Traffic Management Overview

Scope

Read this configuration guide to understand the overall architecture that powers the Cisco Quality of Service (QoS) technology, and also how to use its features to configure and manage the traffic bandwidth and packet loss parameters on your network.

Traditional Traffic Management

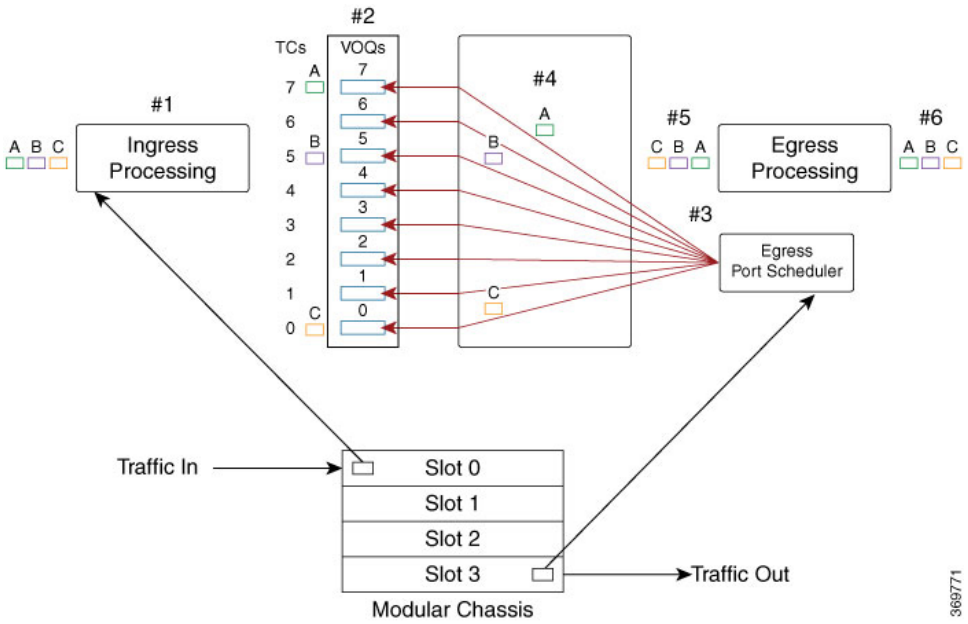
In traditional methods of traffic management, traffic packets are sent to the egress output queues without taking into consideration the egress interface availability to transmit.

Therein lies the problem as well. In case of a traffic congestion, the traffic packets may get dropped at the egress port. Which means that the network resources spent getting the packets from the ingress input queue across the switch fabric to the output queues at egress are wasted. That is not all—input queues buffer traffic meant for different egress ports, so congestion on one egress port could affect traffic on another port, an event referred to as head-of-line-blocking.

Traffic Management on Your Router

Your router's Network Processing Unit (NPU) uses a coupled ingress-egress virtual output queuing (VoQ)-based forwarding architecture to manage traffic.

Figure 1: Traffic flow from ingress port on slice 0 to an egress port on slot 3



Here, each ingress traffic class has a one-to-one VoQ mapping from each ingress slice (pipeline) to each egress port. Which means that every egress interface (#5 in the figure) has earmarked buffer space on every ingress pipeline (#1 in the figure) for each of its VoQs.

Here is how the story of packet travel in times of congestion on your router system unravels:

#1: Packets A (colored green), B (colored pink), and C (colored brown) are at the ingress interface. This is where packet marking, classification, and policing takes place. (For details, see Mark Packets to Change Priority Settings, Classify Packets to Identify Specific Traffic, and Congestion Management .)

#2: These packets are stored in separate buffer storage spaces in dedicated VoQs. This is where queuing, VoQ transmit, and drop packet and byte counters come into play. (For details, see Congestion Avoidance .)

#3: Depending on the bandwidth available on the egress interface, these packets are subjected to egress scheduling, where egress credit and transmit schedulers are configured. In other words, the packets and the sequence in which they will now proceed towards the egress interface is determined here. This is where the fabric bandwidth is taken into consideration for egress scheduling.

#4: The packets are switched through the fabric.

#5: In the final phase, the egress marking and classification takes place, and the congestion is managed in a way that at this stage there is no packet dropped, and all the packets are transmitted to the next hop.

Limitations of the VoQ Model

While the VoQ model of traffic management offers distinct advantages (reducing memory bandwidth requirements, providing end-to-end QoS flow), it has this limitation:

The total egress queue scale is lower because each egress queue must be replicated as an ingress VoQ on each slice of each NPU/ASIC in the system. This means that with the addition of 1 NPU with 20 interfaces, the number of VoQs used on each and every NPU in the system will increase by 20 x 8 (queue/interface) = 160.

There is also an increase in the number of credit connectors from each scheduler for each egress port on pre-existing NPUs to each slice in the newly inserted NPU.

