Troubleshooting Output Drops with Priority Queueing

Document ID: 10105

Contents

Introduction
Prerequisites
Requirements
Components Used
Conventions
Drops with ip rtp priority and LLQ
Drops with Legacy Priority Queueing
Traffic Measurement with a Token Bucket
Troubleshooting Steps to Diagnose Drops
Step 1 − Collect Data
Step 2 − Ensure Sufficient Bandwidth
Step 3 − Ensure Sufficient Burst Size
Step 4 − debug priority
Other Causes for Drops
Priority Queues Drops and Frame Relay
Related Information

Introduction

This document provides tips on how to troubleshoot output drops that result from a priority queueing mechanism configuration on a router interface.

Prerequisites

Requirements

Readers of this document should be familiar with these concepts:

- **priority−group** or **frame−relay priority−group**− Enables Cisco legacy priority queueing mechanism. Supports up to four levels of priority queues.
- **ip rtp priority** or **frame−relay ip rtp priority** − Matches on UDP port numbers for Real−Time Protocol (RTP) traffic encapsulating VoIP packets and places these packets in a priority queue.
- **priority** − Enables Cisco's low latency queueing (LLQ) feature and uses the command structure of the modular quality of service QoS command−line interface (CLI).

A router can report output drops when any of these methods are configured, but there are important functional differences between the methods and the reason for drops in each case.

The information presented in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If you are working in a live network, ensure that you understand the potential impact of any command before you use it.

Components Used

This document is not restricted to specific software and hardware versions.

The information presented in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If you are working in a live network, ensure that you understand the potential impact of any command before using it.

Conventions

For more information on document conventions, refer to Conventions Used in Cisco Technical Tips.

Drops with ip rtp priority and LLQ

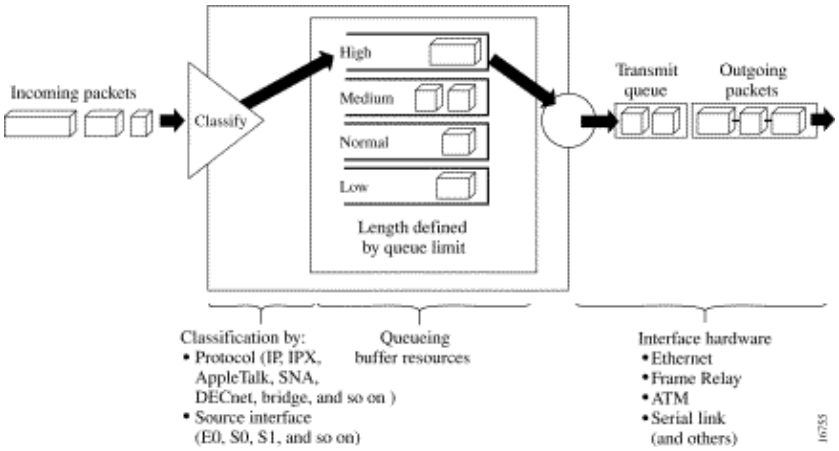
The Cisco IOS Configuration Guide warns against output drops with these priority queueing mechanisms:

- **ip rtp priority**: Because the **ip rtp priority** command gives absolute priority over other traffic, it should be used with care. In the event of congestion, if the traffic exceeds the configured bandwidth, then all the excess traffic is dropped.
- **priority** command and LLQ: When you specify the **priority** command for a class, it takes a bandwidth argument that gives maximum bandwidth. In the event of congestion, policing is used to drop packets when the bandwidth is exceeded.

These two mechanisms use a built−in policer to meter the traffic flows. The purpose of the policer is to ensure that the other queues are serviced by the queueing scheduler. In the cisco original priority queueing feature, which uses the **priority−group** and **priority−list** commands, the scheduler always serviced the highest−priority queue first. If there was always traffic in the high priority queue, the lower−priority queues were starved of bandwidth and packets going to non−priority queues.

