Congestion Management

Congestion Management Overview

Congestion management features allow you to control congestion by determining the order in which a traffic flow (or packets) is sent out an interface based on priorities assigned to packets. Congestion management entails the creation of queues, assignment of packets to those queues based on the classification of the packet, and scheduling of the packets in a queue for transmission.

The types of traffic regulation mechanisms supported are:

- Low-Latency Queueing with Strict Priority Queueing, on page 18
- Traffic Shaping, on page 20
- Traffic Policing, on page 22

Important Notes About Congestion Management

The following points are applicable to the line card 88-LC1-36EH:

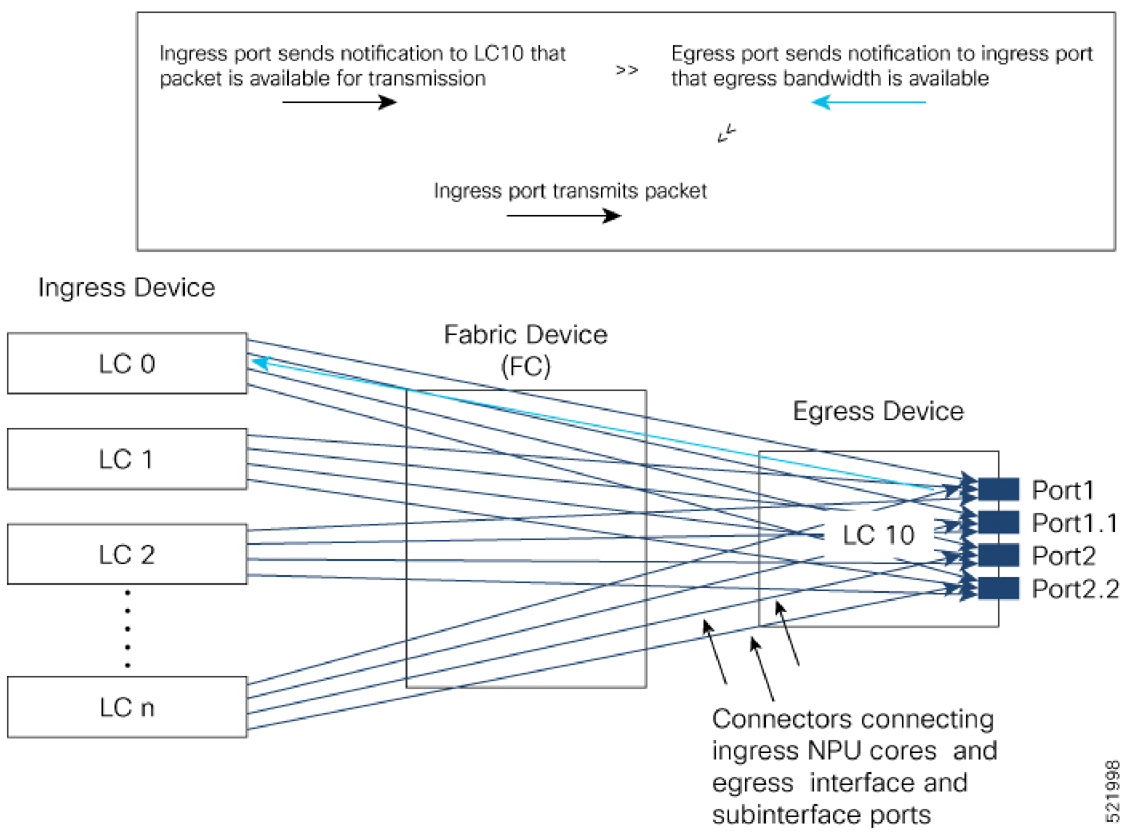
- The default scheduling hierarchy operates in strict-priority mode, with traffic prioritized on classes TC7 >TC6 > TC5 > TC4 > TC3 > TC2 > TC1 > TC0. You can change this default behaviour of strict-priority mode by applying an egress queuing policy map.
- Priority flow control (PFC), explicit congestion notification (ECN), and 4 VOQ mode aren't supposed.
- Egress queuing policy map on the subinterface (physical subinterface or bundle subinterface) isn't supported.

Ingress Traffic Management Model

The ingress traffic management model relies on packet queueing on the egress interface using Virtual Output Queueing (VOQ) on the ingress. In this model, buffering takes place at ingress. Here's how the VOQ process works.

Your routers support up to eight output queues per main interface or physical port. For every egress output queue, the VOQ model earmarks buffer space on every ingress pipeline. This buffer space is in the form of dedicated VOQs. These queues are called virtual because the queues physically exist on the ingress interface only when the line card actually has packets enqueued to it. To support the modular model of packet distribution, each network processing unit (NPU) core at the ingress needs connectors to every egress main interface and subinterface. The ingress traffic management model thus requires a mesh of connectors to connect the ingress NPU cores to the egress interfaces, as shown in **The Ingress Traffic Management Model**.

Figure 1: The Ingress Traffic Management Model



In the figure, every ingress interface (LC 0 through LC n) port has eight VOQs for the single egress line card LC 10.

Here's how packet transmission takes place:

1. When a packet arrives at ingress port (say on LC 0), the forwarding lookup on ingress line card points to the egress interface. Based on the egress interface (say it is on LC10), the packet is enqueued to the VOQ of LC 10. The egress interface is always mapped to a physical port.
2. Once egress bandwidth is available, the LC 10 ports ready to receive the packets (based on the packet marking and distribution model) send grants to the ingress ports via the connectors. (The figure shows a separate line for the grant for the sake of visual representation. In reality, the same connector is used for requests, grants, and transmission between an NPU core at the ingress and the egress port on LC 10.)
3. The ingress ports respond to this permission by transmitting the packets via FC to the LC 10 ports. (The time it takes for the ingress ports to request for egress port access, the egress port to grant access, and the packet to travel across FC is the round-trip time.)

The VOQ model thus operates on the principle of storing excess packets in buffers at ingress until bandwidth becomes available. Based on the congestion that builds up and the configured threshold values, packets begin to drop at the ingress itself, instead of having to travel all the way to the egress interface and then getting dropped.

Hardware Limitation:

In a scale scenario where 1000+ VoQs (created using egress QoS policies) store packets due to active traffic flows and may consume all the available on-chip buffer (OCB), unexpected traffic drops will be seen even though the traffic rate at the VoQ level is less than that of the VoQ shaper.

Egress Traffic Management Model

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Egress Traffic Management	Release 24.3.1	Introduced in this release on: Modular Systems (8800 [LC ASIC: P100])(select variants only*) By employing a two-pass traffic-forwarding model, you can conserve the Virtual Output Queue (VOQ) connector resources and thus increase the QoS queuing policy map scale. The egress traffic management (ETM) model splits VOQ resources into global and local VOQs, where the global VOQs are used in the first pass, and traffic in the second pass is resolved to local VOQs that are mapped to the main or subinterface. *This feature is enabled by default and supported on: • 88-LC1-12TH24FH-E • 88-LC1-52Y8H-EM

Egress traffic management is a feature designed to improve the QoS queuing policy map scale that

- splits the VOQs into global and local VOQs
- uses a two-pass model to manage traffic, where the first pass utilizes global VOQs and the second pass resolves traffic to local VOQs, and
- uses a recycle model at the egress NPU to ensure packet buffering occurs at the ingress interface during the first pass and at the egress NPU during the second pass.

