

Implementing Cisco Service Provider Next-Generation Core Network Services

Volume 2

Version 1.01

Student Guide

Text Part Number: 97-3154-02



Americas Headquarters

Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters

Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters

Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

DISCLAIMER WARRANTY: THIS CONTENT IS BEING PROVIDED "AS IS" AND AS SUCH MAY INCLUDE TYPOGRAPHICAL, GRAPHICS, OR FORMATTING ERRORS. CISCO MAKES AND YOU RECEIVE NO WARRANTIES IN CONNECTION WITH THE CONTENT PROVIDED HEREUNDER, EXPRESS, IMPLIED, STATUTORY OR IN ANY OTHER PROVISION OF THIS CONTENT OR COMMUNICATION BETWEEN CISCO AND YOU. CISCO SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES, INCLUDING WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE, OR ARISING FROM A COURSE OF DEALING, USAGE OR TRADE PRACTICE. This learning product may contain early release content, and while Cisco believes it to be accurate, it falls subject to the disclaimer above.

Table of Contents

Volume 2

<i>QoS Classification and Marking</i>	<i>4-1</i>
Overview	4-1
Module Objectives	4-1
Understanding Classification and Marking	4-3
Overview	4-3
Objectives	4-3
Classification and Marking	4-4
Classification	4-4
Marking	4-5
Classification and Marking at the Data Link Layer	4-5
Ethernet 802.1Q Class of Service	4-5
Cisco ISL Class of Service	4-6
Frame Relay DE and ATM CLP	4-6
MPLS EXP	4-7
Classification and Marking at the Network Layer	4-7
QoS Traffic Models	4-9
Enterprise to Service Providers QoS Service Classes Mapping at the Network Edge	4-13
Example: Enterprise to Service Provider Edge Service Class Mapping Using Four Service Classes	4-14
Trust Boundaries	4-16
Summary	4-19
Using Modular QoS CLI	4-21
Overview	4-21
Objectives	4-21
Using MQC for Classification	4-22
Access Control List	4-26
VLAN	4-26
Destination MAC Address	4-27
Source MAC Address	4-27
Input Interface	4-27
IP RTP Port Range	4-27
QoS Group	4-28
Discard Class	4-28
IP Precedence	4-29
DSCP	4-29
CoS	4-30
MPLS EXP	4-30
Frame Relay DE bit	4-30
Configuring Classification using MQC	4-31
Cisco IOS and IOS XE Software	4-32
Cisco IOS XR Software	4-32
Using MQC for Class-Based Marking	4-33
IP Precedence	4-34
DSCP	4-34
QoS Group	4-35
MPLS EXP	4-35
CoS	4-35
Frame Relay DE Bit	4-35
Configuring Class-Based Marking using MQC	4-36
Summary	4-39
Implementing Advanced QoS Techniques	4-41
Overview	4-41
Objectives	4-41
Network-Based Application Recognition	4-42
Configuring MQC Traffic Classification Using NBAR (match protocol)	4-55

QoS Tunneling Techniques	4-57
Configuring QoS Pre-Classify	4-60
QoS Policy Propagation via BGP	4-63
Configuring QPPB	4-66
Hierarchical QoS	4-68
Summary	4-72
Module Summary	4-73
Module Self-Check	4-75
Module Self-Check Answer Key	4-77

QoS Congestion Management and Avoidance **5-1**

Overview	5-1
Module Objectives	5-1

Managing Congestion **5-3**

Overview	5-3
Objectives	5-3
Queuing Introduction	5-4
FIFO Queuing	5-6
Priority Queuing	5-7
Round Robin Queuing	5-8
Weighted Round Robin Queuing	5-9
Deficit Round Robin Queuing	5-10
Modified Deficit Round Robin Queuing	5-11
Cisco IOS and IOS XR Queue Types	5-13
Cisco IOS XR Forwarding Architecture	5-14
Configuring CBWFQ	5-16
Configuring LLQ	5-23
Summary	5-27

Implementing Congestion Avoidance **5-29**

Overview	5-29
Objectives	5-29
Congestion Avoidance Introduction	5-30
TCP Congestion Management	5-31
Tail Drop and TCP Global Synchronization	5-35
Random Early Detection (RED) Introduction	5-38
Configuring WRED	5-41
Summary	5-48
Module Summary	5-49
References	5-50
Module Self-Check	5-51
Module Self-Check Answer Key	5-53

QoS Traffic Policing and Shaping **6-1**

Overview	6-1
Module Objectives	6-1

Understanding Traffic Policing and Shaping **6-3**

Overview	6-3
Objective	6-3
Traffic Policing and Shaping	6-4
Comparing Traffic Policing vs. Shaping	6-9
Traffic Policing Token Bucket Implementations	6-10
Example: Token Bucket as a Coin Bank	6-11
Example: Dual-Rate Token Bucket as a Coin Bank	6-17
Traffic Shaping Token Bucket Implementation	6-18
Traffic Policing and Shaping in IP NGN	6-19
Traffic Policing and Shaping with Cisco Telepresence	6-20
Summary	6-22

Implementing Traffic Policing	6-23
Overview	6-23
Objectives	6-23
Class-Based Policing	6-24
Single-Rate, Single Token Bucket Policing Configuration	6-26
Single-Rate, Dual Token Bucket Policing Configuration	6-27
Multiaction Policing Configuration	6-28
Dual Rate Policing Configuration	6-30
Percentage Based Policing Configuration	6-31
Hierarchical Policing Configuration	6-32
Monitoring Class-Based Policing Operations	6-33
Cisco Access Switches Policing Configuration	6-34
Cisco Access Switches Aggregate Policer Configuration	6-35
Local Packet Transport Services	6-36
Summary	6-42
Implementing Traffic Shaping	6-43
Overview	6-43
Objectives	6-43
Class-Based Shaping	6-44
Single-Level Shaping Configuration	6-46
Hierarchical Shaping Configuration	6-47
Monitoring Class-Based Shaping Operations	6-50
Summary	6-51
Module Summary	6-53
References	6-54
Module Self-Check	6-55
Module Self-Check Answer Key	6-57

QoS Classification and Marking

Overview

In any network in which networked applications require differentiated levels of service, traffic must be sorted into different classes to which quality of service (QoS) is applied. Classification and marking are two critical functions of any successful QoS implementation. Classification allows network devices to identify traffic as belonging to a specific class with specific QoS requirements as determined by an administrative QoS policy. After network traffic is sorted, individual packets are colored or marked so that other network devices can apply QoS features uniformly to those packets that are in compliance with the defined QoS policy. This module introduces classification and marking, and the different methods of performing these critical QoS functions on service provider and enterprise devices.

Module Objectives

Upon completing this module, you will be able to successfully classify and mark network traffic to implement a policy according to QoS requirements. This ability includes being able to meet these objectives:

- Define the purpose of classification and marking, and how they can be used to define a QoS service class
- Use MQC for classification and marking configuration
- Use NBAR for traffic classification, use QoS preclassification, and implement classification and marking in an interdomain network using QPPB

Understanding Classification and Marking

Overview

Quality of service (QoS) offers the ability to provide different levels of treatment to specific classes of traffic. Before any QoS applications or mechanisms can be applied, traffic must be identified and sorted into different classes. QoS is applied to these different traffic classes. Network devices use classification to identify traffic as belonging to a specific class. After network traffic is sorted, marking can be used to color (tag) individual packets so that other network devices can apply QoS features uniformly to those packets as they travel through the network.