Drops with Legacy Priority Queueing

Priority queueing (PQ) is the cisco legacy priority queueing mechanism. As illustrated below, PQ supports up to four levels of queues: high, medium, normal, and low.



Enabling priority queueing on an interface changes the Output queue display, as illustrated below. Before priority queueing the Ethernet interface is using a single output hold queue with the default queue size of 40 packets.

```
R6−2500# show interface ethernet0
```

```
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c4e.59b1 (bia 0000.0c4e.59b1)
  Internet address is 42.42.42.2/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:03, output 00:00:02, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     239407 packets input, 22644297 bytes, 0 no buffer
     Received 239252 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 input packets with dribble condition detected
     374436 packets output, 31095372 bytes, 0 underruns
     0 output errors, 1 collisions, 13 interface resets
     0 babbles, 0 late collision, 8 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```

After enabling PQ the Ethernet interface now is using four priority queues with varying queue limits, as shown in the output below:

```
R6-2500(config)# interface ethernet0
R6-2500(config-if)# priority-group 1
R6-2500(config-if)# end
R6-2500# show interface ethernet 0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c4e.59b1 (bia 0000.0c4e.59b1)
  Internet address is 42.42.42.2/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:03, output 00:00:03, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0 (size/max/drops); Total output drops: 0
  Queueing strategy: priority-list 1
  Output queue (queue priority: size/max/drops):
     high: 0/20/0, medium: 0/40/0, normal: 0/60/0, low: 0/80/0
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     239411 packets input, 22644817 bytes, 0 no buffer
     Received 239256 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 input packets with dribble condition detected
     374440 packets output, 31095658 bytes, 0 underruns
     0 output errors, 1 collisions, 14 interface resets
     0 babbles, 0 late collision, 8 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```

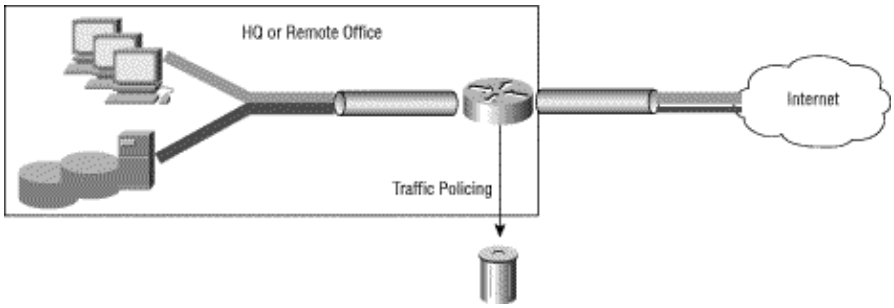
The **priority-list {list-number}** command is used to assign traffic flows to a specific queue. As the packets arrive at an interface, the priority queues on that interface are scanned for packets in a descending order of priority. The high priority queue is scanned first, then the medium priority queue, and so on. The packet at the head of the highest priority queue is chosen for transmission. This procedure is repeated every time a packet is to be sent.

Each queue is defined by a maximum length or by the maximum number of packets the queue can hold. When an arriving packet would cause the current queue depth to exceed the configured queue limit, the packet is dropped. Thus, as noted above, output drops with PQ typically are due to exceeding the queue limit and not to an internal policer, as is the typical case with LLQ. The **priority-list** *list-number* **queue-limit** command

changes the size of a priority queue.

Traffic Measurement with a Token Bucket

LLQ and IP RTP Priority implement the built−in policer by using a token bucket as a traffic measurement system. This section looks at the token bucket concept.



A token bucket itself has no discard or priority policy. The token bucket metaphor works along the following lines:

- Tokens are put into the bucket at a certain rate.
- Each token signifies permission for the source to send a certain number of bits into the network.
- To send a packet, the traffic regulator must be able to remove a number of tokens from the bucket equal in representation to the packet size.
- If there are not enough tokens are in the bucket to send a packet, the packet either waits until the bucket has enough tokens (in the case of a shaper) or the packet is discarded or marked down (in the case of a policer).
- The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded and are not available to future packets. Thus, at any time, the largest burst an application can send into the network is roughly proportional to the size of the bucket. A token bucket permits burstiness, but bounds it.