ETM is enabled by default on specific Cisco Silicon One P100-based line cards and modifies the existing Ingress Traffic Management Model to support higher QoS scales. For more information on the Ingress Traffic Management buffering mechanism, see Ingress Traffic Management Model, on page 2.

This table helps you understand the difference between the Egress and Ingress Traffic Management models based on key attributes.

Table 2: Ingress and Egress Traffic Management Models - Key Differences

Attributes	Ingress Traffic Management	Egress Traffic Management
Queue Location	Buffers packets at the ingress interface using Virtual Output Queues (VOQs).	Buffers packets at the ingress interface during the first pass and at the egress NPU during the second pass.
VOQ Structure	Requires a mesh of connectors to connect ingress NPU cores to egress interfaces.	Splits VOQs into global and local categories, conserving VOQ connectors.
Scalability	Faces significant scale issues due to the need for each ingress device to represent queueing for every destination.	Supports higher scales of egress QoS policies by reducing the number of required VOQs.
Feature Support	Limited ability to use output features such as egress ACLs and policies.	Allows for the use of output features like egress ACLs, policies, and egress policing.
Multicast Traffic Control	Multicast traffic is not scheduled, leading to less control over multicast traffic scheduling.	Provides enhanced control over multicast traffic by applying queuing policy map parameters in the second pass.
Resource Utilization	May consume all available on-chip buffer (OCB) in high-scale scenarios, leading to unexpected traffic drops.	Optimizes resource utilization by buffering packets at both the ingress interface (first pass) and the egress NPU (second pass).
Subinterface QoS Policy	Implementing queueing at the egress sub-interface level requires a large number of VOQs, increasing costs.	Supports class-level rate limiters on subinterface queuing policy maps, enabling granular traffic control.

Usage Guidelines and Limitations for the Egress Traffic Management Model

- **Which routers and line cards support the egress traffic management model?**

  The egress traffic management model is supported on the following line card:

  Table 3: Egress Traffic Management—Supported Line Cards

Line Cards	ETM on LC or Port Level
88-LC1-12TH24FH-E	ETM is enabled on all the ports in the LC by default. You can’t disable ETM on the port level in this LC.
88-LC1-52Y8H-EM	

- **What benefits does the egress traffic management model offer?**
  - Configuring the egress traffic management functionality offers you **higher scales of egress QoS policies**. This is achieved by conserving VoQ connectors through the segregation of VoQs into global VoQs and local VoQs.
  - The multicast traffic is now scheduled in the second pass, which means the egress queuing policy map parameters such as traffic shaping, priority, queuing, and so on, can be applied to the multicast traffic, and hence there’s more control over the multicast traffic scheduling.

Note The multicast traffic in the first pass is unscheduled, which is similar to the non-ETM scenario.

  - With ETM, the subinterface queuing policy map can have class-level rate limiters (traffic shapers) on each class. This helps to granularly control the traffic flow on each class in the queuing policy map.

- **What is the impact of the traffic recycling due to two-pass forwarding?** The impact is **reduced throughput**. The total available network system port bandwidth is reduced to at least 50%. The recycle port per interface group (IFG) can schedule the traffic at a rate of up to 550 Gbps in the first pass.
- **How do I handle Link Aggregation Group (LAG) configurations?** A mix of ETM-enabled and non-ETM-enabled ports in an LAG is not supported.
- **What is the default scheduling hierarchy in the first pass?** The first pass scheduling hierarchy works in the strict-priority mode, which means traffic on classes TC7>TC6>TC5>TC4>TC3>TC2>TC1>TC0. This isn’t configurable using a policy map.
- **What is the default scheduling hierarchy in the second pass?** The second pass scheduling hierarchy operates in strict-priority mode by default, which means traffic on classes TC7 >TC6 > TC5 > TC4 > TC3 > TC2 > TC1 > TC0. This default behaviour of strict priority mode can be changed to the required configuration by applying the egress queuing policy map.
- **Is VoQ mode configuration and counter sharing mode supported on the ETM line card?** Only 8 VoQ mode is supported on ETM line card, and counter sharing mode isn’t supported.
- **Is egress queuing and classification supported on both L2 and L3 interfaces on the ETM line card?** Egress queuing and classification based on traffic classes is only supported on L3 interfaces.

Note ECN remarking and priority flow control (PFC) aren’t supported.

