxo de pacotes de 1500 bytes vindos de 7.0.0.1.

O gerador de tráfego envia um burst de tráfego a 100 Mbps para o Roteador 2 na seguinte ordem:

1. Dez pacotes de 1500 bytes de 7.0.0.1.
2. Dez pacotes de 500 bytes a partir de 7.0.0.200.
3. Dez pacotes de 1.500 bytes de 7.0.0.201.

FIFO é configurado em cada classe.

Verificando o peso do fluxo

O passo seguinte é verificar o peso aplicado para os fluxos classificados:

```
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) 15/128/0/0/0
     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) 8/256/0/0/0
     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) 8/128/0/0/0
     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) 5/256/0/0/0
     Conversation 26, linktype: ip, length: 1494
     source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63,
```

Como você pode ver na saída acima, os fluxos de 7.0.0.200 e 7.0.0.201 receberam a mesma relevância (128). Esse peso é metade do tamanho do peso atribuído ao fluxo a partir de 7.0.0.1. Isso demonstra o fato de que a classe pequena tem uma largura de banda que é o dobro do tamanho da classe grande.

Verificando a distribuição de largura de banda

Podemos gerar as seguintes depurações a partir do dispositivo de depósito:

```
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
```

Nesse cenário, os fluxos em nossa pequena classe não têm o mesmo tamanho de pacote. Por isso, a distribuição de pacotes não é tão trivial quando para o Teste A, acima.

Tempos de programação

Analisemos mais atentamente as programações de tempo para cada pacote. O tempo de agendamento dos pacotes é calculado usando a seguinte fórmula:

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

Para tamanhos diferentes de pacotes, o tempo de programação utiliza a seguinte fórmula:

```
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
```

Com base nessas fórmulas, podemos ver que seis pacotes de 500 bytes de nossa classe pequena são transmitidos para cada pacote de 1.500 bytes de nossa classe grande (mostrada na saída de depuração acima).

Também podemos ver que os dois pacotes de 1.500 bytes de nossa classe pequena são enviados para um pacote de 1.500 bytes de nossa classe grande (mostrada na saída da depuração acima).

Considerando nossos testes acima, podemos concluir o seguinte:

- O tamanho do anel de transmissão (TX-ring-limit) determina com que rapidez o mecanismo de enfileiramento começa a funcionar. Podemos ver o impacto com o aumento do RTT de ping quando o limite do anexo de transmissão aumenta. Assim, se você implementar o CBWFQ ou LLQ [Enfileiramento de latência baixa]), considere a reduzir o limite do anel de transmissão.
- O CBWFQ permite o compartilhamento justo da largura de banda da interface entre diferentes classes.

Informações Relacionadas

- Class-Based Weighted Fair Queuing por VC (CBWFQ por VC) em Roteadores Cisco 7200, 3600 e 2600
- Per-VC Class-Based Weighted Fair Queuing on RSP-based Platforms
- Compreendendo a Weighted Fair Queuing em ATM
- Suporte à tecnologia de classe de serviço IP para ATM

- Suporte à tecnologia ATM
- Suporte Técnico e Documentação - Cisco Systems