QoS Output Scheduling on Catalyst 6500/6000 Series Switches Running Cisco IOS System Software

Document ID: 10587

Contents

Introduction
Prerequisites
Requirements
Components Used
Conventions
Background Information
Output Queue Drops
Output Queueing Capability of Different Line Cards on the Catalyst 6500/6000
Understand the Queue Capability of a Port
Configuration, Monitor, and Example Output Scheduling on the Catalyst 6500/6000
Configuration
Monitor Output Scheduling and Verify Configurations
Example of Output Scheduling
Use Output Scheduling to Reduce Delay and Jitter
Reduce Delay
Reduce Jitter
Conclusion
Related Information

Introduction

The use of output scheduling ensures that important traffic is not dropped in the event of heavy oversubscription. This document discusses all of the techniques and algorithms involved in output scheduling on the Catalyst 6500/6000 switch. This document also explains how to configure and verify the operation of output scheduling on the Catalyst 6500/6000 that runs Cisco IOS® Software.

Refer to QoS Output Scheduling on Catalyst 6500/6000 Series Switches Running CatOS System Software for more information on Weighted Random Early Detection (WRED), Weighted Round Robin (WRR), and tail drop.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

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

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Background Information

Output Queue Drops

Output drops are caused by a congested interface. A common cause of this might be traffic from a high bandwidth link that is switched to a lower bandwidth link, or traffic from multiple inbound links that is switched to a single outbound link.

For example, if a large amount of bursty traffic comes in on a gigabit interface and is switched out to a 100Mbps interface, this might cause output drops to increment on the 100Mbps interface. This is because the output queue on that interface is overwhelmed by the excess traffic due to the speed mismatch between the incoming and outgoing bandwidths. The traffic rate on the outgoing interface cannot accept all packets that should be sent out.

In order to resolve the problem, the best solution is to increase the line speed. However, there are ways to prevent, decrease, or control output drops when you do not want to increase the line speed. You can prevent output drops only if output drops are a consequence of short bursts of data. If output drops are caused by a constant high−rate flow, you cannot prevent the drops. However, you can control them.

Output Queueing Capability of Different Line Cards on the Catalyst 6500/6000

If you are unsure about the queueing capability of a port, issue the **show queueing interface {gigabitethernet | fastethernet}** *mod/port* command. Shown here are the first lines of output of a **show queueing interface** command. The port is on a Supervisor Engine 1A line card:

```
cosmos#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round−Robin

  QoS is disabled globally
  Trust state: trust DSCP
  Default COS is 0
  Transmit group−buffers feature is enabled
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
       1         WRR low             2
       2         WRR high            2
       3         Priority            1

!−−− Output suppressed.
```

The output shows that this port has an output queueing type known as 1p2q2t.

Another way to see the type of queueing available on a specific port is to issue the **show interface capabilities** command:

```
la−orion#show interface gigabitethernet 6/2 capabilities
GigabitEthernet6/2
  Model:                 WS−SUP720−BASE
  Type:                  No GBIC
```

```
Speed:                  1000
Duplex:                 full
Trunk encap. type:      802.1Q,ISL
Trunk mode:             on,off,desirable,nonegotiate
Channel:                yes
Broadcast suppression:  percentage(0-100)
Flowcontrol:            rx-(off,on,desired),tx-(off,on,desired)
Membership:             static
Fast Start:             yes
QOS scheduling:         rx-(1p1q4t), tx-(1p2q2t)
CoS rewrite:            yes
ToS rewrite:            yes
Inline power:           no
SPAN:                   source/destination
UDLD                    yes
Link Debounce:          yes
Link Debounce Time:     yes
Ports on ASIC:          1-2
```

Understand the Queue Capability of a Port

There are several type of queues available on the Catalyst 6500/6000 switches. This table explains the notation of the port QoS architecture:

Transmit (Tx)/Receive (Rx)	Queue Notation	No. of Queues	Priority Queue	No. of WRR Queues	No. and Type of Threshold for WRR Queues
Tx	2q2t	2		2	2 configurable tail drop
Tx	1p2q2t	3	1	2	2 configurable WRED
Tx	1p3q1t	4	1	3	1 configurable WRED
Tx	1p2q1t	3	1	2	1 configurable WRED
Rx	1q4t	1			4 configurable tail drop
Rx	1p1q4t	2	1	1	4 configurable tail drop
Rx	1p1q0t	2	1	1	Not configurable
Rx	1p1q8t	2	1	1	8 configurable WRED
Tx	1p3q8t	4	1	3	8 configurable WRED or tail drop
Tx	1p7q8t	8	1	7	8 configurable WRED or tail drop
Rx	1q2t	1			1 configurable tail drop = 1

					nonconfigurable
Rx	1q8t	1			8 configurable tail drop
Rx	2q8t	2		2	8 configurable tail drop

The next table lists some of the modules and queue types in the Rx and Tx sides of the interface or port. If your module is not listed here, use the **show interface capabilities** command to determine the queue capability that is available. The **show interface capabilities** command is described in the Output Queueing Capability of Different Line Cards on the Catalyst 6500/6000 section.

Module	Rx Queues	Tx Queues
WS−X6K−S2−PFC2	1p1q4t	1p2q2t
WS−X6K−SUP1A−2GE	1p1q4t	1p2q2t
WS−X6K−SUP1−2GE	1q4t	2q2t
WS−X6501−10GEX4	1p1q8t	1p2q1t
WS−X6502−10GE	1p1q8t	1p2q1t
WS−X6516−GBIC	1p1q4t	1p2q2t
WS−X6516−GE−TX	1p1q4t	1p2q2t
WS−X6416−GBIC	1p1q4t	1p2q2t
WS−X6416−GE−MT	1p1q4t	1p2q2t
WS−X6316−GE−TX	1p1q4t	1p2q2t
WS−X6408A−GBIC	1p1q4t	1p2q2t
WS−X6408−GBIC	1q4t	2q2t
WS−X6524−100FX−MM	1p1q0t	1p3q1t
WS−X6324−100FX−SM	1q4t	2q2t
WS−X6324−100FX−MM	1q4t	2q2t
WS−X6224−100FX−MT	1q4t	2q2t
WS−X6548−RJ−21	1p1q0t	1p3q1t
WS−X6548−RJ−45	1p1q0t	1p3q1t
WS−X6348−RJ−21	1q4t	2q2t
WS−X6348−RJ21V	1q4t	2q2t
WS−X6348−RJ−45	1q4t	2q2t
WS−X6348−RJ−45V	1q4t	2q2t
WS−X6148−RJ−45V	1q4t	2q2t
WS−X6148−RJ21V	1q4t	2q2t
WS−X6248−RJ−45	1q4t	2q2t
WS−X6248A−TEL	1q4t	2q2t
WS−X6248−TEL	1q4t	2q2t

WS−X6024−10FL−MT	1q4t	2q2t

Configuration, Monitor, and Example Output Scheduling on the Catalyst 6500/6000

Configuration

This section describes all the steps necessary to configure output scheduling on a Catalyst 6500/6000 that runs Cisco IOS Software. For the Catalyst 6500/6000 default configuration, see the Case 1: QoS Is Enabled and a Default Parameter Is Used section of this document.

Configuration of the Catalyst 6500/6000 implies these five steps:

1. Enable QoS
2. Map each possible class of service (CoS) value to a queue and a threshold (optional)
3. Configure the WRR weight (optional)
4. Configure the buffers that are assigned to each queue (optional)
5. Configure the threshold level for each queue (optional)

Note: Each of these steps is optional, with the exception of Step 1. You can decide to leave the default value for one or more parameters.

Step 1: Enable QoS

First, enable QoS. Remember that QoS is disabled by default. When QoS is disabled, the CoS mapping you have configured does not affect the outcome. There is one queue served in a first in, first out (FIFO) manner, and all packets are dropped there.

```
cosmos#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
cosmos(config)#mls qos

QoS is enabled globally
  Microflow policing is enabled globally

QoS global counters:
    Total packets: 552638
    IP shortcut packets: 0
    Packets dropped by policing: 0
    IP packets with TOS changed by policing: 0
    IP packets with COS changed by policing: 0
    Non−IP packets with CoS changed by policing: 0
```

Step 2: Map Each Possible CoS Value to a Queue and a Threshold

For all queue types, assign the CoS to a queue and a threshold. The mapping defined for a 2q2t type of port is not applied to any 1p2q2t port. Also, the mapping for 2q2t is applied to all ports having a 2q2t queueing mechanism. Issue these **cos−map** commands under the interface:

```
wrr−queue  cos−map Q_number_(1−2) threshold_number_(1−2) cos_value_1
 cos_value_2

priority−queue cos−map Q_number_(always 1) cos_value_1 cos_value_2
```

Note: Each of these commands should be on one line.