This lesson introduces the concepts of classification and marking, explains the different markers that are available at the data-link and network layers, and identifies where classification and marking should be used in a network. In addition, the concept of a QoS service class, and how a service class can be used to represent an application or set of applications, is discussed. At the end of the lesson, trust boundaries in service provider and enterprise environments are defined, as well as why it is important to know the trust boundary for defining QoS classes and policies.

Objectives

Classification is the process of identifying traffic and categorizing that traffic into different classes, while marking allows network devices to classify a packet or frame based on a specific traffic descriptor. Upon completing this lesson, you will be able to meet these objectives:

- Describe classification and marking concepts
- Explain how traffic are typically classified into the different QoS service classes
- Provide an example showing the mapping between the Enterprise and Service Provider QoS service classes at the network edge
- Describe trust boundaries in enterprise and service provider environments

Classification and Marking

This topic describes classification and marking concepts.

Classification and Marking

Classification:

- Identifying and categorizing traffic into different classes
- Without classification, all packets are treated the same
- Should be performed close to the network edge

Marking:

- "Coloring" packet using traffic descriptors
- Easily distinguish the marked packet belonging specific class
- Commonly used markers: CoS, DSCP, MPLS EXP

© 2012 Cisco and/or its affiliates. All rights reserved.SPCORE v1.01--4.3

Classification

Classification is the process of identifying traffic and categorizing that traffic into different classes. Packet classification process uses various criteria to categorize a packet within a specific group in order to define that packet. Typically used traffic descriptors include class of service (CoS), incoming interface, IP precedence, differentiated services code point (DSCP), source or destination address, application, and Multiprotocol Label Switching experimental bits (MPLS EXP). After the packet has been defined (that is, classified), the packet is then accessible for QoS handling on the network.

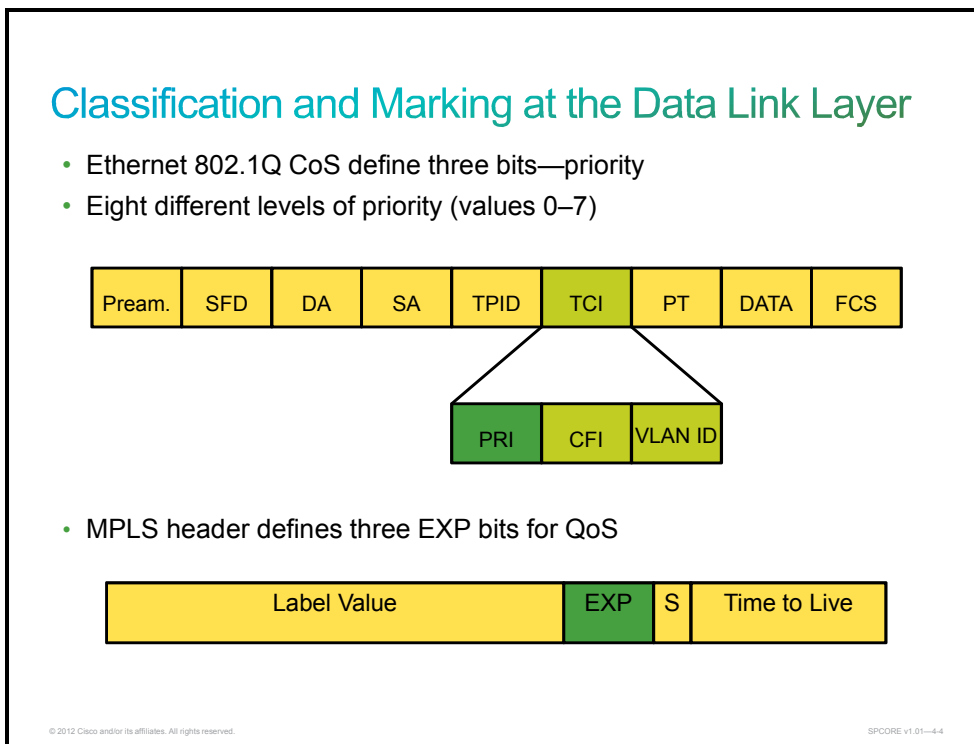
Using packet classification, you can partition network traffic into multiple priority levels or classes of service. When traffic descriptors are used to classify traffic, the source agrees to adhere to the contracted terms and the network promises a QoS. Different QoS mechanisms, such as traffic policing, traffic shaping, and queuing techniques use the traffic descriptor of the packet (that is, the classification of the packet) to ensure adherence to that agreement.

Classification should take place at the network edge, typically in the wiring closet, in IP phones, or at network endpoints. It is recommended that classification occur as close to the source of the traffic as possible.

Marking

Marking is related to classification. Marking allows network devices to classify a packet or frame based on a specific traffic descriptor. Typically used traffic descriptors include CoS, DSCP, IP precedence, and MPLS EXP. Marking can be used to set information in the Layer 2 or Layer 3 packet headers.

Marking a packet or frame with its classification allows network devices to easily distinguish the marked packet or frame as belonging to a specific class. After the packets or frames are identified as belonging to a specific class, QoS mechanisms can be uniformly applied to ensure compliance with administrative QoS policies.



Classification and Marking at the Data Link Layer

Several Layer 2 classification and marking options exist depending on the technology, encapsulation, and transport protocol used:

- Ethernet 802.1Q CoS
- Cisco ISL CoS
- Frame Relay discard eligible (DE)
- ATM cell loss priority (CLP)
- MPLS EXP

Ethernet 802.1Q Class of Service

The 802.1Q standard is an IEEE specification for implementing VLANs in Layer 2 switched networks. The 802.1Q specification defines two 2-byte fields, Tag Protocol Identifier (TPID) and Tag Control Information (TCI), which are inserted within an Ethernet frame following the source address field. The TPID field is currently fixed and assigned the value 0x8100. The TCI field is composed of these three fields, of which the following field is of interest when implementing QoS at Layer 2:

User priority bits (3 bits): These bits can be used to mark packets as belonging to a specific CoS. The CoS marking uses the three 802.1p user priority bits and allows a Layer 2 Ethernet frame to be marked with eight different levels of priority (values 0–7). Three bits allow for eight levels of classification, allowing a direct correspondence with IPv4 (IP precedence) type of service (ToS) values. The 802.1p specification defines these standard definitions for each CoS:

- CoS 7 (111): network
- CoS 6 (110): Internet
- CoS 5 (101): critical
- CoS 4 (100): flash override
- CoS 3 (011): flash
- CoS 2 (010): immediate
- CoS 1 (001): priority
- CoS 0 (000): routine

One disadvantage of using CoS markings is that frames will lose their CoS markings when transiting a non-802.1Q or non-802.1p link, including any type of non-Ethernet WAN link. Therefore, a more permanent marking should be used for network transit, such as Layer 3 IP DSCP marking. This is typically accomplished by translating a CoS marking into another marker or simply using a different marking mechanism.

Cisco ISL Class of Service

Inter-Switch Link (ISL) is a proprietary Cisco protocol for interconnecting multiple switches and maintaining VLAN information as traffic travels between switches. ISL was created prior to the standardization of 802.1Q. However, ISL is compliant with the 802.1p standard.

The ISL frame header contains a 4-bit User field that carries 802.1p CoS values in the three least significant bits. When an ISL frame is marked for priority, the three 802.1p CoS bits are set to a value from 0 to 7.

