Informazioni su Weighted Fair Queuing su ATM

Sommario

Introduzione

Questo documento offre un'introduzione alle code di traffico che utilizzano la tecnologia WFQ (Weighted Fair Queuing).

La WFQ è stata introdotta per consentire collegamenti a bassa velocità, come i collegamenti seriali per fornire un trattamento equo per tutti i tipi di traffico. A tale scopo, WFQ classifica il traffico in diversi flussi (noti anche come conversazioni) in base alle informazioni di livello tre e quattro, quali gli indirizzi IP e le porte TCP. Questa operazione viene eseguita senza che l'utente debba definire gli elenchi degli accessi. Ciò significa che il traffico a bassa larghezza di banda ha in effetti la priorità sul traffico a elevata larghezza di banda, perché il traffico a elevata larghezza di banda condivide i supporti di trasmissione in proporzione al loro peso assegnato.

Tuttavia, WFQ presenta alcune limitazioni:

- Non è scalabile se il valore del flusso aumenta considerevolmente.

- Il modulo WFQ nativo non è disponibile sulle interfacce ad alta velocità come le interfacce ATM.

CBWFQ (Class-based weighted fair queuing) fornisce una soluzione a queste limitazioni.

A differenza delle WFQ standard, CBWFQ consente di definire le classi di traffico. È inoltre possibile applicare parametri, ad esempio limiti di larghezza di banda e di coda. La larghezza di banda assegnata a una classe viene utilizzata per calcolare il peso della classe. Anche il peso di ogni pacchetto che soddisfa i criteri della classe viene calcolato da questo. WFQ viene quindi applicato alle classi, che possono includere diversi flussi, anziché i flussi stessi.

Per ulteriori informazioni su come configurare CBWFQ, consultare i seguenti documenti:

- CBWFQ (Weighted Fair Queuing) basato su classi per-VC su router Cisco 7200, 3600 e 2600
- Accodamento equo ponderato basato su classi per-VC su piattaforme basate su RSP

Le interfacce ATM non supportano i WFQ nativi basati sul flusso configurati direttamente su un'interfaccia con il comando **fair-queue**. Tuttavia, con il software che supporta CBWFQ, è possibile configurare WFQ basato sul flusso all'interno della classe predefinita, come mostrato nell'esempio:

```
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
```

Prerequisiti

Requisiti

Nessun requisito specifico previsto per questo documento.

Componenti usati

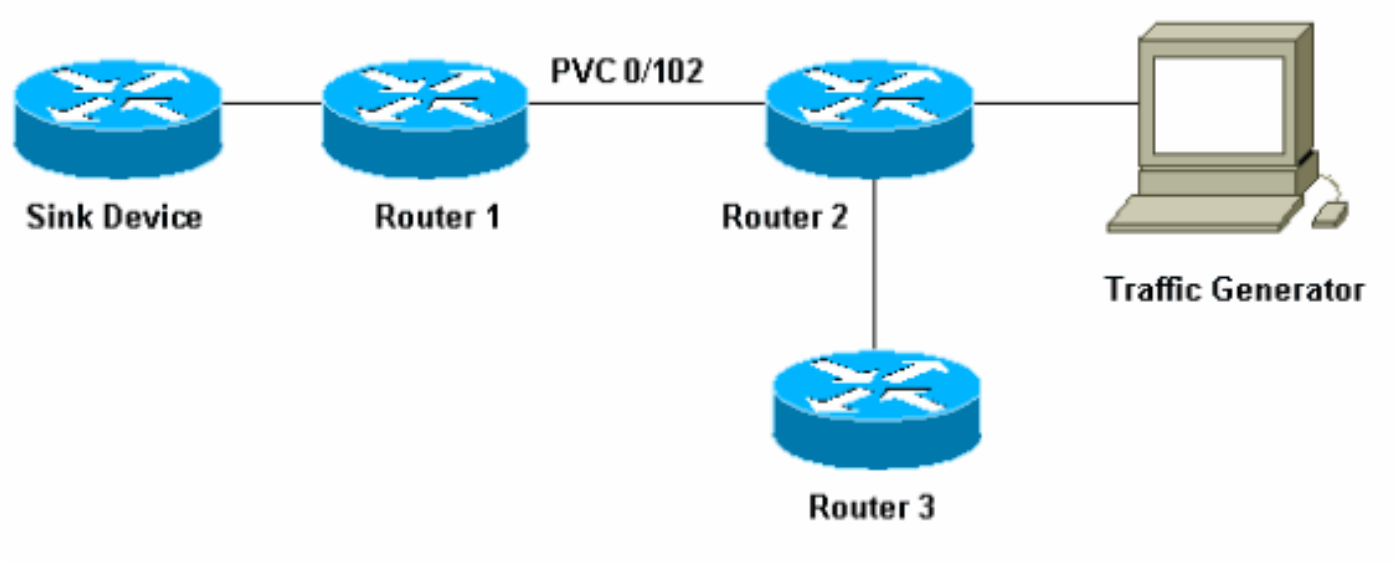
Il documento può essere consultato per tutte le versioni software o hardware.