Let's look at an example using packets and a committed information rate (CIR) of 8000 bps.

- In this example, the initial token buckets starts full at 1000 bytes.
- When a 450 byte packet arrives, the packet conforms because enough bytes are available in the conform token bucket. The packet is sent, and 450 bytes are removed from the token bucket, leaving 550 bytes.
- When the next packet arrives 0.25 seconds later, 250 bytes are added to the token bucket ((0.25 * 8000)/8), leaving 700 bytes in the token bucket. If the next packet is 800 bytes, the packet exceeds, and is dropped. No bytes are taken from the token bucket.

Troubleshooting Steps to Diagnose Drops

Step 1 − Collect Data

The steps to collect data are shown below.

1. Execute the following commands several times and determine how quickly and how often the drops increment. Use the output to establish a baseline of your traffic patterns and traffic levels. Figure out what the "normal" drop rate is on the interface.

    ♦ **show queueing interface**

```
router# show queueing interface hssi 0/0/0
          Interface Hssi0/0/0 queueing strategy: priority

          Output queue utilization (queue/count)

           high/12301 medium/4 normal/98 low/27415
```

- **show interface** − Monitor the load value displayed in the output. In addition, make sure that the sum of the per−queue drop counts in the **show interface** output is equivalent to the output drops count. The **show interface** output drops counter should display the total aggregate of all drops on output, including WRED discard, discard because of buffer shortage ("no buffer" errors), and even discards in on−board port adapter memory.

```
router# show interface serial 4/1/2

Serial4/1/2 is up, line protocol is up
Hardware is cyBus Serial
Description: E1 Link to 60W S9/1/2 Backup
Internet address is 169.127.18.228/27
MTU 1500 bytes, BW 128 Kbit, DLY 21250 usec, rely 255/255, load 183/255
Encapsulation HDLC, loopback not set, keepalive set (10 sec)
Last input 00:00:00, output 00:00:00, output hang never
Last clearing of "show interface" counters 5d10h
Input queue: 0/75/0 (size/max/drops); Total output drops: 68277
Queueing strategy: priority−list 7
Output queue: high 0/450/0, medium 0/350/143, normal 0/110/27266, low 0/100/4
5 minute input rate 959000 bits/sec, 419 packets/sec
5 minute output rate 411000 bits/sec, 150 packets/sec
144067307 packets input, 4261520425 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
42 input errors, 34 CRC, 0 frame, 0 overrun, 1 ignored, 8 abort
69726448 packets output, 2042537282 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 output buffer failures, 46686454 output buffers swapped out
0 carrier transitions
```

Note: Some interfaces display separate "txload" and "rxload" values.

```
Hssi0/0/0 is up, line protocol is up
 Hardware is cyBus HSSI
 MTU 1500 bytes, BW 7500 Kbit, DLY 200 usec,
 reliability 255/255, txload 138/255, rxload 17/255
 Encapsulation FRAME−RELAY IETF, crc 16, loopback not set
 Keepalive set (5 sec)
 LMI enq sent 4704, LMI stat recvd 4704, LMI upd recvd 0, DTE LMI up
 LMI enq recvd 0, LMI stat sent 0, LMI upd sent 0
 LMI DLCI 1023 LMI type is CISCO frame relay DTE
 Broadcast queue 0/256, broadcasts sent/dropped 8827/0, interface
 broadcasts 7651
 Last input 00:00:00, output 00:00:00, output hang never
 Last clearing of "show interface" counters 06:31:58
 Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 84
 Queueing strategy: priority−list 1
 Output queue (queue priority: size/max/drops):
 high: 0/20/0, medium: 0/40/0, normal: 0/60/0, low: 0/80/84
 5 minute input rate 524000 bits/sec, 589 packets/sec
 5 minute output rate 4080000 bits/sec, 778 packets/sec
 11108487 packets input, 1216363830 bytes, 0 no buffer
 Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
 0 parity
 0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
 15862186 packets output, 3233772283 bytes, 0 underruns
 0 output errors, 0 applique, 1 interface resets
 0 output buffer failures, 2590 output buffers swapped out
 0 carrier transitions
```

```
LC=down CA=up TM=down LB=down TA=up LA=down
```

