

ATM에서 가중치 공정 대기열 이해

목차

[소개](#)

[사전 요구 사항](#)

[요구 사항](#)

[사용되는 구성 요소](#)

[표기 규칙](#)

[네트워크 다이어그램](#)

[전송 링 제한을 설정하는 방법](#)

[전송 링 제한의 영향](#)

[예 A](#)

[예 B](#)

[가중치 계산 방법](#)

[예약 시간 계산 방법](#)

[WFQ 작동 방식](#)

[입자란?](#)

[테스트 A](#)

[테스트 B](#)

[1단계](#)

[2단계](#)

[3단계](#)

[4단계](#)

[요약](#)

[관련 정보](#)

소개

이 문서에서는 WFQ(Weighted Fair Queuing) 기술을 사용하는 트래픽 대기열을 소개합니다.

모든 유형의 트래픽에 대해 공정한 처리를 제공하기 위해 직렬 링크와 같은 저속 링크를 활성화하기 위해 WFQ가 도입되었습니다. 이를 위해 WFQ는 IP 주소 및 TCP 포트와 같은 레이어 3 및 레이어 4 정보를 기반으로 트래픽을 서로 다른 흐름(대화라고도 함)으로 분류합니다. 액세스 목록을 정의할 필요 없이 이 작업을 수행합니다. 즉, 고대역폭 트래픽이 할당된 가중치에 비례하여 전송 미디어를 공유하기 때문에 저대역폭 트래픽이 고대역폭 트래픽보다 효과적으로 우선순위를 갖게 됩니다.

그러나 WFQ에는 몇 가지 제한 사항이 있습니다.

- 플로우 양이 상당히 증가하면 확장성이 없습니다.
- 네이티브 WFQ는 ATM 인터페이스와 같은 고속 인터페이스에서 사용할 수 없습니다.

CBWFQ(Class-based weighted fair queuing)는 이러한 제한 사항을 해결할 수 있는 솔루션을 제공

합니다.

표준 WFQ와 달리 CBWFQ에서는 트래픽 클래스를 정의할 수 있습니다. 대역폭 및 대기열 제한 같은 매개변수를 해당 매개변수에 적용할 수도 있습니다. 클래스에 할당하는 대역폭은 해당 클래스의 가중치를 계산하는 데 사용됩니다. 클래스 기준과 일치하는 각 패킷의 가중치도 여기에서 계산됩니다. 그런 다음 WFQ가 클래스에 적용되며, 여기에는 흐름 자체가 아닌 여러 플로우가 포함될 수 있습니다.

CBWFQ 구성 방법에 대한 자세한 내용은 다음 문서를 참조하십시오.

- [Cisco 7200, 3600 및 2600 라우터의 VC 클래스 기반, CBWFQ\(Per-VC CBWFQ\)](#)
- [RSP 기반 플랫폼의 VC 클래스 기반 가중치 공정 대기열](#)

ATM 인터페이스는 **fair-queue** 명령을 사용하여 인터페이스에 직접 구성된 네이티브 흐름 기반 WFQ를 지원하지 않습니다. 그러나 CBWFQ를 지원하는 소프트웨어를 사용하면 다음 예와 같이 기본 클래스 내에서 플로우 기반 WFQ를 구성할 수 있습니다.

```
policy-map test
  class class-default
    fair-queue
!
interface ATMx/y.z point-to-point
  ip address a.b.c.d M.M.M.M
  pvc A/B
  service-policy output test
```

[사전 요구 사항](#)

[요구 사항](#)

이 문서에 대한 특정 요건이 없습니다.

[사용되는 구성 요소](#)