Convenzioni

Per ulteriori informazioni sulle convenzioni usate, consultare il documento Cisco sulle convenzioni nei suggerimenti tecnici.

Esempio di rete

Utilizzare questa impostazione per illustrare il funzionamento di WFQ:



In questa configurazione, i pacchetti possono essere archiviati in una di queste due code:

- La coda FIFO (First In First Out) dell'adattatore della porta e del modulo di rete
- La coda nel software Cisco IOS®, sulla memoria [I/O] di input/output del router, a cui è possibile applicare le funzionalità QoS (Quality of Service), ad esempio CBWFQ

La coda FIFO sull'adattatore porta memorizza i pacchetti prima che vengano segmentati in celle per la trasmissione. Quando la coda è piena, l'adattatore della porta o il modulo di rete segnala al software IOS che la coda è congestionata. Questo meccanismo è chiamato contropressione. Alla ricezione del segnale, il router si interrompe per inviare i pacchetti alla coda FIFO di interfaccia e archivia i pacchetti nel software IOS finché la coda non viene nuovamente congestionata. Quando i pacchetti sono archiviati in IOS, il sistema può applicare QoS.

Come impostare il limite dell'anello di trasmissione

Uno dei problemi è che, maggiore è la dimensione della coda FIFO sull'interfaccia, maggiore è il ritardo prima della trasmissione dei pacchetti alla fine della coda. Ciò può causare gravi problemi di prestazioni per il traffico sensibile al ritardo, ad esempio il traffico vocale.

Il comando **tx-ring-limit** del circuito virtuale permanente (PVC) consente di ridurre le dimensioni della coda 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
```

La *<dimensione>* che è possibile specificare qui è un numero di pacchetti, per i router Cisco 2600 e 3600, o una quantità di particelle, per i router Cisco 7200 e 7500.

La riduzione delle dimensioni dell'anello di trasmissione presenta due vantaggi:

- Riduce il tempo di attesa dei pacchetti nella coda FIFO prima che venga segmentata.
- Accelera l'uso di QoS nel software IOS.

Impatto del limite dell'anello di trasmissione

Osservare l'impatto del limite dell'anello di trasmissione che utilizza la configurazione mostrata nel diagramma di rete precedente. Si supponga quanto segue:

- Il generatore di traffico invia il traffico (pacchetti da 1500 byte) al dispositivo sink e questo traffico sovraccarica il PVC 0/102 tra il router1 e il router2.
- Il router3 tenta di eseguire il ping tra il router1.
- WFQ è abilitato sul router2.

Esaminare due configurazioni che utilizzano limiti diversi degli anelli di trasmissione per verificare l'impatto.

Esempio A

In questo esempio si imposta l'anello di trasmissione su tre (tx-ring-limit=3). Questo è ciò che si verifica quando si esegue il ping tra router1 e router3:

```
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/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
   !--- ping (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
   !--- This is traffic from the traffic generator.
```

Esempio B

In questo esempio, si imposta l'anello di trasmissione su 40 (tx-ring-limit=40). Questo è quanto si verifica quando si utilizza lo stesso comando ping dell'esempio 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 ms
```

Come si può vedere, maggiore è il limite degli anelli di trasmissione, maggiore è il tempo di andata e ritorno del ping (RTT). Da ciò si può dedurre che un grande limite di anello di trasmissione può portare a ritardi significativi nella trasmissione.