You can separately configure the WRR queue. If there is a priority queue, you can configure it with the **priority−queue** command.

Note: The queues are always numbered starting with the lowest−priority queue possible and ending with the strict−priority queue that is available. For example:

- Queue 1 is the low−priority WRR queue.
- Queue 2 is the high−priority WRR queue.
- Queue 3 is the strict−priority queue.

Repeat this operation for all types of queues, or else the default CoS assignment remains. This is an example configuration for 1p2q2t:

```
cosmos#configure terminal
cosmos(config)#interface gigabitethernet 1/1
cosmos(config−if)#priority−queue cos−map 1 5

!−−− Assign a CoS of 5 to priority queue.

  cos−map configured on:  Gi1/1 Gi1/2
cosmos(config−if)#wrr−queue cos−map 1 1 0 1

!−−− Assign CoS 0 and 1 to the first threshold of low−priority WRR queue.

  cos−map configured on:  Gi1/1 Gi1/2
cosmos(config−if)#wrr−queue cos−map 1 2 2 3

!−−− Assign CoS 2 and 3 to the second threshold of low−priority WRR queue.

  cos−map configured on:  Gi1/1 Gi1/2
cosmos(config−if)#wrr−queue cos−map 2 1 4 6

!−−− Assign CoS 4 and 6 to the first threshold of high−priority WRR queue.

  cos−map configured on:  Gi1/1 Gi1/2
cosmos(config−if)#wrr−queue cos−map 2 2 7

!−−− Assign CoS 7 to the first threshold of high−priority WRR queue.

  cos−map configured on:  Gi1/1 Gi1/2
```

Check the configuration:

```
cosmos#show queueing interface gigabitethernet 1/1

!−−− Output suppressed.


queue thresh cos−map
    ---------------------------------------
    1     1      0 1
    1     2      2 3
    2     1      4 6
    2     2      7
    3     1      5

!−−− Output suppressed.
```

Step 3: Configure the WRR Weight

Configure the WRR weight for the two WRR queues. Issue this interface command:

```
wrr-queue bandwidth weight_for_Q1 weight_for_Q2
```

Weight 1 relates to queue 1, which should be the low−priority WRR queue. Always keep this weight one level lower than weight 2. The weight can take any value between 1 and 255. Use these formulas to assign the percentage:

- To queue 1 [weight 1 / (weight 1 + weight 2)]
- To queue 2 [weight 2 / (weight 1 + weight 2)]

You must define the weight for all types of queues. These weight types do not need to be the same. This is an example for 2q2t, where queue 1 is served 20 percent of the time and queue 2 is served 80 percent of the time:

```
cosmos#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
cosmos(config)#interface gigabitethernet 1/1
cosmos(config-if)#wrr-queue bandwidth ?
  <1-255>  enter bandwidth weight between 1 and 255
cosmos(config-if)#wrr-queue bandwidth 20 80

!--- Queue 1 is served 20% of the time, and queue 2 is served
!--- 80% of the time.

cosmos(config-if)#
```

Check the configuration:

```
cosmos#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Port is untrusted
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    -----------------------------------------
       1         WRR low            2
       2         WRR high           2
       3         Priority           1

    WRR bandwidth ratios:   20[queue 1]  80[queue 2]
    queue-limit ratios:     90[queue 1]   5[queue 2]

!--- Output suppressed.
```

Note: You can configure different WRR weights for each interface when it is not possible to use CatOS Software.

Step 4: Configure the Buffers That Are Assigned to Each Queue

You must define the transmit queue ratio. This determines how the buffers are split among the different queues.

```
wrr-queue queue-limit percentage_WRR_Q1 percentage_WRR_Q2

cosmos(config)#interface gigabitethernet 1/2
```

```
cosmos(config-if)#wrr-queue queue-limit 70 15

!--- Queue 1 has 70% of the buffers.
!--- Queues 2 and 3 both have 15% of the buffers.

  queue-limit configured on:  Gi1/1 Gi1/2
```

Note: If the queueing capability of your gigabit port is 1p1q2t, you need to use the same level for the strict−priority queue and for the high−priority WRR queue. These levels cannot differ for hardware reasons. Only the bandwidth for the two WRR queues is configured. You automatically use the same value for the high−priority WRR queue and the strict−priority queue, if there is any.