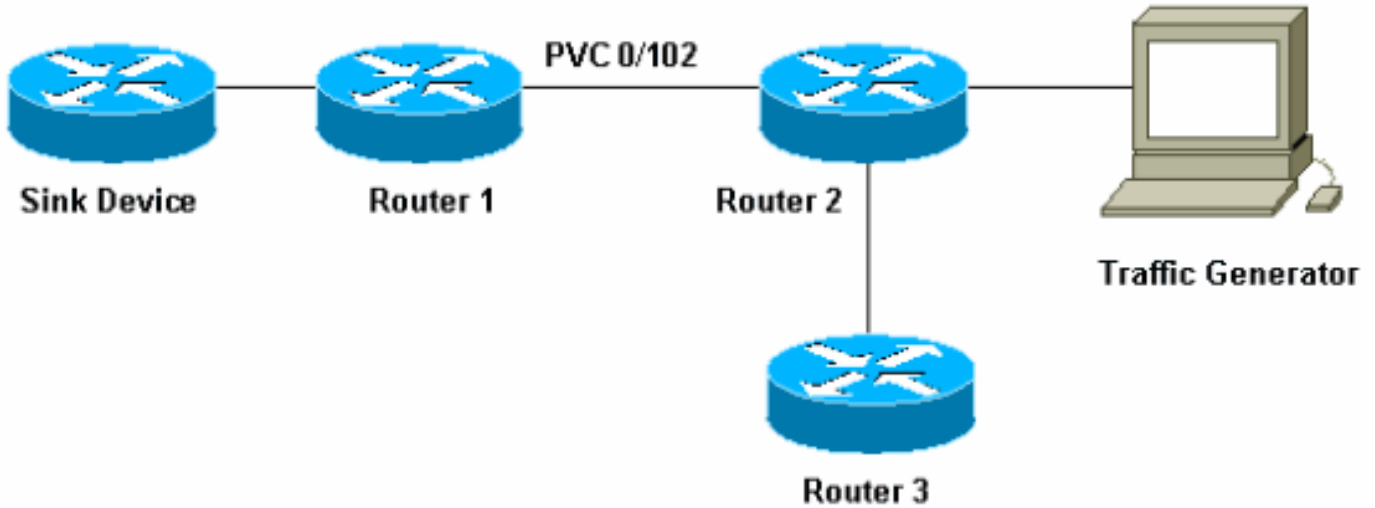
이 문서는 특정 소프트웨어 및 하드웨어 버전으로 한정되지 않습니다.

[표기 규칙](#)

문서 규칙에 대한 자세한 내용은 [Cisco 기술 팁 표기 규칙을 참고하십시오](#).

[네트워크 다이어그램](#)

이 설정을 사용하여 WFQ의 작동 방식을 설명합니다.



이 설정에서 패킷은 다음 두 큐 중 하나에 저장할 수 있습니다.

- 포트 어댑터 및 네트워크 모듈의 FIFO(First In First Out) 대기열 하드웨어
- Cisco IOS[®] 소프트웨어의 대기열은 라우터 입력/출력[I/O] 메모리에서 CBWFQ와 같은 QoS(Quality of Service) 기능을 적용할 수 있습니다.

포트 어댑터의 FIFO 대기열은 전송을 위해 패킷이 셀로 분할되기 전에 패킷을 저장합니다. 이 대기열이 가득 차면 포트 어댑터 또는 네트워크 모듈이 IOS 소프트웨어에 대기열이 혼잡함을 알립니다. 이 메커니즘을 백압력이라고 합니다. 이 신호를 수신하면 라우터가 패킷을 인터페이스 FIFO 대기열에 전송하기 위해 중지하고 대기열이 다시 정체 상태가 될 때까지 패킷을 IOS 소프트웨어에 저장합니다. 패킷이 IOS에 저장되면 시스템에서 QoS를 적용할 수 있습니다.

전송 링 제한을 설정하는 방법

이 대기열 메커니즘의 한 가지 문제는 인터페이스에서 FIFO 대기열이 클수록 이 대기열 끝에 있는 패킷이 전송되기 전에 지연이 길어진다는 것입니다. 이로 인해 음성 트래픽과 같이 지연에 민감한 트래픽에 심각한 성능 문제가 발생할 수 있습니다.

PVC(Permanent Virtual Circuit) **tx-ring-limit** 명령을 사용하면 FIFO 대기열의 크기를 줄일 수 있습니다.

```
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
```

여기서 지정할 수 있는 <size>는 Cisco 2600 및 3600 라우터의 경우 패킷 수, Cisco 7200 및 7500 라우터의 경우 입자 수량입니다.

전송 링 크기를 줄이면 두 가지 이점이 있습니다.

- FIFO 대기열에서 패킷을 분리하기 전에 대기하는 시간을 줄입니다.
- IOS 소프트웨어에서 QoS 사용을 가속화합니다.

