Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

Congestion avoidance techniques monitor traffic flow in an effort to anticipate and avoid congestion at common network bottlenecks. Avoidance techniques are implemented before congestion occurs as compared with congestion management techniques that control congestion after it has occurred.

Congestion avoidance is achieved through packet dropping. Cisco IOS XR software supports the following quality of service (QoS) congestion avoidance techniques that drop packets:

- Random early detection (RED)
- Weighted random early detection (WRED)
- Tail drop

The module describes the concepts and tasks related to these congestion avoidance techniques.

Feature History for Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 3.2</td>
<td>The Congestion Avoidance feature was introduced.</td>
</tr>
<tr>
<td>Release 3.6.0</td>
<td>Weighted random early detection (WRED) support for IPv4 multicast egress QoS traffic was introduced on 2.5 Gbps IP Services Engine (Engine 3) linecards.</td>
</tr>
<tr>
<td>Release 4.0.0</td>
<td>In calculations for the average queue size, indicated that the exponential weight factor is not configurable (CSCeg75763).</td>
</tr>
</tbody>
</table>

Contents

- Prerequisites for Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software, page 102
- How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR Software, page 105
- Configuration Examples for Configuring Policy Maps, page 116
- Additional References, page 117
Prerequisites for Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

The following prerequisite is required for configuring QoS congestion avoidance on your network:

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

To configure QoS congestion avoidance techniques in this document you must understand the following concepts:

- Random Early Detection and TCP, page 102
- Weighted Random Early Detection for Preferential Traffic Handling, page 102
- WRED Support for IPv4 Multicast Egress QoS Traffic, page 104
- Tail Drop and the FIFO Queue, page 105

Random Early Detection and TCP

The RED congestion avoidance technique takes advantage of the congestion control mechanism of TCP. By randomly dropping packets prior to periods of high congestion, RED tells the packet source to decrease its transmission rate. Assuming the packet source is using TCP, it decreases its transmission rate until all packets reach their destination, indicating that the congestion is cleared. You can use RED as a way to cause TCP to slow transmission of packets. TCP not only pauses, but it also restarts quickly and adapts its transmission rate to the rate that the network can support.

RED distributes losses in time and maintains normally low queue depth while absorbing traffic bursts. When enabled on an interface, RED begins dropping packets when congestion occurs at a rate you select during configuration.

Weighted Random Early Detection for Preferential Traffic Handling

WRED provides the ability to define multiple RED profiles within a single class, based on certain match criteria (DSCP, discard class and so on), so that different drop precedences can be configured based on the relative importance of packets. WRED can selectively discard lower priority traffic when the interface begins to get congested and provide differentiated performance characteristics for different classes of service. You can configure WRED to ignore IP precedence when making drop decisions so that nonweighted RED behavior is achieved.

WRED makes early detection of congestion possible and provides for multiple classes of traffic. It also protects against global synchronization. For these reasons, WRED is useful on any output interface in which you expect congestion to occur.
However, WRED is usually used in the core routers of a network, rather than at the edge of the network. Edge routers assign IP precedences to packets as they enter the network. WRED uses these precedences to determine how to treat different types of traffic.

WRED provides separate drop thresholds (minimum and maximum) for different classification criteria (such as IP precedences, MPLS EXP values), allowing you to provide different qualities of service in regard to packet dropping for different traffic types. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

WRED treats non-IP traffic as precedence 0, the lowest precedence. Therefore, non-IP traffic, in general, is more likely to be dropped than IP traffic.

WRED is useful only when the bulk of the traffic is TCP/IP traffic. With TCP, dropped packets indicate congestion, so the packet source reduces its transmission rate. With other protocols, packet sources may not respond or may resend dropped packets at the same rate. Thus, dropping packets does not decrease congestion.

Figure 1 illustrates how WRED works.

**Figure 1  Weighted Random Early Detection**

![Weighted Random Early Detection Diagram](image)

**Average Queue Size for WRED**

The router automatically determines the parameters to use in the WRED calculations. The average queue size is based on the previous average and current size of the queue. The formula is:

\[
\text{average} = (\text{old\_average} \times (1-2^{-x})) + (\text{current\_queue\_size} \times 2^{-x})
\]

where \( x \) is the exponential weight factor.