Some queueing types do not have tunable queue size. An example is the 1p3q1t, which is available on WS−X6548RJ45. These queue types are fixed, and you cannot modify them.

Check the configuration:

```
cosmos#show queueing interface gigabitethernet 1/2
Interface GigabitEthernet1/2 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Port is untrusted
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    -----------------------------------------
       1         WRR low            2
       2         WRR high           2
       3         Priority           1

    WRR bandwidth ratios:    5[queue 1] 255[queue 2]
    queue-limit ratios:     70[queue 1]  15[queue 2]
```

Note: It is best to leave the largest part of the buffers for the low−priority WRR queue. This is the queue where you need to enable additional buffering. The other queues are served with higher priority.

Step 5: Configure the Threshold Level for Each Queue

As a final step, configure the threshold level for the WRED queue or for the tail drop queue. This list provides the commands:

- For queues that use WRED as the drop mechanism for the threshold, issue these commands:

  ```
  wrr-queue random-dtect min-threshold Q_number
   threshold_1_value threshold_2_value

  wrr-queue random-dtect max-threshold Q_number
   threshold_1_value threshold_2_value
  ```

  Note: Each of these commands should be on one line.
- For queues that use tail drop as the drop mechanism, issue this command:

  ```
  wrr-queue threshold Q_number
   threshold_1_value threshold_2_value
  ```

  Note: This command should be on one line.

Configuration for a WRED queue:

```
cosmos(config)#interface gigabitethernet 1/1
cosmos(config-if)#wrr-queue random-detect min-threshold  1 20 50

!--- This sets the threshold of queue 1 to 20 and 50% minimum threshold
!--- configured on Gi1/1 Gi1/2.

cosmos(config-if)#wrr-queue random-detect min-threshold  2 20 50

!--- This sets the threshold of queue 2 to 20 and 50% minimum threshold
!--- configured on Gi1/1 Gi1/2.

cosmos(config-if)#wrr-queue random-detect max-threshold  1 50 80

!--- This sets the threshold of queue 1 to 50 and 80% maximum threshold
!--- configured on Gi1/1 Gi1/2.

cosmos(config-if)#wrr-queue random-detect max-threshold  2 40 60

!--- This sets the threshold of queue 2 to 49 and 60% maximum threshold
!--- configured on Gi1/1 Gi1/2.
```

Configuration for a tail drop queue:

```
cosmos(config)#interface fastethernet 3/1
cosmos(config-if)#wrr-queue threshold ?
  <1-2> enter threshold queue id (1-2)
cosmos(config-if)#wrr-queue threshold 1 ?
  <1-100>  enter percent of queue size between 1 and 100
cosmos(config-if)#wrr-queue threshold 1 50 100

!--- This sets the tail drop threshold for this 2q2t interface for
!--- queue 1 (low-priority) to 50 and 100% of the buffer.

threshold configured on: Fa3/1 Fa3/2 Fa3/3 Fa3/4 Fa3/5 Fa3/6 Fa3/7 Fa3/8
 Fa3/9 Fa3/10 Fa3/11 Fa3/12
cosmos(config-if)#
cosmos(config-if)#
cosmos(config-if)#wrr-queue threshold 2 40 100

!--- This sets the tail drop threshold for this 2q2t interface for
!--- queue 2 (high-priority) to 40 and 100% of the buffer.

threshold configured on: Fa3/1 Fa3/2 Fa3/3 Fa3/4 Fa3/5 Fa3/6 Fa3/7 Fa3/8
 Fa3/9 Fa3/10 Fa3/11 Fa3/12
cosmos(config-if)#
```