전송 링 제한의 영향

이전 네트워크 다이어그램에 표시된 설정을 사용하는 전송 링 제한의 영향을 확인합니다. 다음을 가정합니다.

- 트래픽 생성기는 트래픽(1500바이트 패킷)을 싱크 디바이스로 전송하며, 이 트래픽은 router1과 router2 간에 PVC 0/102를 오버로드합니다.
- Router3에서 router1에 ping을 시도합니다.
- WFQ는 router2에서 활성화됩니다.

이러한 영향을 파악하기 위해 서로 다른 전송 링 제한을 사용하는 두 가지 컨피그레이션을 살펴봅니다.

예 A

이 예에서는 전송 링을 3으로 설정합니다(tx-ring-limit=3). router3에서 router1을 ping할 때 표시되는 내용은 다음과 같습니다.

```
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
```

```
router2#show queue atm 4/0.102
Interface ATM4/0.102 VC 0/102
Queuing strategy: weighted fair
Total output drops per VC: 1505772
Output queue: 65/512/64/1505772 (size/max total/threshold/drops)
Conversations 2/3/16 (active/max active/max total)
Reserved Conversations 0/0 (allocated/max allocated)

(depth/weight/discard/tail drops/interleaves) 1/32384/0/0/0
Conversation 2, linktype: ip, length: 58
source: 8.0.0.1, destination: 6.6.6.6, id: 0x2DA1, ttl: 254, prot: 1
!--- ping (depth/weight/discard/tail drops/interleaves) 64/32384/1505776/0/0
Conversation 15, linktype: ip, length: 1494
source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
!--- This is traffic from the traffic generator.
```

예 B

이 예에서는 전송 링을 40(tx-ring-limit=40)으로 설정합니다. 예제 A와 동일한 ping을 사용할 때 표시되는 내용은 다음과 같습니다.

```
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 ms

여기서 볼 수 있듯이 전송 링 한도가 클수록 ping 왕복 시간(RTT)이 큼니다. 따라서 전송 링 한도가 많으면 전송 지연이 클 수 있습니다.

가중치 계산 방법

예제 A의 **show queue atm** 출력에서 각 대화에 가중치가 할당되어 있음을 확인할 수 있습니다. 자세한 내용은 다음을 참조하십시오.

```
router2#show queue ATM 4/0.102
Interface ATM4/0.102 VC 0/102
Queuing strategy: weighted fair
Total output drops per VC: 1505772
Output queue: 65/512/64/1505772 (size/max total/threshold/drops)
Conversations 2/3/16 (active/max active/max total)
Reserved Conversations 0/0 (allocated/max allocated)

(depth/weight/discards/tail drops/interleaves) 1/32384/0/0/0
Conversation 2, linktype: ip, length: 58
source: 8.0.0.1, destination: 6.6.6.6, id: 0x2DA1, ttl: 254, prot: 1

(depth/weight/discards/tail drops/interleaves) 64/32384/1505776/0/0
Conversation 15, linktype: ip, length: 1494
source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
```

WFQ를 사용할 때 다음 공식을 사용하여 각 대화의 가중치를 계산할 수 있습니다.

- $weight=32384/(precedence+1)$ - Cisco IOS Software 릴리스 12.0(5)T 이상용.
- $weight=4096/(precedence+1)$ - 12.0(5)T 이전 Cisco IOS 소프트웨어 릴리스의 경우

예약 시간 계산 방법

이제 패킷이 IOS 대기열에서 포트 어댑터 또는 네트워크 모듈 FIFO 대기열로 전달되는 경우 각 패킷의 예약 시간을 계산하기 위해 이러한 가중치를 사용할 수 있습니다.

이 공식을 사용하여 출력 스케줄링 시간을 계산할 수 있습니다. 여기서 **queue_tail_time**은 현재 스케줄링 시간입니다.

출력 예약 시간 = queue_tail_time + pktsize*weight

WFQ 작동 방식

이 섹션에서는 WFQ의 작동 방식을 설명합니다. WFQ의 원칙은 작은 가중치 또는 작은 패킷의 패킷이 전송될 때 우선순위가 부여되어야 한다는 것입니다.

이를 확인하기 위해 트래픽 생성기를 사용하는 10개의 패킷과 4개의 더 작은 패킷(82바이트)으로 구성된 흐름을 생성합니다.

이 예에서 router2는 PA-A3(ATM Port Adapter)이 있는 Cisco 7200 라우터입니다. 포트 어댑터의 출력 FIFO 대기열의 크기는 패킷이 아니라 입자로 표현되기 때문에 이 점이 중요합니다. 입자란 [을 보시겠습니까?](#) 섹션을 참조하십시오.

입자란?

입자 버퍼는 버퍼에 대해 연속적인 메모리 한 조각을 할당하는 대신 입자라고 하는 불연속적인(분산형) 메모리 조각을 할당한 다음 하나의 논리적 패킷 버퍼를 형성하기 위해 이를 연결합니다. 이것을 입자 버퍼라고 합니다. 이러한 체계에서 패킷은 여러 입자에 걸쳐 분산될 수 있습니다.

7200 라우터에서 입자 크기는 512바이트입니다.

Cisco 7200 라우터가 입자를 사용하는지 확인하려면 show buffers 명령을 사용합니다.