How Packet Flow Works in the Egress Traffic Management Model

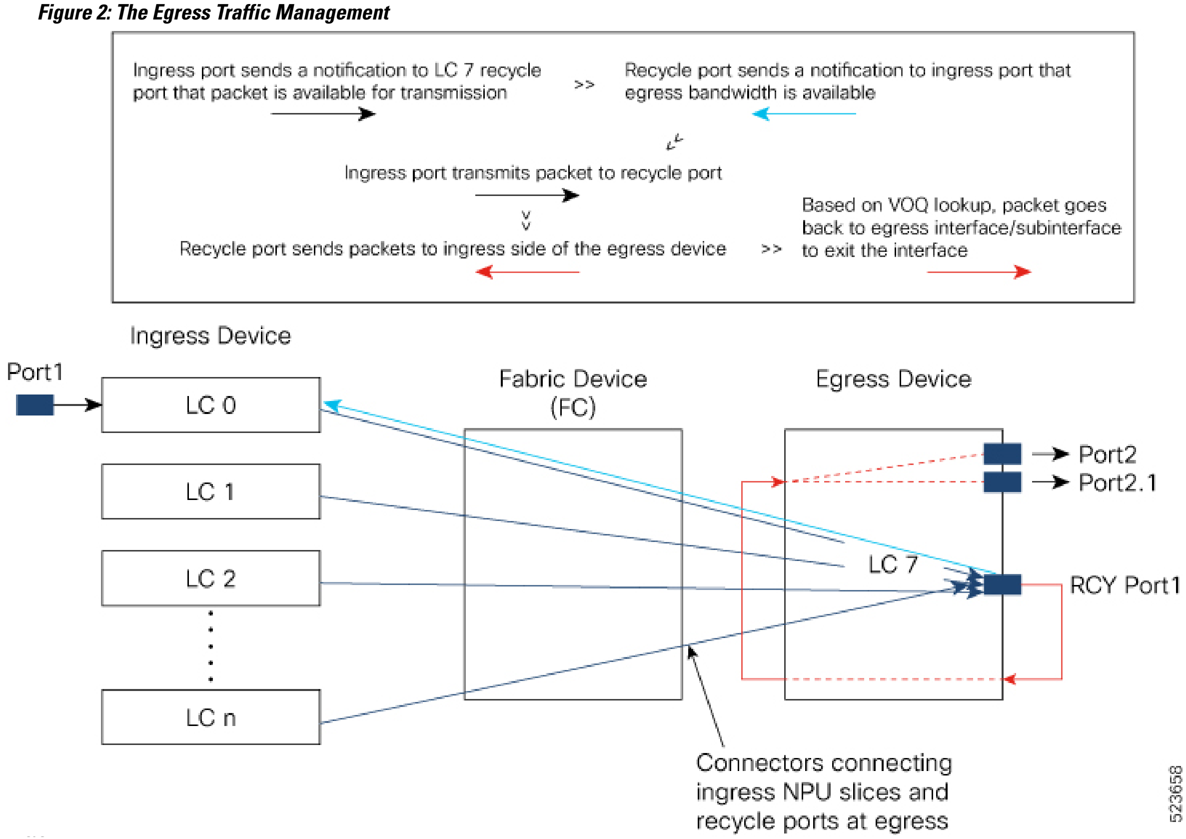
Summary

This topic describes how ETM-enabled line cards transmit packets between an ingress device and an egress device through a fabric device in the egress traffic management model. It also shows the role of the recycle ports in enabling a VOQ lookup on the egress.

Stages

These stages describe the packet flow in ETM.

Figure 2: The Egress Traffic Management



In the figure, let us assume a packet arrives at port 1 in ingress LC 0, and the packet has to be forwarded out via port 2 in egress LC 7.

Here’s how the packet transmission takes place:

1. When a packet arrives at ingress port 1 (on LC 0), the forwarding lookup on the ingress line card points to the egress interface (port 2) that could be physical, subinterface, Link aggregation Group (LAG), or LAG subinterface. In the case of LAG, hashing takes place at the ingress line card to select the LAG member from where the packet are sent out.

2. In contrast to single-pass forwarding, where the traffic gets enqueued to the VoQ of the egress port 2 that is replicated in the ingress slice of the LC0, ETM involves a two-pass forwarding process for traffic. The traffic from ingress is forwarded to the recycle port, also called as Edge recycle port (ERP). The ERP is allotted per interface group (IFG). The traffic gets enqueued in the first pass VOQs called as global VOQs.

   About VOQs on ETM system: Each port on the ETM-enabled line card has two sets of queues (global and local). The port itself is referred to as Edge port (EP). An EP main interface (physical or LAG) will have both global (GVOQ) and local (LVOQ). An EP subinterface uses the local VOQ of the main interface by default. If a queuing policy map is applied on the EP subinterface, then the new set of LVOQs is created for that EP subinterface for the second-pass forwarding.

   The global VOQs are replicated across all the slices in the system. In this example, all the slices present in LC1 to LCn. The local VOQs are replicated only on the slice where the egress interface is associated.

   To send the traffic from any ingress LC port to an egress port, the global VOQs should have connectors from each ingress LC to the egress port. This consumes connectors in proportion to the number of slices present in the system. Whereas since the local VOQs are local to a particular slice, it’s sufficient to use the connectors on that slice only. This change helps to conserve the VOQ connectors and hence increase in the queuing policy map scale.

3. First-pass forwarding: In this example, traffic on ingress port 1 of LC0 gets enqueued into the GVOQ of the EP port 2. This GVOQ is replicated in the LC0 slice of the ingress port1. The credit request from the ingress scheduler is sent to the ERP to forward the traffic. Once the credit is granted by the ERP, the traffic from GVOQ can be dequeued toward the ERP.

4. Second-pass forwarding: The traffic from ERP is forwarded to the ingress side of the slice where EP (port 2 in LC7) is associated. Here in the second pass the LVOQ resolution happens and traffic gets enqueued into the LVOQ on the slice where the Edge Port (EP) is present. The credit request from the ingress scheduler from this slice is sent to the EP for traffic forwarding. Once the credit grant is provided by the EP, the traffic from LVOQ can be dequeued towards the EP (port 2 of LC7).

   An example for second-pass traffic on the egress subinterface (Physical or LAG):

   - No egress queueing policy map on the subinterface: In this case, traffic destined for egress interface port 2.1 on LC7 uses the local VOQs from the main interface port 2.
   - Egress queueing policy map on the subinterface: When an egress queueing policy map is applied to the subinterface, the traffic will use the Local VOQs created for that subinterface on port 2.1 after the application of the queueing policy map.

QoS Policy Configuration Rules

- **How to enqueue the traffic in the right VoQ in the first and second pass?**
  - First pass: To enqueue the traffic in the first pass VoQs from TC7 to TC1, use **set traffic-class** <> in the ingress policy map. This configuration is similar to the configuration on non-ETM cards.
  - Second pass: To enqueue the traffic in the second pass VoQs from TC7 to TC1, you must use **set traffic-class** <> in the egress marking policy map. If this **set traffic-class** <> isn’t used in the egress marking policy map, then traffic uses the TC0 VoQ.
  - Also, use **set discard-class** <> alongside the traffic class value in the egress marking policy map if DQL is employed in the queuing policy map.

> **Note** TC value defines the offset value from the base VoQ of that interface.

- The other configuration changes with ETM are that subinterface queuing policy maps can have shaper configuration at the class level. Use the **shape average** <> command for this purpose.
- You can use classification based on DSCP/EXP/QG for egress marking policy map classification.

Configure Egress Traffic Management

Configure egress traffic management queuing and marking policies on interfaces to improve the QoS queuing policy map scale on an ETM-enabled line card.

Before you begin

Before you configure egress queuing and marking QoS policies, verify the interface is in the etm mode. For more details on the CLI command to use, see step1 in the procedure below.

Step 1 Verify if an interface is in the **etm** mode using the **show controllers npu voq-usage interface** *interface***instance all location** *location* command.

Example:

```
Router#show controllers npu voq-usage interface fourHundredGigE 0/3/0/9 instance all location 0/3/CPU0

-------------------------------------------------------------------
Node ID: 0/3/CPU0
Intf         Intf     NPU Slice IFG  Sys   VOQ   Flow       VOQ      Port
name         handle    #    #    #   Port  base  base       port     speed
             (hex)                                          type
----------------------------------------------------------------------
FH0/3/0/9    1800270   0    0    0   161   40128      0    etm_local    400G
FH0/3/0/9    1800270   0    0    0   161   15248      0   etm_global    400G
```

In this output, **etm_local** refers to LVOQ, and **etm_global** refers to GVOQ.

Step 2 Apply the egress marking policy map on the egress port (EP).

a) Configure class map for egress marking policy.

Example:

```
Router#conf
Router(config)#class-map match-any QG2
Router(config-cmap)# match qos-group 2
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any QG5
Router(config-cmap)# match qos-group 5
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any QG6
Router(config-cmap)# match qos-group 6
Router(config-cmap)# end-class-map
```

```
Router(config)#!
Router(config)#class-map match-any QG4
Router(config-cmap)# match qos-group 4
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any QG1
Router(config-cmap)# match qos-group 1
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any QG3
Router(config-cmap)# match qos-group 3
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any QG7
Router(config-cmap)# match qos-group 7
Router(config-cmap)# end-class-map
Router(config)# commit
```

b) Configure egress marking policy map.