Frame Relay DE and ATM CLP

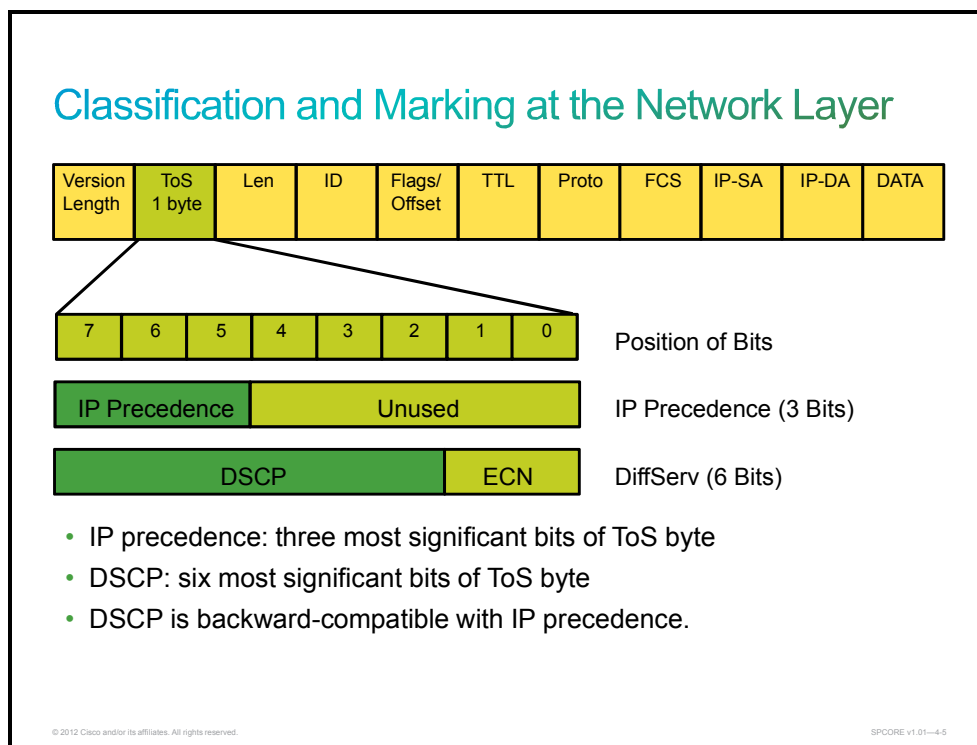
One component of Frame Relay QoS is packet discard when congestion is experienced in the network. Frame Relay will allow network traffic to be sent at a rate exceeding its committed information rate (CIR). Frames sent that exceed the committed rate can be marked as DE. If congestion occurs in the network, frames marked DE will be discarded prior to frames that are not marked.

ATM cells consist of 48 bytes of payload and 5 bytes of header. The ATM header includes the 1-bit CLP field, which indicates the drop priority of the cell if that cell encounters extreme congestion as it moves through the ATM network. The CLP bit represents two values: 0 to indicate higher priority and 1 to indicate lower priority. Setting the CLP bit to 1 lowers the priority of the cell, increasing the likelihood that the cell will be dropped when the ATM network experiences congestion.

MPLS EXP

When a customer transmits IP packets from one site to another, the IP Precedence field (the first three bits of the DSCP field in the header of an IP packet) specifies the CoS. Based on the IP precedence marking, the packet is given the desired treatment, such as guaranteed bandwidth or latency. If the service provider network is an MPLS network, the IP precedence bits are copied into the MPLS experimental field at the edge of the network. However, the service provider might want to set an MPLS packet QoS to a different value that is determined by the service offering.

The MPLS experimental field allows the service provider to provide QoS without overwriting the value in the customer IP Precedence field. The IP header remains available for customer use, and the IP packet marking is not changed as the packet travels through the MPLS network.



Classification and Marking at the Network Layer

At the network layer, IP packets are typically classified based on source or destination IP address, or the contents of the ToS byte. Link-layer media often changes as a packet travels from its source to its destination. Because a CoS field does not exist in a standard Ethernet frame, CoS markings at the link layer are not preserved as packets traverse nontrunked or non-Ethernet networks. Using marking at the network layer (Layer 3) provides a more permanent marker that is preserved from source to destination.

IP Precedence

Originally, only the first three bits of the ToS byte were used for marking, referred to as IP precedence. However, newer standards have made the use of IP precedence obsolete in favor of using the first six bits of the ToS byte for marking, referred to as DSCP.

The header of an IPv4 packet contains the ToS byte. IP precedence uses three precedence bits in the ToS field of the IPv4 header to specify CoS for each packet. IP precedence values range from 0 to 7 and allow you to partition traffic in up to six usable classes of service. (Settings 6 and 7 are reserved for internal network use.)

DiffServ

Differentiated services (DiffServ) is a new model that supersedes—and is backward-compatible with—IP precedence. DiffServ redefines the ToS byte as the DiffServ field and uses six prioritization bits that permit classification of up to 64 values (0 to 63), of which 32 are commonly used. A DiffServ value is called a DSCP.

With DiffServ, packet classification is used to categorize network traffic into multiple priority levels or classes of service. Packet classification uses the DSCP traffic descriptor to categorize a packet within a specific group to define that packet. After the packet has been defined (classified), the packet is then accessible for QoS handling on the network.

Mapping Data Link-to-Network Layer Markings

IP headers are preserved end-to-end when IP packets are transported across a network, but data link layer headers are not preserved. This means that the IP layer is the most logical place to mark packets for end-to-end QoS. However, there are edge devices that can only mark frames at the data link layer, and there are many other network devices that only operate at the data link layer. To provide true end-to-end QoS, the ability to map QoS marking between the data link layer and the network layer is essential.

Service providers offering IP services have a requirement to provide robust QoS solutions to their customers. The ability to map network layer QoS to link layer CoS allows these providers to offer a complete end-to-end QoS solution that does not depend on any specific link-layer technology.

Compatibility between an MPLS transport layer QoS and network layer QoS is also achieved by mapping between MPLS EXP bits and the IP precedence or DSCP bits. A service provider can map the customer network layer QoS marking as is, or change it to fit an agreed-upon service level agreement (SLA). The information in the MPLS EXP bits can be carried end-to-end in the MPLS network, independent of the transport media. In addition, the network layer marking can remain unchanged so that when the packet leaves the service provider MPLS network, the original QoS markings remain intact. Thus, a service provider with an MPLS network can help provide a true end-to-end QoS solution.

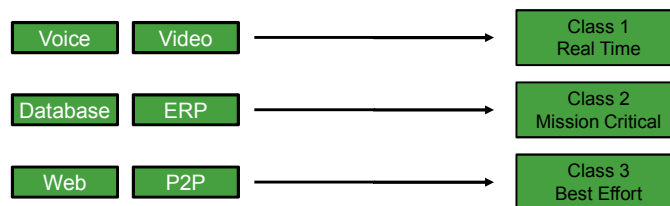
QoS Traffic Models

This topic explains how traffic are typically classified into the different QoS service classes.