For high values of \( x \), the previous average becomes more important. A large factor smooths out the peaks and lows in queue length. The average queue size is unlikely to change very quickly, avoiding a drastic change in size. The WRED process is slow to start dropping packets, but it may continue dropping packets for a time after the actual queue size has fallen below the minimum threshold. The slow-moving average accommodates temporary bursts in traffic.
Note

The exponential weight factor, x, is fixed and is not user configurable.

Note

If the value of x gets too high, WRED does not react to congestion. Packets are sent or dropped as if WRED were not in effect.

For low values of x, the average queue size closely tracks the current queue size. The resulting average may fluctuate with changes in the traffic levels. In this case, the WRED process responds quickly to long queues. Once the queue falls below the minimum threshold, the process stops dropping packets.

If the value of x gets too low, WRED overreacts to temporary traffic bursts and drops traffic unnecessarily.

WRED Support for IPv4 Multicast Egress QoS Traffic

Weighted random early detection (WRED) is supported for IPv4 multicast egress QoS traffic. The following features are supported:

- Classification based on precedence and differentated services code point (DSCP)
- Queue selection
- Maximum and minimum guaranteed bandwidth
- Detection of remaining bandwidth
- Priority queuing
- Queue limit
- Random early detection (RED)
- Traffic shaping
- Layer 2 set on ATM and Ethernet

Policy map configurations that use Random Early Detection (RED) must comply with the following requirements. These requirements must be met across all interfaces in the line card for precedence and DSCP-based WRED to be supported for multicast traffic. The policy map configuration requirements are specific to a line card and not across the line cards:

- Multicast must be enabled.
- The same class maps must be used in all the policy maps.
- The class maps must be configured in the same order in all of the policy maps.
- The same RED statements must be configured in a class map across policy maps. The minimum and maximum threshold values can be different for each policy map.
- If a RED profile is used by more than one DSCP or precedence in a class, the precedence or DSCP values share the same RED profile in all the different policy maps.

If the policy map requirements are not met for all interfaces in the line card and are not met for at least one policy map, RED configuration is not applied to the multicast traffic and all multicast traffic in the line card is tail-dropped. See the “Tail Drop and the FIFO Queue” section on page 105.

Interfaces with no policy maps and interfaces with policy maps that do not use RED function normally.
For hierarchal policy maps that use WRED, only the child policies should conform to the policy map configuration requirements.

**Restrictions**

The following features are not supported by WRED for IPv4 multicast egress QoS traffic:

- Discard-class-based RED. WRED is supported fully on ATM line cards and is supported with restrictions on POS and Ethernet line cards.
- Classification other than DSCP or precedence.
- Layer 3 set.
- QoS is not active when packets go to the slow path. All packets go to the default queue.
- IPv6 multicast.
- Policing traffic. The `hw-module qos multicast priorityq disable` command is used to prevent traffic from reaching the priority queue.

**Supported Platforms**

WRED is supported for IPv4 multicast egress QoS traffic on the following Cisco XR 12000 Series Router platforms:

- POS line cards
  - Cisco XR 12000 Series 4xOC12c/STM4c POS Rev B
  - Cisco XR 12000 Series 16xOC3c/STM1c POS Rev B
  - Cisco XR 12000 Series 8xOC3c/STM1c POS
  - Cisco XR 12000 Series 4xOC3c/STM1c POS
  - Cisco XR 12000 Series 1xOC48c/STM16c POS
- Ethernet line card
  - Cisco XR 12000 Series 4xGE

**Tail Drop and the FIFO Queue**

Tail drop is a congestion avoidance technique that drops packets when an output queue is full until congestion is eliminated. Tail drop treats all traffic flow equally and does not differentiate between classes of service. It manages the packets that are unclassified, placed into a first-in, first-out (FIFO) queue, and forwarded at a rate determined by the available underlying link bandwidth.

See the “Default Traffic Class” section of the “Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software” module.