Example:

```
Router(config)#policy-map EGRESS_MARK_TC
Router(config-pmap #class QG2
Router(config-pmap-c)#set traffic-class 2
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG5
Router(config-pmap-c)#  set traffic-class 5
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG6
Router(config-pmap-c)#  set traffic-class 6
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG7
Router(config-pmap-c)#  set traffic-class 7
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG4
Router(config-pmap-c)#  set traffic-class 4
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG1
Router(config-pmap-c)#  set traffic-class 1
Router(config-pmap-c)# !
Router(config-pmap-c)# class QG3
Router(config-pmap-c)#  set traffic-class 3
Router(config-pmap-c)# !
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# !
Router(config-pmap-c)# end-policy-map
Router(config)#commit
Router(config)#end
```

Step 3 Apply the egress queuing policy map on EP.

a) Configure class map for egress queuing policy.

Example:

```
Router#conf
Router(config)#class-map match-any TC2
Router(config-cmap)# match traffic-class 2
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any TC3
Router(config-cmap)# match traffic-class 3
Router(config-cmap)# end-class-map
```

```
Router(config)#!
Router(config)#class-map match-any TC4
Router(config-cmap)# match traffic-class 4
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any TC5
Router(config-cmap)# match traffic-class 5
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any TC7
Router(config-cmap)# match traffic-class 7
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any TC6
Router(config-cmap)# match traffic-class 6
Router(config-cmap)# end-class-map
Router(config)#!
Router(config)#class-map match-any TC1
Router(config-cmap)# match traffic-class 1
Router(config-cmap)# end-class-map
Router(config)#commit
```

b) Configure policy map for egress queuing policy.

Example:

```
Router(config)#policy-map EGRESS_QUEUING
Router(config-pmap)# class TC2
Router(config-pmap-c)#  bandwidth remaining ratio 15
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC3
Router(config-pmap-c)#  bandwidth remaining ratio 20
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC4
Router(config-pmap-c)#  bandwidth remaining ratio 15
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC5
Router(config-pmap-c)#  bandwidth remaining ratio 30
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC7
Router(config-pmap-c)#  priority level 1
Router(config-pmap-c)#  shape average percent 40
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC6
Router(config-pmap-c)#  shape average percent 30
Router(config-pmap-c)#  priority level 2
Router(config-pmap-c)# !
Router(config-pmap-c)# class TC1
Router(config-pmap-c)#  bandwidth remaining ratio 10
Router(config-pmap-c)# !
Router(config-pmap-c)# class class-default
Router(config-pmap-c)#  bandwidth remaining ratio 10
Router(config-pmap-c)# !
Router(config-pmap-c)# end-policy-map
Router(config)#!
Router(config)#commit

Router(config)#policy-map PARENT_EGRESS_QUEUING
Router(config-pmap)# class class-default
Router(config-pmap-c)#  service-policy EGRESS_QUEUING
Router(config-pmap-c)# !
Router(config-pmap-c)# end-policy-map
```

```
Router(config)#
Router(config)#commit
```

c) Attach the policy map to the interface.

Example:

```
Router(config)#int fourHundredGigE 0/3/0/9
Router(config-if)#service-policy output PARENT_EGRESS_QUEUING
Router(config-if)#commit

Router(config)#int fourHundredGigE 0/3/0/9
Router(config-if)#service-policy output EGRESS_MARK_TC
Router(config-if)#commit
```

Step 4 **Running Configuration**—evaluate the configurations are in effect.

Example:

```
class-map match-any class-default
 end-class-map
!
class-map match-any TC2
 match traffic-class 2
 end-class-map
!
class-map match-any TC3
 match traffic-class 3
 end-class-map
!
class-map match-any TC4
 match traffic-class 4
 end-class-map
!
class-map match-any TC5
 match traffic-class 5
 end-class-map
!
class-map match-any TC7
 match traffic-class 7
 end-class-map
!
class-map match-any TC6
 match traffic-class 6
 end-class-map
!
class-map match-any TC1
 match traffic-class 1
 end-class-map
!
policy-map EGRESS_QUEUING
 class TC2
  bandwidth remaining ratio 15
 !
 class TC3
  bandwidth remaining ratio 20
 !
 class TC4
  bandwidth remaining ratio 15
 !
 class TC5
  bandwidth remaining ratio 30
 !
```

```
 class TC7
  priority level 1
  shape average percent 40
 !
 class TC6
  shape average percent 30
  priority level 2
 !
 class TC1
  bandwidth remaining ratio 10
 !
 class class-default
  bandwidth remaining ratio 10
 !
 end-policy-map
!
policy-map PARENT_EGRESS_QUEUING
 class class-default
  service-policy EGRESS_QUEUING
 !
 end-policy-map
!

class-map match-any QG2
 match qos-group 2
 end-class-map
!
class-map match-any QG5
 match qos-group 5
 end-class-map
!
class-map match-any QG6
 match qos-group 6
 end-class-map
!
class-map match-any QG7
 match qos-group 7
 end-class-map
!
class-map match-any QG4
 match qos-group 4
 end-class-map
!
class-map match-any QG1
 match qos-group 1
 end-class-map
!
class-map match-any QG3
 match qos-group 3
 end-class-map
!
policy-map EGRESS_MARK_TC
 class QG2
  set traffic-class 2
 !
 class QG5
  set traffic-class 4
 !
 class QG6
  set traffic-class 6
 !
 class QG7
  set traffic-class 7
 !
```

```
 class QG4
  set traffic-class 4
 !
 class QG1
  set traffic-class 1
 !
 class QG3
  set traffic-class 3
 !
 class class-default
 !
 end-policy-map
!
RP/0/RP0/CPU0:Router#
```

Step 5 To verify the egress queuing and marking QoS policies are in effect, use the **show qos int** *interface* **output** command.