Check the configuration:

```
cosmos#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Port is untrusted
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    -----------------------------------------
       1         WRR low             2
       2         WRR high            2
       3         Priority            1

    WRR bandwidth ratios:   20[queue 1]  80[queue 2]
    queue-limit ratios:     70[queue 1]  15[queue 2]
```

```
    queue random-detect-min-thresholds
    ----------------------------
      1    20[1] 50[2]
      2    20[1] 50[2]

    queue random-detect-max-thresholds
    ----------------------------------
      1    50[1] 80[2]
      2    40[1] 60[2]

cosmos#show queueing interface fastethernet 3/1
Interface FastEthernet3/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Port is untrusted
  Default cos is 0
  Transmit queues [type = 2q2t]:
    Queue Id    Scheduling  Num of thresholds
    ---------------------------------------
       1         WRR low             2
       2         WRR high            2

    WRR bandwidth ratios:  100[queue 1] 255[queue 2]
    queue-limit ratios:     90[queue 1]  10[queue 2]

    queue tail-drop-thresholds
    --------------------------
    1     50[1] 100[2]
    2     40[1] 100[2]
```

You cannot configure the threshold and assign the CoS to the queue per port. All changes are applied to a set of contiguous ports:

- Four ports for gigabit line cards ports 1 through 4 are together, and ports 5 through 8 are together.
- Twelve ports for 10/100 ports or 100 fiber ports based on 1q4t/2q2t queueing¡ to 12, 13 to 24, 25 to 36, and 36 to 48.
- In order to determine the exact port that belongs to the same ASIC, use the **show interface capabilities** command.

Monitor Output Scheduling and Verify Configurations

The easiest command to issue in order to verify the current run-time configuration for a port with regard to output scheduling is the **show queueing interface {gigabitethernet | fastethernet}** *slot/port* command. This command displays the type of queueing on the port, the mapping of CoS to the different queues and thresholds, the buffer sharing, and the WRR weight. Here, it is 20 percent WRR for queue 1 and 80 percent WRR for queue 2. The command also displays all the configured information for output scheduling and the number of packets that are dropped in each queue for each threshold:

```
cosmos#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Port is untrusted
  Default COS is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    ---------------------------------------
       1         WRR low             2
       2         WRR high            2
       3         Priority            1
    WRR bandwidth ratios:   20[queue 1]  80[queue 2]
    queue-limit ratios:     70[queue 1]  15[queue 2]

    queue random-detect-max-thresholds
```

```
  ---------------------------------
    1    50[1] 80[2]
    2    40[1] 60[2]

  queue thresh cos-map
  ---------------------------------------
  1     1      0 1
  1     2      2 3
  2     1      4 6
  2     2      7
  3     1      5

Receive queues [type = 1p1q4t]:
  Queue Id    Scheduling  Num of thresholds
  -----------------------------------------
     1         Standard            4
     2         Priority            1


  queue tail-drop-thresholds
  --------------------------
  1     100[1] 100[2] 100[3] 100[4]

  queue thresh cos-map
  ---------------------------------------
  1     1      0 1
  1     2      2 3
  1     3      4
  1     4      6 7
  2     1      5


Packets dropped on Transmit:
  BPDU packets:  0

  queue thresh    dropped  [cos-map]
  ---------------------------------------------------
  1     1              0  [0 1 ]
  1     2              0  [2 3 ]
  2     1              0  [4 6 ]
  2     2              0  [7 ]
  3     1              0  [5 ]

Packets dropped on Receive:
  BPDU packets:  0

  queue thresh    dropped  [cos-map]
  ---------------------------------------------------
  1     1              0  [0 1 ]
  1     2              0  [2 3 ]
  1     3              0  [4 ]
  1     4              0  [6 7 ]
  2     1              0  [5 ]
```

Example of Output Scheduling

This traffic is injected on the Catalyst 6500/6000:

- In port gigabit 1/2: one gigabit of traffic with precedence of zero
- In port gigabit 5/2:
    - ♦ 133MB of traffic with precedence of seven
    - ♦ 133MB of traffic with precedence of six

- ♦ 133MB of traffic with precedence of five
- ♦ 133MB of traffic with precedence of four
- ♦ 133MB of traffic with precedence of three
- ♦ 133MB of traffic with precedence of two
- ♦ 133MB of traffic with precedence of one

All the unicast traffic exits the switch per port gigabit 1/1, which is very oversubscribed.

