

Extreme SLX-OS QoS and Traffic Management Configuration Guide, 16r.1.01

Supporting the ExtremeRouting SLX 9850 Router

© 2018, Extreme Networks, Inc. All Rights Reserved.

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries. All other names are the property of their respective owners. For additional information on Extreme Networks Trademarks please see www.extremenetworks.com/company/legal/trademarks. Specifications and product availability are subject to change without notice.

© 2017, Brocade Communications Systems, Inc. All Rights Reserved.

Brocade, the B-wing symbol, and MyBrocade are registered trademarks of Brocade Communications Systems, Inc., in the United States and in other countries. Other brands, product names, or service names mentioned of Brocade Communications Systems, Inc. are listed at www.brocade.com/en/legal/brocade-Legal-intellectual-property/brocade-legal-trademarks.html. Other marks may belong to third parties.

Notice: This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any equipment, equipment feature, or service offered or to be offered by Brocade. Brocade reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact a Brocade sales office for information on feature and product availability. Export of technical data contained in this document may require an export license from the United States government.

The authors and Brocade Communications Systems, Inc. assume no liability or responsibility to any person or entity with respect to the accuracy of this document or any loss, cost, liability, or damages arising from the information contained herein or the computer programs that accompany it.

The product described by this document may contain open source software covered by the GNU General Public License or other open source license agreements. To find out which open source software is included in Brocade products, view the licensing terms applicable to the open source software, and obtain a copy of the programming source code, please visit <http://www.brocade.com/support/oscd>.

Contents

Preface.....	5
Document conventions.....	5
Notes, cautions, and warnings.....	5
Text formatting conventions.....	5
Command syntax conventions.....	6
Extreme resources.....	6
Document feedback.....	6
Contacting Extreme Technical Support.....	7
About This Document.....	9
Supported hardware and software.....	9
What's new in this document.....	9
Traffic Policing	11
Rate limiting and traffic policing overview.....	11
Service policies - policy maps, class maps, and policers	11
Policy maps.....	11
Class maps.....	12
Service policy configuration rules.....	12
Policy maps.....	12
Policy map configuration rules.....	12
QoS shaping rate.....	13
Committed information rate and committed burst size.....	13
Excess information rate and excess burst size.....	13
Traffic policing behavior.....	13
Traffic management egress buffer thresholds.....	14
Class maps.....	14
Class map configuration rules.....	15
Class map policer configuration parameters.....	15
Traffic policer configuration rules for class maps.....	16
Default class map traffic policing	16
Single rate three color marker.....	17
Implementation	18
Two rate three color marker.....	18
Implementation.....	19
Match access-group - class map policing	19
Match access-group - class map policing rules and limitations.....	20
Match access-group class map - DoS mitigation use cases.....	20
ACL-based rate limiting use cases.....	20
Use case 1 - protection against TCP SYN attacks.....	21
Use case 2 - protection against TCP RST attacks.....	21
Use case 3 - protection against ping flood attacks.....	21
Use case 4 - protection against UDP flood attacks.....	21
Configure all the use cases for ACL traffic filtering.....	22
Storm control - rate limiting broadcast, unknown unicast, and multicast traffic.....	22
Storm control considerations and limitations.....	22
Configuring traffic policing	23

Configuring a class map using an ACL	23
Configuring a policy map	24
Configuring port-based traffic policing.....	25
Configuring ACL-based rate limiting.....	25
Configuring storm control	38
Quality of Service.....	41
QoS overview.....	41
QoS features.....	41
IEEE 802.1q ToS-DSCP header fields.....	43
Congestion control.....	43
Scheduling.....	46
Configure flow-based QoS.....	51
QoS ingress data buffer management.....	53
QoS mutation overview.....	53
Multiprotocol Label Switching QoS overview.....	55
Configuring QoS.....	56
Configuring QoS for control traffic	56
Configuring CoS-to-traffic class mappings	56
Configuring DSCP mappings	58
Configuring DSCP-to-CoS mappings.....	60
Configuring DSCP-to-traffic class mappings.....	62
Concept.....	66
Configuring congestion control.....	68
Configuring scheduling.....	70
Configuring port-based QoS.....	71
Configuring virtual output queueing	79
Configuring MPLS QoS DSCP mutation maps.....	80
Configuring MPLS QoS EXP mutation maps.....	82
Configuring MPLS QoS traffic class mutation maps.....	85
Traffic Management Counters and Statistics	89
Counters and statistics overview	89
Statistics collection mechanisms.....	89
Traffic management counter types.....	89
TM device counters.....	89
TM VOQ counter.....	90
Traffic management counters.....	90
TM global statistics command.....	90
TM device level statistics commands.....	90
TM VOQ commands.....	91

Preface

- Document conventions..... 5
- Extreme resources..... 6
- Document feedback..... 6
- Contacting Extreme Technical Support..... 7

Document conventions


The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Extreme technical documentation.


Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

 **CAUTION**
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

 **DANGER**
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
<i>italic text</i>	Identifies text to enter in the GUI.
	Identifies emphasis.
	Identifies variables.
Courier font	Identifies document titles.
	Identifies CLI output.

Format**Description**

Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention**Description**

bold text

Identifies command names, keywords, and command options.

italic text

Identifies a variable.

[]

Syntax components displayed within square brackets are optional.

{ x | y | z }

Default responses to system prompts are enclosed in square brackets.

A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.

x | y

A vertical bar separates mutually exclusive elements.

< >

Nonprinting characters, for example, passwords, are enclosed in angle brackets.

...

Repeat the previous element, for example, *member[member...]*.

\

Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Extreme resources

Visit the Extreme website to locate related documentation for your product and additional Extreme resources.

White papers, data sheets, and the most recent versions of Extreme software and hardware manuals are available at www.extremenetworks.com. Product documentation for all supported releases is available to registered users at www.extremenetworks.com/support/documentation.

Document feedback

Quality is our first concern at Extreme, and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you.

You can provide feedback in two ways:

- Use our short online feedback form at <http://www.extremenetworks.com/documentation-feedback-pdf/>
- Email us at internalinfodev@extremenetworks.com

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Contacting Extreme Technical Support

As an Extreme customer, you can contact Extreme Technical Support using one of the following methods: 24x7 online or by telephone. OEM customers should contact their OEM/solution provider.

If you require assistance, contact Extreme Networks using one of the following methods:

- [GTAC \(Global Technical Assistance Center\)](#) for immediate support
 - Phone: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact.
 - Email: support@extremenetworks.com. To expedite your message, enter the product name or model number in the subject line.
- [GTAC Knowledge](#) - Get on-demand and tested resolutions from the GTAC Knowledgebase, or create a help case if you need more guidance.
- [The Hub](#) - A forum for Extreme customers to connect with one another, get questions answered, share ideas and feedback, and get problems solved. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
- [Support Portal](#) - Manage cases, downloads, service contracts, product licensing, and training and certifications.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

About This Document

- Supported hardware and software.....9
- What's new in this document.....9

Supported hardware and software

In those instances in which procedures or parts of procedures documented here apply to some devices but not to others, this guide identifies exactly which devices are supported and which are not.

Although many different software and hardware configurations are tested and supported by Extreme Networks, Inc. for SLX-OS Release 16r.1.01, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release:

- ExtremeRouting SLX 9850-4
- ExtremeRouting SLX 9850-8

What's new in this document

This is a rebranding release.

On October 30, 2017, Extreme Networks, Inc. acquired the data center networking business from Brocade Communications Systems, Inc. This document has been updated to remove or replace references to Brocade Communications, Inc. with Extreme Networks, Inc., as appropriate.

Traffic Policing

• Rate limiting and traffic policing overview.....	11
• Service policies – policy maps, class maps, and policers	11
• Policy maps.....	12
• Class maps.....	14
• Single rate three color marker.....	17
• Two rate three color marker.....	18
• Match access-group – class map policing	19
• ACL-based rate limiting use cases.....	20
• Storm control – rate limiting broadcast, unknown unicast, and multicast traffic.....	22
• Configuring traffic policing	23

Rate limiting and traffic policing overview

The purpose of rate limiting is to control the amount of bandwidth consumed by an individual flow, or an aggregate of flows.

Rate limiting is applicable to both inbound and outbound traffic, where packets are dropped that are above the configured committed rates. Rate shaping controls traffic bursts applicable to egress traffic by buffering and queueing of excess packets above the committed rate.

Traffic policing refers to class based rate limiters applicable to ingress and egress traffic.

Rate limiting on the SLX platform implements the single rate three color mechanism based on RFC 2697 and two rate three color mechanism based on RFC 4115.

Service policies – policy maps, class maps, and policers

Traffic policing is accomplished by applying a service policy to an interface.

A service policy consists of a policy map that specifies traffic policing and QoS parameters based on matching associated class maps.

One service policy can be applied per interface per direction.

Policy maps

A policy map includes a set of class maps and policer values for the classified traffic. The policy map configuration allows you to specify policers in a single location that can be applied to multiple ports and to make changes to that policy.

When using the traffic policing policies available from previous versions, the policy parameters are provided explicitly for each port during port configuration. In this version, the policies must be defined using a policy map.

You can configure up to 1024 policy maps per system, but one policy map can be specified per service policy.

See the [Policy maps](#) on page 12 section for more information.

Class maps

A class map is used to determine the traffic properties subject to policing. The port-based rate limit, applied using the default class map, is applicable to all traffic and may be used for ingress and egress service policies. An IP standard or extended ACL are also supported to classify traffic for ingress only service policies.

The default and **match access-group** class maps may not be combined in a single policy map.

See the [Class maps](#) on page 14 section for more information.

Service policy configuration rules

You must follow these binding rules when configuring or deleting a service policy:

- All policy map and class map names used in a service policy must be unique among all maps of that type.
- You can bind a service policy to multiple interfaces.
- A service policy can only be bound to physical ports and port-channel interfaces (LAGs). It cannot be bound to virtual interfaces.
- A service policy cannot be bound to interface if a class map is not associated with the policy map referenced in the service policy.
- If a service policy is bound to interface, and the policy class map lacks mandatory policer attributes (CIR), then the traffic on that interface is treated as conformed traffic. The packets on that interface are marked as green, no meter is allocated and no statistics are available.
- Broadcast, unknown unicast and multicast (BUM) storm control and service policy port based rate limiting are mutually exclusive features. Only one can be enabled at a time on a given interface.

Policy maps

Policy maps allow you to set a policy in a single location that affects multiple ports and to make changes to that policy.

The policy map configuration includes a set of class maps and QoS parameters.

A policy map allows you to specify policers in a single location that can be applied to multiple ports and to make changes to that policy.

When using the traffic policing policies available from previous versions, the policy parameters are provided explicitly for each port during port configuration. In this version, the policies must be defined using a policy map. One policy map can be specified per service-policy. You can configure up to 1024 policy maps per system.

Policy map configuration rules

Follow these rules when configuring traffic policing:

- A policer map (policy map or class map) name must be unique among all maps of that type.
- A policer name must begin with a-z or A-Z . An underscore, hyphen, and numeric values 0-9 can be used in the body of the name but not as the first character.
- You can configure a maximum of 1024 policy maps.
- The default and **match access-group** class maps cannot be combined in a single policy map.
- For an ingress or egress service policy, one default class map can be specified per policy map.
- For an ingress only service policy, 16 **match access-group** class maps are supported per policy map.
- Broadcast, unknown unicast and multicast (BUM) policies are counted separately.

- You cannot delete a policy map if it is referenced in an active service policy (applied on an interface).

QoS shaping rate

You can specify the shaping rate per port attached to the policy map to smooth the traffic that egresses an interface. This configuration is allowed only for egress traffic.

Committed information rate and committed burst size

The committed information rate (CIR) bucket is defined by two separate parameters: the CIR rate, and the committed burst size (CBS) rate.

The CIR is the maximum number of bits a port is allowed to receive or send during a one-second interval. The rate of the traffic that matches the traffic policing policy cannot exceed the CIR. The CIR represents a portion of an interface's line rate (bandwidth), expressed in bits per second (bps) and it cannot be larger than the port's line rate. CIR-defined traffic that does not use the CIR available to it accumulates credits until the credit reaches to CBS, these credits can be used later in circumstances where it temporarily exceeds the CIR.

When traffic exceeds the bandwidth that has been reserved for it by the CIR defined in its policy, it becomes subject to the CBS rate. The CBS rate provides a rate higher than the CIR to traffic that exceeded the CIR. The bandwidth in the CBS rate, as expressed in bytes, is accumulated during periods when policy-defined traffic does not use the full CIR available to it. Traffic is allowed to pass through the port for a short period of time at the CBS rate.

The traffic rate limited by the CIR bucket can have its priority, traffic class, and DSCP values changed.

Excess information rate and excess burst size

When inbound or outbound traffic exceeds the bandwidth available for the defined CIR and CBS, it is either dropped, or made subject to the conditions set in the excess information rate (EIR) and excess burst size (EBS).

The EIR bucket provides an option for traffic that has exceeded the conditions set by policy for the CIR bucket. The EIR and EBS operate exactly like the CIR and CBS except that they only act upon traffic that has been passed to the EIR bucket because it could not be accommodated by the CIR bucket. Like the CIR, the EIR provides an initial bandwidth allocation to accommodate inbound and outbound traffic. Like the CBS, the bandwidth available for burst traffic from the EBS is subject to the amount of bandwidth that is accumulated during periods of time when traffic that has been allocated by the EIR policy is not used. When inbound or outbound traffic exceeds the bandwidth available (the accumulated credits or tokens), it is dropped.

The traffic rate limited by the EIR bucket can have its priority, traffic class, and DSCP values changed.

Traffic policing behavior

Take note of these behaviors when configuring traffic policing.

- Policer actions are applicable only to data traffic.
- When a Layer 2 control protocol is not enabled on an interface, those packets are dropped at ingress and are subject to ingress policing.
- Layer 3 control packets are policed at egress.
- If user configured CBS value is less than $2 \times \text{MTU}$ value then $2 \times \text{MTU}$ will be programmed as CBS on hardware. For example if you configure CBS as 4000 Bytes and MTU on an interface is 3000 Bytes. When a policy map is applied on this interface, the CBS value that is programmed on hardware is $2 \times \text{MTU} = 6000$ Bytes)

- If the CBS and EBS values are not configured, then these values are derived from the CIR and EIR respectively. The burst size calculation is: Burst size (CBS or EBS) = (1.2 × information rate (CIR or EIR)) ÷ 8
- If you do not configure EIR and EBS then single rate two color scheme is applied (packets are marked as either green or red).
- You have the responsibility to configure rate limit threshold values on an interface based on interface speed. No validation is performed for user configured values against the interface speed.
- Since CIR is mandatory policer attribute, you cannot delete the CIR parameter. If you want to delete the CIR, then you should execute the **no police** command in policy-map-class sub-mode, which deletes all policer attributes.
- Packet drops caused by any action other than the ACL are included in the policer counter.
- Layer 3 control packets get policed at the egress side.

Traffic management egress buffer thresholds

During the traffic management (TM) initialization process, TM egress buffer thresholds are configured and set to the values listed in the following tables.

The following table shows the egress thresholds for unicast and multicast traffic that are configured during the TM initialization process at the device level.

TABLE 1 TM egress thresholds on the device

Threshold	Packet descriptor (PD) flow control (FC)	PD Drop	DB (Data buffers) FC	DB Drop
Unicast	6100	6000	6100	6000
Multicast		26000		6000

The following table shows the egress thresholds for unicast and multicast traffic that are configured during the TM initialization process at port level on the device.

TABLE 2 Local port TM egress thresholds on the device

Threshold	PD FC	PD Drop	DB FC	DB Drop
Unicast priority low	1024	4000	84	6000
Unicast priority high	1024	4000	84	6000
Unicast port	167	6000	167	6000
Multicast priority low		722		7220
Multicast priority high		722		7220
Multicast port		135		1350

NOTE

The burst size on special CPU ports (202 and 203) is set to 600.

Class maps

A class map is used to determine the traffic properties subject to policing.

An ACL can be used for match criteria while port-based policing is only implemented to match any criteria. Class maps can be used in a policy map to apply policing and QoS policies to a particular class. You can also define a default class map that matches any criteria.

The default class map, is applicable to all traffic and may be used for ingress and egress service policies. In addition an IP standard or extended ACL are supported to classify traffic for ingress only service policies.

Class map configuration rules

Follow these rules when configuring class maps:

- A class map name must be unique among all maps of this type.
- A class-map name must begin with a-z or A-Z .
- An underscore, hyphen, and numeric values 0-9 can be used in the body of the name but not as the first character.
- A class map cannot be deleted if it is referenced in a policy map.
- A class map cannot be deleted from a policy map when the policy map is bound to an active service policy.
- You can configure a maximum of 32K class maps.
- The default and **match access-group** class maps cannot be combined in a single policy map.
- One default class map can be specified per policy map.

Class map policer configuration parameters

Use these values when setting the CIR, CBS, EIR, and EBS parameters.

TABLE 3 Map parameters for rate limiting

Parameter	Values	Range	Increments of
cir - Committed information rate	bits per second	18000 through 300000000000	Starts at 18000 then is rounded up to next achievable rate.
cbs - Committed burst size	Bytes per second	1250 through 37500000000	1 Byte
eir - Excess information rate	bits per second	18000 through 300000000000	Starts at 18000 then is rounded up to next achievable rate.
ebs - Excess burst size	Bytes per second	1250 through 37500000000	1 Byte

NOTE

The parameters cir and eir are configured in bits per second, cbs and ebs are configured in Bytes per second.

The possible combinations when entering policer values are:

```
device(config-policymap-class)# police cir 600000000
device(config-policymap-class)# police cir 700000000 cbs 8000000000
device(config-policymap-class)# police cir 7000000000 cbs 70000000 eir 500000000
device(config-policymap-class)# police cir 7000000000 cbs 70000000 eir 500000000 ebs 90000000
device(config-policymap-class)# police cir 700000 eir 800000
device(config-policymap-class)# police cir 700000 eir 800000 ebs 6000000
```

Follow these rules when configuring the parameters:

- The cir value must be specified, all other parameters are optional.
- Default values will be calculated if not specified by the user.
- Configured values take priority over default values.
- If you only specify the cir value, a default value is calculated and set to cbs.
- If you specify the values of both cir and cbs, in police the configured value takes priority over the default values.

- Should the cir value be updated, the configured cbs value is retained, the default value is not restored.
- If you want to revert to the default cbs value, you must first remove the configured value of cbs.

Traffic policer configuration rules for class maps

The following are rules for configuring traffic policing for classified traffic in a policy map.

- A service policy map or class map name must be unique among all maps of that type.
- You cannot delete a service policy map or class map if it is active on an interface.
- Operational values that are programmed in the hardware are displayed as part of show policy-map interface ethernet slot/port command.
- A policer name must begin with a to z, or A to Z. Underscore, hyphen, and numeric values 0-9 are permitted, except as the first character of the name.
- The configurable CIR and EIR range starts from 18000 bits per second (bps) and are rounded up to the next achievable rate.
- Percentage values are not supported as a policer parameter.
- Policer actions are not supported.
- If a service policy map is applied to an interface and no policer attributes are present in that service policy map, then ingress and egress packets on that interface are marked as green (conforming).
- If the configured CBS value is less than $2 \times \text{MTU}$ value, then $2 \times \text{MTU}$ is programmed as the CBS in the hardware. For example, if you configure CBS at 4000 bytes and the MTU on an interface is 3000 bytes, when a service policy map is applied on this interface, the CBS programmed in the hardware is $2 \times \text{MTU} = 6000$ bytes.
- If CBS and EBS values are not configured, then these values are derived from CIR and EIR values, respectively. Burst size calculation is as follows: Burst size (CBS or EBS) = $(1.2 \times \text{information rate (CIR or EIR)}) \div 8$.
- If you do not configure EIR and EBS, then the single rate, two color scheme is applied. Packets are marked as either green or red.
- You must configure rate limit threshold values on an interface based on interface speed.
- No validation is performed for user-configured values against interface speed.
- You can configure up to 1024 service policy maps. Broadcast, unknown unicast, unknown multicast policy are counted separately.