Example:

```
Router#show qos int fourHundredGigE 0/3/0/9 output
NOTE:- Configured values are displayed within parentheses
Interface FourHundredGigE0/3/0/9 ifh 0x1800270 -- output policy
NPU Id:                        0
Total number of classes:       8
Interface Bandwidth:           400000000 kbps
Policy Name:                   EGRESS_MARK_TC
VOQ Base:                      0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
VOQ Base:                      8
Shared Counter Mode:           1
------------------------------------------------------------------------------
Level1 Class                            =   QG2
New traffic class                       =   2

Level1 Class                            =   QG5
New traffic class                       =   4

Level1 Class                            =   QG6
New traffic class                       =   6

Level1 Class                            =   QG7
New traffic class                       =   7

Level1 Class                            =   QG4
New traffic class                       =   4

Level1 Class                            =   QG1
New traffic class                       =   1

Level1 Class                            =   QG3
New traffic class                       =   3

Level1 Class                            =   class-default
Interface FourHundredGigE0/3/0/9 ifh 0x1800270  -- output policy
NPU Id:                        0
Total number of classes:       9
Interface Bandwidth:           400000000 kbps
Policy Name:                   PARENT_EGRESS_QUEUING
VOQ Base:                      40128
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
VOQ Mode:                      8
Shared Counter Mode:           1
------------------------------------------------------------------------------
Level1 Class                            =   class-default
Queue Max. BW.                          =   no max (default)
```

```
      Inverse Weight / Weight                  =   0 / (BWR not configured)

         Level2 Class                             =   TC2
         Egressq Queue ID                         =   40130 (LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   4 / (15)
         Guaranteed service rate                  =   18000000 kbps
         TailDrop Threshold                       =   13498368 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class                             =   TC3
         Egressq Queue ID                         =   40131 (LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   3 / (20)
         Guaranteed service rate                  =   23999999 kbps
         TailDrop Threshold                       =   17995776 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class                             =   TC4
         Egressq Queue ID                         =   40132 (LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   4 / (15)
         Guaranteed service rate                  =   18000000 kbps
         TailDrop Threshold                       =   13498368 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class                             =   TC5
         Egressq Queue ID                         =   40133 (LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   2 / (30)
         Guaranteed service rate                  =   36000000 kbps
         TailDrop Threshold                       =   26996736 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class (HP1)                       =   TC7
         Egressq Queue ID                         =   40135 (HP1 queue)
         Queue Max. BW.                           =   160000000 kbps (40 %)
         Guaranteed service rate                  =   160000000 kbps
         TailDrop Threshold                       =   119998464 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class (HP2)                       =   TC6
         Egressq Queue ID                         =   40134 (HP2 queue)
         Queue Max. BW.                           =   120000000 kbps (30 %)
         Guaranteed service rate                  =   120000000 kbps
         TailDrop Threshold                       =   89997312 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class                             =   TC1
         Egressq Queue ID                         =   40129 (LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   6 / (10)
         Guaranteed service rate                  =   11999999 kbps
         TailDrop Threshold                       =   8994816 bytes / 6 ms (default)
         WRED not configured for this class

         Level2 Class                             =   class-default
         Egressq Queue ID                         =   40128 (Default LP queue)
         Queue Max. BW.                           =   no max (default)
         Inverse Weight / Weight                  =   6 / (10)
         Guaranteed service rate                  =   11999999 kbps
         TailDrop Threshold                       =   8994816 bytes / 6 ms (default)
         WRED not configured for this classRouter#
```

Step 6 To verify the QoS policy map statistics, use the **show policy-map** *interface* **output** command.

Example:

```
Router#show policy-map int fourHundredGigE 0/3/0/9 output

FourHundredGigE0/3/0/9 output: EGRESS_MARK_TC

Class QG2
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520780/65520780000         6194677
    Transmitted         :           65520780/65520780000         6194677
    Total Dropped       :                  0/0                   0
Class QG5
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520854/65520854000         6194683
    Transmitted         :           65520854/65520854000         6194683
    Total Dropped       :                  0/0                   0
Class QG6
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520850/65520850000         6194683
    Transmitted         :           65520850/65520850000         6194683
    Total Dropped       :                  0/0                   0
Class QG7
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520850/65520850000         6194683
    Transmitted         :           65520850/65520850000         6194683
    Total Dropped       :                  0/0                   0
Class QG4
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520846/65520846000         6194683
    Transmitted         :           65520846/65520846000         6194683
    Total Dropped       :                  0/0                   0
Class QG1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520848/65520848000         6194683
    Transmitted         :           65520848/65520848000         6194683
    Total Dropped       :                  0/0                   0
Class QG3
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520844/65520844000         6194683
    Transmitted         :           65520844/65520844000         6194683
    Total Dropped       :                  0/0                   0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           65520842/65520842000         6194684
    Transmitted         :           65520842/65520842000         6194684
    Total Dropped       :                  0/0                   0
Policy Bag Stats time: 1705937324806  [Local Time: 01/22/24 15:28:44.806]

FourHundredGigE0/3/0/9 output: PARENT_EGRESS_QUEUING

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          523227880/528983327628        50113244
    Transmitted         :          523227880/528983327628        50113244
    Total Dropped       :                  0/0                   0

  Policy EGRESS_QUEUING Class TC2
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :           65403419/66122856609         6264151
      Transmitted         :           65403419/66122856609         6264151
      Total Dropped       :                  0/0                   0
    Queueing statistics
```

```
    Queue ID                             : 40130
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC3
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403485/66122923335          6264156
    Transmitted         :            65403485/66122923335          6264156
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40131
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC4
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403490/66122928390          6264155
    Transmitted         :            65403490/66122928390          6264155
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40132
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC5
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403479/66122917269          6264155
    Transmitted         :            65403479/66122917269          6264155
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40133
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC7
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403545/66122924943          6264156
    Transmitted         :            65403545/66122924943          6264156
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40135
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC6
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403479/66122917269          6264156
    Transmitted         :            65403479/66122917269          6264156
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40134
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class TC1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403489/66122927379          6264157
    Transmitted         :            65403489/66122927379          6264157
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40129
    Taildropped(packets/bytes)           : 0/0

Policy EGRESS_QUEUING Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            65403494/66122932434          6264158
    Transmitted         :            65403494/66122932434          6264158
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                             : 40128
```

```
      Taildropped(packets/bytes)           : 0/0
Policy Bag Stats time: 1705937324635  [Local Time: 01/22/24 15:28:44.635]
Router#
```

Low-Latency Queueing with Strict Priority Queueing

The Low-Latency Queueing (LLQ) feature brings strict priority queuing (PQ) to the CBWFQ scheduling mechanism. Priority Queueing (PQ) in strict priority mode ensures that one type of traffic is sent, possibly at the expense of all others. For PQ, a low-priority queue can be detrimentally affected, and, in the worst case, never allowed to send its packets if a limited amount of bandwidth is available or the transmission rate of critical traffic is high. Strict PQ allows delay-sensitive data, such as voice, to be de-queued and sent before packets in other queues are de-queued.

Configure Low Latency Queueing with Strict Priority Queueing

Configuring low latency queueing (LLQ) with strict priority queuing (PQ) allows delay-sensitive data such as voice to be de-queued and sent before the packets in other queues are de-queued.

Guidelines

- Only priority level 1 to 7 is supported, with 1 being the highest priority and 7 being the lowest. However, the default CoSQ 0 has the lowest priority among all.
- Egress policing is not supported. Hence, in the case of strict priority queuing, there are chances that the other queues do not get serviced.
- You can configure **shape average** and **queue-limit** commands along with **priority**.
- You can configure an optional shaper to cap the maximum traffic rate. This is to ensure the lower priority traffic classes are not starved of bandwidth.