QoS Policy Inheritance

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
QoS Policy Inheritance	Release 7.3.15	To create QoS policies for subinterfaces, you had to apply the policy on each subinterface manually. From this release, all you do is create and apply a single QoS policy on the main interface and the subinterfaces automatically inherit the policy. The inheritance model provides an easily maintainable method for applying policies, enabling you to create targeted policies for a group of interfaces and their subinterfaces. This model saves your time and resources while creating QoS policies.

- **What is this functionality all about?**—As the name suggests, the functionality is based on an inheritance model, where you create and apply a QoS policy to a main interface. The subinterfaces that are attached to the main interface automatically inherit the policy.

  The inheritance model applies to all QoS operations including:

  - Classification
  - Marking
  - Policing
  - Shaping

- **How does the inheritance model help?**—Previously, if you had, for example, eight subinterfaces, you created and applied policies to each of those subinterfaces separately. With the inheritance model, you save on time and resources, with just one policy automatically applied across a main interface and its subinterfaces.
- **Do I need to do anything more to enable the inheritance model?**—No, you don't. The inheritance model is the default option.
- **What if I want to override the inheritance option?**—Technically, you can't override this option. However, you can remove the policy from main interface and add policies to the subinterfaces except to the ones where you don't want the policy to be inherited.

- **What about policy-map statistics?**—There's no change to this behavior. Running the show policy-map interface command displays the cumulative statistics for an interface, and these numbers include the subinterfaces as well.
- **Any limitations I need to be aware of?**—There's no support for ECN marking and egress marking policy on the same interface and subinterface combination. However, the QoS policy inheritance functionality accepts these multiple policies causing the ECN markings to fail. To prevent such failures:
  - Don't configure an egress marking policy on a subinterface and apply an ECN-enabled policy on the main interface.
  - Don't apply an ECN policy on a subinterface and configure an egress marking policy on the main interface.

Cisco Modular QoS CLI to Deploy QoS

Cisco Modular QoS CLI (MQC) framework is the Cisco IOS QoS user language that enables:

- a standard Command Line Interface (CLI) and semantics for QoS features.
- simple and accurate configurations.
- QoS provisioning within the context of an extensible language.

For your router, in the egress direction, two types of MQC policies are supported: queuing and marking. You use the queuing policy to configure credit scheduling hierarchy, rates, priority, buffering, and congestion avoidance. You use the marking policy to classify and mark packets that have been scheduled for transmission. Even when a queuing policy is not applied, there is an implicit queuing policy with TC7 - P1, TC6 - P2, TC5 - TC0 (6 x Pn), so packets marked with TC7 and control inject packets are always prioritized over other packets. In the ingress, only one policy is supported for classification and marking.

You can apply queuing and marking policy independent of each other or together in the egress direction. If you apply both policies together, the queuing policy actions are provisioned first, followed by marking policy actions.

Important Points about MQC Egress Queuing Policy

These are important points that you must know about the MQC egress queuing policy:

- The MQC queuing policy consists of a set of class maps, which are added to a policy map. You control queuing and scheduling parameters for that traffic class by applying actions to the policy.
- **class-default** matches **traffic-class 0**, and no other class can match **traffic-class 0**. When **set traffic-class** is used in the ingress policy-map, then it should match the **traffic-class** in the egress queueing policy-map to have a predictable behaviour.
- Each unique combination of traffic classes that match **class-default** require a separate traffic class (TC) profile. The number of TC profiles are limited to 8 for main interfaces and 8 for sub-interfaces.
- You cannot configure multiple traffic classes with the same priority level.
- Each priority level, when configured, must be configured to the class that matches the corresponding TC as shown in the following table.

Priority Level	Traffic Class
P1	7
P2	6
P3	5
P4	4
P5	3
P6	2
P7	1

- If all the priority levels configured in a policy-map are sorted, they must be contiguous. In other words, you cannot skip a priority level. For example, P1 P2 P4 (skipping P3), is not allowed.
- From IOS XR Release 7.3.1 onwards, you can create a single set of contiguous priority TCs. Ensure that you assign priority levels that increase for every TC or remain the same, but don’t decrease. Also, make sure that you assign priority level 1 for traffic class 7. You need not configure unused traffic classes, so you can create only those many TCs that you require on the egress policy-map.
- MQC supports up to two levels (parent, child) of queuing policy. The parent level aggregates all the traffic classes and whereas the child level differentiates traffic classes using MQC classes.
- Only these actions are supported in the queuing policy:
  - priority
  - shape
  - bandwidth remaining ratio
  - queue-limit
  - Random Early Detection (RED)
  - Priority flow control
- You can have only one match traffic-class value in the class map.
- You cannot apply a queuing policy to a main interface and its sub-interfaces.