- **show policy−map interface** *interface−name* − Look for a non−zero value for the "pkts discards" counter.

```
Router# show policy−map interface s1/0
 Serial1/0.1: DLCI 100 −
 output : mypolicy
  Class voice
   Weighted Fair Queueing
       Strict Priority
       Output Queue: Conversation 72
         Bandwidth 16 (kbps) Packets Matched 0
        (pkts discards/bytes discards) 0/0
  Class immediate−data
   Weighted Fair Queueing
       Output Queue: Conversation 73
         Bandwidth 60 (%) Packets Matched 0
         (pkts discards/bytes discards/tail drops) 0/0/0
         mean queue depth: 0
         drops: class  random   tail     min−th   max−th   mark−prob
                0      0        0        64       128      1/10
                1      0        0        71       128      1/10
                2      0        0        78       128      1/10
                3      0        0        85       128      1/10
                4      0        0        92       128      1/10
                5      0        0        99       128      1/10
                6      0        0        106      128      1/10
                7      0        0        113      128      1/10
                rsvp   0        0        120      128      1/10
```

Note: The following example output displays matching values for the "packets" and "pkts matched" counters. This condition indicates that a large number of packets are being process switched or that the interface is experiencing extreme congestion. Both of these conditions can lead to exceeding a class's queue limit and should be investigated.

```
router# show policy−map interface

Serial4/0

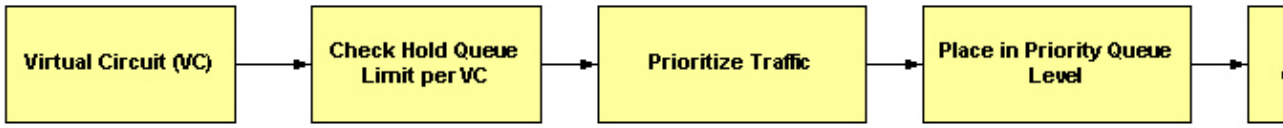
Service−policy output: policy1

Class−map: class1 (match−all)
189439 packets, 67719268 bytes
5 minute offered rate 141000 bps, drop rate 0 bps
Match: access−group name ds−class−af3
Weighted Fair Queueing
Output Queue: Conversation 265
Bandwidth 50 (%) Max Threshold 64 (packets)
(pkts matched/bytes matched) 189439/67719268
(depth/total drops/no−buffer drops) 0/0/0
```

2. Characterize the traffic flows and the packets in those flows.

   - What is the average packet size?
   - In which direction are the MTU−sized frame flowing? Many traffic flows are asynchronous with respect to load. For example, with an FTP download, most of the MTU−sized packets flow from the FTP server to client. Packets from the FTP client to the server are simple TCP ACKs.
   - Are the packets using TCP or UDP? TCP allows each flow to send an authorized number of packets before the source needs to suspend transmission and wait for the destination to acknowledge the transmitted packets.

3. With Frame Relay, determine whether the drops are occuring at the interface queue or at a per−VC queue. The following diagram illustrates the flow of packets through a Frame Relay virtual circuit:



4. Priority Queueing supports up to four output queues, one per priority queue level and each queue is defined by a queue limit. The queueing system checks the size of the queue against the configured queue limit before placing the packet in a queue. If the selected queue is full, the router drops the packet. Try increasing the queue size with the **priority−list {#} queue−limit** command and resume monitoring.