Configuration Example

You have to accomplish the following to complete the LLQ with strict priority queuing:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Specifying priority to the traffic class
4. (Optional) Shaping the traffic to a specific bit rate
5. Attaching the policy-map to an output interface

```
Router# configure
Router(config)# policy-map test-priority-1
Router(config-pmap)# class qos1
Router(config-pmap-c)# priority level 7
Router(config-pmap-c# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/6/0/18
```

```
Router(config-if)# service-policy output test-priority-1
Router(config-if)# no shutdown
Router(config-if)# commit
```

Running Configuration

```
policy-map strict-priority
class tc7
  priority level 1
  queue-limit 75 mbytes
!
 class tc6
  priority level 2
  queue-limit 75 mbytes
!
 class tc5
  priority level 3
  queue-limit 75 mbytes
 !
 class tc4
  priority level 4
  queue-limit 75 mbytes
!
 class tc3
  priority level 5
  queue-limit 75 mbytes
!
 class tc2
  priority level 6
  queue-limit 75 mbytes
 !
 class tc1
  priority level 7
  queue-limit 75 mbytes
 !
 class class-default
  queue-limit 75 mbytes
 !
 end-policy-map
!



class-map match-any tc1
match traffic-class 1
 end-class-map
!
class-map match-any tc2
match traffic-class 2
 end-class-map
!
class-map match-any tc3
match traffic-class 3
 end-class-map
!
class-map match-any tc4
match traffic-class 4
 end-class-map
!
class-map match-any tc5
match traffic-class 5
 end-class-map
!
```

```
class-map match-any tc6
match traffic-class 6
 end-class-map
!
class-map match-any tc7
match traffic-class 7
 end-class-map
!

interface HundredGigE0/6/0/18
 service-policy input 100g-s1-1
 service-policy output test-priority-1
!
```

Verification

```
Router#  show qos int hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class (HP7)                       =   qos-1
Egressq Queue ID                         =   11177 (HP7 queue)
TailDrop Threshold                       =   125304832 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class (HP6)                       =   qos-2
Egressq Queue ID                         =   11178 (HP6 queue)
TailDrop Threshold                       =   125304832 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   11176 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
TailDrop Threshold                       =   1253376 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Congestion Management Overview, on page 1
- Traffic Shaping, on page 20

Traffic Shaping

Traffic shaping allows you to control the traffic flow exiting an interface to match its transmission to the speed of the remote target interface and ensure that the traffic conforms to policies contracted for it. Traffic adhering to a particular profile can be shaped to meet downstream requirements, thereby eliminating bottlenecks in topologies with data-rate mismatches.

Note Traffic shaping is supported only in egress direction.

Configure Traffic Shaping

The traffic shaping performed on outgoing interfaces is done at the Layer 1 level and includes the Layer 1 header in the rate calculation.

Guidelines

- Only egress traffic shaping is supported.
- It is mandatory to configure all the eight qos-group classes (including class-default) for the egress policies.
- You can configure **shape average** command along with **priority** command.
- It is recommended that you configure all the eight <traffic-class classes> (including **class-default**) for the egress policies. A limited number of < traffic-class class> combinations are supported, but unicast and multicast traffic-class may not be handled consistently. Hence, such configurations are recommended, when the port egress has only unicast traffic.

Configuration Example

You have to accomplish the following to complete the traffic shaping configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Shaping the traffic to a specific bit rate
4. Attaching the policy-map to an output interface

```
Router# configure
Router(config)# policy-map egress_policy1
Router(config-pmap)# class c5
Router(config-pmap-c)# shape average percent 40
Router(config-pmap-c# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/1/0/0
Router(config-if)# service-policy output egress_policy1
Router(config-if)# commit
```

Running Configuration

```
policy-map egress_policy1
 class c5
  shape average percent 40
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
```

```
 service-policy input 100g-s1-1
 service-policy output egress_policy1
!
```

Verification

```
Router#  show qos interface hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   c5
Egressq Queue ID                         =   11177 (LP queue)
Queue Max. BW.                           =   40329846 kbps (40 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
Guaranteed service rate                  =   40000000 kbps
TailDrop Threshold                       =   50069504 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   11176 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
Guaranteed service rate                  =   50000000 kbps
TailDrop Threshold                       =   62652416 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Congestion Management Overview, on page 1

Traffic Policing

Traffic policing allows you to control the maximum rate of traffic sent or received on an interface and to partition a network into multiple priority levels or class of service (CoS).Traffic policing manages the maximum rate of traffic through a token bucket algorithm. The token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on an interface at a given moment in time. The token bucket algorithm is affected by all traffic entering or leaving the interface (depending on where the traffic policy with traffic policing is configured) and is useful in managing network bandwidth in cases where several large packets are sent in the same traffic stream. By default, the configured bandwidth value takes into account the Layer 2 encapsulation that is applied to traffic leaving the interface.

The router supports the following traffic policing mode:

- Single-Rate Two-Color (SR2C) in color-blind mode. See Single-Rate Policer, on page 23.

Restrictions

- 1R3C policers are not supported.
- Up to eight policers are supported per ingress policy.
- Policers are allocated in multiples of 2, so any request for allocating odd number of policers is internally rounded up by a factor of one.
- Only one conditional marking action is supported. You can set discard class to 0/1, that is used to set either virtual output queueing (VOQ) limits or Random Early Detection (RED) profiles.
- If you configure discard-class explicitly at the class level and not under policer, then the explicit mark action is applied to all the transmitted policer traffic.
- Egress policing is not supported.

Committed Bursts and Excess Bursts

Unlike a traffic shaper, a traffic policer does not buffer excess packets and transmit them later. Instead, the policer executes a “send or do not send” policy without buffering. Policing uses normal or committed burst (bc) values and excess burst values (be) to ensure that the router reaches the configured committed information rate (CIR). Policing decides if a packet conforms or exceeds the CIR based on the burst values you configure. Burst parameters are based on a generic buffering rule for routers, which recommends that you configure buffering to be equal to the round-trip time bit-rate to accommodate the outstanding TCP windows of all connections in times of congestion. During periods of congestion, proper configuration of the excess burst parameter enables the policer to drop packets less aggressively.

For more details, see Committed Bursts, on page 28 and Excess Bursts, on page 29.