```
router#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
ATM2/0 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
```

테스트 A

다음은 WFQ 기능을 설명하기 위한 몇 가지 테스트입니다. 첫 번째 테스트에서는 서로 다른 대화 간에 대역폭을 공유할 수 있는지 확인합니다.

이 테스트에서는 router1과 router2 간에 PVC 0/102를 오버로드할 수 있을 만큼 트래픽 생성기를 신속하게 전송하도록 했습니다. 동일한 PVC에서 router3에서 router1으로 ping을 수행합니다.

```
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:
..... (WFQ is enabled here)!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.
Success rate is 98 percent (604/613), round-trip min/avg/max = 164/190/232 ms
```

표시된 출력에서 볼 수 있듯이, 인터페이스에서 WFQ가 활성화될 때까지 트래픽은 다른 트래픽이 통과할 수 없게 하고 회선을 시작합니다. WFQ가 활성화되면 ping이 성공적으로 수행됩니다.

WFQ를 사용하면 다른 대화를 차단하지 않고도 다른 대화 간에 대역폭을 공유할 수 있다는 것을 알

수 있습니다.

테스트 B

이것이 대역폭 공유 방법입니다.

트래픽 생성기가 전송하는 폴로우는 10개의 큰 패킷으로 구성된 버스트이며, 그 뒤에 82바이트의 작은 패킷 4개가 옵니다. 이를 라우터2에 100Mbps로 전송합니다. 버스트를 전송하면 router2 ATM 인터페이스의 출력 대기열이 비어 있습니다. Router2는 출력 대기열에서 혼잡이 발생하도록 10KB PVC(매우 느린 PVC)를 통해 이러한 패킷을 전송합니다.

이 프로세스를 간소화하기 위해 여러 단계에서 이 테스트를 수행합니다.

1단계

큰 트래픽은 10개의 482바이트 패킷으로 구성됩니다. PA-A3의 입자는 512바이트이므로, 크고 작은 패킷은 포트 어댑터 출력 대기열에 저장될 때 하나의 입자를 가져와야 합니다. 라우터의 전송 링 제한 값은 3입니다(tx-ring-limit=3). 다음은 싱크 디바이스에서 볼 수 있는 예시입니다.

```
.Nov 7 15:39:13.776: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, len 482, rcvd 4
.Nov 7 15:39:13.776: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:39:14.252: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:39:14.252: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:39:14.732: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:39:14.732: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:39:15.208: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:39:15.208: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
```