Default class map traffic policing

The default class map (port-based) policer feature controls the amount of bandwidth consumed by an individual flow or aggregate of flows.

The default class map is a port-based policer feature that controls the inbound (ingress) and outbound (egress) traffic rate on an individual port according to criteria that you define.

Default class map traffic policing considerations and limitations

- You may configure up to 1024 policy maps.
- You may configure one class-maps per policy-map.
- You may configure up to 32656 policers (ingress - egress).
- Traffic filtered by an ACL is not subject to default service policy policer.
- Match default class map service policy is supported on ingress and egress interfaces.

- A class based service policy and storm control policy (BUM) are mutually exclusive applications. Only one can be enabled at a time on a given interface.
- Control protocols are not rate-limited by the default class map service policy.
- The default class service policy does support remarking or internal queue assignment.
- Metering is performed on the packet as received on the wire. For example, including IPG and preamble, excluding CRC.

Single rate three color marker

Single rate three color marker (SrTCM) meters an IP packet stream and marks its packets either green, yellow, or red.

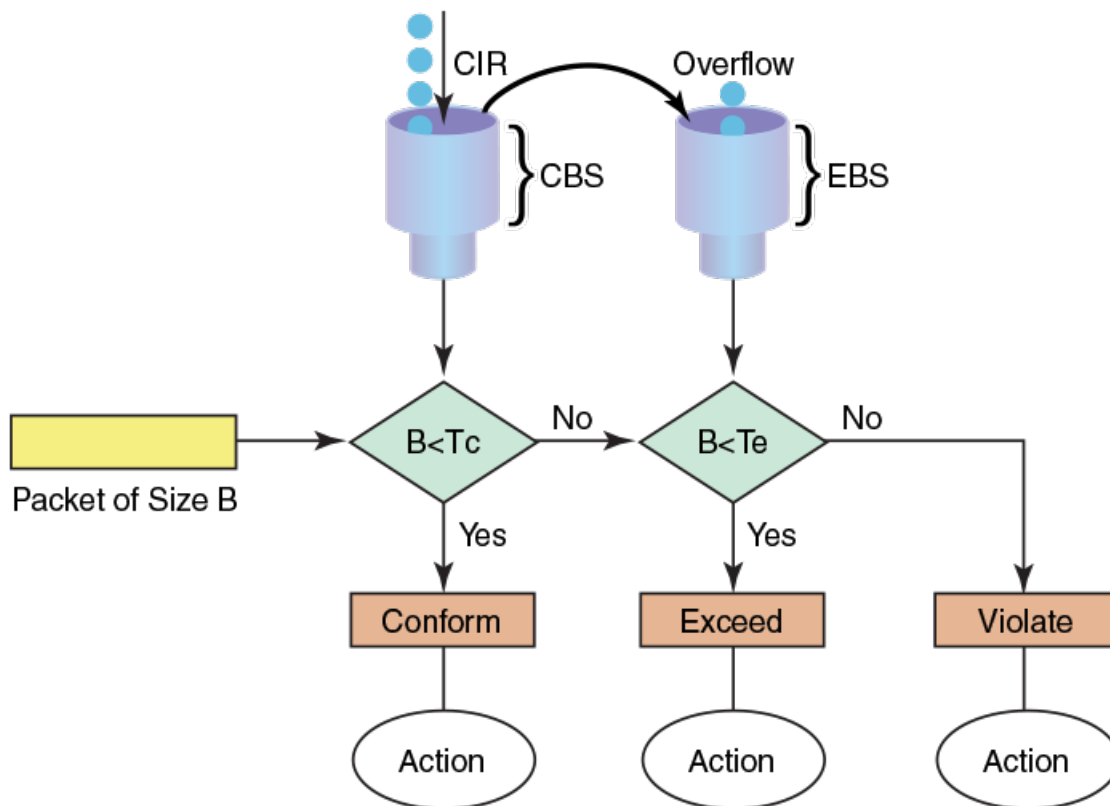
Single-rate traffic contract has three parameters, the, CIR, CBS and EBS. associated with this type of contract:

Marking is based on CIR and two associated burst sizes, CBS and EBS. Packets are marked:

- Green - if it does not exceed the CBS
- Yellow - if it does exceed the CBS, but not the EBS
- Red - otherwise

The SrTCM is useful for ingress policing of a service, where only the length, not the peak rate, of the burst determines service eligibility.

FIGURE 1 Single rate three color marker



Implementation

SrTCM traffic policing is implemented by tracking the current burst size using token-buckets, and discarding packets that exceed the CIR. In SrTCM the three color scheme has any incoming burst classified as either conforming (green, under CBS), exceeding (yellow, over CBS but under EBS), or violating (red, over EBS).

Every arriving packet is first compared to CBS and then to the EBS to determine the next action. There is a single flow of the tokens that fills the CBS bucket first and then continues to filling the EBS bucket. The second bucket is only filled if there was enough idle time to let the first bucket fill up completely.

The drawback of single-rate traffic contracts is that the service provider should be cautious assigning CIR bandwidth, by offering less bandwidth than it can service at any moment. The reason for this is that not all customers send traffic simultaneously, so network links may effectively become underutilized even at weak spots.

Two rate three color marker

Two rate three color marker (TrTCM) meters an IP packet stream and marks its packets either green, yellow, or red.

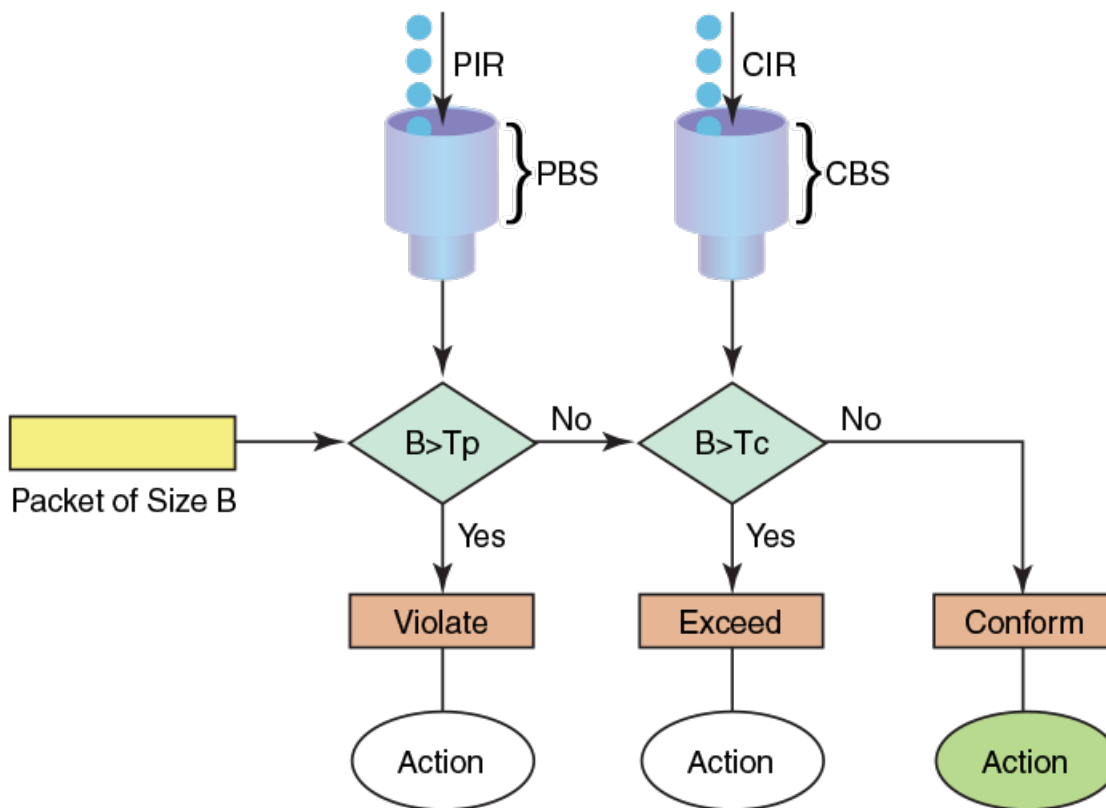
There are four main parameters in a dual-rate traffic contract. CIR, CBS, PIR, and EBS.

Marking is based on CIR. Packets are marked:

- Green - if it does not exceed the CIR
- Yellow - if it exceeds the CIR
- Red - if it exceeds the peak information rate (PIR)

The TrTCM is useful for ingress policing of a service, where a peak rate needs to be enforced separately from a committed rate.

FIGURE 2 Two rate three color marker



Implementation

Dual-rate traffic contract: supplies customers with two sending rates, but only guarantee the smaller one. In case of congestion in the network, discard traffic that exceeds the committed rate more aggressively and signal the customer to slow down to the committed rate. This principle was first widely implemented in Frame-Relay networks, but could be easily replicated using any packet-switching technology. There are four main parameters in a dual-rate traffic contract.

Match access-group - class map policing

Access groups are used for Layer 2 and Layer 3 ACL-based ingress rate limit and denial of service (DoS) mitigation.

ACL-based rate limiting is built on top of ACL and policer features, it rate limits the Layer 3 traffic that matches the permit conditions specified in an IPv4 access list. The ACL-based policer feature controls the amount of bandwidth consumed by an individual flow or aggregate of inbound flows by limiting the traffic rate on an individual port according to criteria defined by the **match access-group** class map. This ACL-based rate limiting feature can serve as a hardware solution to prevent DoS attacks.

Match access-group - class map policing rules and limitations

Consider these rules and limitations when you are configuring **match access-group** class map policing:

- You can configure:
 - 1024 policy maps
 - 16 class maps
 - 6144 ACL Content Addressable Memory (CAM) entries for use with rate limiting
 - 16356 ingress policers
- For protection against:
 - PING attacks
 - TCP Reset attacks
 - TCP SYN attacks
 - UDP attack
- Only Layer 3 IPv4 ACL-based rate limiting is supported.
- ACL-based rate limiting is applicable only to ingress traffic.
- There is one policer per ACL, it applies to all the rules for that ACL
- Control protocols are rate-limited if they match the configured ACL clause.
- When a **match access-group** class map rate limit is applied to a LAG logical port, and all LAG ports belong to the same tower, then MAX CIR value is the interface speed × number of physical ports. For example: if 1/1, 1/2 are LAG member ports, then MAX CIR will be 2 × 10Gbps.
- When a **match access-group** class map rate limit is applied to LAG logical port, MAX rate on that port is the number of the tower in that LAG × CIR. For example: if 1/1, 1/2, 2/1, 3/1 are LAG member ports, then MAX rate is 3 × CIR.

Match access-group class map - DoS mitigation use cases

The **match access-group** ACL may be configured for protection against these use cases:

[Use case 1 - protection against TCP SYN attacks](#) on page 21 and [Configuring use case 1 - protection against TCP SYN attacks](#) on page 26.

[Use case 2 - protection against TCP RST attacks](#) on page 21 and [Configuring use case 2 - protection against TCP RST attacks](#) on page 28.

[Use case 3 - protection against ping flood attacks](#) on page 21 and [Configuring use case 3 - protection against ping flood attacks](#) on page 30.

[Use case 4 - protection against UDP flood attacks](#) on page 21 and [Configuring use case 4 - protection against UDP flood attacks](#) on page 33.

ACL-based rate limiting use cases

The following describes 4 common DoS attacks and how to protect against them using ACL based rate limiting.

See the topics:

[Use case 1 - protection against TCP SYN attacks](#) on page 21 and [Configuring use case 1 - protection against TCP SYN attacks](#) on page 26.

[Use case 2 - protection against TCP RST attacks](#) on page 21 and [Configuring use case 2 - protection against TCP RST attacks](#) on page 28.

[Use case 3 - protection against ping flood attacks](#) on page 21 and [Configuring use case 3 - protection against ping flood attacks](#) on page 30.

[Use case 4 - protection against UDP flood attacks](#) on page 21 and [Configuring use case 4 - protection against UDP flood attacks](#) on page 33.

Use case 1 - protection against TCP SYN attacks

A TCP SYN attack, also known as a SYN flood, is a form of denial-of-service (DoS) attack where an attacker sends a series of SYN requests to a system in an attempt to consume enough server resources so that the system is unresponsive to other traffic.

TCP SYN attacks disrupt normal traffic by exploiting the way TCP connections are established. These attacks attempt to exhaust the target system's half open TCP queue, which is a limited resource to service new connection requests. The attacker creates a random source address for each packet and a SYN flag is set in each packet to request to open a new connection. The TCP IP stack of the victim responds to the spoofed IP with SYN ACK and waits for a return ACK from the sender which never comes.

See the topic, [Configuring use case 1 - protection against TCP SYN attacks](#) on page 26.

Use case 2 - protection against TCP RST attacks

A TCP RST (reset) attack is meant to abnormally terminate legitimate TCP connections by sending a random packet with the RST bit set.

In the packet stream of a TCP connection, each packet contains a TCP header and every header contains an RST bit. If this bit is set to 1, it instructs the receiving computer to immediately terminate the TCP connection. Following this instruction, the sending computer does not forward any more packets through the connection's ports, and discards any further packets it receives with headers indicating they should be sent to that connection.

A TCP reset terminates a TCP connection instantly.

See the topic, [Configuring use case 2 - protection against TCP RST attacks](#) on page 28.

Use case 3 - protection against ping flood attacks

A ping flood is a DoS attack that is based on sending the targeted system an overwhelming number of ICMP Echo Request (ping) packets.

The attack uses the ping flood option, which sends ICMP packets as fast as possible without waiting for replies. In a successful attack, the target system responds to the ping requests with ICMP Echo Reply packets, consuming both outgoing bandwidth as well as incoming bandwidth. If the target system is slow enough, it is possible to consume enough of its CPU cycles for a user to notice a significant slowdown.

See the topic, [Configuring use case 3 - protection against ping flood attacks](#) on page 30.

Use case 4 - protection against UDP flood attacks

A User Datagram Protocol (UDP) flood is a brute force DoS attack where a large number of UDP packets are sent by the attacker to random ports on a remote host.

In a UDP attack, the targeted system is forced to reply to the UDP packets with ICMP Destination Unreachable packets, eventually leading the target system becomes unreachable to other clients. The targeted system responds to a UDP flood by:

Checking for the application listening at that port > Seeing that no application listens at that port > Replies with an iCMP Destination Unreachable packet

The attacker may also spoof the IP address of the UDP packets, ensuring that the excessive ICMP return packets do not reach them, and anonymize their network location.

See the topic, [Configuring use case 4 - protection against UDP flood attacks](#) on page 33.

Configure all the use cases for ACL traffic filtering

You can configure all four use cases and apply them to an ingress port by following these high level steps.

1. Create an ACL, with criteria that matches the potential attack
 - A standard ACL table provides option to filter only based on source address information.
 - An extended ACL table provides option to filter based on most of the fields in the packet header
2. Create a Class Map, associate to that ACL
3. Create a Policy Map using the Class Map created in step 2, and assign a Policier.
4. Associate that Policy Map to an ingress port.

See the topic, [Configuring and applying all four use cases for ACL-based traffic filtering](#) on page 35.

Storm control - rate limiting broadcast, unknown unicast, and multicast traffic

A broadcast, unknown unicast, and multicast (BUM) traffic storm occurs when packets flood the LAN, creating excessive traffic and degrading network performance.

BUM storm control can prevent disruptions on Layer 2 physical ports. This feature is supported only at the interface level.

BUM storm control allows you to limit the amount of broadcast, unknown unicast, and multicast ingress traffic on a specified interface. All traffic received in excess of the configured rate is discarded. You also have the option to specify whether to shut down an interface if the maximum defined rate is exceeded within a ten second sampling period. When a port is shut down, you receive a log message.

Storm control considerations and limitations

- BUM storm control applies only to ingress traffic.
- BUM can only be configured on physical interfaces.
- BUM storm control and input service policy are mutually exclusive features. Only one can be enabled at a time on a given interface.
- For LAG ports, BUM rate limiting needs to be enabled on each LAG member port.
- A single rate two color marking scheme is used.
- Metering is performed on the packet size as received on the wire (including IPG and preamble), ignoring CRC.
- If BUM traffic is also classified by an ACL, then BUM rate limiting is not effective
- The configured rate in bits per second (bps) is rounded up to next achievable rate.
- Only FWD and DROP counters are supported:
 - Conformed - Shows FWD packets including green and yellow color packets.
 - Violated - Shows DROP packets including red color packets.
 - Exceed - is always ZERO.

- FWD and DROP counters must use a counter profile other than default.

Configuring traffic policing

Follow these tasks to configure traffic policing.

Configuring a class map using an ACL

To configure a classification or class map by using an ACL, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an IP access list to define the traffic.

- a) Create and name a standard IP access list and enter IP ACL configuration mode.

```
device(config)# ip access-list standard ip_acl
```

- b) Allow traffic from a specific IP address.

```
device(conf-ipacl-std)# permit host 10.10.10.0
```

- c) Exit IP ACL configuration mode to global configuration mode.

```
device(conf-ipacl-std)# exit
```

Refer to the *Brocade SLX-OS Security Configuration Guide* for details on creating access lists.

3. Verify the IP ACL.

```
device(config)# do show running-config | include ip_acl
ip access-list standard ip_acl
```

4. Create and name a class map.

```
device(config)# class-map class_1
```

5. Provide match criteria for the class.

```
device(config-classmap)# match access-group ip_acl
```

6. Return to privileged exec mode.

```
device(config-classmap)# end
```

7. Verify the class configuration.

```
device# show running-config | include class
...
class-map cee
class-map class_1
class-map default
```

8. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Class map using an ACL configuration example

```
device# configure terminal
device(config)# ip access-list standard IP_acl
device(conf-ipacl-std)# permit host 10.10.10.0
device(conf-ipacl-std)# exit
device(config)# do show running-config | include ip_acl
device(config)# class-map class_1
device(config-classmap)# match access-group ip_acl
device(config-classmap)# end
device# show running-config | include class
device# copy running-config startup-config
```

Configuring a policy map

Add a class map to a policy map and set policing parameters to the class map.

See the topic, Policy maps for policing parameter ranges.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name a policy map.

```
device(config)# policy-map policy_2
```

3. Add a class map to the policy map.

```
device(config-policymap)# class default
```

4. Create a policy map class police instance and set the committed information rate (cir), committed burst rate (cbs), excess information rate (eir), and the excess burst rate (ebs).

```
device(config-policymap-class)# police cir 3000000 cbs 375000000 eir 300000000 ebs 37500000
```

5. Return to privileged exec mode.

```
device(config-policymap-class)# end
```

6. Verify the configuration.

```
device# show policy-map detail policy_2

Policy-Map policy_2
  Class default
    Police cir 3000000 cbs 375000000 eir 300000000 ebs 37500000

  Bound To:None
```

7. Save the configuration.

```
device# copy running-config startup-config
```


Policy map configuration example

```
device# configure terminal
device(config)# policy-map policy_2
device(config-policy-map)# class default
device(config-policy-map-class)# police cir 3000000 cbs 375000000 eir 300000000 ebs 37500000
device(config-policy-map-class)# set-priority priorityMap
device(config-policy-map-class)# end
device# show policy-map detail policy_2
device# copy running-config startup-config
```

Configuring port-based traffic policing

Follow these steps to associate the policy map with the Interface. By associating the policy map, the policing parameters are applied to the port.