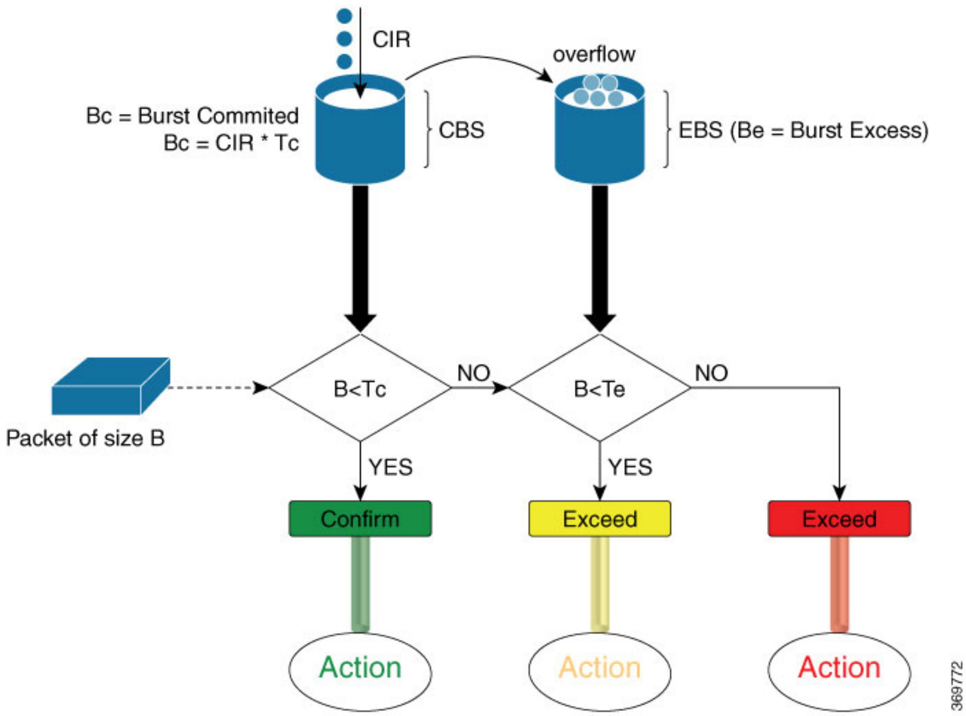
Single-Rate Policer

This section explains the concept of the single-rate two-color policer.

Single-Rate Two-Color Policer

A single-rate two-color (1R2C) policer provides one token bucket with two actions for each packet: a conform action and an exceed action.

Figure 3: Workflow of Single-Rate Two-Color Policey



Based on the committed information rate (CIR) value, the token bucket is updated at every refresh time interval. The Tc token bucket can contain up to the Bc value, which can be a certain number of bytes or a period of time. If a packet of size B is greater than the Tc token bucket, then the packet exceeds the CIR value and a default action is performed. If a packet of size B is less than the Tc token bucket, then the packet conforms and a different default action is performed.

Configure Traffic Policing (Single-Rate Two-Color)

Traffic policing is often configured on interfaces at the edge of a network to limit the rate of traffic entering or leaving the network. The default conform action for single-rate two color policer is to transmit the packet and the default exceed action is to drop the packet. Users cannot modify these default actions.

Configuration Example

You have to accomplish the following to complete the traffic policing configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. (Optional) Specifying the marking action
4. Specifying the policy rate for the traffic
5. Attaching the policy-map to an input interface

```
Router# configure
Router(config)# policy-map test-police-1R2C
Router(config-pmap)# class dscp1
```

```
Router(config-pmap-c)# police rate 10 gbps
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface hundredGigE 0/0/0/18
Router(config-if)# service-policy input test-police-1R2C
Router(config-if)# commit
```

Running Configuration

```
class-map match-any dscp1
 match dscp ipv4 1
 end-class-map
!
!
policy-map test-police-1R2C
 class dscp1
  police rate 10 gbps
  !
 !
 class class-default
  !
 !
 end-policy-map
!
!
interface HundredGigE0/0/0/8
service-policy input test-police-1R2C
!
```

Verification

```
Router# show qos interface hundredGigE 0/0/0/8 input
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/0/0/8 ifh 0xf0001e8  -- input policy
NPU Id:                        0
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                   test-police-1R2C
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   dscp1
Policer committed rate                   =   10000000 kbps (10 gbits/sec)
Policer conform burst                    =   1024000 bytes (default)
Policer conform action                   =   Just TX
Policer exceed action                    =   DROP PKT

Level1 Class                             =   class-default
Policer not configured for this class
```

Related Topics

- Traffic Policing, on page 22

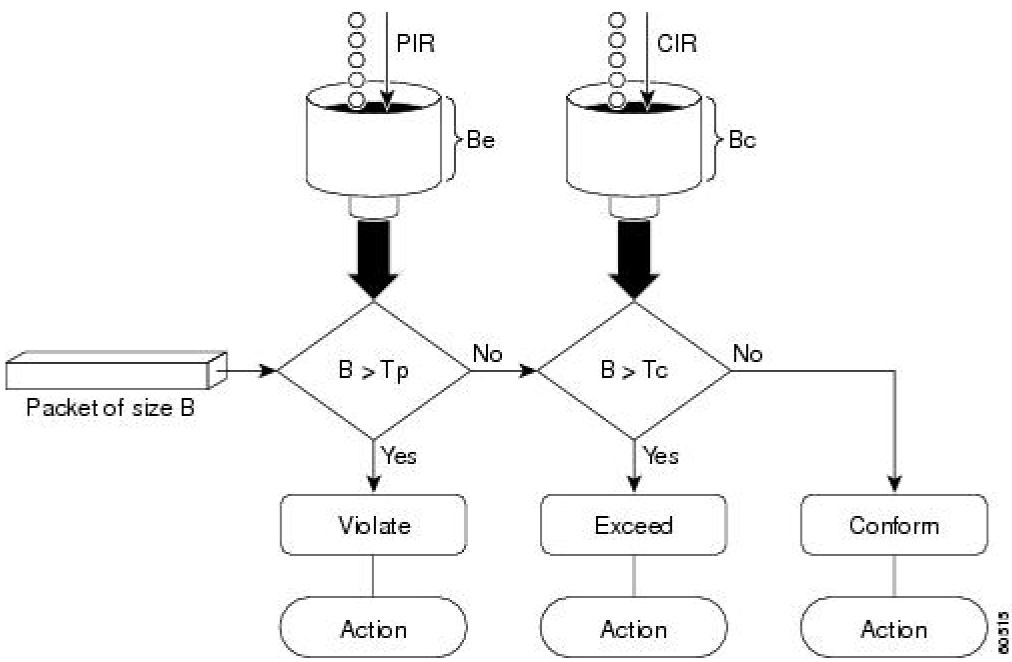
Two-Rate Policer

The two-rate policer manages the maximum rate of traffic by using two token buckets: the committed token bucket and the peak token bucket. The dual-token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on a queue at a given moment. In this way, the two-rate policer can meter traffic at two independent rates: the committed information rate (CIR) and the peak information rate (PIR).

The dual-token bucket algorithm provides users with three actions for each packet—a conform action, an exceed action, and an optional violate action. Traffic entering a queue with the two-rate policer configured is placed into one of these categories. The actions are pre-determined for each category. The default conform and exceed actions are to transmit the packet, and the default violate action is to drop the packet.

This figure shows how the two-rate policer marks a packet and assigns a corresponding action to the packet.

Figure 4: Marking Packets and Assigning Actions—Two-Rate Policer



Also, see Two-Rate Policer Details, on page 29.

The router supports Two-Rate Three-Color (2R3C) policer.