Case 1: QoS Is Enabled and a Default Parameter Is Used

The **show queueing interface gigabitethernet 1/1** command configures all output in this example. The command provides additional information about input scheduling. However, as this document only covers output scheduling, it suppresses that output.

When QoS is globally enabled and all default parameters are in use, this output results after a few minutes:

```
nelix#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Trust state: trust DSCP
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    -----------------------------------------
       1         WRR low            2
       2         WRR high           2
       3         Priority           1

    WRR bandwidth ratios:  100[queue 1] 255[queue 2]
    queue-limit ratios:     90[queue 1]   5[queue 2]

    queue random-detect-max-thresholds
    ----------------------------------
      1    40[1] 100[2]
      2    40[1] 100[2]

    queue thresh cos-map
    ---------------------------------------
    1     1      0 1
    1     2      2 3
    2     1      4
    2     2      6 7
    3     1      5

  Packets dropped on Transmit:
    BPDU packets:  0

    queue thresh    dropped  [cos-map]
    ---------------------------------------------------
    1     1       149606424  [0 1 ]
    1     2               0  [2 3 ]
    2     1        16551394  [4 ]
    2     2         4254446  [6 7 ]
    3     1               0  [5 ]
```

In this output, the default values are:

- WRR weight for queue 1¡00 / (100 + 255) = 28%
- WRR weight for queue 2¢55 / (255 + 100) = 72%
- Buffer sharing: 90% for queue 1, 5% for queue 2, and 5% for strict-priority queue

Most of the packets in the low−priority WRR queue are dropped, but some are still dropped in the high−priority WRR queue for both thresholds. There is a total of 170,412,264 drops (149,606,424 + 16,551,394 + 4,254,446). These drops are split as follows:

- 149,606,424 / 170,412,264 = 88% of drops in queue 1 (first threshold packet with CoS 0 and 1)
- 16,551,394 / 170,412,264 = 10% of drops in queue 2 (first threshold packet with CoS 4)
- 4,254,446 / 170,412,264 = 2% of drops in queue 2 (second threshold packet with CoS of 6 or 7)

Note: You do not see any drops in the strict−priority queue.

Case 2: Modify WRR Weight

As noted in the Case 1: QoS Is Enabled and a Default Parameter Is Used section, packets in queue 2 are still being dropped. Modify the WRR weight to give more bandwidth to queue 2. Now, queue 1 is emptied 4 percent of the time, and queue 2 is emptied 96 percent of the time:

```
show run interface gigabitethernet 1/1
interface GigabitEthernet1/1
 no ip address
 wrr−queue bandwidth 10 255
 mls qos trust dscp
 switchport
 switchport mode access
end

nelix#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round−Robin
  Port QoS is enabled
  Trust state: trust DSCP
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
       1         WRR low             2
       2         WRR high            2
       3         Priority            1

    WRR bandwidth ratios:   10[queue 1] 255[queue 2]
    queue−limit ratios:     90[queue 1]   5[queue 2]

    queue random−detect−max−thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
      1    40[1] 100[2]
      2    40[1] 100[2]

    queue thresh cos−map
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
    1     1      0 1
    1     2      2 3
    2     1      4
    2     2      6 7
    3     1      5

  Packets dropped on Transmit:
    BPDU packets:  0

    queue thresh    dropped  [cos−map]
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
    1     1         2786205  [0 1 ]
    1     2               0  [2 3 ]
    2     1           11363  [4 ]
    2     2              69  [6 7 ]
    3     1               0  [5 ]
```

As seen in this output, the percentage of drops in queue 2 is now much lower. A total of 2,797,637 drops are split in this way:

- 2,786,205 / 2,797,637 = 99.591% of drops in queue 1 (with packet of CoS 0 and 1)
- 11,363 / 2,797,637 = 0.408% of drops in queue 2 (first threshold with packet CoS 4)
- 69 / 2,797,637 = 0.001% of drops in queue 2 (second threshold for packet with CoS 6 and 7)

If you use various WRR weights, it ensures more QoS in queue 2.