An ingress or egress policy map has been created and populated with policing parameters.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode

```
device(config)# interface ethernet 2/2
```

3. Attach an input or output policy map.

```
device(config-if-eth-2/2)# service-policy out policy_2
```

For rate limiting ingress traffic just replace the **out** keyword with **in** and use an ingress policy map.

4. Return to privileged exec mode.

```
device(config-if-eth-2/2)# end
```

5. Verify the configuration.

```
device# show policy-map interface ethernet 2/2 out

Egress Direction :
Policy-Map policy_2
Class default
Police cir 4000000 cbs 50000 eir 800000 ebs 400000
Stats:
Operational cir:4006912 cbs:50000 eir:0 ebs:400000
Conform Byte:0 Exceed Byte:0 Violate Byte:0
```

Port-based traffic policing configuration example

```
device# configure terminal
device(config)# interface ethernet 2/2
device(config-if-eth-2/2)# service-policy out policy_2
device(config-if-eth-2/2)# end
device# show policy-map interface ethernet 2/2 out
```

Configuring ACL-based rate limiting

Configuring use case 1 - protection against TCP SYN attacks

Follow these steps to configure an ACL that can be used to protect against TCP SYN DoS attacks.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an extended IP ACL.

```
device(config)# ip access-list extended acl1
2015/04/01-13:18:15, [SSMD-1400], 2315, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 is created.
```

The system message is generated when you create an ACL. If you are configuring an existing ACL, no message is generated.

3. Configure the extended ACL to permit TCP traffic from any source to any destination while filtering packets for which the **sync** (synchronize) flag is set.

```
device(conf-ipacl-ext)# permit tcp any any sync
2015/04/01-13:22:16, [SSMD-1404], 2316, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 10 is added.
```

4. Return to privileged exec mode.

```
device(conf-ipacl-ext)# end
```

5. Verify the ACL.

```
device# show running-config ip access-list extended acl1
ip access-list extended acl1
seq 10 permit tcp any any sync
```

Protection against TCP SYN attacks - ACL configuration example

```
device# configure terminal
device(config)# ip access-list extended acl1
device(conf-ipacl-ext)# permit tcp any any sync
device(conf-ipacl-ext)# end
device# show running-config ip access-list extended acl1
```

Configuring use case 1 - bind the TCP SYN ACL to an interface

To protect against TCP SYN DoS attacks, bind ACL-based protection against TCP SYN attacks to an interface.

You have configured an extended Layer 3 ACL-based rate limit matching TCP SYN.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a class map.

```
device(config)# class-map aclFilter
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

3. While in class map mode associate the class map with an ACL.

```
device(config)# match access-group acl1
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

4. Return to privileged exec mode.

```
device(config-classmap)# end
```

5. Verify the class map to ACL association.

```
device# show running-config class-map aclFilter
class-map aclFilter
  match access-group acl1
!
```

6. Create a policy map with a policer.

```
device(config)# policy-map policyAclFilter
```

A policy map is used to apply policer and QoS attributes to a particular interface.

7. Associate a class map with the policy map.

```
device(config-policymap)# class aclFilter
```

Each policy map can have different class maps. Each class map in the policy map can be associated to separate policing and QoS parameters.

8. Populate the class map policer parameters.

```
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
```

CIR and EIR are in increments of 18000 bps.

9. Return to privileged exec mode.

```
device(config-policymap-class-police)# end
```

10. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000

    Bound To:None
```

11. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

12. Bind the policy map to the port.

```
device(conf-if-eth-1/2)# service-policy in policyAclFilter
2015/04/02-14:13:31, [SSMD-1405], 2511, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 configured on interface Ethernet 1/2 at Ingress by FbQos_9_11.
```

13. Return to privileged exec mode.

```
device(conf-if-eth-1/2)# end
```

14. Verify the configuration.

```
device# show policy-map detail policyAclFilter
Policy-Map policyAclFilter
Class aclFilter
Police cir 180000 cbs 50000 eir 36000 ebs 400000
Bound To: Et 1/2(in)
```

15. Save the configuration.

```
device# copy running-config startup-config
```

ACL-based protection against TCP SYN attacks applied to an interface configuration example

```
device# configure terminal
device(config)# class-map aclFilter
device(config)# match access-group acl1
device(config-classmap)# end
device# show running-config class-map aclFilter
device(config)# policy-map policyAclFilter
device(config-policymap)# class aclFilter
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
device(config-policymap-class-police)# end
device# show policy-map detail policyAclFilter
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in policyAclFilter
device(conf-if-eth-1/2)# end
device# show policy-map detail policyAclFilter
device# copy running-config startup-config
```

Configuring use case 2 - protection against TCP RST attacks

Follow these steps to configure an ACL that can be used to protect against TCP RST DoS attacks.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create or invoke an extended IP ACL.

```
device(config)# ip access-list extended acl1
2015/04/01-13:18:15, [SSMD-1400], 2315, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 is created.
```

The system message is generated when you create an ACL. If you are configuring an existing ACL, no message is generated.

3. Configure the extended ACL to permit TCP traffic from any source to any destination while filtering packets for which the **rst** flag is set.

```
device(conf-ipacl-ext)# permit tcp any any rst
2015/04/01-13:22:16, [SSMD-1404], 2316, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 10 is added.
```

4. Return to privileged exec mode.

```
device(conf-ipacl-ext)# end
```

5. Verify the ACL.

```
device# show running-config ip access-list extended acl1
ip access-list extended acl1
seq 10 permit tcp any any rst
```

Protection against TCP RST attacks - ACL configuration example

```
device# configure terminal
device(config)# ip access-list extended acl1
device(conf-ipacl-ext)# permit tcp any any rst
device(conf-ipacl-ext)# exit
device# show running-config ip access-list extended acl1
```

Configuring use case 2 - bind the TCP RST ACL to an interface

To protect against TCP RST DoS attacks, bind an extended Layer 3 ACL based rate limit matching TCP RST to an interface.

A TCP RST matching ACL has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a class map.

```
device(config)# class-map aclFilter
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

3. While in class map mode associate the class map with an ACL.

```
device(config)# match access-group acl1
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

4. Return to privileged exec mode.

```
device(config-classmap)# end
```

5. Verify the class map to ACL association.

```
device# show running-config class-map aclFilter
class-map aclFilter
  match access-group acl1
!
```

6. Create a policy map with a policer.

```
device(config)# policy-map policyAclFilter
```

A policy map is used to apply policer and QoS attributes to a particular interface.

7. Associate a class map with the policy map.

```
device(config-policymap)# class aclFilter
```

Each policy map can have different class maps. Each class map in the policy map can be associated to separate policing and QoS parameters.

8. Populate the class map policer parameters.

```
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
```

CIR and EIR are in increments of 18000 bps.

9. Return to privileged exec mode.

```
device(config-policymap-class-police)# end
```

10. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000

  Bound To:None
```

11. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

12. Bind the policy map to the port.

```
device(conf-if-eth-1/2)# service-policy in policyAclFilter
2015/04/02-14:13:31, [SSMD-1405], 2511, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 configured on interface Ethernet 1/2 at Ingress by FbQos_9_11.
```

13. Return to privileged exec mode.

```
device(conf-if-eth-1/2)# end
```

14. Verify the configuration.

```
device# show policy-map detail policyAclFilter
Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000
  Bound To: Et 1/2(in)
```

15. Save the configuration.

```
device# copy running-config startup-config
```

ACL-based protection against TCP RST attacks applied to an interface configuration example

```
device# configure terminal
device(config)# class-map aclFilter
device(config)# match access-group acl1
device(config-classmap)# end
device# show running-config class-map aclFilter
device(config)# policy-map policyAclFilter
device(config-policymap)# class aclFilter
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
device(config-policymap-class-police)# end
device# show policy-map detail policyAclFilter
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in policyAclFilter
device(conf-if-eth-1/2)# end
device# show policy-map detail policyAclFilter
device# copy running-config startup-config
```

Configuring use case 3 - protection against ping flood attacks

Follow these steps to configure an ACL that can be used to protect against ping flood attack.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create or invoke an extended IP ACL.

```
device(config)# ip access-list extended acl1
2015/04/01-13:18:15, [SSMD-1400], 2315, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 is created.
```

The system message is generated when you create an ACL. If you are configuring an existing ACL, no message is generated.

3. Configure the extended ACL to filter ICMP packets.

```
device(conf-ipacl-ext)# permit icmp any any
2015/04/02-11:44:45, [SSMD-1404], 2501, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 10 is added.
```

4. Return to privileged exec mode.

```
device(conf-ipacl-ext)# end
```

5. Verify the ACL.

```
device# show running-config ip access-list extended acl1
ip access-list extended acl1
seq 10 permit icmp any any
```

Protection against ping attacks - ACL configuration example

```
device# configure terminal
device(config)# ip access-list extended acl1
device(conf-ipacl-ext)# permit icmp any any
device(conf-ipacl-ext)# end
device# show running-config ip access-list extended acl1
```

Configuring use case 3 - bind the ping flood attack ACL to an interface

To protect against ping flood DoS attacks, bind an extended Layer 3 ACL based rate limit to filter ICMP packets and bind it to an interface.

You have configured an extended Layer 3 ACL-based rate limit to filter ICMP packets.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a class map.

```
device(config)# class-map aclFilter
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

3. While in class map mode associate the class map with an ACL.

```
device(config)# match access-group acl1
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

4. Return to privileged exec mode.

```
device(config-classmap)# end
```

5. Verify the class map to ACL association.

```
device# show running-config class-map aclFilter
class-map aclFilter
  match access-group acl1
!
```

6. Create a policy map with a policer.

```
device(config)# policy-map policyAclFilter
```

A policy map is used to apply policer and QoS attributes to a particular interface.

7. Associate a class map with the policy map.

```
device(config-policymap)# class aclFilter
```

Each policy map can have different class maps. Each class map in the policy map can be associated to separate policing and QoS parameters.

8. Populate the class map policer parameters.

```
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
```

CIR and EIR are in increments of 18000 bps.

9. Return to privileged exec mode.

```
device(config-policymap-class-police)# end
```

10. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000

    Bound To:None
```

11. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

12. Bind the policy map to the port.

```
device(conf-if-eth-1/2)# service-policy in policyAclFilter
2015/04/02-14:13:31, [SSMD-1405], 2511, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 configured on interface Ethernet 1/2 at Ingress by FbQos_9_11.
```

13. Return to privileged exec mode.

```
device(conf-if-eth-1/2)# end
```

14. Verify the configuration.

```
device# show policy-map detail policyAclFilter
Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000
    Bound To: Et 1/2(in)
```

15. Save the configuration.

```
device# copy running-config startup-config
```


ACL-based protection against ping attacks applied to an interface configuration example

```
device# configure terminal
device(config)# class-map aclFilter
device(config)# match access-group acl1
device(config-classmap)# end
device# show running-config class-map aclFilter
device(config)# policy-map policyAclFilter
device(config-policymap)# class aclFilter
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
device(config-policymap-class-police)# end
device# show policy-map detail policyAclFilter
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in policyAclFilter
device(conf-if-eth-1/2)# end
device# show policy-map detail policyAclFilter
device# copy running-config startup-config
```

Configuring use case 4 - protection against UDP flood attacks

Follow these steps to configure an ACL that can be used to protect against UDP flood attacks.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create or invoke an extended IP ACL.

```
device(config)# ip access-list extended acl1
2015/04/01-13:18:15, [SSMD-1400], 2315, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 is created.
```

The system message is generated when you create an ACL. If you are configuring an existing ACL, no message is generated.

3. Configure the extended ACL to filter UDP packets.

```
device(conf-ipacl-ext)# permit udp any any
2015/04/02-11:44:45, [SSMD-1404], 2501, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 10 is added.
```

4. Return to privileged exec mode.

```
device(conf-ipacl-ext)# end
```

5. Verify the ACL.

```
device(config)# do show running-config ip access-list extended acl1
ip access-list extended acl1
seq 10 permit udp any any
```

Protection against UDP flood attacks - ACL configuration example

```
device# configure terminal
device(config)# ip access-list extended acl1
device(conf-ipacl-ext)# permit udp any any
device(conf-ipacl-ext)# end
device# show running-config ip access-list extended acl1
```

Configuring use case 4 - bind the UDP ACL to an interface

A UDP flood attack is a brute force type of DoS attack where a large number of UDP packets are sent to random ports on the targeted system

You have configured an extended Layer 3 UDP ACL.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a class map.

```
device(config)# class-map aclFilter
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

3. While in class map mode associate the class map with an ACL.

```
device(config)# match access-group acl1
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

4. Return to privileged exec mode.

```
device(config-classmap)# end
```

5. Verify the class map to ACL association.

```
device# show running-config class-map aclFilter
class-map aclFilter
  match access-group acl1
!
```

6. Create a policy map with a policer.

```
device(config)# policy-map policyAclFilter
```

A policy map is used to apply policer and QoS attributes to a particular interface.

7. Associate a class map with the policy map.

```
device(config-policymap)# class aclFilter
```

Each policy map can have different class maps. Each class map in the policy map can be associated to separate policing and QoS parameters.

8. Populate the class map policer parameters.

```
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
```

CIR and EIR are in increments of 18000 bps.

9. Return to privileged exec mode.

```
device(config-policymap-class-police)# end
```

10. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000

  Bound To:None
```

11. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

12. Bind the policy map to the port.

```
device(conf-if-eth-1/2)# service-policy in policyAclFilter
2015/04/02-14:13:31, [SSMD-1405], 2511, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 configured on interface Ethernet 1/2 at Ingress by FbQos_9_11.
```

13. Return to privileged exec mode.

```
device(conf-if-eth-1/2)# end
```

14. Verify the configuration.

```
device# show policy-map detail policyAclFilter
Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000
  Bound To: Et 1/2(in)
```

15. Save the configuration.

```
device# copy running-config startup-config
```

ACL-based protection against UDP flood attacks applied to an interface configuration example

```
device# configure terminal
device(config)# class-map aclFilter
device(config)# match access-group acl1
device(config-classmap)# end
device# show running-config class-map aclFilter
device(config)# policy-map policyAclFilter
device(config-policymap)# class aclFilter
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
device(config-policymap-class-police)# end
device# show policy-map detail policyAclFilter
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in policyAclFilter
device(conf-if-eth-1/2)# end
device# show policy-map detail policyAclFilter
device# copy running-config startup-config
```

Configuring and applying all four use cases for ACL-based traffic filtering

Follow these steps to apply ACLs for traffic filtering.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an ACL.

```
device(config)# ip access-list extended acl1
2015/04/02-13:22:39, [SSMD-1400], 2506, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 is created.
```

The system message is generated when you create an ACL. If you are configuring an existing ACL, no message is generated.

3. Configure the extended ACL to filter packets for which the **sync** (synchronize) flag is set.

```
device(config-ipacl-ext)# permit tcp any any sync
2015/04/02-13:25:28, [SSMD-1404], 2507, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 10 is added.
```

This step provides protection from TCP SYN attacks.

4. Configure the extended ACL to filter packets for which the **rst** flag is set.

```
device(config-ipacl-ext)# permit tcp any any
rst
2015/04/02-13:26:48, [SSMD-1404], 2508, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 20 is added.
```

This step provides protection from TCP RST attacks.

5. Configure the extended ACL to filter ICMP packets.

```
device(config-ipacl-ext)# permit icmp any any
2015/04/02-13:28:20, [SSMD-1404], 2509, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 30 is added.
```

This step protects against ping flood attacks.

6. Configure the extended ACL to filter UDP packets.

```
device(config-ipacl-ext)# permit udp any any
2015/04/02-13:30:15, [SSMD-1404], 2510, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 rule sequence number 40 is added.
```

This step protects against UDP flood attacks.

7. Return to global configuration mode.

```
device(config-ipacl-ext)# exit
```

8. Verify the ACL.

```
device(config)# do show running-config ip access-list extended acl1
ip access-list extended acl1
 seq 10 permit tcp any any sync
 seq 20 permit tcp any any rst
 seq 30 permit icmp any any
 seq 40 permit udp any any
!
```

9. Create a class map.

```
device(config)# class-map aclFilter
```

The class map is used to classify the traffic; different match conditions, including an ACL, can be used to match the traffic properties.

10. While in class map mode associate the class map with an ACL.

```
device(config-classmap)# match access-group acl1
```

11. Return to global configuration mode.

```
device(config-classmap)# exit
```

12. Verify the class map to ACL association.

```
device(config)# do show running-config class-map aclFilter
class-map aclFilter
  match access-group acl1
!
```

13. Create a policy map with a policer.

```
device(config)# policy-map policyAclFilter
```

A policy map is used to apply policer and QoS attributes to a particular interface.

14. Associate a class map with the policy map.

```
device(config-policymap)# class aclFilter
```

Each policy map can have different class maps. Each class map in the policy map can be associated to separate policing and QoS parameters.

15. Populate the class map policer parameters.

```
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
```

CIR and EIR are in increments of 18000 bps.

16. Return to privileged exec mode.

```
device(config-policymap-class-police)# end
```

17. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 4000000 cbs 50000 eir 800000 ebs 400000

  Bound To:None
```

18. Enter global configuration mode.

```
device# configure terminal
```

19. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

20. Bind the policy map to the port.

```
device(conf-if-eth-1/2)# service-policy in policyAclFilter
2015/04/02-14:13:31, [SSMD-1405], 2511, SW/device | Active | DCE, INFO, device, IPv4 access list
acl1 configured on interface Ethernet 1/2 at Ingress by FbQos_9_11.
```

21. Return to privileged exec mode.

```
device(conf-if-eth-1/2)# end
```

22. Verify the configuration.

```
device# show policy-map detail policyAclFilter

Policy-Map policyAclFilter
  Class aclFilter
    Police cir 180000 cbs 50000 eir 36000 ebs 400000

Bound To: Et 1/2(in)
```

23. Save the configuration.

```
device# copy running-config startup-config
```

ACL-based traffic filtering to protect from DoS attacks configuration example

```
device# configure terminal
device(config)# ip access-list extended acl1
device(conf-ipacl-ext)# permit tcp any any sync
device(conf-ipacl-ext)# permit tcp any any rst
device(conf-ipacl-ext)# permit icmp any any
device(conf-ipacl-ext)# permit udp any any
device(config)# do show running-config ip access-list extended acl1
device(config)# class-map aclFilter
device(config-classmap)# match access-group acl1
device(config-classmap)# exit
device(config)# do show running-config class-map aclFilter
device(config)# policy-map policyAclFilter
device(config-policymap)# class aclFilter
device(config-policymap-class)# police cir 180000 cbs 50000 eir 36000 ebs 400000
device(config-policymap-class-police)# end
device# show policy-map detail policyAclFilter
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# service-policy in policyAclFilter
device(conf-if-eth-1/2)# end
device# show policy-map detail policyAclFilter
device# copy running-config startup-config
```

Configuring storm control

This example configures BUM storm control on a 10-gigabit Ethernet interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Specify the Ethernet interface for the traffic you want to control.

```
device(config)# interface ethernet 2/2
```

3. Issue the storm control ingress command to set a traffic limit for broadcast traffic on the interface..