```
!--- Congestion occurs at this point. .Nov 7 15:39:15.512: IP: s=7.0.0.200 (FastEthernet0/1),
d=6.6.6.6, Len 82, rcvd 4 .Nov 7 15:39:15.516: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len
82, unknown protocol .Nov 7 15:39:15.644: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82,
rcvd 4 .Nov 7 15:39:15.644: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown
protocol .Nov 7 15:39:15.776: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4 .Nov
7 15:39:15.776: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol .Nov 7
15:39:15.904: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4 .Nov 7 15:39:15.904:
IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol .Nov 7 15:39:16.384: IP:
s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:16.384: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol .Nov 7 15:39:16.860: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:16.860: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol .Nov 7 15:39:17.340: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:17.340: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol .Nov 7 15:39:17.816: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:17.816: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol .Nov 7 15:39:18.296: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:18.296: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol .Nov 7 15:39:18.776: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4 .Nov 7 15:39:18.776: IP: s=7.0.0.1
(FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
```

82바이트 패킷보다 먼저 전송된 482바이트의 패킷 4개를 볼 수 있습니다. 이 패킷은 일반적으로 우선 순위를 갖습니다. 이것이 바로 이런 일이 일어나는 이유입니다.

버스트는 기본적으로 10개의 482바이트 패킷으로 구성되므로, 이러한 패킷은 먼저 라우터에 도달하고 그 뒤에 82바이트 패킷이 옵니다. 트래픽이 없으므로 482바이트 패킷이 혼잡이 없는 시점에 도달하기 때문에 한 패킷은 즉시 포트 어댑터 세그멘테이션 및 리어셈블리(SAR)로 대기하여 셀에 청크되어 전선으로 전송됩니다. 즉, 전송 링이 여전히 비어 있습니다.

하나의 482바이트 패킷을 전송하는 데 필요한 시간이 총 버스트를 전송하기 위해 트래픽 생성기에 필요한 시간보다 길다는 것을 계산할 수 있습니다. 따라서 첫 번째 482바이트 패킷이 포트 어댑터로 대기될 때 버스트의 482바이트 패킷이 라우터에 이미 있다고 가정할 수 있습니다. 따라서 더 많은 482바이트 패킷을 전송 링으로 대기열에 추가할 수 있습니다. 482바이트의 3개 이상의 패킷이 대기열에 추가되어 3개의 여유 입자를 사용합니다.

참고: 패킷을 저장하려면 둘 이상의 파티클이 필요한 경우에도 프리파티클이 있는 즉시 전송 링으로 패킷이 대기됩니다.

이 시점에서, 세 개의 입자가 가득 차있기 때문에 혼잡함이 있습니다. 따라서 큐잉은 IOS에서 시작됩니다. 4개의 82바이트 패킷이 마지막으로 라우터에 도달하면 혼잡이 발생합니다. 이 4개의 패킷은 대기열에 있으며 WFQ는 두 플로우에 사용됩니다. **show queue ATM** 명령을 사용하는 ATM 대기열에서 다음을 확인합니다.

```
router2#show queue ATM 4/0.102 vc 0/102
  Interface ATM4/0.102 VC 0/102
  Queuing strategy: weighted fair
  Total output drops per VC: 0
  Output queue: 10/512/64/0 (size/max total/threshold/drops)
    Conversations 2/4/16 (active/max active/max total)
    Reserved Conversations 0/0 (allocated/max allocated)

(depth/weight/total drops/no-buffer drops/interleaves) 4/32384/0/0/0
  Conversation 6, linktype: ip, length: 82
  source: 7.0.0.200, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255

(depth/weight/total drops/no-buffer drops/interleaves) 6/32384/0/0/0
  Conversation 15, linktype: ip, length: 482
  source: 7.0.0.1, destination: 6.6.6.6, id: 0x0000, ttl: 63, prot: 255
```

디버그에서 482바이트의 처음 4개의 패킷 뒤에 82바이트 패킷이 오는 것을 확인할 수 있습니다. 이 작은 패킷은 큰 패킷보다 먼저 라우터에서 이동합니다. 이는 혼잡이 발생하면 작은 패킷이 큰 패킷보다 우선 순위를 갖게 됨을 나타냅니다.

이를 확인하려면 가중치 계산 섹션에 제공된 [가중치](#) 및 스케줄링 시간 공식을 사용합니다.

2단계

전송 링 제한을 5로 늘리고 대용량 패킷은 482바이트라면 이전 출력에 따라 혼잡이 발생하기 전에 6 패킷 482바이트, 82바이트의 4 패킷, 482바이트의 4 패킷, 482바이트의 또 다른 4 패킷을 볼 수 있습니다.