Case 3: Additional WRR Weight Modification

You can be even more aggressive with the WRR weight. In this sample output, only 0.39 percent of the weight is given to queue 1:

```
show run interface gigabitethernet 1/1
interface GigabitEthernet1/1
 no ip address
 wrr-queue bandwidth 1 255
 mls qos trust dscp
 switchport
 switchport mode access
end

nelix#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round-Robin
  Port QoS is enabled
  Trust state: trust DSCP
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    -----------------------------------------
       1         WRR low            2
       2         WRR high           2
       3         Priority           1

    WRR bandwidth ratios:    1[queue 1] 255[queue 2]
    queue-limit ratios:     90[queue 1]   5[queue 2]

    queue random-detect-max-thresholds
    ----------------------------------
      1    40[1] 100[2]
      2    40[1] 100[2]

    queue thresh cos-map
    ---------------------------------------
    1     1      0 1
    1     2      2 3
    2     1      4
    2     2      6 7
    3     1      5

  Packets dropped on Transmit:
    BPDU packets:  0

    queue thresh    dropped  [cos-map]
    ---------------------------------------------------
    1     1         2535315  [0 1 ]
    1     2               0  [2 3 ]
    2     1             705  [4 ]
    2     2              73  [6 7 ]
    3     1               0  [5 ]
```

Even with the aggressive WRR weight, packets are still being dropped in queue 2. However, in comparison, it is not many packets. There is now only a 0.03 percent packet drop in queue 2.

Case 4: Modify Queue Limit Buffer Assignment

As seen in the Case 2: Modify WRR Weight and Case 3: Additional WRR Weight Modification sections, packets are still dropping in queue 2, although the WRR percentage assures you that the drop is minimal. However, when the second threshold (which is set to 100 percent) is reached in queue 2, some packets continue to be dropped.

In order to improve this, change the queue limit (size of the buffer assigned to each queue). In this example, the queue limit is set to 70 percent for queue 1, 15 percent for queue 2, and 15 percent for strict−priority queue:

```
show run gigabitethernet 1/1
interface GigabitEthernet1/1
 no ip address
 wrr−queue bandwidth 1 255
 wrr−queue queue−limit 70 15
 mls qos trust dscp
 switchport
 switchport mode access
end

nelix#show queueing interface gigabitethernet 1/1
Interface GigabitEthernet1/1 queueing strategy:  Weighted Round−Robin
  Port QoS is enabled
  Trust state: trust DSCP
  Default cos is 0
  Transmit queues [type = 1p2q2t]:
    Queue Id    Scheduling  Num of thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
       1         WRR low            2
       2         WRR high           2
       3         Priority           1

    WRR bandwidth ratios:    1[queue 1] 255[queue 2]
    queue−limit ratios:     70[queue 1]  15[queue 2]

    queue random−detect−max−thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
      1    40[1] 100[2]
      2    40[1] 100[2]

    queue thresh cos−map
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
    1     1      0 1
    1     2      2 3
    2     1      4
    2     2      6 7
    3     1      5

  Receive queues [type = 1p1q4t]:
    Queue Id    Scheduling  Num of thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
       1         Standard           4
       2         Priority           1

    queue tail−drop−thresholds
    −−−−−−−−−−−−−−−−−−−−−−−−−
    1     100[1] 100[2] 100[3] 100[4]

    queue thresh cos−map
```

```
  ---------------------------------------
  1     1      0 1
  1     2      2 3
  1     3      4
  1     4      6 7
  2     1      5

Packets dropped on Transmit:
  BPDU packets:  0

  queue thresh    dropped  [cos-map]
  ---------------------------------------------------
  1     1       154253046  [0 1 ]
  1     2               0  [2 3 ]
  2     1               0  [4 ]
  2     2               0  [6 7 ]
  3     1               0  [5 ]
```

Now, the drops only occur in queue 1.

Use Output Scheduling to Reduce Delay and Jitter

The cases studies in the Example of Output Scheduling section demonstrate the benefit of implementing output scheduling to avoid a drop of VoIP or mission−critical traffic in the event of oversubscription of the output port. Oversubscription does not occur very frequently in a normal network (particularly on a gigabit link). Oversubscription should only happen during peak times of traffic or during traffic bursts that occur within a very short period of time.

Even without any oversubscription, output scheduling can be of great help in a network where QoS is implemented end to end. This section provides examples of how output scheduling can help reduce delay and jitter.