QoS Service Class

- Logical grouping of packets that are to receive same level of applied quality
- QoS service class can be:
 - Single user (MAC address, IP address)
 - Specific customer or set of customers
 - Specific application or set of applications

Example of QoS service classes by set of applications:



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4.6

When an administrative policy requiring QoS is created, you must determine how network traffic is to be treated. As part of that policy definition, network traffic must be associated with a specific service class. QoS classification mechanisms are used to separate traffic and identify packets as belonging to a specific service class. QoS marking mechanisms are used to tag each packet as belonging to the assigned service class. After the packets are identified as belonging to a specific service class, QoS mechanisms such as policing, shaping, and queuing techniques can be applied to each service class to meet the specifications of the administrative policy. Packets belonging to the same service class are given the same treatment with regard to QoS.

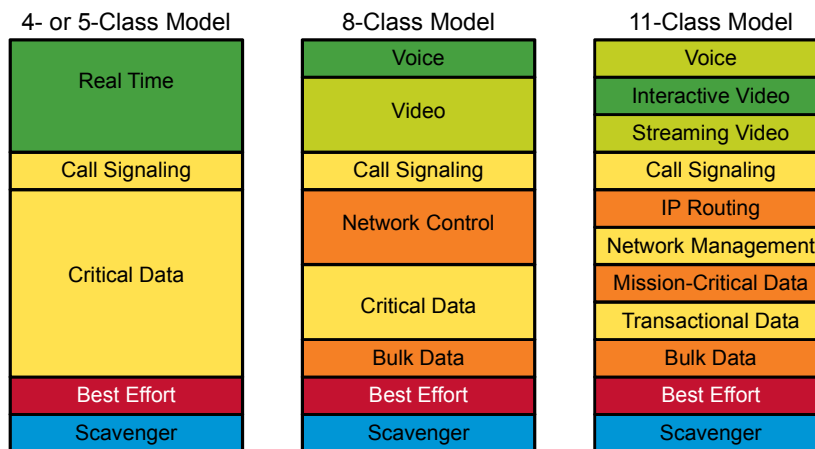
A QoS service class, being a logical grouping, can be defined in many ways, including these:

- Organization or department (marketing, engineering, sales, and so on)
- A specific customer or set of customers
- Specific applications or set of applications (Telnet, FTP, voice, Service Advertising Protocol [SAP], Oracle, video, and so on)
- Specific users or sets of users (based on MAC address, IP address, LAN port, and so on)
- Specific network destinations (tunnel interfaces, VPNs, and so on)

Specifying an administrative policy for QoS requires that a specific set of service classes be defined. QoS mechanisms are uniformly applied to these individual service classes to meet the requirements of the administrative policy. There are many different methods in which service classes can be used to implement an administrative policy. The first step is to identify the traffic that exists in the network and the QoS requirements for each traffic type. Then, traffic can be grouped into a set of service classes for differentiated QoS treatment in the network.

Models of Enterprise QoS Service Classes

- Three models are defined: 4- to 5-, 8-, and 11-class models
- More granularity in differentiation of traffic—more classes



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.7

The number of traffic classes used by enterprises has increased over the past few years, from four classes to between five and seven classes. The reason for this increase is that enterprises are using more and more applications and increasingly want more granularity in QoS differentiation among applications. The Cisco QoS baseline has suggested an 11-class model. This 11-class model is not mandatory, but merely an example of traffic classification based on various types of applications in use and their QoS requirements from an enterprise perspective.

Layer 2 and Layer 3 Marking of Service Classes

Application	Layer 3 Classification			Layer 2 Classification
	IPP	PHB	DSCP	CoS / MPLS EXP
Voice	5	EF	46	5
Interactive Video	4	AF41	34	4
Streaming Video	4	CS4	32	4
Call Signaling	3	CS3	24	3
IP Routing	6	CS6	48	6
Network Management	2	CS2	16	2
Mission-Critical Data	3	AF31	26	3
Transactional Data	2	AF21	18	2
Bulk Data	1	AF11	10	1
Best Effort	0	BE	0	0
Scavenger	1	CS1	8	1

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.8

Although there are several sources of information that can be used as guidelines for determining a QoS policy, none of them can determine exactly what is proper for a specific network. Each network presents its own unique challenges and administrative policies. To properly implement QoS, measurable goals must be declared, and then a plan for achieving these goals must be formulated and implemented.

QoS must be implemented consistently across the entire network. It is not so important whether call signaling is marked as DSCP 34 or 26, but it is important that DSCP 34 and 26 are treated in a manner that will accomplish the QoS policy. It is also important that data marked DSCP 34 is treated consistently across the network.

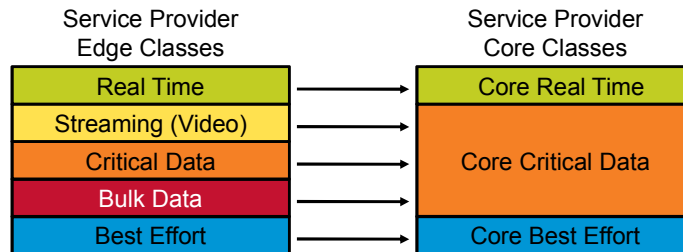
Originally, Cisco marked call signaling traffic as Assured Forwarding (AF) 31, and call signaling traffic was originally marked by Cisco IP telephony equipment to DSCP AF31. However, the AF classes, as defined in RFC 2597, were intended for flows that could be subject to markdown and, subsequently, the aggressive dropping of marked-down values. Marking down and aggressively dropping call signaling could result in noticeable delay-to-dial-tone (DDT) and lengthy call setup times, both of which generally translate to poor user experiences.

The Cisco QoS baseline changed the marking recommendation for call signaling traffic to DSCP CS3 because class selector code points, as defined in RFC 2474, were not subject to markdown or aggressive dropping.

Service Provider QoS Service Classes

- Service provider service class types: edge and core
- Core service classes:
 1. Core real time
 2. Core critical data
 3. Core best effort
- Edge service class models: Three to six service classes

Example of mapping between service provider core and edge:



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.0

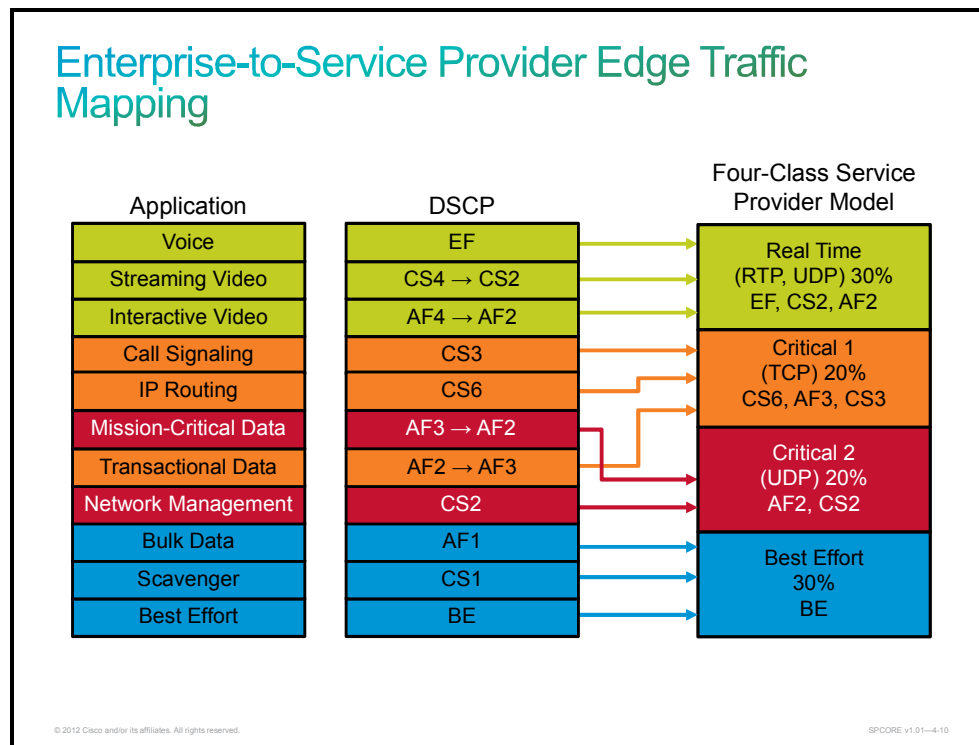
It is not necessary to ensure that the backbone network supports the same number of DiffServ classes as the edge, assuming that proper design principles are in place to support the given SLAs. One example of this is to provision three DiffServ classes in the backbone network, while five classes are provisioned at the provider edges, as shown in the figure.