```
.Nov 7 15:49:57.365: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:57.365: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:57.841: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:57.845: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:58.321: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:58.321: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:58.797: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:58.801: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:59.277: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:59.277: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:59.757: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:49:59.757: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:49:59.973: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:49:59.973: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
```



```

.Nov 7 15:50:00.105: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:50:00.105: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:50:00.232: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:50:00.232: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:50:00.364: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:50:00.364: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:50:00.840: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:50:00.844: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:50:01.320: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:50:01.320: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:50:01.796: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:50:01.800: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol
.Nov 7 15:50:02.276: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, rcvd 4
.Nov 7 15:50:02.276: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 482, unknown protocol

```

여러분이 여기서 볼 수 있듯이, 이것은 실제로 일어나는 것입니다.

3단계

입자 크기는 512바이트입니다. 따라서 전송 링이 입자로 표현되고 입자 크기보다 약간 큰 패킷을 사용하는 경우 각 하나는 두 개의 입자를 사용합니다. 이는 582바이트의 패킷과 3의 전송 링을 사용하여 설명합니다. 이러한 매개변수를 사용하면 582바이트의 3개의 패킷이 표시됩니다. 하나는 ATM 인터페이스에 트래픽이 없는 상태로 전송되며, 이것은 3개의 입자를 자유로워 줍니다. 따라서 두 개의 패킷을 더 큐에 추가할 수 있으며, 그 뒤에 82바이트의 패킷 4개와 582바이트의 패킷 7개가 옵니다.

```

.Nov 7 15:51:34.604: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:34.604: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:35.168: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:35.168: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:35.732: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:35.736: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:35.864: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:51:35.864: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:51:35.996: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:51:35.996: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:51:36.124: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:51:36.124: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:51:36.256: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 7 15:51:36.256: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 7 15:51:36.820: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:36.820: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:37.384: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:37.388: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:37.952: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:37.952: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:38.604: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:38.604: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:39.168: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:39.168: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:39.732: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:39.736: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol
.Nov 7 15:51:40.300: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, rcvd 4
.Nov 7 15:51:40.300: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 582, unknown protocol

```

4단계

패킷 크기를 1482(3개의 입자)로 설정하고 전송 링을 5로 정의합니다. 전송 링이 입자로 정의된 경우 다음과 비슷한 것이 표시됩니다.

- 한 패킷이 즉시 전송됨
- 5개의 입자 중 3개를 차지하는 패킷 1개
- 두 개의 입자가 사용 가능하므로 한 개의 패킷이 대기됨

```
.Nov 8 07:22:41.200: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:41.200: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:42.592: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:42.592: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:43.984: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:43.984: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:44.112: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 8 07:22:44.112: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 8 07:22:44.332: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 8 07:22:44.332: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 8 07:22:44.460: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 8 07:22:44.460: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 8 07:22:44.591: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, rcvd 4
.Nov 8 07:22:44.591: IP: s=7.0.0.200 (FastEthernet0/1), d=6.6.6.6, Len 82, unknown protocol
.Nov 8 07:22:45.983: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:45.983: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:47.371: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:47.375: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:48.763: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:48.767: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:50.155: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:50.155: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:51.547: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:51.547: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:53.027: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:53.027: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
.Nov 8 07:22:54.415: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, rcvd 4
.Nov 8 07:22:54.419: IP: s=7.0.0.1 (FastEthernet0/1), d=6.6.6.6, Len 1482, unknown protocol
```

요약

수행한 테스트에서 다음과 같은 결과를 얻을 수 있습니다.

- WFQ가 없는 느린 PVC에서 대량 트래픽은 WFQ가 활성화될 때까지 중단된 ping과 같은 작은 트래픽에 영향을 줍니다.
- 전송 링 크기(tx-ring-limit)는 큐잉 메커니즘이 작업을 시작하는 속도를 결정합니다. 전송 링 한도가 증가할 때 ping RTT가 증가하면 이러한 영향을 확인할 수 있습니다. 따라서 WFQ 또는 LLQ를 구현해야 하는 경우 전송 링 제한을 줄이는 것이 좋습니다.
- CBWFQ를 사용하는 WFQ는 소규모 트래픽에 벌크 트래픽보다 우선 순위를 지정합니다.

관련 정보

- [ATM 기술 지원 페이지](#)
- [혼잡 관리 개요](#)
- [기술 지원 및 문서 - Cisco Systems](#)