```
device(conf-if-eth-2/2)# storm-control ingress broadcast limit-bps 400000 shutdown
```

In this example you set a control on the inbound broadcast traffic, limiting the rate to 400000 bits per second (bps), with a parameter set to shutdown the port in case of a limit violation.

4. Issue the storm control ingress command to set a traffic limit for unknown-unicast traffic on the interface..

```
device(config-if-eth-2/2)# storm-control ingress unknown-unicast limit-bps 50000000 monitor
```

In this example you set a control on the inbound unknown-unicast traffic, limiting the rate to 50000000 bps, with a parameter set to monitor the port in case of a limit violation.

5. Issue the storm control ingress command to set a traffic limit for multicast traffic on the interface..

```
device(config-if-eth-2/2)# storm-control ingress multicast limit-percent 3 shutdown
```

In this example you set a control on the inbound multicast traffic, limiting the rate to 3% of traffic, with a parameter set to monitor the port in case of a limit violation.

6. Return to privileged exec mode.

```
device(config-if-eth-2/2)# end
```

7. Verify the storm control configuration.

```
device# show run | include storm-control
storm-control ingress broadcast limit-bps 400000 shutdown
storm-control ingress multicast limit-percent 3 shutdown
storm-control ingress unknown-unicast limit-bps 50000000 monitor
```

8. Save the configuration.

```
device# save running-config startup-config
```

BUM storm control configuration example

```
device# configure terminal
device(config)# interface ethernet 2/2
device(config-if-eth-2/2)# storm-control ingress broadcast limit-bps 400000 shutdown
device(config-if-eth-2/2)# storm-control ingress unknown-unicast limit-bps 50000000 monitor
device(config-if-eth-2/2)# storm-control ingress multicast limit-percent 3 shutdown
device(config-if-eth-2/2)# end
device# show storm-control
device# save running-config startup-config
```


Quality of Service

- QoS overview..... 41
- Configuring QoS..... 56

QoS overview

Quality of Service (QoS) provides preferential treatment to specific traffic.

By offering preferential treatment to specific traffic, other traffic may be stopped or slowed. Without QoS, the device offers best-effort service to each packet and transmits packets without any assurance of reliability, delay bounds, or throughput. Implementing QoS in a network makes performance more predictable and bandwidth utilization more effective.

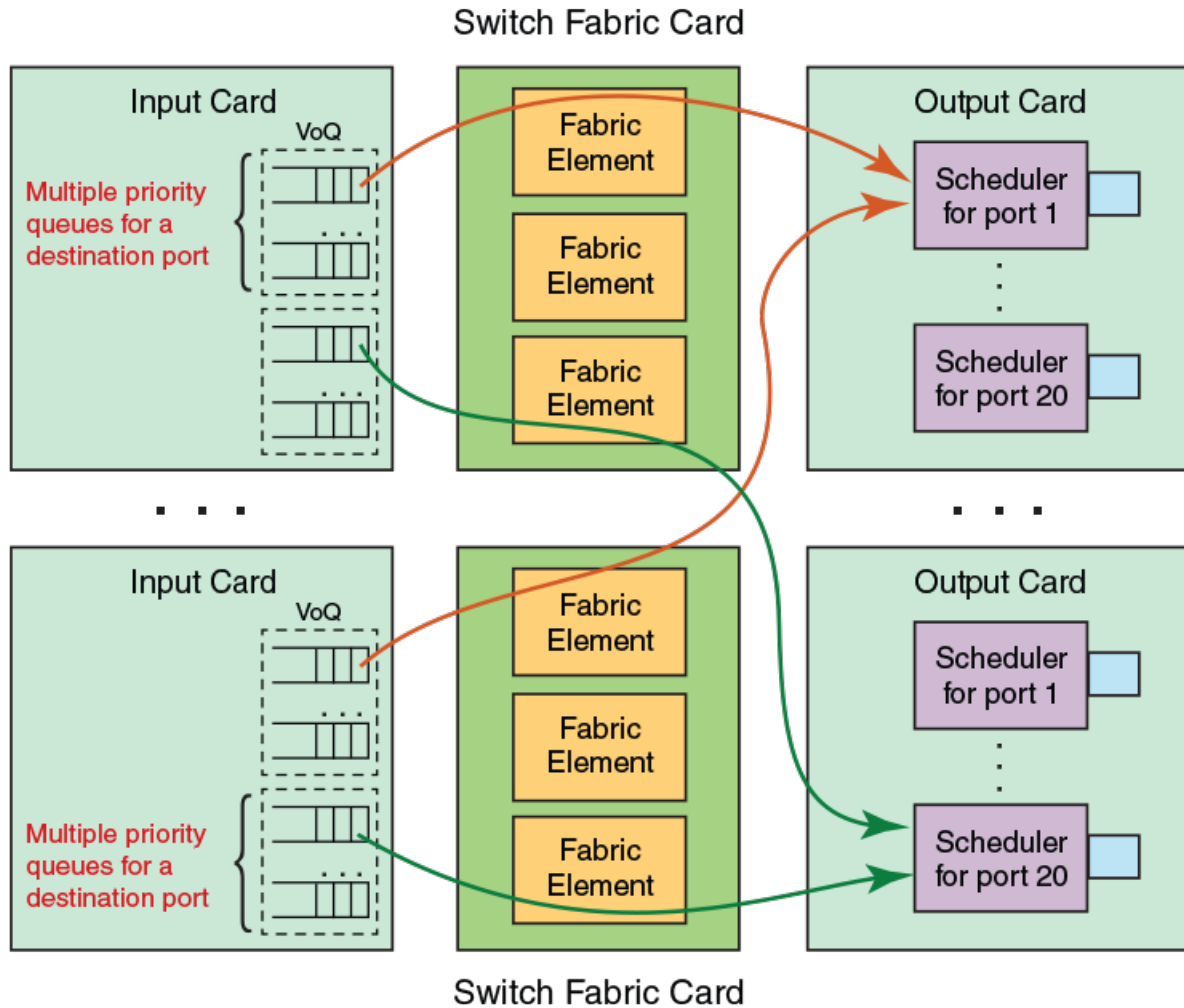
QoS features

The principal QoS features are as follows.

QoS for unicast traffic

In the Traffic Manager, QoS for unicast traffic follows the process seen in the below figure.

FIGURE 3 QoS for unicast traffic



- The input is buffered through Virtual Output Queues (VOQ) with output port driven scheduling.
- Each ingress Traffic Manager maintains a set of eight distinct priority queues for every output port on the system. Incoming packets are enqueued to a VOQ corresponding to the destination output port and classified with an internal priority.
- Packets are dequeued by an output port on the egress card when the output port is ready to send a packet.
- Switch fabric messaging is used to maintain a tight coupling between ingress and egress cards.
- Egress traffic manager dequeues packets from appropriate VOQs by sending VOQ transmission credits to the respective ingress traffic managers.

QoS for multicast traffic

While managing multicast traffic, consider the following:

- There are four fabric multicast queues (FMQs) for multicast traffic.
- The Traffic Manager (TM) maps incoming packets to these queues based on the traffic class (TC) or drop precedence (DP) received from the packet processor (PP).

- The ingress TM pushes multicast traffic to the fabric either by strict priority (SP) or by mixed SP and weighted priority.
- There is CLI to configure ingress multicast shaping. By default it is opened to the maximum rate of the tower.
- The egress TM maps these multicast packets to two egress queues (EGQ).
- EGQs share memory from a 3MB pool.

IEEE 802.1q ToS-DSCP header fields

The Type of Service (ToS), now known as Differentiated Services (DS), defines a mechanism for assigning a priority to each IP packet as well as a mechanism to request specific treatment such as high throughput, high reliability or low latency.

The 8 bit ToS field originally defined a mechanism for assigning priority to each IP packet as well as a way to request treatment such as high throughput, high reliability or low latency.

The definition of this field was changed in RFC 2474 . The 6 bit field is now called the DS (Differentiated Services) field and the upper 6 bits contain a value called the Differentiated Services Code Point (DSCP). The remaining two least significant bits are used for Explicit Congestion Notification (ECN).

DSCP

The ToS field is now used by Differentiated Services and is called the Differentiated Services Code Point (DSCP) .

DSCP values range from 0 through 63 that map in groups of 8 to the user priority values.

TABLE 4 Default DSCP mappings

DSCP IP precedence	User priority
0-7	0
8-15	1
16-23	2
24-31	3
32-39	4
40-47	5
48-55	6
56-63	7

Congestion control

Congestion control covers features that define how the system responds when congestion occurs or active measures taken to prevent the network from entering a congested state.

Sustained, large queue buildups generally indicate congestion in the network and can affect application performance through increased queueing delays and frame loss. Queues can begin filling up for a number of reasons, such as over-subscription of a link or back pressure from a downstream device. When queues begin filling up and all buffering is exhausted, frames are dropped. This has a detrimental effect on application throughput. Congestion control techniques are used to reduce the risk of queue overruns without adversely affecting network throughput.

Congestion control covers features that define how the system responds when congestion occurs or active measures taken to prevent the network from entering a congested state. These features include link level flow control (LLFC) and Weighted random early detection (WRED).

Weighted random early detection

Weighted random early detection (WRED) is a traffic control feature that uses IP precedence to determine how it treats or drops traffic.

On the device, queues are provided to buffer traffic levels that exceed the bandwidth of individual ports. For each output port, a set of eight priority queues is allocated. When traffic exceeds the bandwidth of a port, packets are dropped randomly as long as the congestion persists. Under these conditions, traffic of greater priority can be dropped instead of traffic with a lesser priority.

Instead of being subject to random selection, you can configure a device to monitor traffic congestion and drop packets according to a WRED algorithm. This algorithm enables the system to detect the onset of congestion and take corrective action. In practice, WRED causes a device to start dropping packets as traffic in the device starts to back up. WRED provides various control points that can be configured to change a system's reaction to congestion. The following variables are used when calculating whether to drop or forward packets:

Statistical Average-Q-Size - The statistical average size of the queue calculated over time on the device.

Current-Q-Size - The current size of the queue as calculated on the device.

Wq - This variable specifies the weights that should be given to the current queue size and the statistical average-q-size when calculating the size for WRED calculations.

Max-Instantaneous-Q-Size - The maximum size up to which a queue is allowed to grow. Packets that cause the queue to grow beyond this point are unconditionally dropped. This variable is user configured.

Min-Average-Q-Size - The average queue size below which all packets are accepted. This variable is user configured.

Max-Average-Q-Size - The average queue size above which all packets are dropped. This variable is user configured.

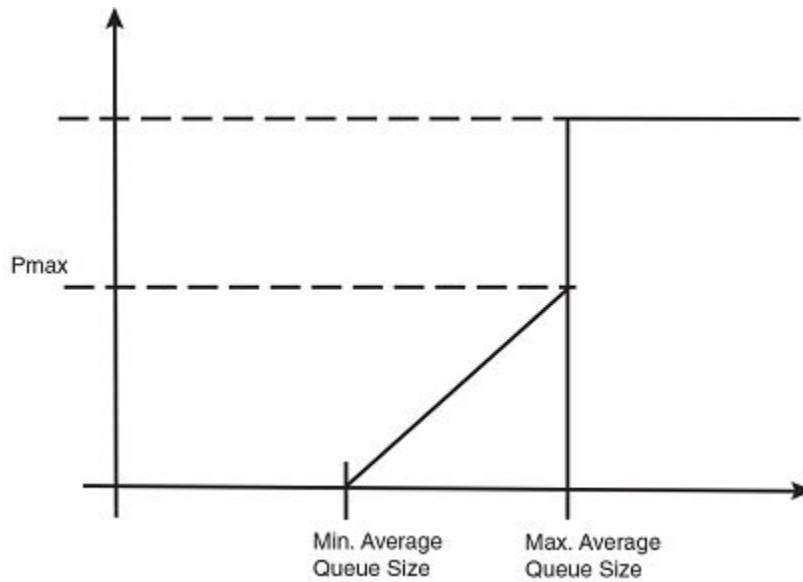
Pmax - The maximum drop probability when queue-size is at Max-Average-Q-Size. This variable is user configured.

Pkt-Size-Max - The packet size to which the current packet's size is compared as shown in the algorithm below. This variable is user configured.

How WRED works

The WRED operation graph below describes the interaction of the previously described variables in the operation of WRED. When a packet arrives at a device, the average queue size (*avg-q-size*) is calculated as described below (note that this is not the statistical average queue size). If *avg-q-size* as calculated, is below the configured Min. Average Queue Size, then the packet is accepted. If the average queue size is above the Max. configured Average Queue Size threshold, the packet is dropped. If the instantaneous queue size exceeds the value configured for the Max-Instantaneous-Q-Size, the packet is dropped. If the Average Queue size falls between the Min. Average Queue Size and the Max. Average Queue Size, packets are dropped according to the calculated probability described below.

FIGURE 4 WRED operation graph



Calculating avg-q-size

The algorithm first calculates the *avg-q-size* through the following equation.

$$\text{avg-q-size} = [(1 - Wq) \times \text{Statistical Average-Q-Size}] + (Wq \times \text{Current-Q-Size})$$

The user-configured *Wq* value is instrumental to the calculation and can be:

- equal to the statistical average queue size ($Wq == 0$), or
- equal to the current queue size ($Wq == 1$) or
- be between 0 and 1 ($0 < Wq < 1$).

Lower *Wq* values cause the *avg-q-size* to lean towards the statistical average queue size, reducing WRED's sensitivity to the current state of the queue and thus reducing WRED's effectiveness. On the other hand, higher *Wq* values cause the *avg-q-size* to lean towards the instantaneous queue size, which exposes WRED to any change in the instantaneous queue size and thus may cause WRED to overreact in cases of bursts. Thus, the value of *Wq* should be carefully chosen according to the application at hand.

Calculating packets that are dropped

The *Pdrop* value, as calculated in the following equation, is the probability that a packet will be dropped in a congested device.

$$P_{\text{drop}} = (\text{pkt-size} \div \text{pkt-size-max}) \times P_{\text{max}} \times [(\text{avg-q-size} - \text{min-avg-q size}) \div (\text{max-avg-q-size} - \text{min-avg-q size})]$$

Apply WRED

Packets are assigned to an ingress queue type based on their individual destination port and one of the 8 (0 - 7) internal priorities. Each of these priorities is assigned a queue type from 0 - 7 according to the internal priority it belongs to.

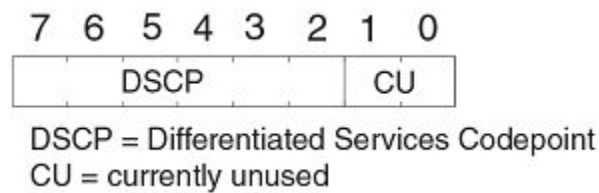
TABLE 5 Internal priority to queue type mapping

Internal priority	0	1	2	3	4	5	6	7

TABLE 5 Internal priority to queue type mapping (continued)

Queue type	0	1	2	3	4	5	6	7
------------	---	---	---	---	---	---	---	---

The WRED algorithm is applied to traffic on these individual queues based upon parameters configured for its assigned queue type. When traffic arrives at a queue, it is passed or dropped as determined by the WRED algorithm. Packets in an individual queue are further differentiated by one of four drop precedence values which are determined by the value of bits 3:2 of the DSCP bits in the IPv4 or IPv6 packet header.

FIGURE 5 DSCP bits in packet header

The user configurable values applied per queue type and per drop precedence value are:

- Maximum Drop Probability
- Minimum and Maximum Average Queue Size
- Maximum Packet Size

Link level flow control

Link level flow control (LLFC) is a way to alleviate system congestion by pausing data transmission.

When a receiving device is congested, it communicates with the transmitting device by sending a PAUSE frame that instructs the device to stop data transmission for a specified period of time. This feature is available per port in all front ports and applies to all the traffic on the link. However, Extreme supports the receive direction only, we do not support the generation of the PAUSE frame.

By default, the feature is disabled for both directions.

See the procedure, [Configuring link level flow control](#) on page 69 for the steps to configure LLFC.

Scheduling

Scheduling arbitrates among multiple queues waiting to transmit a frame.

The device supports Strict Priority (SP) scheduling, Weighted fair queue traffic scheduling (WFQ), and mixed SP and WFQ scheduling.

Scheduling types

TABLE 6 Scheduling comparisons

Scheduling type	Description
SP (Strict priority)	SP handles the scheduling of the packets following a priority-based model where packets are classified and placed into different queues with different priorities. Packets are sent from the head of a given queue for processing only if the queues with higher priorities are empty.
WRR (Weighted round robin)	WRR addresses the priority queue problem in which one queue can starve other queues that are not as high a priority. WRR does this by allowing at least one packet to be removed from each queue containing packets in each scheduling turn. This scheme is best used with server queues with different processing capacities.

TABLE 6 Scheduling comparisons (continued)

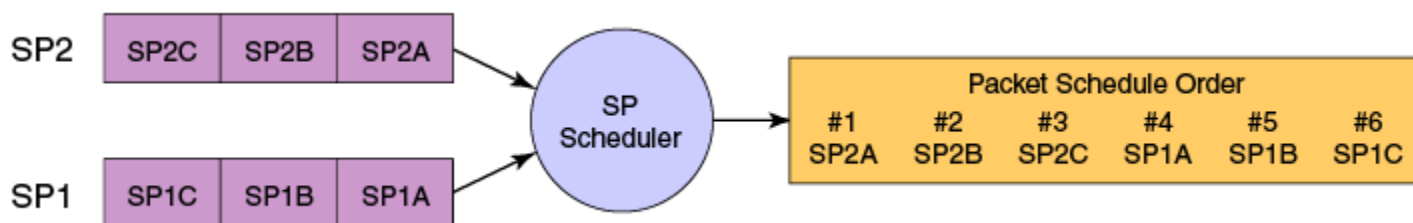
Scheduling type	Description
WFQ (Weighted fair queueing)	In WFQ big packets do not get more scheduling time than smaller packets, as the WFQ foci is on bits and not packets as in WRR.
DWRR (Deficit weighted round robin)	DWRR is a modified WRR scheduling type that addresses the limitations of WRR. The algorithm handles packets with variable sizes. A maximum packet size number is subtracted from the packet length, and packets that exceed that number are held back until the next scheduling turn
Mixed SP and WFQ	With this type of scheduling the top scheduler inputs are SP and the bottom scheduler inputs are WFQ . Usually it is the top three are SP and the bottom five are WFQ.

QoS strict priority egress traffic scheduling

Egress traffic scheduling allows you to selectively manage traffic based on the forwarding queue to which it is mapped.

Strict priority scheduling (SP) scheduling is used to facilitate support for latency sensitive traffic. A strict priority scheduler drains all frames queued in the highest-priority queue before continuing on to service lower-priority traffic classes.

The following figure displays the frame scheduling order for an SP scheduler servicing two SP queues. The higher-numbered queue, SP2, has a higher priority.

FIGURE 6 Strict priority schedule — two queues

The disadvantage of strict priority-based scheduling is that lower-priority traffic can be starved of any access.

The devices classify packets into one of eight internal priorities. For each egress port, there are 8 Virtual output queues (VOQ) allocated on each ingress TM core to support 8 priorities. SP queue input values map to traffic classes and range from 0 through 7. These are:

- 0 - No strict priority queue.
- 1 - Traffic Class 7 strict priority queue.
- 2 - Traffic Class 6 through 7 strict priority queues.
- 3 - Traffic Class 5 through 7 strict priority queues.
- 4 - Traffic Class 4 through 7 strict priority queues.
- 5 - Traffic Class 3 through 7 strict priority queues.
- 6 - Traffic Class 2 through 7 strict priority queues.
- 7 - Traffic Class 1 through 7 strict priority queues.

When configuring egress traffic scheduling you use credit request and grant mechanisms to perform QoS. The credit size is 1024B.

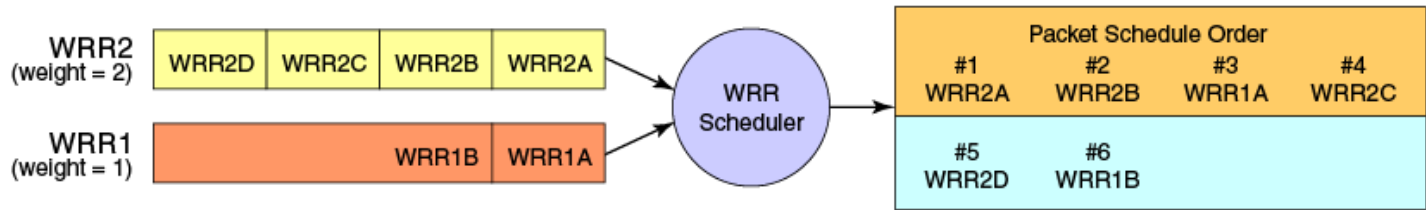
Weighted round robin egress traffic scheduling

In the weighted round robin (WRR) destination-based scheduling enabled scheme, some weight-based bandwidth is allocated to all queues.

WRR scheduling is used to facilitate controlled sharing of the network bandwidth. WRR assigns a weight to each queue; that value is then used to determine the amount of bandwidth allocated to the queue. The round robin aspect of the scheduling allows each queue to be serviced in a set order, sending a limited amount of data before moving onto the next queue and cycling back to the highest-priority queue after the lowest-priority queue is serviced.

The following figure displays the frame scheduling order for a WRR scheduler servicing two WRR queues. The higher-numbered queue is considered higher priority (WRR2), and the weights indicate the network bandwidth should be allocated in a 2:1 ratio between the two queues. In this figure WRR2 receives 66 percent of the bandwidth and WRR1 receives 33 percent. The WRR scheduler tracks the extra bandwidth used and subtracts it from the bandwidth allocation for the next cycle through the queues. In this way, the bandwidth utilization statistically matches the queue weights over longer time periods.

FIGURE 7 WRR schedule — two queues



Deficit Weighted Round Robin (DWRR) is an improved version of WRR. DWRR remembers the excess used when a queue goes over its bandwidth allocation and reduces the queue’s bandwidth allocation in the subsequent rounds. This way the actual bandwidth usage is closer to the defined level when compared to WRR.

Fair queue egress traffic scheduling

There are two types of fair queue egress traffic scheduling, weighted fair queue (WFQ) and mixed strict priority (SP) and WFQ.

Weighted fair queue

With WFQ destination-based scheduling enabled, some weight-based bandwidth is allocated to all queues. With this scheme, the configured weight distribution is guaranteed across all traffic leaving an egress port and an input port is guaranteed allocation in relationship to the configured weight distribution. You can specify weighted for each VOQ if in WFQ mode.

Mixed SP and WFQ egress traffic scheduling

This scheme provides a mixture of SP for the three highest priority queues and WFQ for the five remaining priority queues.

Multicast queue scheduling

A fixed mapping from multicast traffic class to equivalent unicast traffic class is applied to select the queue scheduling behavior.

The multicast traffic classes are numbered from 0 to 7; higher numbered traffic classes are considered higher priority. The Multicast traffic class equivalence mapping table below presents the multicast traffic class with the equivalence mapping applied.

Once the multicast traffic class equivalence mapping has been applied, then scheduling and any scheduler configuration are inherited from the equivalent unicast traffic class. Refer to the table below for details on exact mapping equivalencies.

TABLE 7 Multicast traffic class equivalence mapping

Multicast traffic class	Equivalent unicast traffic class
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7

Unicast ingress and egress queueing utilizes a hybrid scheduler that simultaneously supports SP+WRR service and multiple physical queues with the same service level. Multicast adds additional multicast expansion queues. Because multicast traffic classes are equivalent to unicast service levels, they are treated exactly as their equivalent unicast service policies.

Control protocol packet prioritization

Certain control packets are handled with certain priorities by default and hence those priorities cannot be lowered with any of the QoS configuration commands.

The following table lists the protocol packets that are internally and automatically prioritized for IPv4, Layer 2, and IPv6.

TABLE 8 Default prioritized protocol table

Protocol Packets
ARP
BFD (Bidirectional Forwarding Detection)
BGP / BGP in 6to4
BGP / BGP over GRE
BOOTP/DHCP
ES-IS
GARP
IGMP
IPv4 Router Alert
IPv4/L2
IPv6
IS-IS
ISIS over GRE or GRE
Keep Alive Packets
LACP
LDP basic
LDP extended
MLD
MRP

TABLE 8 Default prioritized protocol table (continued)

Protocol Packets
MSDP / MSDP over GRE
ND6 / ND6 in 6to4
OSPF / OSPF in 6to4
OSPF / OSPF over GRE
PIM / PIM in 6to4
PIM / PIM over GRE
RIP
RIPNG
RSVP
STP/RSTP/BPDU
UDLD
VRRP
VRRP
VRRPE
VRRPEPIM / PIM in 6to4BFD (Bidirectional Forwarding Detection)
VSRP
Y.1731

Enhanced control packet prioritization

The Traffic Manager (TM) allows prioritization and scheduling of packets destined for the CPU to guarantee optimal control packet processing and to reduce protocol flapping.

The TM achieves physical separation of CPU-bound data and control packets. The hierarchical structure supports four sets of eight priority queues. The four sets are as follows:

- Protocol set - Protocol packets that are prioritized by the network processor.
- Management set - Packets destined for the router; for example, Ping and Telnet.
- Flow set - Flow-driven packets to the CPU; for example, Unknown DA, DPA, Layer 2 broadcast and multicast, Multicast, VPLS SA Learning packets.
- Snoop set - CPU copy packets; for example, Regular Layer 2 SA learning, sFlow, ACL Logging, RPF Logging. The **rl-cpu-copy** command defines the rate shaping value for the snoop queues.

Priority queue 7 is given the highest rate, priority queues 6 to 2 are given the same rate, and priority queues 1 and 0 are each given a lower rate. Each of the four sets of CPU queues is given equal weights. With the flow-driven packets given the same weight as the other packets, the protocol packets and flow-driven packets are queued separately so that the protocol packets are processed at a faster rate compared to the flow-driven packets.

The following table lists the protocol packets of the network processor prioritized protocol set based on their priorities.

TABLE 9 Network processor prioritized protocol packets

Priority categorization	Protocols
P7	LACP, UDLD (802.3ah), STP, RSTP, BPDU, VSRP, MRP, BFD, GRE-KA, IS-IS over GRE, G. 8032, LLDP, non-CCM 802.1ag (Ethernet + MPLS Encapsulated), BFD (Single-hop, Multi-hop, and MPLS), IS-IS Hello, OSPFv2 Hello (GRE + Ethernet), OSPFv3 Hello (6to4 + Ethernet).

TABLE 9 Network processor prioritized protocol packets (continued)

Priority categorization	Protocols
P6	IS-IS Non-Hello, OSPFv2 and OSPFv3 Non-Hello (Ethernet, GRE, 6to4), IPsec ESP Packets (for OSPFv3 over IPsec), OSPF, OSPF over GRE or 6to4, IS-IS, RIP, RIPNG, VRRP (Version 4 and Version 6), VRRP-E (Version 4 and Version 6).
P5	BGP, BGP over GRE or 6to4, PIM, PIM over GRE or 6to4, LDP (basic and extended), RSVP, CCP (MCT), 802.1ag CCM (Ethernet + MPLS Encapsulated).
P4	VPLS Encapsulated PIM, MSDP, MSDP over GRE, MSDP over VPLS.
P3	IGMP, VPLS Encapsulated IGMP, GRE Encapsulated IGMP, ARP, MLD, DHCP, BOOTP, ND6 and ND6 in 6to4.
P2	IPv4 Router Alert.
P1	New unassigned protocols.
P0	Existing unassigned protocols: GARP, L2-Trace.

Prioritize management traffic to achieve QoS value

This feature introduces the ability to classify and prioritize the management traffic (SSH and Telnet) so that the outgoing management protocol packets from the device can receive a particular QoS value.

In congested networks, where management protocol packets may be dropped, or management protocol packets require a higher priority, the DSCP or IP precedence value from the incoming packets needs to be copied to the outgoing packets.

This feature is designed to change the per-hop behavior (PHB) of packets that are routed through the network. This will not affect the treatment of the packets inside the device.

Feature details

- CoS values from the received management protocol packet is copied to the transmitted packet. It is implemented by using a CLI command **copy-received-cos**. The command is supported per application, irrespective of IP address, TCP or UDP port, etc.
- This feature supports IPv4 and IPv6.
- Each individual received packet is evaluated and its CoS value is copied to the transmitted packet. The priority changes during the flow, if the client changes the CoS values.
- This feature supports on SSH and Telnet.

Limitations and pre-requisites

If the DSCP encoding or PCP encoding is on, then the default encoding map must be used to achieve the expected behavior.

If user-defined Encode policy map is configured to mark the packets as they exit the device, then the DSCP or IP precedence value and PCP value may get changed based on the policy map and hence may not remain the same as the incoming packets.

Configure flow-based QoS

Follow these high level steps to configure flow-based QoS.

1. Configure a class map to classify traffic according to the traffic properties required for your flow-based QoS needs.
2. Configure a policy map and associate it to the class map.

NOTE

Policy maps can be bound in both the ingress and egress directions.

3. Add the QoS action to be applied on the type of flow determined by the class map.
4. Bind the policy map to a specific interface.

Match access-group - class map policing

Access groups are used for Layer 2 and Layer 3 ACL-based ingress rate limit and denial of service (DoS) mitigation.

ACL-based rate limiting is built on top of ACL and policer features, it rate limits the Layer 3 traffic that matches the permit conditions specified in an IPv4 access list. The ACL-based policer feature controls the amount of bandwidth consumed by an individual flow or aggregate of inbound flows by limiting the traffic rate on an individual port according to criteria defined by the **match access-group** class map. This ACL-based rate limiting feature can serve as a hardware solution to prevent DoS attacks.

Match access-group - class map policing rules and limitations

Consider these rules and limitations when you are configuring **match access-group** class map policing:

- You can configure:
 - 1024 policy maps
 - 16 class maps
 - 6144 ACL Content Addressable Memory (CAM) entries for use with rate limiting
 - 16356 ingress policers
- For protection against:
 - PING attacks
 - TCP Reset attacks
 - TCP SYN attacks
 - UDP attack
- Only Layer 3 IPv4 ACL-based rate limiting is supported.
- ACL-based rate limiting is applicable only to ingress traffic.
- There is one policer per ACL, it applies to all the rules for that ACL
- Control protocols are rate-limited if they match the configured ACL clause.
- When a **match access-group** class map rate limit is applied to a LAG logical port, and all LAG ports belong to the same tower, then MAX CIR value is the interface speed × number of physical ports. For example: if 1/1, 1/2 are LAG member ports, then MAX CIR will be 2 × 10Gbps.
- When a **match access-group** class map rate limit is applied to LAG logical port, MAX rate on that port is the number of the tower in that LAG × CIR. For example: if 1/1, 1/2, 2/1, 3/1 are LAG member ports, then MAX rate is 3 × CIR.

Policy maps

Policy maps allow you to set a policy in a single location that affects multiple ports and to make changes to that policy.

The policy map configuration includes a set of class maps and QoS parameters.

A policy map allows you to specify policers in a single location that can be applied to multiple ports and to make changes to that policy.

When using the traffic policing policies available from previous versions, the policy parameters are provided explicitly for each port during port configuration. In this version, the policies must be defined using a policy map. One policy map can be specified per service-policy. You can configure up to 1024 policy maps per system.

Policy map configuration rules

Follow these rules when configuring traffic policing:

- A policer map (policy map or class map) name must be unique among all maps of that type.
- A policer name must begin with a-z or A-Z . An underscore, hyphen, and numeric values 0-9 can be used in the body of the name but not as the first character.
- You can configure a maximum of 1024 policy maps.
- The default and **match access-group** class maps cannot be combined in a single policy map.
- For an ingress or egress service policy, one default class map can be specified per policy map.
- For an ingress only service policy, 16 **match access-group** class maps are supported per policy map.
- Broadcast, unknown unicast and multicast (BUM) policies are counted separately.
- You cannot delete a policy map if it is referenced in an active service policy (applied on an interface).

QoS shaping rate

You can specify the shaping rate per port attached to the policy map to smooth the traffic that egresses an interface. This configuration is allowed only for egress traffic.

QoS ingress data buffer management

Buffer management consists of the following.

- Packets arrived at ingress are stored in a data buffer (DB).
- A DB can be an on-chip buffer (OCB, 128 MB) or external DRAM (up to 8 GB).
- Each OCB size is 256 B and external DRAM buffer size is 2 KB.
- The Virtual output queue (VOQ) holds packet descriptors, which are a list of buffer descriptors (BD).
- The entire packet buffer can be configured for a given VOQ to 1.5 GB for D cards and 2 GB for M cards.

IMPORTANT



Specifying a value over 1.5 GB for a D card generates an error:

```
QSizeLimit LC Type -D ifIdx: 0x80000000, slot_id: 1, maxQueueSize: 1536.
```

- The DB and BD pools are managed, per-license, as follows:
 - Avoid starving high priority traffic by allocating too many resources to high rate low priority traffic.
 - Each VOQ has its own minimum guaranteed BD and DB (10% queue size). The default VOQ size is 1 MB.
 - The non-guaranteed BD and DB are allocated from a shared pool.

See the topic Configuring virtual output queueing.

QoS mutation overview

Ingress QoS mutation

The QoS operation on ingress traffic involves reception and processing of packets based upon priority information contained within the packet.

When packets are processed through the device, there are several opportunities to influence the processing by configuration as described in the steps below. The processes performed to map packet priority to internal priority and drop precedence can be described as following:

- Collect priority and drop precedence information from various portions of the packet header:
 - If a packet's EtherType matches 8100, derive a priority value and drop precedence by decoding the PCP value.
 - For MPLS packets, you derive priority value and drop precedence by decoding the EXP bits.
 - For IPv4 or IPv6 packets, derive priority value and drop precedence by decoding the DSCP bits.
 - For untagged Layer 2 packet, derive traffic class and drop precedence using the port's default value.
 - The derived values for PCP, EXP and DSCP are mapped using either a default map or a configured ingress decode policy map.
 - To assist the device in the decoding process described, decode map tables are defined.
- The priority and drop precedence values are obtained in descending order of priority, as follows:
 1. If tag exists and packet is switched, by decoding the PCP value from the tag.
 2. For IPv4 or IPv6 packets, and when the packet is routed, by decoding the DSCP field from the IP header.
 3. For MPLS packets, by decoding the EXP value from MPLS header.
 4. Physical port default value.

Egress QoS mutation

The QoS operation on egress traffic involves marking packets as they leave the chip on the egress port. As the packets are prepared to exit the device you can set the PCP, DSCP, and CoS values in the packet headers.

802.1P priority mapping (traffic class-to-PCP mutation)

The internal traffic class can be mapped to the outgoing PCP value when the packet egresses the switch. You can create a priority mapping table using a CoS mutation map. This CoS mutation map can then be applied to an egress interface to effect the priority re-mapping. This feature only maps the internal traffic class to outgoing priority.

DSCP remarking

This feature allows the user to remark the DSCP value of the egressing IP packet using the ingress DSCP value. The configuration must be bound to an egress interface.

DSCP-to-CoS mutation mapping

The ingress DSCP value can be mapped to outgoing 802.1P values by configuring a DSCP-to-CoS mutation map on the egress interface.

Apply QoS mutation maps

Several types of QoS mutation maps can be applied to ethernet ports.

Specifying the mutation map used on a port can lead to contradictions if there are other user-defined classes used in the same policy map that have a set CoS action configured. In this case the defined CoS takes priority over the mutation map.

The available mutations are :

- **cos-cos**
- **cos-traffic-class**
- **dscp-dscp**
- **dscp-cos**
- **dscp-mutation**
- **dscp-traffic-class**
- **traffic-class-cos**

Multiprotocol Label Switching QoS overview

Multiprotocol Label Switching MPLS is a data carrying technique that directs data from one network node to the next based on labels rather than network addresses or prefixes.

Label edge routers

A label edge router (LER) is a router that operates at the edge of an MPLS network that is the entry and exit points for the network.

Ingress label edge routers

The process followed by a LER consists of the following high level steps:

1. Layer 2 or Layer 3 traffic enters the edge of the MPLS network at the edge LER (PE1).
2. The LER receives the traffic from the input interface and uses the PCP or the IP DSCP bits to determine the EXP bits.
3. The appropriate label is pushed (imposition) into the packet, and the EXP value resulting from the QoS decision is copied into the MPLS EXP field in the label header.
4. The labeled packets (marked by EXP) are sent to the core MPLS network.

Egress label edge routers

The default behavior of an egress LER is to pop the outermost MPLS label, and send the frame with intact DSCP and PCP fields.

- Ingress interface of egress LER - The mapping {pkt-EXP} => {TC, DP} is derived just like for the ingress LSR interface.
- EXP-to-DSCP mapping in egress LER - The EXP can be used to derive the DSCP value of egressing Layer 3 frames. This can be done by configuring an EXP-to-DSCP map on the egress interface of the LER.
- TC-to-PCP mapping in egress LER - For Layer 2 switched frames, traffic class is used to derive the PCP value going out. This is done by configuring a traffic class-to-PCP map on the egress interface of the LER.

Label switch routers

A label switch router (LSR) is an MPLS router that performs routing based only on the label.

The LSR is located in the middle of an MPLS network. It is responsible for switching the labels used to route packets.

The LSR operates under these rules:

- Incoming MPLS-labeled packets from a LER or another LSR arrive at the core LSR.
- The LSR receives the traffic from the input interface and uses the EXP bits to perform classification, marking, and policing.

- Once the next hop LSR is determined, an appropriate label is placed (swapped) into the packet and the derived MPLS EXP bits are copied into the label header.
- The labeled packets (marked with EXP) are sent to another LSR in the core MPLS network or to an LER at the egress edge.

Configuring QoS

QoS configuration involves multiple procedures for QoS processing as described in the following sections.

Configuring QoS for control traffic

Configure the Traffic Manager (TM) CPU port shaper rate (all towers) to the line card (LC) CPU.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Set the TM CPU port shaper on slot 1 to 4000 Kbps with a burst size of 1KB.

```
device(config)# qos cpu slot 1 port shaper rate 4000 burst 1
```

3. Return to privileged exec mode,

```
device(config)# exit
```

4. Verify the configuration.

```
device# show run qos cpu
qos cpu slot 1 port shaper rate 4000 burst 1
```

5. Save the configuration.

```
device# copy running-config startup-config
```

QoS for control traffic configuration example

```
device# configure terminal
device(config)# qos cpu slot 1 port shaper rate 4000 burst 1
device(config)# exit
device# show run qos cpu
device# copy running-config startup-config
```

Configuring CoS-to-traffic class mappings

Follow these tasks to configure QoS CoS-to-traffic class mappings.

Configuring a CoS-to-traffic class mutation map

Follow these steps to configure a QoS CoS-to-traffic class mutation map.

The ingress 802.1p priority values can be used to classify traffic to a specific traffic class (priority queue) and drop precedence. This can be done by configuring a cos-to-traffic class map. The drop precedence is optional in all below commands.

If a CoS-to-traffic class mutation map is not defined, the default CoS is used as value of the traffic class, and 0 is used for the drop precedence.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name a CoS-to-traffic class map.