Backbone-network classes are defined as follows:

- **Core real time:** This class targets applications such as VoIP and interactive video, which require low loss, low delay, and low jitter, and have a defined availability. This class may also support per-flow sequence preservation. This class should always be engineered for the worst-case delay to support the real-time traffic. Excess traffic in this class is typically dropped. This class should be associated to expedited forwarding with a priority queue to ensure that the delay and jitter contracts are met.
- **Core critical data:** This class represents business-critical interactive applications. It is defined in terms of delay (round-trip time [RTT] should be less than 250 ms—the threshold for human delay perception) and loss (less than 1 percent loss rate is typical, with targets as low as 0.1 percent also available), with an availability. Throughput is derived from loss and RTT. Jitter is not important for this service class and is not defined. Excess in this class is typically re-marked with an out-of-contract identifier (re-marking of EXP to a lower value) and transmitted. This class may also support per-flow sequence preservation.
- **Core best effort:** This class represents all other customer traffic that has not been classified as real-time or critical data. It is defined as a loss rate with availability. Throughput is derived from loss. Delay and jitter are not important for this service and are not defined—therefore, only 10 percent of remaining link capacity (after the priority queue has been served) should be allocated to this queue.

Enterprise to Service Providers QoS Service Classes Mapping at the Network Edge

This topic provides an example showing the mapping between the Enterprise and Service Provider QoS service classes at the network edge.



Most service providers offer only a limited number of classes within their MPLS VPN clouds. At times, this might require enterprises to collapse the number of classes that they have provisioned to integrate into the QoS models of their service provider. The following caveats should be considered when deciding how best to collapse and integrate enterprise classes into various service provider QoS models.

Voice and Video

Service providers typically offer only one real-time class or priority class of service. If an enterprise wants to deploy both voice and IP/VC (each of which should be provisioned with strict priority treatment) over the MPLS VPN, they might be faced with a dilemma of which one should be assigned to the real-time class. There may be complications if both are assigned to the real-time class.

Call Signaling

VoIP requires provisioning not only of RTP bearer traffic, but also of call-signaling traffic, which is very lightweight and requires only a moderate amount of guaranteed bandwidth. Because the service levels applied to call-signaling traffic directly affect delay to the dial tone, it is important that call signaling be protected. Service providers might not always offer a suitable class for call-signaling traffic itself. Therefore, the enterprise must determine which other traffic classes to mix with call signaling.

Mixing TCP with UDP

It is a general best practice to avoid mixing TCP-based traffic with UDP-based traffic (especially streaming video) within a single service provider class, because of the behaviors of these protocols during periods of congestion. Specifically, TCP transmitters throttle flows when drops are detected. Although some UDP applications have application-level windowing, flow control, and retransmission capabilities, most UDP transmitters ignore drops and, thus, never lower transmission rates because of dropping.

When TCP flows are combined with UDP flows within a single service provider class and the class experiences congestion, TCP flows continually lower their transmission rates, potentially giving up their bandwidth to UDP flows that will ignore drops. This effect is called TCP starvation and UDP dominance.

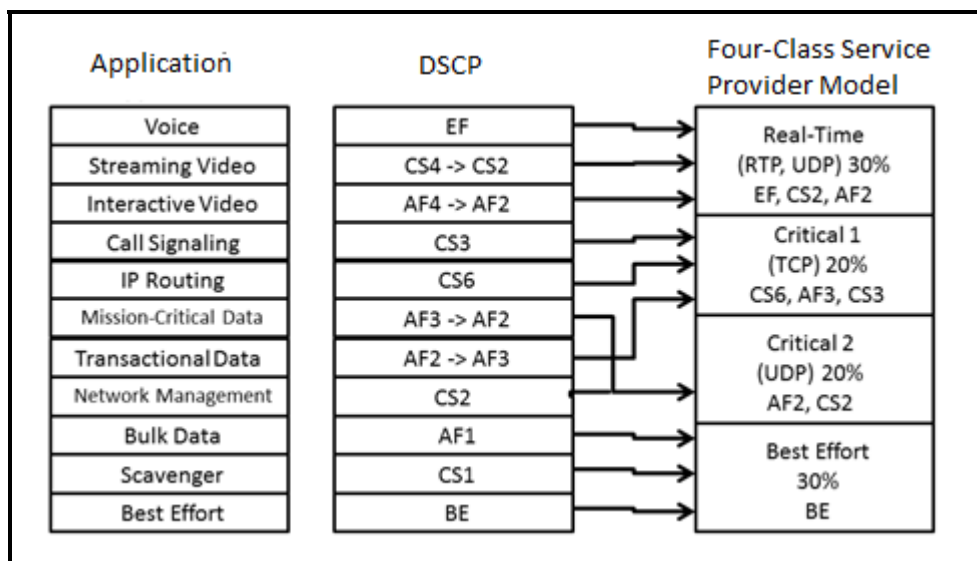
Marking and Re-Marking

Most service providers use the Layer 3 marking attributes (IP precedence or DSCP) of packets that are sent to them to determine the service provider class of service to which a packet should be assigned. Therefore, enterprises must mark or re-mark their traffic in a way that is consistent with the service provider admission criteria. Additionally, service providers might re-mark at Layer 3 out-of-contract traffic within their cloud. This can affect enterprises that require consistent end-to-end Layer 3 markings.

A general DiffServ principle is to mark or trust traffic as close to the source as administratively and technically possible. However, certain traffic types might need to be re-marked before handoff to the service provider to gain admission to the correct class. If such re-marking is required, it is recommended that the re-marking be performed at the egress edge of the customer edge (CE), rather than within the campus. This is because service provider service offerings are likely to evolve or expand over time, and adjusting to such changes will be easier to manage if re-marking is performed only at the CE egress edge.

Example: Enterprise to Service Provider Edge Service Class Mapping Using Four Service Classes

In the model shown in the figure, the service provider offers four classes of service. Because there are so few classes to choose from in this example, interactive video may need to be combined with another application.



It is highly recommended not to combine interactive video with any unbounded application (an application without admission control) within a single service provider class, because doing so could lead to class congestion and result in drops of video packets. This will occur with or without weighted random early detection (WRED) enabled on the service provider class. Therefore, there are two options in such a design:

- Assign interactive video to the service provider real-time class along with voice.
- Assign interactive video to a dedicated non-priority service-provider class.

In this example, interactive video is assigned to the service provider real-time class.