Reduce Delay

The delay of a packet increases because of the time "lost" in the buffer of each switch while it waits to be transmitted. For example, a small voice packet with a CoS of 5 is sent out of a port during a large backup or file transfer. Assume there is no QoS for the output port, and the small voice packet is queued after 10 large 1500−byte packets. In this case, you can easily calculate that the gigabit speed time that is needed to transmit the 10 large packets is:

- (10 x 1500 x 8) = 120,000 bits transmitted in 120 microseconds

If this packet needs to cross eight or nine switches while passing through the network, a delay of about 1 millisecond can result. This includes only delays in the output queue of the switch crossed in the network.

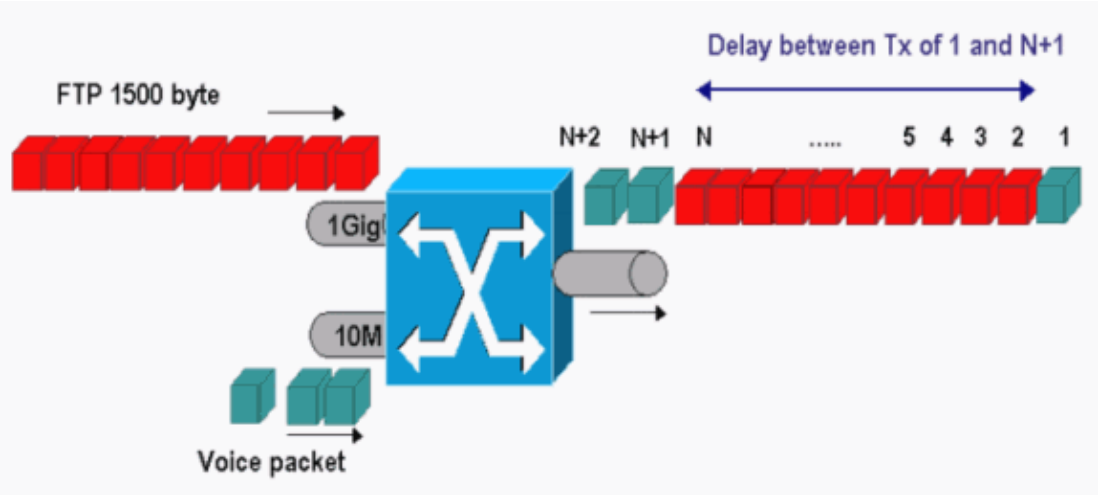
Note: If you need to queue the same 10 large packets on a 10MB interface (for example, connected to an IP phone and a PC), the delay introduce is:

- (10 x 1500 x 8) = 120,000 bits transmitted in 12 milliseconds

The implementation of output scheduling ensures voice packets with a CoS of 5 are put in the strict−priority queue and are sent before any packets with a CoS of less than 5. This reduces delay.

Reduce Jitter

Another important benefit of output scheduling is the reduction of jitter. Jitter is the variation in delay for packets within the same flow. This example scenario shows how output scheduling can reduce jitter:



In this scenario, the same output port needs to send two streams:

- One incoming voice stream on a 10MB Ethernet port.
- One incoming FTP stream on a 1 Gigabit Ethernet uplink port.

Both streams leave the switch through the same output port. This example shows what can occur without the use of output scheduling. All large data packets can be interleaved between two voice packets. This creates jitter in the reception of the voice packet from the same stream. There is a larger delay between the reception of packet 1 and packet *n* + 1 as the switch transmits the large data packet. However, the delay between *n* + 1 and *n* + 2 is negligible. This results in jitter in the voice traffic stream. You can easily avoid this problem with the use of a strict−priority queue. Ensure that you map the CoS value of the voice packets to the strict−priority queue.

Conclusion

In this document, you have seen case studies of how to configure and troubleshoot output queue scheduling on a Catalyst 6500/6000 that runs Cisco IOS Software. You have also seen the advantages of output scheduling in most networks with voice traffic:

- Avoids the drop of critical traffic in the case of oversubscription of the output port.
- Reduces the delay.
- Reduces the jitter.

Related Information

- **QoS Output Scheduling on Catalyst 6500/6000 Series Switches Running CatOS System Software**
- **Understanding Quality of Service on Catalyst 6000 Family Switches**
- **LAN Product Support Pages**
- **LAN Switching Support Page**
- **Technical Support & Documentation − Cisco Systems**

Updated: Sep 02, 2005 Document ID: 10587