```
device(config)# qos map cos-traffic-class cosTCMap
```

3. Map ingress CoS value to the CoS-to-traffic class map traffic-class and drop precedence values.

```
device(config-cos-traffic-class-cosTCMap)# map cos 4 to traffic-class 3 drop-precedence 0
device(config-cos-traffic-class-cosTCMap)# map cos 5 to traffic-class 5 drop-precedence 1
device(config-cos-traffic-class-cosTCMap)# map cos 6 to traffic-class 6 drop-precedence 0
device(config-cos-traffic-class-cosTCMap)# map cos 7 to traffic-class 6 drop-precedence 1
```

4. Return to privileged exec mode.

```
device(config-cos-traffic-class-cosTCMap)# end
```

5. Verify the configuration.

```
device# show qos maps cos-traffic-class
```

```
Cos-to-Traffic Class map 'cosTCMap'
  In-Cos   : 0  1  2  3  4  5  6  7
-----
TrafficClass : 0  1  2  3  3  6  6  6
DropPrecedence: 0  0  0  0  0  1  0  1
```

```
Enabled on the following interfaces:
```

QoS CoS-to-traffic class map configuration example

```
device# configure terminal
device(config)# qos map cos-traffic-class cosTCMap
device(config-cos-traffic-class-cosTCMap)# map cos 4 to traffic-class 3 drop-precedence 0
device(config-cos-traffic-class-cosTCMap)# map cos 5 to traffic-class 5 drop-precedence 1
device(config-cos-traffic-class-cosTCMap)# map cos 6 to traffic-class 6 drop-precedence 0
device(config-cos-traffic-class-cosTCMap)# map cos 7 to traffic-class 6 drop-precedence 1
device(config-cos-traffic-class-cosTCMap)# end
device# show qos maps cos-traffic-class
```

Applying a CoS-to-traffic class mutation map to an interface

Follow these steps to apply a QoS CoS-to-traffic class map to an interface.

You have configured a QoS CoS-to-traffic class map.

The internal traffic class can be mapped to the outgoing PCP value when the packet egresses the switch. A user can create a priority mapping table using a CoS mutation map. This CoS mutation map can then be applied to an egress interface to effect the priority remapping. This feature only maps the incoming priority to outgoing priority.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode..

```
device(config)# interface ethernet 1/5
```

3. Apply the CoS-to-traffic class map to an ingress interface and return to privileged exec mode.

```
device(conf-if-eth-1/5)# qos cos-traffic-class tc_1
```

4. Return to privileged exec mode.

```
device(conf-if-eth-1/5)# end
```

5. Verify the configuration.

```
device# show qos maps cos-traffic-class tc_1

Cos-traffic-class map 'tc_1'
  In-Cos      : 0  1  2  3  4  5  6  7
  -----
Traffic-class: 5  5  5  5  5  5  5  5

Enabled on the following interfaces: Eth 1/5
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS CoS-to-traffic class mutation map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# qos cos-traffic-class tc_1
device(conf-if-eth-1/5)# end
device# show qos maps cos-traffic-class tc_1
device# copy running-config startup-config
```

Configuring DSCP mappings

Follow the tasks below to configure DSCP mappings.

Configuring a DSCP-to-DSCP mutation map

Follow these steps to create a DSCP mutation map and remap the incoming DSCP value of the ingress packet to egress DSCP values.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create the DSCP-to-DSCP mutation map by specifying a map name, which places the system in DSCP mutation mode so that you can map to traffic classes.

```
device(config)# qos map dscp-mutation dscpMap
```

3. Map ingress DSCP values to egress DSCP values.

- a) Set the DSCP input value 24 to output as DSCP value 50.

```
device(dscp-mutation-dscpMap)# map dscp 24 to dscp 50
```

- b) Set the DSCP input value 33 to output as DSCP value 35.

```
device(dscp-mutation-dscpMap)# map dscp 33 to dscp 35
```

- c) Set the DSCP input value 53 to output as DSCP value 61.

```
device(dscp-mutation-dscpMap)# map dscp 53 to dscp 61
```

- d) Set the DSCP input value 60 to output as DSCP value 40.

```
device(dscp-mutation-dscpMap)# map dscp 60 to dscp 40
```

4. Return to privileged exec mode.

```
device(dscp-mutation-dscpMap# end
```

5. Verify the configuration.

```
device# show qos map dscp-mutation dscpMap
Dscp-to-Dscp Mutation map 'dscpMap' (dscp= d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 :    00 01 02 03 04 05 06 07 08 09
1 :    10 11 12 13 14 15 16 17 18 19
2 :    20 21 22 23 24 25 26 27 28 29
3 :    30 31 32 33 34 35 36 37 38 39
4 :    40 41 42 43 44 45 46 47 48 49
5 :    50 51 52 53 54 55 56 57 58 59
6 :    40 61 62 63
```

Enabled on the following interfaces:

6. Save the running-config file to the startup-config file

```
device# copy running-config startup-config
```

QoS DSCP-to-DSCP mutation map configuration example

```
device# configure terminal
device(config)# qos map dscp-mutation dscpMap
device(dscp-mutation-dscpMap)# map dscp 60 to dscp 40
device(dscp-mutation-dscpMap)# map dscp 24 to dscp 50
device(dscp-mutation-dscpMap)# map dscp 33 to dscp 35
device(dscp-mutation-dscpMap)# map dscp 53 to dscp 61
device(dscp-mutation-dscpMap# end
device# show qos map dscp-mutation dscpMap
device# copy running-config startup-config
```

Applying a DSCP-to-DSCP mutation map to an egress interface

Follow these steps to apply a QoS DSCP-to-DSCP mutation map to an egress interface.

This feature allows you to take the normalized QoS in-DSCP value of the egressing IP packet, and bind it to an egress interface.

A QoS DSCP-to-DSCP mutation map has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/5
```

3. Enable the DSCP-to-DSCP mutation map on the interface.

```
device(conf-if-eth-1/5)# qos dscp-mutation dscpMap
```

4. Return to privileged exec mode.

```
device(conf-if-eth-1/5)# end
```

5. Verify the configuration.

```
device# show qos maps dscp-mutation

Dscp-to-Dscp Mutation map 'dscpMap+' (dscp= d1d2)
d1 : d2 0  1  2  3  4  5  6  7  8  9
-----
0 :      00 01 02 03 04 05 06 07 08 09
1 :      10 11 12 13 14 15 16 17 18 19
2 :      20 21 22 23 24 25 26 27 28 29
3 :      30 31 32 33 34 35 36 37 38 39
4 :      40 41 42 43 44 45 46 47 48 49
5 :      50 51 52 53 54 55 56 57 58 59
6 :      40 61 62 63
```

Enabled on the following interfaces: Eth 1/5

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS DSCP-to-DSCP mutation map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# qos dscp-mutation dscpMap
device(conf-if-eth-1/5)# end
device# show qos maps dscp-mutation
device# copy running-config startup-config
```

Configuring DSCP-to-CoS mappings

Follow these tasks to configure DSCP-to-CoS mappings.

Configuring a DSCP-to-CoS mutation map

Follow these steps to use the DSCP value of ingress packets to remap the egress 802.1p CoS priority values.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a named QoS DSCP-to-CoS mutation map.

```
device(config)# qos map dscp-cos dscpCosMap
```

This also places the system in dscp-cos map mode so that you can map DSCP values to CoS values.

3. Map ingress DSCP values to egress CoS values.

- a) DSCP value 23 is set to output as CoS priority 4.

```
device(dscp-cos-dscpCosMap)# map dscp 23 to cos 4
```

- b) DSCP values 43 are set to output as CoS priority 4.

```
device(dscp-cos-dscpCosMap)# map dscp 43 to cos 5
```

- c) DSCP value 53 is set to output as CoS priority 6.

```
device(dscp-cos-dscpCosMap)# map dscp 53 to cos 6
```

- d) DSCP value 63 is set to output as CoS priority 7.

```
device(dscp-cos-dscpCosMap)# map dscp 63 to cos 7
```

4. Return to privileged exec mode.

```
device(dscp-cos-dscpCosMap)# end
```

5. Verify the configuration.

```
device# show qos maps dscp-cos

Dscp-to-CoS map 'dscpCosMap' (dscp= d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 :    00 00 00 00 00 00 00 00 00 01 01
1 :    01 01 01 01 01 01 02 02 02 02 02
2 :    02 02 02 04 03 03 03 03 03 03 03
3 :    03 03 04 04 04 04 04 04 04 04 04
4 :    05 05 05 05 05 05 05 05 05 06 06
5 :    06 06 06 06 06 06 07 07 07 07 07
6 :    07 07 07 07
```

Enabled on the following interfaces:

6. Save the running-config file to the startup-config file

```
device# copy running-config startup-config
```

QoS DSCP-to-CoS mutation map configuration example

```
device# configure terminal
device(config)# qos map dscp-cos dscpCosMap
device(dscp-cos-dscpCos)# map dscp 43 to cos 4
device(dscp-cos-dscpCos)# map dscp 63 to cos 6
device(dscp-cos-dscpCos)# map dscp 53 to cos 5
device(dscp-cos-dscpCos)# map dscp 23 to cos 2
device(dscp-cos-dscpCosMap)# end
device# show qos maps dscp-cos
device# copy running-config startup-config
```

Applying a DSCP-to-CoS mutation map to an interface

Follow these steps to map an ingress DSCP value to an outgoing 802.1p value. This can be done by configuring a DSCP-to-CoS mutation map on the ingress interface.

A QoS DSCP-to-CoS mutation map has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/5
```

3. Enable the DSCP mutation map on the interface.

```
device(conf-if-eth-1/5)# qos dscp-cos dscpCosMap
```

4. Return to privileged exec mode.

```
device(conf-if-eth-1/5)# end
```

5. Verify the configuration.

```
device# show qos maps dscp-cos
```

```
Dscp-to-CoS map 'dscpCosMap' (dscp= d1d2)
d1 : d2 0  1  2  3  4  5  6  7  8  9
-----
0 :      00 00 00 00 00 00 00 00 01 01
1 :      01 01 01 01 01 01 02 02 02 02
2 :      02 02 02 04 03 03 03 03 03 03
3 :      03 03 04 04 04 04 04 04 04 04
4 :      05 05 05 05 05 05 05 05 06 06
5 :      06 06 06 06 06 06 06 07 07 07
6 :      07 07 07 07
```

Enabled on the following interfaces: Eth 1/5

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS DSCP-to-CoS mutation map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/5
device(conf-if-eth-1/5)# qos dscp-cos dscp_cos_1
device(conf-if-eth-1/5)# end
device# show qos interface ethernet 1/5
device# copy running-config startup-config
```

Configuring DSCP-to-traffic class mappings

Configuring a DSCP-to-traffic class mutation map

Follow these steps to configure a QoS DSCP-to-traffic class map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name a QoS DSCP-to-traffic class map.

```
device(config)# qos map dscp-traffic-class dscpTcMap
```

3. Define the QoS DSCP-to-traffic class values.

```
device(config-dscp-traffic-class-dscpTcMap)# map dscp 10 to traffic-class 3
device(config-dscp-traffic-class-dscpTcMap)# map dscp 40 to traffic-class 4
device(config-dscp-traffic-class-dscpTcMap)# map dscp 45 to traffic-class 5
device(config-dscp-traffic-class-dscpTcMap)# map dscp 52 to traffic-class 3
```

The default value is used for those DSCP that are not explicitly defined.

4. Return to privileged exec mode.

```
device(config-dscp-traffic-class-dscpTcMap)# end
```

5. Verify the configuration.

```
device# show qos maps dscp-traffic-class

Dscp-to-Traffic-Class map 'dscpTcMap'
{x/y: traffic-class = x, drop-precedence = y & dscp = d1d2}
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 : 0/0 0/0 0/0 0/0 0/0 0/0 0/0 0/0 4/2 1/0
1 : 1/0 1/0 1/0 1/0 1/0 1/0 2/0 2/0 2/0 2/0
2 : 2/0 2/0 2/0 2/0 3/0 3/0 3/0 3/0 3/0 3/0
3 : 3/0 3/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0
4 : 5/0 5/0 5/0 5/0 5/0 5/0 5/0 5/0 6/0 6/0
5 : 6/0 6/0 6/0 6/0 6/0 6/0 7/0 7/0 7/0 7/0
6 : 7/0 7/0 7/0 7/0
```

Enabled on the following interfaces:

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

QoS DSCP-to-traffic class and drop precedence map configuration example

```
device# configure terminal
device(config)# qos map dscp-traffic-class dscpTcMap
device(config-dscp-traffic-class-dscpTcMap)# map dscp 10 to traffic-class 3
device(config-dscp-traffic-class-dscpTcMap)# map dscp 40 to traffic-class 4
device(config-dscp-traffic-class-dscpTcMap)# map dscp 45 to traffic-class 5
device(config-dscp-traffic-class-dscpTcMap)# map dscp 52 to traffic-class 3
device(config-dscp-traffic-class-dscpTcMap)# end
device# show qos maps dscp-traffic-class
device# copy running-config startup-config
```

Applying a DSCP-to-traffic class mutation map to an interface

Follow these steps to apply a QoS DSCP-to-traffic class map to an ingress interface.

A QoS DSCP-to-traffic class map has been configured

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

3. Activate the QoS DSCP-to-traffic class mutation map on the interface.

```
device(config-if-eth-1/2)# qos map dscp-traffic-class dscpTcMap
```

4. Return to privileged exec mode.

```
device(config-if-eth-1/2)# end
```

5. Verify the configuration

```
device# show qos maps dscp-traffic-class
```

```
Dscp-to-Traffic-Class map 'dscpTcMap'
{x/y: traffic-class = x, drop-precedence = y & dscp = d1d2}
d1 : d2  0   1   2   3   4   5   6   7   8   9
-----
0 :      0/0 0/0 0/0 0/0 0/0 0/0 0/0 0/0 4/2 1/0
1 :      1/0 1/0 1/0 1/0 1/0 1/0 2/0 2/0 2/0 2/0
2 :      2/0 2/0 2/0 2/0 3/0 3/0 3/0 3/0 3/0 3/0
3 :      3/0 3/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0
4 :      5/0 5/0 5/0 5/0 5/0 5/0 5/0 5/0 6/0 6/0
5 :      6/0 6/0 6/0 6/0 6/0 6/0 7/0 7/0 7/0 7/0
6 :      7/0 7/0 7/0 7/0
```

Enabled on the following interfaces: Eth 1/2

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS DSCP-to-traffic class map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/2
device(config-if-eth-1/2)# qos map dscp-traffic-class dscpTcMap
device(config-if-eth-1/2)# end
device# show qos maps dscp-traffic-class
device# copy running-config startup-config
```

Configuring a DSCP-to-traffic class and drop precedence mutation map

Follow these steps to configure a QoS DSCP to traffic class and drop precedence mutation map.

1. Enter global configuration mode.

```
device# configure terminal
```


2. Create and name a QoS DSCP-to-traffic class and drop precedence mutation map.

```
device(config)# qos map dscp-traffic-class dscpTcDpMap
```

3. Define the QoS DSCP-to-traffic class and drop precedence values.

```
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 10 to traffic-class 3 drop-precedence 1
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 40 to traffic-class 4 drop-precedence 1
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 45 to traffic-class 5 drop-precedence 0
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 52 to traffic-class 3 drop-precedence 1
```

If a default DSCP-to-traffic class map is not defined, then the IP precedence bits (first 3 bits) of the DSCP are used as the traffic class for the map, and drop precedence is given a value of 0.

4. Return to privileged exec mode.

```
device(config-dscp-traffic-class-dscpTcDpMap)# end
```

5. Verify the configuration.

```
device# show qos maps dscp-traffic-class
DSCP-to-TC Map: a1 (x/y: TC = x, DP = y, DSCP = d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 : 0/0 0/0 0/0 0/0 0/0 0/0 0/0 0/0 1/0 1/0
1 : 1/0 1/0 1/0 1/0 1/0 1/0 2/0 2/0 2/0 2/0
2 : 2/0 2/0 2/0 2/0 3/0 3/0 3/0 3/0 3/0 3/0
3 : 3/0 3/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0
4 : 5/0 5/0 5/0 5/0 5/0 5/0 5/0 5/0 6/0 6/0
5 : 6/0 6/0 6/0 6/0 6/0 6/0 7/0 7/0 7/0 7/0
6 : 7/0 7/0 7/0 7/0
```

Enabled on the following interfaces: >>> map a1 is not applied on any interface.

CoS = Class of Service, TC = Traffic Class, DP = Drop Precedence.

6. Save the running-config file to the startup-config file

```
device# copy running-config startup-config
```

QoS DSCP to traffic class and drop precedence mutation map configuration example

```
device# configure terminal
device(config)# qos map dscp-traffic-class dscpTcDpMap
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 10 to traffic-class 3 drop-precedence 1
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 40 to traffic-class 4 drop-precedence 1
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 45 to traffic-class 5 drop-precedence 0
device(config-dscp-traffic-class-dscpTcMap)# map dscp-value 52 to traffic-class 3 drop-precedence 1
device(dscp-traffic-class-dscpTcDpMap)# end
device# show qos maps dscp-traffic-class dscpTcDpMap
device# copy running-config startup-config
```

Applying a DSCP-to-traffic class and drop precedence mutation map to an interface

Follow these steps to apply a DSCP-to-traffic class and drop precedence mutation map to an ingress interface.

A QoS DSCP-to-traffic class and drop precedence mutation map has been configured

The ingress DSCP value can be used to classify traffic in to a specific traffic class and drop precedence by applying a DSCP-to-traffic class and drop precedence mutation map on the ingress interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

3. Enable the QoS DSCP-to-traffic class and drop precedence mutation map on the interface.

```
device(config-if-eth-1/2)# qos dscp-traffic-class dscpTcDpMap
```

4. Return to privileged exec mode.

```
device(config-if-eth-1/2)# end
```

5. Verify the configuration

```
device# show qos maps dscp-traffic-class dscpTcDpMap
DSCP-to-TC Map: dscpTc (x/y: TC = x, DP = y, DSCP = d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 : 0/0 0/0 0/0 0/0 0/0 0/0 0/0 0/0 1/0 1/0
1 : 1/0 1/0 1/0 1/0 1/0 1/0 2/0 2/0 2/0 2/0
2 : 2/0 2/0 2/0 2/0 3/0 3/0 3/0 3/0 3/0 3/0
3 : 3/0 3/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0 4/0
4 : 5/0 5/0 5/0 5/0 5/0 5/0 5/0 5/0 6/0 6/0
5 : 6/0 6/0 6/0 6/0 6/0 6/0 7/0 7/0 7/0 7/0
6 : 7/0 7/0 7/0 7/0
```

Enabled on the following interfaces: Eth 1/2, Eth 3/12

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS DSCP-to-traffic class map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/2
device(config-if-eth-1/2)# qos dscp-traffic-class dscpTcDpMap
device(config-if-eth-1/2)# end
device# show qos maps dscp-traffic-class dscpTcDpMap
device# copy running-config startup-config
```

Concept

Configuring a traffic class-to-CoS mutation map

Follow these steps to configure QoS traffic class-to-CoS mutation map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure the QoS traffic class-to-CoS mutation map.