**How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR Software**

This section contains instructions for the following tasks:
Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR Software

QC-106

Cisco IOS XR Modular Quality of Service Configuration Guide for the Cisco XR 12000 Series Router

OL-24694-01

Configuring Random Early Detection, page 106 (required)
Configuring Weighted Random Early Detection, page 108 (required)
Configuring Tail Drop, page 111 (required)
Configuring Multicast Egress QoS, page 114 (optional)

Configuring Random Early Detection

You can configure Random Early Detection (RED) by not specifying IP precedence or any other match criteria. In this way, a single RED profile is applied to all packets matching the class.

This configuration task is similar to that used for WRED except that the random-detect precedence command is not configured and the random-detect command with the default keyword must be used to enable RED.

Restrictions

For the random-detect command to take effect, you must configure either the shape average, bandwidth/bandwidth remaining percent command in the user defined policy map class. This dependency is not applicable to the policy map class class-default.

SUMMARY STEPS

1. configure
2. policy-map policy-name
3. class class-name
4. random-detect {discard-class value | dscp value | exp value | precedence value | min-threshold [units] max-threshold [units] } 
5. bandwidth {bandwidth [units] | percent value }
6. bandwidth remaining percent value
7. shape average {percent percentage | value [units]}
8. exit
9. exit
10. interface type interface-path-id
11. service-policy {input | output} policy-map
12. end
   or
   commit
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td><code>policy-map policy-name</code></td>
<td>Enters policy map configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config)# policy-map policy1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td><code>class class-name</code></td>
<td>Enters policy map class configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap)# class class1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>`random-detect {discard-class value</td>
<td>dscp value</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap-c)# random-detect 1000000 2000000</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>`bandwidth {bandwidth [units]</td>
<td>percent value}`</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap-c)# bandwidth percent 30</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td><code>bandwidth remaining percent value</code></td>
<td>(Optional) Specifies how to allocate leftover bandwidth to various classes.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>`shape average {percent percentage</td>
<td>value [units]}`</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap-c)# shape average percent 50</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td><code>exit</code></td>
<td>Returns the router to policy map configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap-c)# exit</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td><code>exit</code></td>
<td>Returns the router to global configuration mode.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap)# exit</td>
<td></td>
</tr>
</tbody>
</table>
Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR Software

Configuring Weighted Random Early Detection

WRED drops packets selectively based on IP precedence. Edge routers assign IP precedences to packets as they enter the network. WRED uses these precedences to determine how to treat different types of traffic.

When a packet arrives, the following actions occur:

- The average queue size is calculated.
- If the average queue size is less than the minimum queue threshold, the arriving packet is queued.
- If the average queue size is between the minimum queue threshold for that type of traffic and the maximum threshold for the interface, the packet is either dropped or queued, depending on the packet drop probability for that type of traffic.
- If the average queue size is greater than the maximum threshold, the packet is dropped.

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 10</td>
<td><code>interface type interface-path-id</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/0/CPU0:router(config)# interface pos 0/2/0/0</code></td>
</tr>
<tr>
<td>Step 11</td>
<td>`service-policy (input</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/0/CPU0:router(config-if)# service-policy output policy1</code></td>
</tr>
<tr>
<td>Step 12</td>
<td><code>end</code> or <code>commit</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td><code>RP/0/0/CPU0:router(config-cmap)# end</code> or <code>RP/0/0/CPU0:router(config-cmap)# commit</code></td>
</tr>
</tbody>
</table>

When you issue the `end` command, the system prompts you to commit changes:

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:
```

- Entering `yes` saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
- Entering `no` exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
- Entering `cancel` leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the `commit` command to save the configuration changes to the running configuration file and remain within the configuration session.
Restrictions

You cannot configure WRED in a class that has been set for priority queueing (PQ).
You cannot use the `random-detect` command in a class configured with the `priority` command.
For the `random-detect` command to take effect, you must configure either the `shape average` or `bandwidth/bandwidth remaining percent` command in the user defined policy map class. This dependency is not applicable to the policy map class `class-default`.

**SUMMARY STEPS**

1. `configure`
2. `policy-map policy-name`
3. `class class-name`
4. `random-detect dscp dscp-value min-threshold [units] max-threshold [units]`
5. `bandwidth {bandwidth [units] | percent value}`
6. `bandwidth remaining percent value`
7. `shape average {percent | value [units]}`
8. `exit`
9. `interface type interface-path-id`
10. `service-policy {input | output} policy-map`
11. `end`
    or `commit`

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>policy-map policy-name</code></td>
<td>Enters policy map configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config)# policy-map policy1</td>
<td>* Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.</td>
</tr>
<tr>
<td><strong>Step 3</strong> <code>class class-name</code></td>
<td>Enters policy map class configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config-pmap)# class class1</td>
<td>* Specifies the name of the class whose policy you want to create or change.</td>
</tr>
</tbody>
</table>
### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 4</td>
<td><code>random-detect dscp dscp-value min-threshold [units] max-threshold [units]</code></td>
<td>Changes the minimum and maximum packet thresholds for the DSCP value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Enables RED.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>dscp-value</code>—Number from 0 to 63 that sets the DSCP value. Reserved keywords can be specified instead of numeric values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>min-threshold</code>—Minimum threshold in the specified units. The value range of this argument is from 512 to 1073741823. When the average queue length reaches the minimum threshold, WRED randomly drops some packets with the specified DSCP value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>max-threshold</code>—Maximum threshold in the specified units. The value range of this argument is from the value of the <code>min-threshold</code> argument to 1073741823. When the average queue length exceeds the maximum threshold, WRED drops all packets with the specified DSCP value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <code>units</code>—Units of the threshold value. This can be bytes, gbytes, kbytes, mbytes, ms (milliseconds), packets, or us (microseconds). The default is packets.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• This example shows that for packets with DSCP AF11, the WRED minimum threshold is 1,000,000 bytes and maximum threshold is 2,000,000 bytes.</td>
</tr>
<tr>
<td>Step 5</td>
<td>`bandwidth bandwidth [units]</td>
<td>percent value)`</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• This example guarantees 30 percent of the interface bandwidth to class class1.</td>
</tr>
<tr>
<td>Step 6</td>
<td><code>bandwidth remaining percent value</code></td>
<td>(Optional) Specifies how to allocate leftover bandwidth to various classes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The remaining bandwidth of 70 percent is shared by all configured classes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In this example, class class1 receives 20 percent of the 70 percent.</td>
</tr>
<tr>
<td>Step 7</td>
<td>`shape average percent percentage</td>
<td>value [units])`</td>
</tr>
<tr>
<td>Step 8</td>
<td><code>exit</code></td>
<td>Returns the router to global configuration mode.</td>
</tr>
</tbody>
</table>

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# random-detect dscp af11 1000000 bytes 2000000 bytes

Example:

RP/0/0/CPU0:router(config-pmap-c)# bandwidth percent 30

Example:

RP/0/0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20

Example:

RP/0/0/CPU0:router(config-pmap-c)# shape average percent 50

Example:

RP/0/0/CPU0:router(config-pmap-c)# exit
Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR Software

Configuring Tail Drop

Packets satisfying the match criteria for a class accumulate in the queue reserved for the class until they are serviced. The queue-limit command is used to define the maximum threshold for a class. When the maximum threshold is reached, enqueued packets to the class queue result in tail drop (packet drop).

The queue-limit value uses the guaranteed service rate (GSR) of the queue as the reference value for the queue_bandwidth. If the class has bandwidth percent associated with it, the queue-limit is set to a proportion of the bandwidth reserved for that class.

When a class has no guaranteed service rate, the default queue limit depends on whether shaping is applied. If shaping is not applied, the default queue limit is 16384 packets. If shaping is applied, the default queue limit is:

default queue limit (in packets) = (200 ms * (queue bandwidth or shaper rate) / 8) / average packet size, which is 250 bytes

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 9</strong> interface type interface-path-id</td>
<td>Enters configuration mode and configures an interface.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/0/CPU0:router(config)# interface pos 0/2/0/0</td>
</tr>
<tr>
<td><strong>Step 10</strong> service-policy (input</td>
<td>output) policy-map</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/0/CPU0:router(config-if)# service-policy output policy1</td>
</tr>
<tr>
<td><strong>Step 11</strong> end</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td>or commit</td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>RP/0/0/CPU0:router(config-cmap)# end or RP/0/0/CPU0:router(config-cmap)# commit</td>
</tr>
</tbody>
</table>

Uncommitted changes found, commit them before exiting(yes/no/cancel)?
[cancel]:

- Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
- Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
- Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.
Restrictions

- When configuring the `queue-limit` command in a class, you must configure one of the following commands: `priority`, `shape average`, `bandwidth`, or `bandwidth remaining`, except for the default class.

SUMMARY STEPS

1. `configure`
2. `policy-map policy-name`
3. `class class-name`
4. `queue-limit value [units]`
5. `priority [level priority-level]`
6. `class class-name`
7. `bandwidth remaining percent value`
8. `exit`
9. `exit`
10. `interface type interface-path-id`
11. `service-policy {input | output} policy-map`
12. `end` or `commit`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode.</td>
</tr>
</tbody>
</table>
| **Example:**  
RP/0/0/CPU0:router# configure | |
| **Step 2** `policy-map policy-name` | Enters policy map configuration mode. |
| **Example:**  
RP/0/0/CPU0:router(config)# policy-map policy1 | • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy. |
| **Step 3** `class class-name` | Enters policy map class configuration mode. |
| **Example:**  
RP/0/0/CPU0:router(config-pmap)# class class1 | • Specifies the name of the class whose policy you want to create or change. |
| **Step 4** `queue-limit value [units]` | Specifies or modifies the maximum the queue can hold for a class policy configured in a policy map. The default value of the `units` argument is `packets`. |
| **Example:**  
RP/0/0/CPU0:router(config-pmap-c)# queue-limit 1000000 bytes | • In this example, when the queue limit reaches 1,000,000 bytes, enqueued packets to the class queue are dropped. |
### Command or Action

**Step 5**

**priority** [level *priority-level*]

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# priority level 1

Specifies priority to a class of traffic belonging to a policy map.

**Step 6**

**class** *class-name*

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# class class2

Specifies the name of the class whose policy you want to create or change.

- In this example, class2 is configured.

**Step 7**

**bandwidth remaining percent** *value*

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20

(Optional) Specifies how to allocate leftover bandwidth to various classes.

- This example allocates 20 percent of the leftover interface bandwidth to class class2.

**Step 8**

**exit**

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# exit

Returns the router to policy map configuration mode.

**Step 9**

**exit**

**Example:**

RP/0/0/CPU0:router(config-pmap-c)# exit

Returns the router to global configuration mode.

**Step 10**

**interface** *type* *interface-path-id*

**Example:**

RP/0/0/CPU0:router(config)# interface pos 0/2/0/0

Enters configuration mode, and configures an interface.

**Step 11**

**service-policy** {input | output} *policy-map*

**Example:**

RP/0/0/CPU0:router(config-if)# service-policy output policy1

Attaches a policy map to an input or output interface to be used as the service policy for that interface.

- In this example, the traffic policy evaluates all traffic leaving that interface.
Configuring Multicast Egress QoS

This task configures multicast egress QoS and disables multicast egress traffic on the priority queue.

Prerequisites

Multicast egress QoS can be configured only on an interface that is already configured for multicast routing. See Cisco IOS XR Multicast Configuration Guide for Cisco XR 12000 Series Router for information on configuring multicast routing.

Restrictions

The “Restrictions” section on page 105 lists the restrictions for configuring multicast egress QoS.

SUMMARY STEPS

1. configure
2. hw-module qos multicast location node-id
3. hw-module qos multicast priorityq disable location node-id
4. end
   or
   commit

Example:
RP/0/0/CPU0:router(config-cmap)# end
or
RP/0/0/CPU0:router(config-cmap)# commit

Step 12

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td></td>
<td>• When you issue the end command, the system prompts you to commit changes:</td>
</tr>
<tr>
<td></td>
<td>Uncommitted changes found, commit them before exiting(yes/no/cancel) ?</td>
</tr>
<tr>
<td></td>
<td>[cancel]:</td>
</tr>
<tr>
<td></td>
<td>– Entering yes saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.</td>
</tr>
<tr>
<td></td>
<td>– Entering no exits the configuration session and returns the router to EXEC mode without committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>– Entering cancel leaves the router in the current configuration session without exiting or committing the configuration changes.</td>
</tr>
<tr>
<td></td>
<td>• Use the commit command to save the configuration changes to the running configuration file and remain within the configuration session.</td>
</tr>
</tbody>
</table>
## Detailed Steps

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router# configure</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> hw-module qos multicast location node-id</td>
<td>Enables multicast egress QoS for a specified location. This command is needed only for multicast QoS on 2.5 Gbps IP Services Engine (Engine 3) linecards.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config)# hw-module qos multicast location 0/2/0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> hw-module qos multicast priorityq disable location node-id</td>
<td>Diverts multicast traffic slated for the priority queue to the default queue QoS for a specified location.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config)# hw-module qos multicast priorityq disable location 0/2/0</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> end or commit</td>
<td>Saves configuration changes.</td>
</tr>
<tr>
<td><strong>Example:</strong> RP/0/0/CPU0:router(config)# end or RP/0/0/CPU0:router(config)# commit</td>
<td></td>
</tr>
</tbody>
</table>

- When you issue the `end` command, the system prompts you to commit changes:
  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:
  - Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
  - Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
  - Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.

- Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.
This section provides the following configuration examples:

- **Enabling WRED for IPv4 Egress Multicast Traffic, page 116**

### Enabling WRED for IPv4 Egress Multicast Traffic

The following example shows a policy map configuration that meets the requirements to enable WRED for IPv4 egress multicast traffic:

```plaintext
policy-map policy_A
  class prec0123
    bandwidth percent 20
    random-detect precedence routine 1000 packets 2000 packets
    random-detect precedence priority 1000 packets 4000 packets
    random-detect precedence immediate 2000 packets 5000 packets
  class prec45
    bandwidth percent 40
    random-detect precedence flash-override 4000 packets 8000 packets

policy-map policy_B
  class prec0123
    bandwidth percent 10
    random-detect precedence routine 1000 packets 4000 packets
    random-detect precedence priority 2000 packets 5000 packets
    random-detect precedence immediate 3000 packets 6000 packets
  class prec45
    bandwidth percent 30
    random-detect precedence flash-override 4000 packets 8000 packets

policy-map policy_C
  class prec012
      bandwidth percent 20
  class prec34
      bandwidth percent 40

interface pos0/1/0/0
  service-policy output policy_A
interface pos 0/1/0/1
  service-policy output policy_B
interface pos 0/1/0/3
  service-policy output policy_C
```

The interfaces mapped to policy_A and policy_B are configured with the same classes and RED statements. The precedences for policy_A and policy_B are mapped to the same RED profile although the minimum and maximum RED thresholds are different. The interface mapped to policy_C is configured differently than the interfaces mapped to policy_A and policy_B.
Additional References

The following sections provide references related to implementing QoS congestion avoidance.

Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial system bootup and configuration</td>
<td>Cisco IOS XR Getting Started Guide for the Cisco XR 12000 Series Router</td>
</tr>
<tr>
<td>Master command reference</td>
<td>Cisco XR 12000 Series Router Master Command Listing</td>
</tr>
<tr>
<td>QoS commands</td>
<td>Cisco IOS XR Modular Quality of Service Command Reference for the Cisco XR 12000 Series Router</td>
</tr>
<tr>
<td>User groups and task IDs</td>
<td>“Configuring AAA Services on Cisco IOS XR Software” module of Cisco IOS XR System Security Configuration Guide</td>
</tr>
</tbody>
</table>

Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.</td>
<td>—</td>
</tr>
</tbody>
</table>

MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: <a href="http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml">http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml</a></td>
</tr>
</tbody>
</table>

RFCs

<table>
<thead>
<tr>
<th>RFCs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.</td>
<td>—</td>
</tr>
</tbody>
</table>
### Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.</td>
<td><a href="http://www.cisco.com/techsupport">http://www.cisco.com/techsupport</a></td>
</tr>
</tbody>
</table>