In the four-class service provider model, there is a real-time class, a default best-effort class, and two additional non-priority traffic classes. In this case, the enterprise administrator may elect to separate TCP-based applications from UDP-based applications by using these two non-priority service provider traffic classes. Specifically, if voice and interactive video are the only applications to be assigned to the service provider real-time class, streaming video and network management traffic (which is largely UDP-based) can all be assigned to the service provider UDP (Critical 2) class. This leaves the other non-priority service provider class (Critical 1) available for control plane applications, such as network control and call signaling, along with TCP-based transactional data applications. The figure shows the per-class re-marking requirements from the CE edge to gain access to the classes within the four-class service provider model, with interactive video assigned to the service provider real-time class, along with voice.

In this example, individual traffic classes must be re-marked on the CE egress edge in order to gain access to the service provider associated class. Some traffic classes, such as best effort, scavenger, and bulk, do not need to be re-marked. Additionally, the relative per-class bandwidth allocations must be aligned, so that the enterprise CE edge queuing policies are consistent with the service provider edge (PE) edge queuing policies to ensure compatible per-hop behaviors (PHBs).

Trust Boundaries

This topic describes trust boundaries in enterprise and service provider environments.

Trust Boundaries

Network edge at which packets are trusted or not

- Packets are treated differently depending on whether they are confined within boundary
- Where classification and marking should take place
- Where to enforce trust boundary?

Should be set as close as possible to the source

- Trust boundary exists from perspective of:
 - Enterprise
 - Service provider

© 2012 Cisco and/or its affiliates. All rights reserved.SPCORE v1.01--4.11

The administrator needs to consider where to enforce the trust boundary, that is, the network edge at which packets are trusted (or not). In line with the strategic QoS classification principle mentioned earlier, the trust boundary should be set as close to the endpoints as technically and administratively feasible.

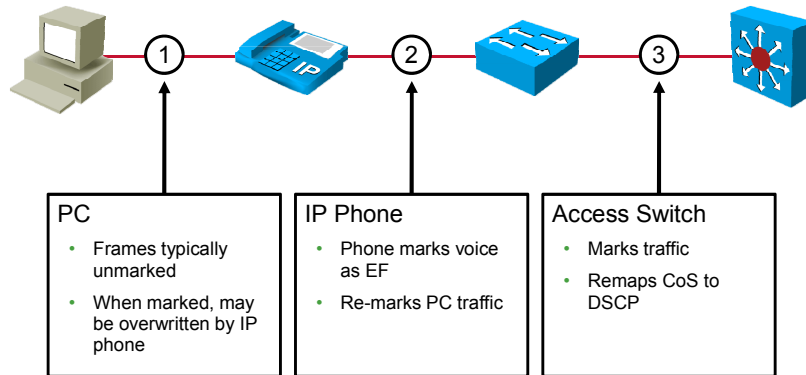
The reason for the "administratively feasible" caveat within this design recommendation is that, while many endpoints (including user PCs) technically support the ability to mark traffic on their network interface cards (NICs), allowing a blanket trust of such markings could easily facilitate network abuse, as users could simply mark all their traffic with Expedited Forwarding, which would allow them to hijack network priority services for their traffic that is not real-time, and thus ruin the service quality of real-time applications throughout the enterprise.

The concept of trust is important and integral to deploying QoS. After the end devices have set CoS or ToS values, the switch has the option of trusting them. If the switch trusts the values, it does not need to reclassify. If the switch does not trust the values, it must perform reclassification for the appropriate QoS.

The notion of trusting or not trusting forms the basis for the trust boundary. Ideally, classification should be done as close to the source as possible. If the end device is capable of performing this function, the trust boundary for the network is at the end device. If the device is not capable of performing this function, or the wiring closet switch does not trust the classification done by the end device, the trust boundary might shift.

Enterprise Trust Boundary

As close as possible to the source of traffic:



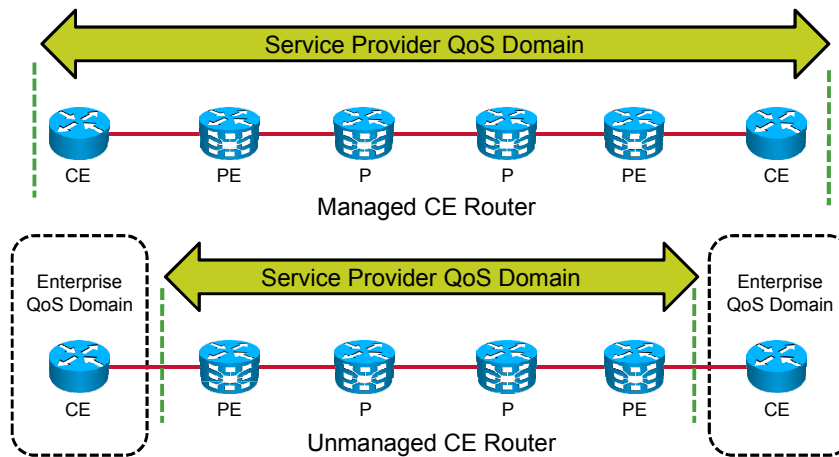
Classification should take place at the network edge, typically in the wiring closet or within endpoints (servers, hosts, video endpoints, or IP telephony devices).

For example, consider the campus network containing IP telephony and host endpoints. Frames can be marked as important by using link-layer CoS settings, or the IP precedence or DSCP bits in the ToS and DiffServ field in the IPv4 header. Cisco IP phones can mark voice packets as high priority using CoS as well as ToS. By default, the IP phone sends 802.1p-tagged packets with the CoS and ToS set to a value of 5 for its voice packets. Because most PCs do not have an 802.1Q-capable NIC, they send packets untagged. This means that the frames do not have an 802.1p field. Also, unless the applications running on the PC send packets with a specific CoS value, this field is zero.

If the end device is not a trusted device, the reclassification function (setting or zeroing the bits in the CoS and ToS fields) can be performed by the access layer switch, if that device is capable of doing so. If the device is not capable, then the reclassification task falls to the distribution layer device.

Service Provider Trust Boundary

- Separates enterprise and service provider QoS domains
- What is not trusted? Traffic class or traffic rate or both



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4-13

Although a CE device is traditionally owned and managed by the customer, a service provider often provides managed CE service to a customer, where the CE is owned and managed by the service provider. The trust boundary for traditional unmanaged service delivery is at the PE–CE boundary, whereas in the case of managed service it lies behind the CE, between the CE and the rest of the enterprise network.

For unmanaged services, the service provider maps enterprise traffic classes to aggregated service provider traffic classes at the PE. Since traffic from multiple customers may be aggregated at a single PE, the PE needs to have separate configurations on a per-customer basis to implement such mappings and to enforce the SLA.

The PE QoS configuration could be more complex in this case, depending on the extent of variations of individual customer QoS policies and SLAs. In a managed service, the service provider owns and operates the CE from a QoS perspective. One advantage of this is that it allows a service provider to distribute the complexity of the enterprise-to-service provider QoS policy mapping to the CE devices.

Since the service provider owns the CE device, the enterprise-to-service provider traffic class mapping, as well as other SLA enforcements like per-class policing, can now be done in the CE itself, offloading the PE and simplifying the PE configuration.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- Classifying packets into different classes is called classification, while marking packets is important to easily distinguish packets
- QoS must be implemented consistently across the entire network
- Most service providers offer only a limited number of classes within their MPLS VPN clouds
- The trust boundary differs depending on whether the CE device is owned by the service provider or not.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-14

Using Modular QoS CLI

Overview

Packet classification identifies the traffic flow and marking identifies traffic flows that require congestion management or congestion avoidance on a data path. The Modular Quality of Service (QoS) CLI (MQC) is used to define the traffic flows that should be classified, where each traffic flow is called a class of service, or class. Subsequently, a traffic policy is created and applied to a class.