Come calcolare il peso

Nell'output **show queue atm** dell'esempio A, viene mostrato come è stato assegnato un peso a ciascuna conversazione. Osservate più dettagliatamente questo aspetto:

```
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
```

Quando si utilizza WFQ, è possibile calcolare il peso di ogni conversazione utilizzando la formula seguente:

- **weight=32384/(precedence+1)** - per il software Cisco IOS versione 12.0(5)T e successive.
- **weight=4096/(precedenza+1)** - per le versioni software Cisco IOS precedenti alla 12.0(5)T.

Come calcolare l'ora di programmazione

Èora possibile usare questi pesi per calcolare l'ora di pianificazione di ciascun pacchetto, quando il pacchetto viene inoltrato dalla coda IOS all'adattatore di porta o alla coda FIFO del modulo di rete.

Èpossibile calcolare l'ora di programmazione dell'output utilizzando questa formula, dove **queue_tail_time** è l'ora di programmazione corrente:

output scheduling time= queue_tail_time + pktsize*weight

Funzionamento di WFQ

In questa sezione viene illustrato il funzionamento di WFQ. Il principio di WFQ è che i pacchetti di

piccole dimensioni, o piccoli pacchetti, devono avere la priorità quando vengono inviati.

Creare un flusso che comprenda dieci pacchetti di grandi dimensioni e quattro pacchetti più piccoli (di 82 byte) che utilizzi un generatore di traffico per verificare questa condizione.

Nell'esempio, il router2 è un router Cisco 7200 con un PA-A3 (ATM Port Adapter). Ciò è importante perché le dimensioni della coda FIFO di output sull'adattatore della porta sono espresse in particelle e non in pacchetti. Vedere la descrizione di Cos'è una particella? per ulteriori informazioni.

Cos'è una particella?

Invece di allocare un pezzo di memoria contigua per un buffer, il buffering delle particelle alloca pezzi di memoria non contigui (sparsi), chiamati particelle, e poi li collega per formare un buffer logico del pacchetto. Questo si chiama buffer di particelle. In uno schema di questo tipo, un pacchetto può quindi essere distribuito su più particelle.

Nel router 7200, le dimensioni delle particelle sono 512 byte.

Usare il comando **show buffers** per verificare se i router Cisco 7200 usano le particelle:

```
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
```

Test A

Di seguito vengono riportati alcuni test che consentono di illustrare la funzionalità WFQ. In questo primo test, verifica se la larghezza di banda può essere condivisa tra conversazioni diverse.

In questo test, il generatore di traffico ha inviato il traffico in modo sufficientemente rapido da sovraccaricare il PVC 0/102 tra il router1 e il router2. Eseguire un ping tra il router3 e il router1 sullo stesso PVC:

```
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)!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!.[break]
```

```
Success rate is 98 percent (604/613), round-trip min/avg/max = 164/190/232 ms
```

Come mostrato nell'output, finché WFQ non è abilitato sull'interfaccia, il traffico impedisce il passaggio di altro traffico e blocca la linea. Quando WFQ è abilitato, il ping ha esito positivo.

Potete vedere da questo che, con l'uso di WFQ, la larghezza di banda può essere condivisa tra diverse conversazioni senza che una blocchi le altre.

Test B

Questa è la *modalità* di condivisione della larghezza di banda.

Il flusso inviato dal generatore di traffico è un burst composto da dieci pacchetti di grandi dimensioni, seguiti da quattro pacchetti più piccoli di 82 byte. Il valore viene inviato a 100 Mbps al router2. Quando si invia il burst, la coda di output sull'interfaccia ATM router2 è vuota. Il router2 invia questi pacchetti tramite un PVC di 10 KB (un PVC molto lento) per garantire che si verifichi una congestione nella coda di output.

Per semplificare questo processo, eseguire il test in più fasi:

Fase 1

Il grande traffico comprende dieci pacchetti da 482 byte. Poiché le particelle sull'adattatore PA-A3 sono di 512 byte, ogni pacchetto, sia grande che piccolo, dovrebbe prendere una particella quando viene memorizzato nella coda di output dell'adattatore di porta. Il router ha un limite di tre squilli di trasmissione (tx-ring-limit=3). Questo è un esempio di quello che si può vedere sul dispositivo di sink:

```
   .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 15:39:17.820: 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
```

ossia i quattro pacchetti da 482 byte inviati prima dei pacchetti da 82 byte, ai quali normalmente dovrebbe essere assegnata la priorità. Ecco perché succede.