Step 2 − Ensure Sufficient Bandwidth

With LLQ, policing allows for fair treatment of other data packets in other class−based weighted fair queuing (CBWFQ) or WFQ queues. To avoid packet drops, be certain to allocate an optimum amount of bandwidth to the priority queue, taking into consideration the type of codec used and interface characteristics. IP RTP Priority will not allow traffic beyond the allocated amount.

It is always safest to allocate slightly more than the known required amount of bandwidth to the priority queue. For example, suppose you allocated 24 kbps bandwidth, the standard amount required for voice transmission, to the priority queue. This allocation seems safe because transmission of voice packets occurs at a constant bit rate. However, because the network and the router or switch can use some of the bandwidth to produce jitter and delay, allocating slightly more than the required amount of bandwidth (such as 25 kbps) ensures constancy and availability.

The bandwidth allocated for a priority queue always includes the Layer 2 encapsulation header. It does not include the cyclic redundancy check (CRC). (Refer to What Bytes Are Counted by IP to ATM CoS Queueing? for more information.) Although it is only a few bytes, the CRC imposes an increasing impact as traffic flows include a higher number of small packets.

In addition, on ATM interfaces, the bandwidth allocated for a priority queue does not include the following ATM cell tax overhead:

- Any padding by the segmentation and reassembly (SAR) to make the last cell of a packet an even multiple of 48 bytes.
- 4−byte CRC of the ATM Adaptation Layer 5 (AAL5) trailer.
- 5−byte ATM cell header.

When you calculate the amount of bandwidth to allocate for a given priority class, you must account for the fact that Layer 2 headers are included. When ATM is used, you must account for the fact that ATM cell tax overhead is not included. You must also allow bandwidth for the possibility of jitter introduced by network devices in the voice path. Refer to the Low Latency Queueing Feature Overview.

When using priority queueing to carry VoIP packets, refer to Voice over IP − Per Call Bandwidth Consumption.

Step 3 − Ensure Sufficient Burst Size

The treatment of a series of packets leaving an interface through a priority queue depends on the size of the packet and the number of bytes remaining in the token bucket. It is important to consider the characteristics of the traffic flow being directed to the priority queue because LLQ uses a policer, not a shaper. A policer uses a token bucket as follows:

- The bucket is filled up with tokens based on the class rate to a maximum of the burst parameter.

- If the number of tokens is greater than or equal to packet size, the packet is sent, and the token bucket is decremented. Otherwise, the packet is dropped.

The default burst value of LLQ's token bucket traffic meter is computed as 200 milliseconds of traffic at the configured bandwidth rate. In some cases, the default value is inadequate, particularly when TCP traffic is going into the priority queue. TCP flows are typically bursty and may require a burst size greater than the default assigned by the queueing system, particularly on slow links.

The following sample output was generated on an ATM PVC with a sustained cell rate of 128 kbps. The queueing system adjusts the burst size as the value specified with the **priority** command changes.

```
7200-17# show policy-map int atm 4/0.500
 ATM4/0.500: VC 1/500 -

Service-policy output: drops

    Class-map: police (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Weighted Fair Queueing
        Strict Priority
        Output Queue: Conversation 24
        Bandwidth 90 (%)
        Bandwidth 115 (kbps) Burst 2875 (Bytes)

!--- Burst value of 2875 bytes is assigned when
        !--- the reserved bandwidth value is 115 kbps.

        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any

7200-17# show policy-map int atm 4/0.500
 ATM4/0.500: VC 1/500 -

  Service-policy output: drops

    Class-map: police (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Weighted Fair Queueing
        Strict Priority
        Output Queue: Conversation 24
        Bandwidth 50 (%)
        Bandwidth 64 (kbps) Burst 1600 (Bytes)

!--- Burst value changes to 1600 bytes when the
        !--- reserved bandwidth value is changed to 64 kbps.

        (pkts matched/bytes matched) 0/0
        (total drops/bytes drops) 0/0

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

The functionality of LLQ was extended to allow a configurable Committed Burst (Bc) size with the Configuring Burst Size in Low Latency Queueing feature. With this new functionality, the network can now accommodate temporary bursts of traffic and handle network traffic more efficiently.

Use the burst parameter with the **priority** command to increase the burst value from 1600 bytes to 3200 bytes.