```
device(config)# qos map traffic-class-cos CoSMap 1 1 2 3 4 4 4 3
```

If the QoS CoS mutation map is not configured, then the default CoS mutation map is used with 1:1 mapping for the traffic class-to-PCP values.

3. Return to privileged exec mode.

```
device(config)# exit
```

4. Verify the configuration.

```
device# show qos maps traffic-class-cos

Traffic Class-to-Cos Mutation map 'CoSMap'
TrafficClass: 0  1  2  3  4  5  6  7
-----
Out-Cos: 1  1  2  3  4  4  4  3

Enabled on the following interfaces:
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Traffic class-to-CoS mutation map configuration example

```
device# configure terminal
device(config)# qos map traffic-class-cos CoSMap 1 1 2 3 4 4 4 3
device(config)# exit
device# show qos maps traffic-class-cos
device# copy running-config startup-config
```

Applying a traffic class-to-CoS mutation map to an egress interface

Follow these steps to apply a QoS traffic class-to-CoS mutation map to an egress interface.

A QoS traffic class-to-CoS mutation map has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 2/2
```

3. Apply the configured QoS traffic class-to-CoS map to the interface.

```
device(conf-if-eth-2/2)# qos traffic-class-cos tcCoSMap
```

4. Return to privileged exec mode.

```
device(conf-if-eth-2/2)# end
```

5. Verify the configuration.

```
device# show qos maps traffic-class-cos tcCoSMap

[Note: CoS = Class of Service, TC = Traffic Class, DP = Drop Precedence]
TC-to-CoS Map: tcCoSMap
      In-TC: 0  1  2  3  4  5  6  7
-----
Out-CoS (DP=0): 0  1  2  3  4  2  6  7
Out-CoS (DP=1): 0  1  2  3  4  5  6  7
Out-CoS (DP=2): 0  1  1  3  4  2  6  7
Out-CoS (DP=3): 0  1  2  3  4  5  6  7

Enabled on the following interfaces:
Eth 2/2
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS traffic class-to-CoS mutation map to an egress interface configuration example

```
device# configure terminal
device(config)# interface ethernet 2/2
device(config-if-eth-2/2)# qos traffic-class-cos tcCoSMap
device(config-if-eth-2/2)# end
device# show qos maps traffic-class-cos tcCoSMap
device# copy running-config startup-config
```

Configuring congestion control

Refer to the section [Congestion control](#) on page 43.

Configuring WRED

WRED is configurable on the ingress side to control when to perform a tail drop or Random Early Drop (RED). Follow these steps to configure WRED.

1. Enter configuration mode.

```
device# configure terminal
```

2. Create a WRED profile identified as profile 1, set the thresholds, and set the drop probability.

```
device(config)# qos red-profile 1 min-threshold 30 max-threshold 60 drop-probability 44
```

3. Verify the WRED configuration.

```
device(config)# do show qos red profiles 1

Red Profile 1
  Minimum Threshold: 30
  Maximum Threshold: 60
  Drop Probability: 44
```

4. Return to privileged exec mode.

```
device(config)# exit
```

5. Save the configuration.

```
device# copy running-config startup-config
```

WRED configuration example

```
device# configure terminal
device(config)# qos red-profile 1 min-threshold 30 max-threshold 60 drop-probability 44
device(config)# do show qos red profiles 1
device(config)# exit
device# copy running-config startup-config
```

Configuring link level flow control

Link level flow control allows a congested receiver to communicate a PAUSE frame to a transmitter to stop data transmission until the congestion is cleared.

Link level flow control can only be configured at the interface level.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 3/18
```

3. Enable linked level flow control in the receive direction for the port.

```
device(conf-eth-3/18)# qos flowcontrol rx on
```

4. Return to privileged exec mode.

```
device(conf-eth-3/18)# end
```

5. Verify the configuration

```
device# show qos flowcontrol interface ethernet 3/18
Interface Ethernet 3/18
Mode 802.3x
  TX      RX      TX Output Paused
Admin  Admin  Frames  512 BitTimes
-----
  Off      On
```

6. Save the configuration

```
device# copy running-config startup-config
```

Link level flow control configuration example

```
device# configure terminal
device(config)# interface ethernet 3/18
device(conf-eth-3/18)# qos flowcontrol rx on
device(conf-eth-3/18)# end
device# show qos flowcontrol interface ethernet 3/18
device# copy running-config startup-config
```

Configuring scheduling

Configuring strict priority egress scheduling

Follow these steps to configure strict priority scheduling.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Select the policy map.

```
device(config)# policy-map policy_1
```

3. Select the classification.

```
device(config-policymap)# class default
```

4. Specify the scheduling attributes.

```
device(config-policymap-class)# scheduler strict-priority 3 dwrr 10 10 10 10 60 TC535000 TC6 36000 TC7 37000
```

For complete information, refer to the *Brocade SLX-OS Layer 2 Configuration Guide*.

5. Return to privileged exec mode.

```
device(config-policymap)# end
```

6. Verify the configuration.

```
device# show running-config | include strict-priority
scheduler strict-priority 3 dwrr 10 10 10 10 60 TC5 40000 TC6 41000 TC7 42000
```

7. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Strict priority scheduling configuration example

```
device# configure terminal
device(config)# policy-map policy_1
device(config-policymap)# class default
device(config-policymap-class)# scheduler strict-priority 3 dwrr 10 10 10 10 60 TC535000 TC6 36000 TC7 37000
device(config-policymap-class)# end
device# show running-config | include strict-priority
device# copy running-config startup-config
```

Configuring the QoS multicast queue to a strict priority on an interface

To configure multicast QoS follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device# interface ethernet 1/1
```

3. Set the QoS scheduler to a strict priority.

```
device(config-if-eth-1/1)# qos rx-queue multicast burst-rate 400000
```

The burst rate ranges from 0 to the maximum tower rate.

4. Return to privileged exec mode.

```
device(config-if-eth-1/1)# end
```

5. Verify the configuration.

```
device# show qos interface 1/1
```

6. Save the running-config file to the startup-config file

```
device# copy running-config startup-config
```

QoS Multicast queue to a strict priority on an interface configuration example.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if-eth-1/1)# qos rx-queue multicast burst-rate 400000
device(config-if-eth-1/1)# end
device# show qos interface 1/1
device# copy running-config startup-config
```

Configuring port-based QoS

Follow these tasks to configure ingress flow-based QoS.

Configuring a class map

The class map classifies traffic based on match criteria. If traffic matches the criteria, it belongs to the class.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a named class map and enter class map configuration mode

```
device(config)# class-map class_map_2
```

3. Provide match criteria for the class.

```
device(config-classmap)# match any
```

The **match any** option is not supported. Only **access-group** *acl-name* can be matched.

4. Return to privileged exec mode.

```
device(config-classmap)# end
```

5. Verify the configuration.

```
device# show running-config | include class_map_2
...
class-map class_map_2
...
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Classification map configuration example

```
device# configure terminal
device(config)# class-map class_map_2
device(config-classmap)# match any
device(config-classmap)# end
device# show running-config | include class_map_2
device# copy running-config startup-config
```

Configuring a class map using an ACL

To configure a classification or class map by using an ACL, follow these steps.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create an IP access list to define the traffic.

- a) Create and name a standard IP access list and enter IP ACL configuration mode.

```
device(config)# ip access-list standard ip_acl
```

- b) Allow traffic from a specific IP address.

```
device(conf-ipacl-std)# permit host 10.10.10.0
```

- c) Exit IP ACL configuration mode to global configuration mode.

```
device(conf-ipacl-std)# exit
```

Refer to the *Brocade SLX-OS Security Configuration Guide* for details on creating access lists.

3. Verify the IP ACL.

```
device(config)# do show running-config | include ip_acl
ip access-list standard ip_acl
```

4. Create and name a class map.

```
device(config)# class-map class_1
```

5. Provide match criteria for the class.

```
device(config-classmap)# match access-group ip_acl
```

6. Return to privileged exec mode.

```
device(config-classmap)# end
```


7. Verify the class configuration.

```
device# show running-config | include class
...
class-map cee
class-map class_1
class-map default
```

8. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Class map using an ACL configuration example

```
device# configure terminal
device(config)# ip access-list standard IP_acl
device(config-ipacl-std)# permit host 10.10.10.0
device(config-ipacl-std)# exit
device(config)# do show running-config | include ip_acl
device(config)# class-map class_1
device(config-classmap)# match access-group ip_acl
device(config-classmap)# end
device# show running-config | include class
device# copy running-config startup-config
```

Configuring a policy map

Follow these steps to create a policy map.

A rate limit policy map is configured and then applied to the type of QoS flow defined by the class map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name a policy map.

```
device(config)# policy-map policyMap1
```

3. Return to privileged exec mode.

```
device(config-policymap)# end
```

4. Verify the configuration

```
device# show policy-map

Number of policy maps : 2

Policy-Map policy
  Bound To:None

Policy-Map policyMap1
  Bound To:None
```

5. Display policy map details.

```
device# show policy-map detail policy
Policy-Map policy
  Class cmap1
    Police cir 43454
    Bound To: ET 1/33(in), Te 5/33(out)
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Policy map configuration example

```
device# configure terminal
device(config)# policy-map policyMap1
device(config-policymap)# end
device# show policy-map
device# copy running-config startup-config
```

Binding the policy map at the system level

Follow these steps to apply policing parameters to an interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Bind the policy map to inbound traffic.

```
device(config)# qos service-policy in policyMap1
```

You cannot use a policy map that is bound to class maps, default, or CEE maps.

3. Return to privileged exec mode.

```
device(config-service-policy-in/policyMap1)# end
```

4. Verify the configuration.

```
device# show policy-map detail policyMap1

Policy-Map policyMap1
  Class class_1
    Police cir 40000 cbs 5000 eir 40000 ebs 3000 conform-tc 6 exceed-tc 2 conform-dscp 61 exceed-
dscp 63

  Bound To: none
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Bind a policy map globally configuration example

```
device# configure terminal
device(config)# qos service-policy in policyMap1
device(config-service-policy-in/policyMap1)# end
device# show policy-map detail policyMap1
device# copy running-config startup-config
```

Binding the policy map to an interface

Follow these step to configure the default remapping priorities.

Consider the following rules when binding a policy map to an interface:

- You can bind the same policy map to multiple interfaces but only one policy per interface per direction is allowed.
- You cannot bind policy maps to an interface if the policy map has no class map associations.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Enter interface configuration mode.

```
device(config)# interface ethernet 1/40
```

3. Bind a policy map to egress traffic on the interface.

```
device(config-if-eth-1/40)# service-policy out policyMap1
```

4. Bind a policy map to ingress traffic on the interface.

```
device(config-if-eth-1/40)# service-policy in policyMap1
```

5. Return to privileged exec mode.

```
device(config-if-eth-1/40)# end
```

6. Verify the configuration.

```
device# show policy-map interface ethernet 1/40
```

```
Ingress Direction :
  Policy-Map policyMap1
    Class class_1
      matches 0 packets
      Police cir 40000 cbs 5000 eir 40000 ebs 3000 conform-tc 6 exceed-tc 2 conform-dscp 61 exceed-
dscp 63
      Stats:
        Operational cir:39856 cbs:5000 eir:39856 ebs:3000
        Conform Byte:0 Exceed Byte:0 Violate Byte:0

Egress Direction :
  Policy-Map policyMap1
    Class class_1
      matches 0 packets
      Police cir 40000 cbs 5000 eir 40000 ebs 3000 conform-tc 6 exceed-tc 2 conform-dscp 61 exceed-
dscp 63
      Stats:
        Operational cir:39856 cbs:5000 eir:39856 ebs:3000
        Conform Byte:0 Exceed Byte:0 Violate Byte:0
```

7. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Bind the policy map to an interface configuration example

```
device# configure terminal
device(config)# interface ethernet 1/40
device(config-if-eth-1/40)# service-policy out policyMap1
device(config-if-eth-1/40)# service-policy in policyMap1
device(config-if-eth-1/40)# end
device# show policy-map interface ethernet 1/40
device# copy running-config startup-config
```

Configuring QoS mutation map actions

Follow these steps to configure a QoS mutation map.

A policy map and a class map have been configured.

Different kinds of mutations can be used depending on the command. For complete information, refer to relevant Command Reference guide. The available commands are **cos-mutation**, **cos-traffic-class**, **dscp-cos**, **dscp-mutation**, and **dscp-traffic-class**.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Select the policy map.

```
device(config)# policy-map policyMap1
```

3. Select the class.

```
device(config-policymap)# class default
```

4. Specify the mutation map.

```
device(config-policyclass)# map dscp-cos all-zero-map
```

In this example a DSCP-to-CoS mutation is configured.

5. Return to privileged exec mode.

```
device(config-policyclass)# end
```

6. Verify the configuration.

```
device# show run policy-map
policy-map policyMap1
  class default
    map dscp-cos all-zero-map
!
```

7. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

QoS mutation map configuration example

```
device# configure terminal
device(config)# policy-map policyMap1
device(config-policymap)# class default
device(config-policyclass)# map dscp-cos all-zero-map
device(config-policyclass)# end
device# show run policy-map
device# copy running-config startup-config
```

Applying QoS mutation maps to an interface

Follow these steps to specify the mutation map to be used on a port.

A mutation map has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Select the policy map.

```
device(config)# policy-map policyMap1
```

3. Enter interface configuration mode.

```
device(config-policymap)# interface ethernet 1/1
```

4. Apply a DSCP-to-DSCP mutation map to the interface.

```
ddevice(conf-if-eth-1/1)# qos dscp-mutation dscpMutMap
```

Different kinds of mutations can be used.

5. Return to privileged exec mode.

```
device(conf-if-eth-1/1)# end
```

6. Verify the configuration.

```
device# show qos map dscp-mutation dscpMutMap

Dscp-to-Dscp Mutation map 'dscpMutMap' (dscp= d1d2)
d1 :  d2 0  1  2  3  4  5  6  7  8  9
-----
0 :   11 11 11 03 11 11 11 11 11 11
1 :   11 11 11 11 11 11 11 11 11 11
2 :   11 11 11 23 24 25 26 27 28 29
3 :   30 31 32 33 34 35 36 37 38 39
4 :   40 41 42 43 44 45 46 47 48 49
5 :   50 51 52 53 54 11 11 11 11 11
6 :   11 11 11 11

Enabled on the following interfaces:
Eth 1/1
```

7. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a QoS mutation map to an interface configuration example

```
device# configure terminal
device(config)# policy-map policyMap1
device(config-policymap)# interface ethernet 1/1
device(conf-if-eth-1/1)# qos dscp-mutation dscpMutMap
device(conf-if-eth-1/1)# end
device# show qos map dscp-mutation dscpMutMap
device# copy running-config startup-config
```

Configuring the QoS policing rate

To configure QoS for rate policing on an interface, you apply a policy map to the interface.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create a policy map and enter policy map configuration mode.

```
device(config)# policy-map policy_1
```

3. Under policy map configuration mode, attach the classification map to the policy map.

```
device(config-policymap)# class default
```

4. Set the QoS action.

```
device(config-policymap-class)# police cir 40000
```

5. Return to privileged exec mode.

```
device(config-policymap-class)# end
```

6. Verify the configuration.

```
device# show policy-map detail policy_1
```

```
Policy-Map P1
  Class default
    Police cir 40000

  Bound To:None
```

7. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

QoS policing rate configuration example

```
device# configure terminal
device(config)# policy-map policy_1
device(config-policymap)# class default
device(config-policymap-class)# police cir 40000
device(config-policymap-class)# end
device# show policy-map detail policy_1
device# copy running-config startup-config
```

Applying the QoS policing rate to an interface

Follow these steps to apply the policing rate to an interface.

A policy map has been configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Switch to interface configuration mode.

```
device(config-policymap-class)# interface ethernet 1/49
```

3. Bind the egress policy map policy map to the interface.

```
device(config-if-eth-1/49)# service-policy out policy_1
```

In this case it is for rate shaping (outbound or egress) traffic.

4. Return to privileged exec mode.

```
device(config-if-eth-1/49)# end
```

In this case it is for rate shaping (outbound or egress) traffic.

5. Verify the configuration.

```
device# show policy-map

Number of policy maps : 2
...
Policy-Map policy_1
  Bound To: Et 1/49(out)
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

QoS policing rate on an interface configuration example

```
device# configure terminal
device(config-policy-map-class)# interface ethernet 1/49
device(config-if-eth-1/49)# service-policy out policy_1
device(config-if-eth-1/49)# end
device# show policy-map
device# copy running-config startup-config
```

Configuring virtual output queueing

Virtual output queueing (VOQ) is a technique where instead of one input traffic queue, multiple queues are maintained.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Configure an interface ingress queue, enter interface configuration mode.

```
device(config)# interface ethernet 1/2
```

3. Use the traffic class to set parameters for unicast packet handling on the interface.

```
device(config-if-eth-1/2)# qos rx-queue unicast traffic-class 3 min-queue-size 128 max-queue-size 1024
```

This command sets the values for the traffic class value, with a range of 0 through 7; minimum queue size, with a range of 0 through 1024 KB per-second (KBps); and the maximum queue size, with a range of 0 through 2048 MB per-second (MBps).

4. Use the traffic class to set parameters for multicast packet handling on the interface.

- a) Configure multicast data best effort rate

```
device(conf-if-eth-1/2)# qos rx-queue multicast best-effort-rate 3000
```

The range of values is from 0 through 6000000000 kilobits per-second (kbps).

- b) Configure multicast data guarantee rate

```
device(conf-if-eth-1/2)# qos rx-queue multicast guarantee-rate 30000
```

The range of values is from 0 through 6000000000 kbps.

- c) Set parameter values, by traffic class, for multicast packet handling on the interface.

```
device(conf-if-eth-1/2)# qos rx-queue multicast traffic-class 3 min-queue-size 512 max-queue-size 1024
```

5. Configure the CoS thresholds for the interface.

```
device(conf-if-eth-1/2)# qos rx-queue cos-threshold 10 10 10 10 10 5 10 5
```

You configure eight percentages for the eight CoS thresholds. The combined total cannot exceed 100%. In the above command the first position configures CoS 0 to 10%, the second configures CoS 1 to 10%, and so on.

6. Return to privileged exec mode

```
device(conf-if-eth-1/2)# end
```

7. Verify the configuration.

```
device# show qos rx-queue interface all

device# show running-config | include queue
qos rx-queue queue-size 512
```

8. View the buffer pool statistics..

```
device# show buffmgr stats slot 1
```

9. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Virtual output queueing configuration example

```
device# configure terminal
device(config)# interface ethernet 1/2
device(conf-if-eth-1/2)# qos rx-queue unicast traffic-class 3 min-queue-size 128 max-queue-size 1024
device(conf-if-eth-1/2)# qos rx-queue multicast best-effort-rate 3000
device(conf-if-eth-1/2)# qos rx-queue multicast guarantee-rate 30000
device(conf-if-eth-1/2)# qos rx-queue multicast traffic-class 3 min-queue-size 512 max-queue-size 1024
device(conf-if-eth-1/2)# qos rx-queue cos-threshold 10 10 10 10 10 5 10 5
device(conf-if-eth-1/2)# end
device# show qos rx-queue interface all
device# copy running-config startup-config
```

Configuring MPLS QoS DSCP mutation maps

Configuring an MPLS QoS DSCP-to-EXP mutation map

Follow these steps to configure an MPLS QoS DSCP-to-EXP mutation map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name an MPLS QoS DSCP-to-EXP mutation map.