Poiché la frammentazione è composta principalmente da dieci pacchetti da 482 byte, questi raggiungono per primi il router, seguiti dai pacchetti da 82 byte. Poiché i pacchetti da 482 byte arrivano in un momento in cui non c'è congestione e non c'è traffico, un pacchetto viene immediatamente inserito in coda alla segmentazione e al riassemblaggio dell'adattatore della porta (SAR) per essere suddiviso in celle e inviato in filo. In altre parole, l'anello di trasmissione è ancora vuoto.

È possibile calcolare che il tempo necessario per inviare un pacchetto da 482 byte è maggiore del tempo necessario al generatore di traffico per inviare lo burst totale. Si può quindi supporre che, quando il primo pacchetto da 482 byte viene accodato all'adattatore di porta, più pacchetti da 482 byte della frammentazione siano già presenti nel router. Pertanto, è possibile accodare all'anello di trasmissione più pacchetti da 482 byte. Altri tre pacchetti di 482 byte vengono messi in coda usando le tre particelle libere presenti.

Nota: i pacchetti vengono accodati nell'anello di trasmissione non appena c'è una particella libera, anche se hanno bisogno di più di una particella per essere memorizzati.

A questo punto, c'è congestione, poiché le tre particelle sono piene. Pertanto, l'accodamento viene avviato in IOS. Quando i quattro pacchetti da 82 byte raggiungono il router, si verifica una congestione. Questi quattro pacchetti vengono accodati e sui due flussi viene utilizzato WFQ. Osservare la coda ATM che usa il comando **show queue ATM** per verificare quanto segue:

```
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
```

I debug mostrano che i primi quattro pacchetti di 482 byte sono seguiti dai pacchetti da 82 byte. Questi pacchetti piccoli escono dal router prima dei pacchetti grandi. Ciò significa che, non appena si verifica una congestione, i pacchetti di piccole dimensioni hanno la priorità sui pacchetti di grandi dimensioni.

Per verificare questa condizione, utilizzare le formule relative al peso e alla durata di programmazione fornite nella sezione Calcolo del peso.

Fase 2

Se si aumenta il limite degli anelli di trasmissione a cinque e i pacchetti di grandi dimensioni sono 482 byte, in base all'output precedente, verranno visualizzati sei pacchetti di 482 byte prima del verificarsi della congestione, seguiti da quattro pacchetti di 82 byte e quindi da altri quattro pacchetti di 482 byte:

```
.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
```

Come potete vedere qui, questo è ciò che accade.

Fase 3

La dimensione della particella è di 512 byte. Pertanto, se l'anello di trasmissione è espresso in particelle e usate pacchetti leggermente più grandi delle dimensioni delle particelle, ognuno prende due particelle. Come si evince dall'uso di pacchetti di 582 byte e di un anello di trasmissione di tre byte. Con questi parametri, dovrebbero essere visualizzati tre pacchetti di 582 byte. Una viene inviata senza traffico sull'interfaccia ATM, lasciando tre particelle libere. Pertanto, è possibile accodare altri due pacchetti, seguiti da quattro pacchetti di 82 byte e quindi da sette pacchetti di 582 byte:

```
.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
```

Fase 4

Prendete un pacchetto delle dimensioni di 1482 (tre particelle) e definite un anello di trasmissione di cinque. Se l'anello di trasmissione è definito in particelle, si nota qualcosa di simile:

- Un pacchetto trasmesso immediatamente
- Un pacchetto che prende tre delle cinque particelle
- Un pacchetto in coda perché due particelle sono libere

```
.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
```

Riepilogo

Dai test effettuati è possibile concludere quanto segue:

- Sui PVC lenti senza WFQ, il traffico di massa influisce sul traffico di piccole dimensioni, ad esempio i ping che vengono bloccati finché non viene abilitato WFQ.
- La dimensione dell'anello di trasmissione (limite dell'anello tx) determina la velocità con cui il meccanismo di accodamento inizia a svolgere il proprio lavoro. Ciò si verifica in particolare con l'aumento del ping RTT quando il limite degli anelli di trasmissione aumenta. Pertanto, se

è necessario implementare WFQ o LLQ, ha senso ridurre il limite dell'anello di trasmissione.
- WFQ che utilizza CBWFQ dà la priorità ai piccoli traffici rispetto al traffico di massa.

Informazioni correlate

- Pagine di supporto per la tecnologia ATM
- Panoramica della gestione delle congestioni
- Documentazione e supporto tecnico – Cisco Systems