This lesson provides the conceptual and configuration information for QoS packet classification and marking options using the MQC.

Objectives

Upon completing this lesson, you will be able to configure classification and marking options using MQC. You will be able to meet these objectives:

- Describe using MQC traffic classification
- Explain how to use MQC to implement traffic classification
- Describe using MQC class-based marking
- Explains how to use MQC to implement class-based marking

Using MQC for Classification

This topic describes using MQC traffic classification.

Classification Options in MQC

- Traffic class contains three major elements:
 - Class name
 - Match statement(s)
 - Match any or match all criteria
- **Match statements** include the following criteria for packet classification:

- Access list	- 802.1q/ISL CoS bits	- Frame Relay DE bit
- IP precedence	- Input interface	- Frame Relay DLCI
- DSCP value	- Source MAC address	- Another class map
- QoS group number	- Destination MAC address	- IP-specific values
- Discard class	- Any packet	
- MPLS EXP bits	- RTP/UDP port range	
- Protocol		

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.3

A traffic class contains three major elements: a name, a series of match commands, and, if more than one **match** command exists in the traffic class, an instruction on how to evaluate these commands.

MQC classification with class maps is extremely flexible and can classify packets by using these classification tools:

- **Access control lists (ACLs):** ACLs for any protocol can be used within the class map configuration mode. The MQC can be used for other protocols, not only IP.
- **IP precedence:** IP packets can be classified directly by specifying IP precedence values.
- **Differentiated services code point (DSCP):** IP packets can be classified directly by specifying IP DSCP values. DiffServ-enabled networks can have up to 64 classes if DSCP is used to mark packets.
- **QoS group:** A QoS group parameter can be used to classify packets in situations where up to 100 classes are needed or the QoS group parameter is used as an intermediate marker—for example, MPLS-to-QoS-group translation on input and QoS-group-to-DSCP translation on output. QoS group markings are local to a single router.
- **Discard class:** A discard-class value has no mathematical significance. For example, the discard class value 2 is not greater than 1. The value simply indicates that a packet marked with discard class 2 should be treated differently than a packet marked with discard class 1. Packets that match the specified discard class value are treated differently from packets marked with other discard class values. The discard class is a matching criterion only, used in defining per-hop behavior (PHB) for dropping traffic.
- **Multiprotocol Label Switching experimental (MPLS EXP) bits:** Packets can be matched based on the value in the experimental bits of the MPLS header of labeled packets.

- **Protocol:** Classification is possible by identifying Layer 3 or Layer 4 protocols. Advanced classification is also available by using the Network-Based Application Recognition (NBAR) tool, which identifies dynamic protocols by inspecting higher-layer information.
- **Class of service (CoS):** Packets can be matched based on the information that is contained in the three CoS bits (when using IEEE 802.1Q encapsulation) or priority bits (when using the Inter-Switch Link [ISL] encapsulation).
- **Input interface:** Packets can be classified based on the interface from which they enter the device.
- **MAC address:** Packets can be matched based on their source or destination MAC addresses.
- **All packets:** MQC can also be used to implement a QoS mechanism for all traffic, in which case classification will put all packets into one class.
- **UDP port range:** Real-Time Transport Protocol (RTP) packets can be matched based on a range of UDP port numbers.
- **Frame Relay discard-eligible (DE) bit:** Packets can be matched based on the value of the underlying Frame Relay DE bit.
- **Frame Relay data-link connection identifier (DLCI):** This match criterion can be used in main interfaces and point-to-multipoint subinterfaces in Frame Relay networks, and it can also be used in hierarchical policy maps.
- **Class map hierarchy:** Another class map can be used to implement template-based configurations.
- **IP-specific values:** These values are used to match on previously defined criteria, such as DSCP, IP precedence, and IP RTP port range values.

Matching Criteria in the Class

- **match-any** matches **ANY** of the match statements
- **match-all** must match **ALL** of the match statements

Example (Cisco IOS XR Software):

class1 must match access list 100
or DSCP 46

class1 must match access list 100
and DSCP 46

```
class-map match-any class1
  match access-group ipv4 100
  match dscp 46
```

```
class-map match-all class1
  match access-group ipv4 100
  match dscp46
```

- **match-any** is default in Cisco IOS XR Software
- **match-all** is default in Cisco IOS and IOS XE Software

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4.4

The traffic class is named in the **class-map** command. The **match** commands are used to specify various criteria for classifying packets. Packets are checked to determine whether they match the criteria specified in the **match** commands. If a packet matches the specified criteria, that packet is considered a member of the class and is forwarded according to the QoS specifications set in the traffic policy. Packets that fail to meet any of the matching criteria are classified as members of the default traffic class.

The instruction on how to evaluate these match commands needs to be specified if more than one match criterion exists in the traffic class. The evaluation instruction is specified with the **class-map** command. If the **match-any** option is specified as the evaluation instruction, the traffic being evaluated by the traffic class must match at least one of the specified criteria. If the **match-all** option is specified, the traffic must match all of the match criteria.

Syntax Description

Parameter	Description
[match-any match-all]	(Optional) Determines how packets are evaluated when multiple match criteria exist. Packets must either meet all of the match criteria (match-all) or one of the match criteria (match-any) to be considered a member of the class. The default in Cisco IOS and IOS XE Software is match-all . The default in Cisco IOS XR Software is match-any .
<i>class-map-name</i>	The name of the class for the class map. The name can be a maximum of 40 alphanumeric characters. The class name is used for both the class map and to configure policy for the class in the policy map.

Options for Classification in MQC

- Options for classification in Cisco IOS and IOS XE Software:
 - match-any** matches all traffic
 - match class-map** for nested classification
- Packets can be classified using **match not** criteria.

Example: Match all IPv4 traffic that does not have QoS group marking 1, 2, or 3.

IOS XR Software:

```
class-map match-all class9
match protocol ipv4
match not qos-group 1 2 3
```