Configure Traffic Policing (Two-Rate Three-Color)

The default conform and exceed actions for two-rate three-color (2R3C) policer are to transmit the packet and the default violate action is to drop the packet. Users cannot modify these default actions.

Configuration Example

You have to accomplish the following to complete the two-rate three-color traffic policing configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Specifying the packet marking

4. Configuring two rate traffic policing
5. Attaching the policy-map to an input interface

```
Router# configure
Router(config)# policy-map test-police-2R3C
Router(config-pmap)# class dscp1
Router(config-pmap-c)# police rate 10 gbps peak-rate 20 gbps
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# iinterface hundredGigE 0/0/0/8
Router(config-if)# service-policy input test-police-2R3C
Router(config-if)# commit
```

Running Configuration

```
class-map match-any dscp1
 match dscp ipv4 1
 end-class-map
!
!
policy-map test-police-2R3C
 class dscp1
  police rate 10 gbps peak-rate 20 gbps
  !
 !
 class class-default
  !
 !
 end-policy-map
!
!
interface HundredGigE0/0/0/8
service-policy input test-police-2R3C
!
```

Verification

```
Router# show qos interface hundredGigE 0/0/0/8 input

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/0/0/8 ifh 0xf0001e8  -- input policy
NPU Id:                        0
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                   test-police-2R3C
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   dscp1
Policer committed rate                   =   10000000 kbps (10 gbits/sec)
Policer peak rate                        =   20000000 kbps (20 gbits/sec)
Policer conform burst                    =   1024000 bytes (default)
Policer exceed burst                     =   2048000 bytes (default)
Policer conform action                   =   Just TX
Policer exceed action                    =   DROP PKT
Policer violate action                   =   DROP PKT
```

```
Level1 Class                              =   class-default
Policer not configured for this class


Router# policy-map interface hundredGigE 0/0/0/8 input

HundredGigE0/0/0/8 input: test-police-2R3C

Class dscp1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          289228439/289228439000        27734775
    Transmitted         :           56422213/56422213000         5410359
    Total Dropped       :          232806226/232806226000        22324416
  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :           56422213/56422213000         5410359
    Policed(exceed)     :           56422215/56422215000         5410358
    Policed(violate)    :          176384011/176384011000        16914058
    Policed and dropped :          232806226/232806226000
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           61136620/61136620000         0
    Transmitted         :           61136620/61136620000         0
    Total Dropped       :                  0/0                   0
Policy Bag Stats time: 1570155764000  [Local Time: 10/04/19 02:22:44.000]
```

Related Topics

• Two-Rate Policer, on page 26

Committed Bursts

The committed burst (bc) parameter of the police command implements the first, conforming (green) token bucket that the router uses to meter traffic. The bc parameter sets the size of this token bucket. Initially, the token bucket is full and the token count is equal to the committed burst size (CBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the conforming token bucket to send packets:

- If sufficient tokens are in the conforming token bucket when a packet arrives, the meter marks the packet green and decrements the conforming token count by the number of bytes of the packet.
- If there are insufficient tokens available in the conforming token bucket, the meter allows the traffic flow to borrow the tokens needed to send the packet. The meter checks the exceeding token bucket for the number of bytes of the packet. If the exceeding token bucket has a sufficient number of tokens available, the meter marks the packet.

  Green and decrements the conforming token count down to the minimum value of 0.

  Yellow, borrows the remaining tokens needed from the exceeding token bucket, and decrements the exceeding token count by the number of tokens borrowed down to the minimum value of 0.
- If an insufficient number of tokens is available, the meter marks the packet red and does not decrement either of the conforming or exceeding token counts.

Note When the meter marks a packet with a specific color, there must be a sufficient number of tokens of that color to accommodate the entire packet. Therefore, the volume of green packets is never smaller than the committed information rate (CIR) and committed burst size (CBS). Tokens of a given color are always used on packets of that color.

Excess Bursts

The excess burst (be) parameter of the police command implements the second, exceeding (yellow) token bucket that the router uses to meter traffic. The exceeding token bucket is initially full and the token count is equal to the excess burst size (EBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the exceeding token bucket to send packets:

- When the first token bucket (the conforming bucket) meets the committed burst size (CBS), the meter allows the traffic flow to borrow the tokens needed from the exceeding token bucket. The meter marks the packet yellow and then decrements the exceeding token bucket by the number of bytes of the packet.
- If the exceeding token bucket does not have the required tokens to borrow, the meter marks the packet red and does not decrement the conforming or the exceeding token bucket. Instead, the meter performs the exceed-action configured in the police command (for example, the policer drops the packets).

Important Points

- User configurable burst values— committed burst size (CBS) and excess burst size (EBS)—are not supported. Instead, they are derived using user configured rates—committed information rates (CIR) and peak information rates (PIR).
- The router supports two burst values: low (10 kilobytes) and high (1 megabyte). For a lower range of configured values (1.6 MBps to less than 1 GBps), the burst value is 10 KB. For a higher range of configured values (1 GBps to 400 GBps), the burst value is 1 MB.

Two-Rate Policer Details

The committed token bucket can hold bytes up to the size of the committed burst (bc) before overflowing. This token bucket holds the tokens that determine whether a packet conforms to or exceeds the CIR as the following describes:

- A traffic stream is conforming when the average number of bytes over time does not cause the committed token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream green.
- A traffic stream is exceeding when it causes the committed token bucket to overflow into the peak token bucket. When this occurs, the token bucket algorithm marks the traffic stream yellow. The peak token bucket is filled as long as the traffic exceeds the police rate.

The peak token bucket can hold bytes up to the size of the peak burst (be) before overflowing. This token bucket holds the tokens that determine whether a packet violates the PIR. A traffic stream is violating when it causes the peak token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream red.

For example, if a data stream with a rate of 250 kbps arrives at the two-rate policer, and the CIR is 100 kbps and the PIR is 200 kbps, the policer marks the packet in the following way:

- 100 kbps conforms to the rate
- 100 kbps exceeds the rate
- 50 kbps violates the rate

The router updates the tokens for both the committed and peak token buckets in the following way:

- The router updates the committed token bucket at the CIR value each time a packet arrives at the interface. The committed token bucket can contain up to the committed burst (bc) value.
- The router updates the peak token bucket at the PIR value each time a packet arrives at the interface. The peak token bucket can contain up to the peak burst (be) value.
- When an arriving packet conforms to the CIR, the router takes the conform action on the packet and decrements both the committed and peak token buckets by the number of bytes of the packet.
- When an arriving packet exceeds the CIR, the router takes the exceed action on the packet, decrements the committed token bucket by the number of bytes of the packet, and decrements the peak token bucket by the number of overflow bytes of the packet.
- When an arriving packet exceeds the PIR, the router takes the violate action on the packet, but does not decrement the peak token bucket.

See Two-Rate Policer, on page 26.

References for Modular QoS Management

Read this section for more information on committed bursts, excess bursts, and two-rate policer.