```
device(config)# qos-mpls map dscp-exp dscpExpMap
```

3. Define the DSCP-to-EXP values.

```
device(dscp-exp-dscpExpMap)# dscp 0 to exp 7
device(dscp-exp-dscpExpMap)# dscp 3 to exp 4
device(dscp-exp-dscpExpMap)# dscp 61 to exp 5
device(dscp-exp-dscpExpMap)# end
```

4. Verify the configuration.

```
device# show qos-mpls maps dscp-exp dscpExpMap
```

```
dscp-exp map 'dscpExpMap' (dscp= d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 : 00 00 00 00 00 00 00 00 01 01
1 : 01 01 01 01 01 01 02 02 02 02
2 : 02 02 02 02 03 03 03 03 03 03
3 : 03 03 04 04 04 04 04 04 04 04
4 : 05 05 05 05 05 05 05 05 06 06
5 : 06 06 06 06 06 06 07 07 07 07
6 : 07 07 07 07
Enabled on the following slots:
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

MPLS QoS DSCP-to-EXP mutation map configuration example

```
device# configure terminal
device(config)# qos-mpls map dscp-exp dscpExpMap
device(dscp-exp-dscpExpMap)# dscp 0 to exp 7
device(dscp-exp-dscpExpMap)# dscp 3 to exp 4
device(dscp-exp-dscpExpMap)# dscp 61 to exp 5
device(dscp-exp-dscpExpMap)# end
device# show qos maps dscp-exp dscpExpMap
device# copy running-config startup-config
```

Applying an MPLS QoS DSCP-to-EXP mutation map globally

Follow these steps to apply an MPLS QoS DSCP-to-EXP mutation map globally.

An MPLS QoS DSCP-to-EXP map is configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Apply an MPLS QoS DSCP-to-EXP map globally.

```
device(config)# qos-mpls map-apply dscp-exp dscpExpMap all
```

- Return to privileged exec mode

```
device(config)# end
```

- Verify the configuration.

```
device# show qos maps dscp-exp

dscp-exp map 'dscpExpMap' (dscp= d1d2)
d1 : d2 0 1 2 3 4 5 6 7 8 9
-----
0 : 00 00 00 00 00 00 00 00 01 01
1 : 01 01 01 01 01 01 02 02 02 02
2 : 02 02 02 02 03 03 03 03 03 03
3 : 03 03 04 04 04 04 04 04 04 04
4 : 05 05 05 05 05 05 05 05 06 06
5 : 06 06 06 06 06 06 07 07 07 07
6 : 07 07 07 07
    Enabled on the following slots:
```

- Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply an MPLS QoS DSCP-to-EXP mutation map globally configuration example

```
device# configure terminal
device(config)# qos-mpls map-apply dscp-exp dscpExpMap all
device(config)# end
device# show qos-mpls maps dscp-exp
device# copy running-config startup-config
```

Configuring MPLS QoS EXP mutation maps

Configuring an MPLS QoS EXP-to-DSCP mutation map

Follow these steps to configure an MPLS QoS EXP-to-DSCP mutation map.

- Enter global configuration mode.

```
device# configure terminal
```

- Create and name an MPLS QoS EXP-to-DSCP mutation map.

```
device(config)# qos-mpls map exp-dscp expDscpMap
```

- Define the EXP-to-DSCP values.

```
device(exp-dscp-expDscpMap)# exp 0 to dscp 7
device(exp-dscp-expDscpMap)# exp 1 to dscp 23
device(exp-dscp-expDscpMap)# exp 3 to dscp 31
device(exp-dscp-expDscpMap)# exp 4 to dscp 62
```

The default value is used for those DSCP that are not explicitly defined.

- Return to privileged exec mode

```
device(exp-dscp-expDscpMap)# end
```

5. Verify the configuration.

```
device# show qos maps exp-dscp expDscpMap

exp-dscp map 'expDscpMap'
  Exp   : 0  1  2  3  4  5  6  7
  -----
  DSCP  : 0  2  4  3  6  4  5  7

Enabled on the following slots:
```

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

MPLS QoS EXP-to-DSCP mutation map configuration example

```
device# configure terminal
device(config)# qos-mpls map exp-dscp expDscpMap
device(exp-dscp-expDscpMap)# exp 0 to priority 7 drop-precedence 0
device(exp-dscp-expDscpMap)# exp 0 to dscp 7
device(exp-dscp-expDscpMap)# exp 1 to dscp 23
device(exp-dscp-expDscpMap)# exp 3 to dscp 31
device(exp-dscp-expDscpMap)# exp 4 to dscp 62
device(exp-dscp-expDscpMap)# end
device# show qos maps exp-dscp expDscpMap
device# copy running-config startup-config
```

Applying an MPLS QoS EXP-to-DSCP mutation map globally

Follow these steps to apply an MPLS QoS EXP-to-DSCP mutation map globally.

An MPLS QoS EXP-to-DSCP mutation map is configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Apply an MPLS QoS EXP-to-DSCP mutation map globally.

```
device(config)# qos-mpls map-apply exp-dscp expDscpMap all
```

3. Return to privileged exec mode

```
device(exp-dscp-expDscpMap) # end
```

4. Verify the configuration.

```
device# show qos-mpls maps exp-dscp

exp-dscp map 'expDscpMap'
  Exp   : 0  1  2  3  4  5  6  7
  -----
  DSCP  : 0  2  4  3  6  4  5  7

Enabled on the following slots:
  ALL
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply an MPLS QoS EXP-to-DSCP mutation map globally configuration example

```
device# configure terminal
device(config)# qos-mpls map-apply exp-dscp expDscpMap all
device(config)# end
device# show qos-mpls maps exp-dscp
device# copy running-config startup-config
```

Configuring an MPLS QoS EXP-to-traffic class mutation map

Follow these steps to configure an MPLS QoS EXP-to-traffic class mutation map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name an MPLS QoS EXP-to-traffic class mutation map.

```
device(config)# qos-mpls map exp-traffic-class expTcMap
```

3. Define the EXP-to-traffic class values.

```
device(exp-traffic-class-expTcMap)# exp 0 to traffic-class 7 drop-precedence 0
device(exp-traffic-class-expTcMap)# exp 1 to traffic-class 7 drop-precedence 1
device(exp-traffic-class-expTcMap)# exp 4 to traffic-class 7 drop-precedence 0
device(exp-traffic-class-expTcMap)# exp 5 to traffic-class 7 drop-precedence 1
device(exp-traffic-class-expTcMap)# exp 6 to traffic-class 7 drop-precedence 2
device(exp-traffic-class-expTcMap)# exp 7 to traffic-class 7 drop-precedence 3
```

You can enter eight mappings corresponding to EXP values of 0 to 7.

4. Return to privileged exec mode

```
device(exp-traffic-class-expTcMap)# end
```

5. Verify the configuration.

```
device# show qos-mpls maps exp-traffic-class
exp-traffic-class map 'expTcMap'
  Exp      :    0  1  2  3  4  5  6  7
-----
Traffic-class: 5  5  4  6  5  5  5  5
Drop-Preced  : 0  1  1  1  0  2  2  1
```

Enabled on the following slots:

6. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

MPLS QoS EXP-to-traffic class mutation map configuration example

```
device# configure terminal
device(config)# qos-mpls map exp-traffic-class expTcMap
device(exp-traffic-class-expTcMap)# exp 0 to traffic-class 7 drop-precedence 0
device(exp-traffic-class-expTcMap)# exp 1 to traffic-class 7 drop-precedence 1
device(exp-traffic-class-expTcMap)# exp 4 to traffic-class 7 drop-precedence 0
device(exp-traffic-class-expTcMap)# exp 5 to traffic-class 7 drop-precedence 1
device(exp-traffic-class-expTcMap)# exp 6 to traffic-class 7 drop-precedence 2
device(exp-traffic-class-expTcMap)# exp 7 to traffic-class 7 drop-precedence 3
device(exp-traffic-class-expTcMap)# end
device# show qos-mpls maps exp-traffic-class
device# copy running-config startup-config
```

Applying an MPLS QoS EXP-to-traffic class mutation map globally

Follow these steps to apply an MPLS QoS EXP-to-traffic class mutation map globally.

A MPLS QoS EXP-to-traffic class map is configured.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Apply the MPLS QoS EXP-to-traffic class map globally.

```
device(config)# qos-mpls map exp-traffic-class expTcMap ALL
```

3. Return to privileged exec mode

```
device(exp-traffic-class-expTcMap)# end
```

4. Verify the configuration.

```
device# show qos-mpls maps exp-traffic-class
```

```
exp-traffic-class map 'expTcMap'
  Exp      :    0  1  2  3  4  5  6  7
  -----
traffic-class : 5  5  4  6  5  5  5  5
drop-precedence: 0  1  1  1  0  2  2  1
```

```
Enabled on the following slots:
```

5. Save the running-config file to the startup-config file.

```
device# copy running-config startup-config
```

Apply a MPLS QoS EXP-to-traffic class map globally configuration example

```
device# configure terminal
device(config)# qos-mpls map exp-traffic-class expTcMap ALL
device(exp-traffic-class-expTcMap)# end
device# show qos-mpls maps exp-traffic-class
device# copy running-config startup-config
```

Configuring MPLS QoS traffic class mutation maps

Configuring an MPLS QoS traffic class-to-EXP mutation map

Follow these steps to configure an MPLS QoS traffic class-to-EXP mutation map.

1. Enter global configuration mode.

```
device# configure terminal
```

2. Create and name an MPLS QoS traffic class-to-EXP mutation map.

```
device(config)# qos-mpls map traffic-class-exp tcExpMap
```

3. Define the traffic class-to-EXP values.

```

device(traffic-class-exp-tcExpMap)# priority 0 drop-precedence 0 to exp 7
device(traffic-class-exp-tcExpMap)# priority 1 drop-precedence 0 to exp 6
device(traffic-class-exp-tcExpMap)# priority 4 drop-precedence 0 to exp 3
device(traffic-class-exp-tcExpMap)# priority 5 drop-precedence 0 to exp 2
device(traffic-class-exp-tcExpMap)# priority 6 drop-precedence 0 to exp 2
device(traffic-class-exp-tcExpMap)# priority 7 drop-precedence 0 to exp 1
device(traffic-class-exp-tcExpMap)# end

```

4. Return to privileged exec mode

```

device(traffic-class-exp-tcExpMap)# end

```

5. Verify the configuration.

```

device# show qos-mpls maps traffic-class-exp

traffic-class-cos map      'TcExpMap' (Drop-Precedence = dp)
dp: traffic-class : 0  1  2  3  4  5  6  7
-----
0: exp              : 7  6  2  3  3  2  2  1
1:                  : 0  1  2  3  4  5  6  7
2:                  : 0  1  2  3  4  5  6  7
3:                  : 0  1  2  3  4  5  6  7

```

Enabled on the following slots :

6. Save the running-config file to the startup-config file.

```

device# copy running-config startup-config

```

MPLS QoS traffic class-to-EXP mutation map configuration example

```

device# configure terminal
device(config)# qos-mpls map traffic-class-exp tcExpMap
device(traffic-class-exp-tcExpMap)# priority 0 drop-precedence 0 to exp 7
device(traffic-class-exp-tcExpMap)# priority 1 drop-precedence 0 to exp 6
device(traffic-class-exp-tcExpMap)# priority 4 drop-precedence 0 to exp 3
device(traffic-class-exp-tcExpMap)# priority 5 drop-precedence 0 to exp 2
device(traffic-class-exp-tcExpMap)# priority 6 drop-precedence 0 to exp 2
device(traffic-class-exp-tcExpMap)# priority 7 drop-precedence 0 to exp 1
device(traffic-class-exp-tcExpMap)# end
device# show qos-mpls maps traffic-class-exp
device# copy running-config startup-config

```

Applying an MPLS QoS traffic class-to-EXP mutation map globally

Follow these steps to apply an MPLS QoS traffic class-to-EXP mutation map globally.

An MPLS QoS traffic class-to-EXP map is configured.

1. Enter global configuration mode.

```

device# configure terminal

```

2. Apply an MPLS QoS traffic class-to-EXP mutation map globally.

```

device(config)# qos-mpls map-apply traffic-class-exp TcExpMap all

```

3. Return to privileged exec mode

```

device(config)# end

```

4. Verify the configuration.

```
device# show qos-mpls maps traffic-class-exp

traffic-class-cos map      'TcExpMap' (Drop-Precedence = dp)
dp: traffic-class : 0  1  2  3  4  5  6  7
-----
0: exp                : 7  6  2  3  3  2  2  1
1:                    : 0  1  2  3  4  5  6  7
2:                    : 0  1  2  3  4  5  6  7
3:                    : 0  1  2  3  4  5  6  7

Enabled on the following slots:
```

5. Save the configuration.

```
device# copy running-config startup-config
```

Apply a MPLS QoS traffic class-to-EXP mutation map globally configuration example

```
device# configure terminal
device(config)# qos-mpls map-apply traffic-class-exp TcExpMap all
device(config)# end
device# show qos-mpls maps traffic-class-exp
device# copy running-config startup-config
```


Traffic Management Counters and Statistics

• Counters and statistics overview	89
• Traffic management counter types.....	89
• Traffic management counters.....	90

Counters and statistics overview

These commands display traffic management counter statistics.

The SLX-OS uses algorithmic and sequential sampling simultaneously. This combination ensures that the entire counter engine database is periodically delivered into the software. You can set the sequential sampling timer to a lower rate as the overflow is prevented by algorithmic sampling.

Statistics collection mechanisms

In this implementation statistics collection uses Counter engines as a collection mechanisms.

With counter engines:

- There are 16 on-chip counter engines with 16k packet and octet counters each.
- Two mini counter engines B0/B1 are dedicated to the egress queue and support 4k 64-bit entries.
- Each counter engine can be individually assigned to a statistics source.
- A statistic flow is mapped to a set of counters within one of the counting engines.
- A packet is mapped to a counter pair within a counter set according to the packet's disposition (drop/forward status) and color.

Traffic management counter types

There are two counter typed that apply to traffic management (TM):

- TM device counter
- TM VOQ counter

TM device counters

This is a device level TM counter to track packet counts on a per chip level. There are Ingress and Egress device counters.

- Ingress counters keep track of:
 - Number of ingress packets
 - IQM enqueued and dequeued packets and bytes
 - Total discard packets
 - Total deleted packet
 - Packets destined to invalid queues (those dropped by the TM)

- Egress counters keep track of:
 - Discarded unicast packets counter
 - Discarded multicast packets
 - EGQ packets

The current poll time for TM devices statistics is 10 seconds.

TM VOQ counter

This counts the packets that are enqueued or discarded for a specific VOQ. For SLX devices, 16k VOQs statistics are available out of which 15k of VOQ statistics are used to track VOQ traffic for each egress port. Entries for the TM VOQ counter engine are 1:1 mapped to the VOQs. Each VOQ ID directly results in a counter engine statistics ID and points to the counter set of the engine associated with the VOQ.

Current polling time for TM VOQ stats is 4 minutes.

Traffic management counters

These commands display traffic management (TM) counter statistics.

TM global statistics command

```
show tm statistics device
```

TM device level statistics commands

Use these commands to display traffic management statistics for interface modules.

```
device# show tm statistics device interface ethernet 4/1
```

```
TM Counters:
```

```
=====
```

```
Ingress Counters:
```

```
-----
```

```
Total Ingress Pkt Count      1164304848
CPU Packet Count              0
Enque Packet Count            953260002
DeQue Packet Count            953260002
Total Discard Pkt Count       211045069
Oldest Discard Pkt Count      0
Resolved to be dropped        437190
```

```
Egress Counters:
```

```
-----
```

```
Unicast Pkt Count             586664425
FQP Pkt Count                  586664425
Discard UC Pkt Count           0
Discard MC Pkt Count           0
MC Packet Count                0
EHP Discard Count              0
```

```
device# show tm statistics device interface ethernet 3/25 details
```

```
TM Counters:
```

```
=====
```

```
Ingress Counters:
```

```

-----
Total Ingress Pkt Count          863033385
CPU Packet Count                 13
NIF Packet Count                 863033385
OAMP Packet Count                0
OLP Packet Count                0
Recycle Packet Count            0
MMU IDR Packet Count            0
Enque Packet Count              850127673
DeQue Packet Count              850127673
Total Discard Pkt Count         12905725
Oldest Discard Pkt Count        0
Resolved to be dropped          0
FDT Packet Count               10837941
CRC Error Count                 0

```

Egress Counters:

```

-----
Reassembly error Discard Count   21538682
  Packet UC Discard Count       10769341
  Packet MC Discard Count       0
  SOP UC Discard Count          0
  SOP MC Discard Count          10769341
Filter Discard Count            10769341
  MC Pruning High Priority       0
  MC Pruning Low Priority        0
  LAG Pruning Discard Count      0
  PMF Discard Count             0
  VLAN Member Discard Count      0
Discard UC Pkt Count            0
Discard MC Pkt Count            334638
UC Pkt Count                    839289732
MC Packet Count                 10434703
FQP Pkt Count                   849724435
Editor Discard Pkt Count        0
Editor Pkt Count                849724435
Total Egress Pkt Count          849724435

```

TM VOQ commands

This command displays a summary count of traffic management VOQ discards.

```
device# show tm voq-stat ingress-device ethernet 2/1 discards
```

```

----- Ports 1/1 - 1/36 -----
Dest Port | Prio | Queue | Discards
-----
3/1       | 0    | 320   | 2473804
2/4       | 0    | 224   | 1867789
4/2       | 2    | 434   | 1023452
4/8       | 4    | 487   | 920349
1/2       | 1    | 120   | 858723
1/3       | 1    | 128   | 75328
2/5       | 0    | 260   | 22234
2/6       | 0    | 268   | 5248

```

NOTE

The entries are sorted by highest number of discards with eight entries displayed by default.

This command displays a summary count of traffic management VOQ discards with priority 0.

```
device# show tm voq-stat ingress-device ethernet 2/1 discards priority 0
```

```

----- Ports 1/1 - 1/36 -----
Dest Port | Prio | Queue | Discards
-----
3/1       | 0    | 320   | 2473804
2/4       | 0    | 224   | 1867789

```

2/5		0		260		22234
2/6		0		268		5248

NOTE

The entries are sorted by highest number of discards with eight entries displayed by default.

Use this command to display TM VOQ statistics for an egress interface.

```
device# show tm voq-stat ingress-device ethernet 2/1 egress-port ethernet 2/7 priority 2
```

VOQ-Counters:

=====

Priority 2

EnQue Pkt Count	67404602
EnQue Bytes Count	1768413221
Total Discard Pkt Count	0
Total Discard Bytes Count	0
Current Queue Depth	0
Maximum Queue Depth since Last read	160

This command displays a summary of TM VOQ **max-queue-depth** parameter statistics.

```
device# show tm voq-stat ingress-device 2/1 max-queue-depth
```

----- Ports 1/1 - 1/36 -----								
Dest Port		Prio		Queue		Max Depth		Max Util

3/1		0		320		1013804		96%
2/4		0		224		902789		86%
4/2		2		434		543440		51%
4/8		4		487		220349		21%
1/2		1		120		138723		13%
1/3		1		128		97328		9%
2/5		0		260		34234		3%
2/6		0		268		11723		1%

This command displays a summary of TM VOQ maximum buffer utilization.

```
device# show tm voq-stat ingress-device 2/1 max-buffer-util
```

----- Ports 1/1 - 1/36 -----	
Max Buffer Size	Max Buffer Util

6007013804	96%