```
policy−map AV
  class AV
  priority percent 50 3200
```

Note: A high value increases the effective bandwidth that the priority class may use and may give the appearance that the priority classes are getting more than their fair share of the bandwidth.

In addition, the queueing system originally assigned an internal queue limit of 64 packets to the low−latency queue. In some cases, when a burst of 64 packets arrived at the priority queue, the traffic meter would determine that the burst conformed to the configured rate, but the number of packets exceeded the queue limit. As a result, some packets were tail−dropped. Cisco bug ID CSCdr51979 (registered customers only) resolves this problem by allowing the priority queue size to grow as deep as allowed by the traffic meter.

The following output was captured on a Frame Relay PVC configured with a CIR of 56 kbps. In the first set of sample output, the combined offered rate of the c1 and c2 classes is 76 kbps. The reason is that the calculated values of offered rates minus drop rates do not represent the actual transmission rates and are not including packets sitting in the shaper before they are transmitted.

```
router# show policy−map int s2/0.1
  Serial2/0.1: DLCI 1000 −

   Service−policy output: p

     Class−map: c1 (match−all)
       7311 packets, 657990 bytes
       30 second offered rate 68000 bps, drop rate 16000 bps
       Match: ip precedence 1
       Weighted Fair Queueing
         Strict Priority
         Output Queue: Conversation 24
         Bandwidth 90 (%)
         Bandwidth 50 (kbps) Burst 1250 (Bytes)
         (pkts matched/bytes matched) 7311/657990
         (total drops/bytes drops) 2221/199890

     Class−map: c2 (match−all)
       7311 packets, 657990 bytes
       30 second offered rate 68000 bps, drop rate 44000 bps
       Match: ip precedence 2
       Weighted Fair Queueing
         Output Queue: Conversation 25
         Bandwidth 10 (%)
         Bandwidth 5 (kbps) Max Threshold 64 (packets)
         (pkts matched/bytes matched) 7310/657900
         (depth/total drops/no−buffer drops) 64/6650/0

     Class−map: class−default (match−any)
       2 packets, 382 bytes
       30 second offered rate 0 bps, drop rate 0 bps
       Match: any
```

In this second set of output, the **show policy−map** interface counters have normalized. On the 56 kbps PVC, the class c1 is sending about 50 kbps, and the class c2 is sending about 6 kbps.

```
router# show policy−map int s2/0.1
```

```
Serial2/0.1: DLCI 1000 −

 Service−policy output: p

   Class−map: c1 (match−all)
     15961 packets, 1436490 bytes
     30 second offered rate 72000 bps, drop rate 21000 bps
     Match: ip precedence 1
     Weighted Fair Queueing
       Strict Priority
       Output Queue: Conversation 24
       Bandwidth 90 (%)
       Bandwidth 50 (kbps) Burst 1250 (Bytes)
       (pkts matched/bytes matched) 15961/1436490
       (total drops/bytes drops) 4864/437760

   Class−map: c2 (match−all)
     15961 packets, 1436490 bytes
     30 second offered rate 72000 bps, drop rate 66000 bps
     Match: ip precedence 2
     Weighted Fair Queueing
       Output Queue: Conversation 25
       Bandwidth 10 (%)
       Bandwidth 5 (kbps) Max Threshold 64 (packets)
       (pkts matched/bytes matched) 15960/1436400
       (depth/total drops/no−buffer drops) 64/14591/0

   Class−map: class−default (match−any)
     5 packets, 1096 bytes
     30 second offered rate 0 bps, drop rate 0 bps
     Match: any
```

Step 4 − debug priority

The **debug priority** command displays priority queueing output if packets are dropped from the priority queue.

Caution: Before you use **debug** commands, refer to Important Information on **Debug** Commands. The **debug priority** command may print a large amount of disruptive debug output on a production router. The amount depends on the level of congestion.

The following sample output was generated on a Cisco 3640.

```
r3−3640−5# debug priority
Priority output queueing debugging is on

r3−3640−5# ping 10.10.10.2

Type escape sequence to abort.
Sending 5, 100−byte ICMP Echos to 10.10.10.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round−trip min/avg/max = 56/57/60 ms
r3−3640−5#
00:42:40: PQ: Serial0/1: ip −> normal
00:42:40: PQ: Serial0/1 output (Pk size/Q 104/2)
00:42:40: PQ: Serial0/1: ip −> normal
00:42:40: PQ: Serial0/1 output (Pk size/Q 104/2)
00:42:40: PQ: Serial0/1: ip −> normal
00:42:40: PQ: Serial0/1 output (Pk size/Q 104/2)
00:42:40: PQ: Serial0/1: ip −> normal
00:42:40: PQ: Serial0/1 output (Pk size/Q 104/2)
00:42:40: PQ: Serial0/1: ip −> normal
```

```
00:42:40: PQ: Serial0/1 output (Pk size/Q 104/2)
00:42:41: PQ: Serial0/1 output (Pk size/Q 13/0)
r3-3640-5#no debug priority
00:42:51: PQ: Serial0/1 output (Pk size/Q 13/0)
Priority output queueing debugging is off
```

In the following **debug priority** output, 64 indicates the actual priority queue depth at the time the packet was dropped.

```
*Feb 28 16:46:05.659:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.671:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.679:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.691:WFQ:dropping a packet from the priority queue 64
```

Other Causes for Drops

The following reasons for output drops with LLQ were discovered by the Cisco Technical Assistance Center (TAC) during case troubleshooting and documented in a Cisco bug report:

- Increasing the weighted random early detection (WRED) maximum threshold values on another class depleted the available buffers and led to drops in the priority queue. To help diagnose this problem, a "no−buffer drops" counter for the priority class is planned for a future release of IOS.
- If the input interface's queue limit is smaller than the output interface's queue limit, packet drops shift to the input interface. These symptoms are documented in Cisco bug ID CSCdu89226 (registered customers only) . Resolve this problem by sizing the input queue and the output queue appropriately to prevent input drops and allow the outbound priority queueing mechanism to take effect.
- Enabling a feature that is not supported in the CEF−switched or fast−switched path causes a large number of packets to be process−switched. With LLQ, process−switched packets are policed currently whether or not the interface is congested. In other words, even if the interface is not congested, the queueing system meters process−switched packets and ensures the offered load does not exceed the bandwidth value configured with the **priority** command. This problem is documented in Cisco bug ID CSCdv86818 (registered customers only) .

Priority Queues Drops and Frame Relay

Frame Relay is a special case with respect to policing the priority queue. The Low Latency Queueing for Frame Relay feature overview notes the following caveat: "The PQ is policed to ensure that the fair queues are not starved of bandwidth. When you configure the PQ, you specify in kbps the maximum amount of bandwidth available to that queue. Packets that exceed that maximum are dropped." In other words, originally, the priority queue of a service−policy configured in a Frame Relay map class was policed during periods of congestion and non−congestion. IOS 12.2 removes this exception. PQ is still policed with FRF.12, but other non−conforming packets are only dropped if there is congestion.

Related Information

- **QoS Support Page**
- **Cisco IOS Software Release 12.2 Traffic Policing Feature Module**
- **Technical Support − Cisco Systems**

Contacts & Feedback | Help | Site Map
© 2014 − 2015 Cisco Systems, Inc. All rights reserved. Terms & Conditions | Privacy Statement | Cookie Policy | Trademarks of Cisco Systems, Inc.