Nested classification in IOS and IOS XE Software:

```
class-map match-all class9
match any
match not qos-group 1 2 3
!
class-map match-any cisco9
match class-map class9
match dscp ef
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.5

These are additional options that give extra power to class maps:

- Any condition can be negated by inserting the keyword **not**.
- A class map can use another class map to match packets (Cisco IOS and IOS XE Software only).
- The **any** keyword can be used to match all packets (Cisco IOS and IOS XE Software only).

In Cisco IOS and IOS XE Software, you can also nest class maps in MQC configurations by using the **match class-map** command within the class map configuration. By nesting class maps, you can create generic classification templates and more sophisticated classifications.

The syntax for the **match not** command is as follows:

match not *match-criteria*

Syntax Description

Parameter	Description
<i>match-criteria</i>	(Required) Specifies the match criterion value that is an unsuccessful match criterion. All other values of the specified match criteria will be considered successful match criteria.

Classification Options by Source and Destination

- Access group: Match all packets that access list permits
`match access-group 101`
- VLAN: Match all packets belonging to specific VLAN
`match vlan 201`
- Destination address: Match all packets destined to specific MAC address
`match destination-address mac 001f.ca6c.45d4`
- Source address: Match all packets sourced from specific MAC address
`match source-address mac 001f.ca6c.45d9`
- Input interface: Match all packets sourced from specific interface
`match input-interface FastEthernet 0/0`
- IP RTP: Match RTP packets with source or destination UDP port range numbers (Cisco IOS and IOS XE Software only)
`match ip rtp 16384 16383`

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4.6

The first set of classification options includes classification that is based on source and destination parameters of the packet, such as source and destination IP address, source and destination port numbers, source and destination MAC address, input interface, and frames belonging to specific VLAN and RTP packets that have the source or destination port within a specific range.

Access Control List

The **match access-group** command specifies a numbered or named ACL whose contents are used as the match criteria. Packets are checked against the contents of the ACL to determine if they belong to the class specified by the class map. To configure the match criteria for a class map based on the specified ACL number or name, use the **match access-group** class map configuration command.

match access-group {access-group | name *access-group-name*}

ACLs are still one of the most powerful classification tools. Class maps can use any type of ACL (not only IP ACLs).

ACLs have a drawback. Compared to other classification tools, they are very CPU-intensive. For this reason, ACLs should not be used for classification on high-speed links where they could severely impact performance of routers. ACLs are typically used on low-speed links at network edges, where packets are classified and marked (for example, with IP precedence). Classification in the core is done based on the IP precedence value.

VLAN

You can specify a single VLAN identification number, multiple VLAN identification numbers that are separated by spaces (for example, 2 5 7), or a range of VLAN identification numbers that are separated by a hyphen.

To match and classify traffic on the basis of the VLAN identification number, use the **match vlan** command in class map configuration mode.

match vlan *vlan-id-number*

Destination MAC Address

To use the destination MAC address as a match criterion, use the **match destination-address mac** command in class map configuration mode.

match destination-address mac *address*

Source MAC Address

To use the source MAC address as a match criterion, use the **match source-address mac** command in QoS class map configuration mode.

match source-address mac *address-destination*

Input Interface

The **match input-interface** command specifies the name of an input interface to be used as the match criterion against which packets are checked to determine if they belong to the class specified by the class map. To configure a class map to use the specified input interface as a match criterion, use the **match input-interface** class map configuration command.

match input-interface *interface-name*

IP RTP Port Range

This command is used to match IP RTP packets that fall within the specified port range. It matches packets that are destined to all even UDP port numbers in the range from the *starting-port-number* argument to the *starting-port-number* plus the *port-range* argument.

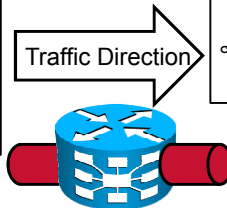
Use of an RTP port range as the match criterion is particularly effective for applications that use RTP, such as voice or video. To configure a class map to use the RTP port as the match criterion, use the **match ip rtp** command in class map configuration mode. To remove the RTP port match criterion, use the **no** form of this command.

match ip rtp *starting-port-number port-range*

Classification Based on Internal Markings

- Internal marking significant only to local device
- Common use: **mark at ingress for easier classification at egress**
- Two classification options:
 - Classification based on **qos-group**
 - Classification based on **discard-class**

```
class-map match-all Premium-in
  match dscp ef
class-map match-all Critical-in
  match dscp af31
policy-map input-policy
  class Premium
    set qos-group 5
    set discard-class 0
  class Critical
    set qos-group 4
    set discard-class 1
```



```
class-map match-any Premium-out
  match qos-group 5
  match discard-class 0
class-map match-any Critical-out
  match qos-group 4
  match discard-class 1
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.7

QoS Group

The **match qos-group** command is used by the class map to identify a specific QoS group value marking on a packet. This command can also be used to convey the received MPLS EXP field value to the output interface.

The *qos-group-value* argument is used as a marking only. The QoS group values have no mathematical significance. For instance, the *qos-group-value* of 2 is not greater than 1. The value simply indicates that a packet marked with the QoS group value of 2 is different than a packet marked with the QoS group value of 1. The treatment of these packets is defined by the user through the setting of QoS policies in QoS policy map class configuration mode.

The QoS group value is local to the router, meaning that the QoS group value that is marked on a packet does not leave the router when the packet leaves the router. If you need a marking that resides in the packet, use the IP precedence setting, the IP DSCP setting, or another method of packet marking.

To identify a specific QoS group value as a match criterion, use the **match qos-group** command in class map configuration mode. To remove a specific QoS group value from a class map, use the **no** form of this command.

match qos-group *qos-group-value*

Discard Class

A discard class value has no mathematical significance. For example, the discard class value 2 is not greater than 1. The value simply indicates that a packet marked with discard class 2 should be treated differently than a packet marked with discard class 1.

Packets that match the specified discard class value are treated differently from packets marked with other discard class values. The discard class is a matching criterion only, used in defining PHB for dropping traffic.

To specify a discard class as a match criterion, use the **match discard-class** command in class map configuration mode. To remove a previously specified discard class as a match criterion, use the **no** form of this command.

match discard-class *class-number*

Classification Based on Packet Markings

Commonly used in the core

- IP precedence: Match packets with certain IP precedence values

```
match precedence critical
```

- DSCP: Match packets with certain DSCP values

```
match dscp af41 af31 af21
```

- CoS: Match tagged Ethernet frames with certain CoS values

```
match cos 5 4
```

- MPLS EXP: Match packets with certain MPLS EXP values

```
match mpls experimental topmost 5
```

- Frame Relay DE: Match Frame Relay frames with DE bit set

```
match fr-de
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.8

Classification based on packet markings is commonly used in the core. Those frames and packets are marked at the network edge. These include classification based on IP precedence value, DSCP value, CoS bits, MPLS EXP bits, and Frame Relay DE bit.

IP Precedence

A much faster method of classification than using ACLs is matching the IP precedence. Up to four separate IP precedence values or names can be used to classify packets based on the IP Precedence field in the IP header on a single match-statement line.

The figure contains a mapping between IP precedence values and names. The running configuration, however, only shows IP precedence values (not names).

The syntax for the **match ip precedence** command is as follows:

```
match ip precedence ip-prec-value [ip-prec [ip-prec [ip-prec]]]
```

DSCP

IP packets can also be classified based on the IP DSCP field. A QoS design can be based on IP precedence marking or DSCP marking. DSCP standards make IP precedence marking obsolete but include backward compatibility with IP precedence by using the Class Selector (CS) values. CS values are 6-bit equivalents to their IP precedence counterparts, and are obtained by setting the three most significant bits of the DSCP to the IP precedence value, while holding the three least significant bits to zero.

The syntax for the **match [ip] dscp** command is as follows:

```
match [ip] dscp ip-dscp-value [ip-dscp-value ...]
```

CoS

Routers can also match the three CoS bits in 802.1Q headers or priority bits in the ISL header. These bits can be used in a LAN-switched environment to provide differentiated quality of service.

The syntax for the **match cos** command is as follows:

```
match cos cos-value [cos-value cos-value cos-value]
```

MPLS EXP

The **match mpls experimental** command specifies the name of an EXP field value to be used as the match criterion against which packets are checked to determine if they belong to the class specified by the class map.

To configure a class map to use the specified value of the EXP field as a match criterion, use the **match mpls experimental** class map configuration command. To remove the EXP field match criterion from a class map, use the **no** form of this command.

```
match mpls experimental number
```

Frame Relay DE bit

Routers can also match frames based on whether the Frame Relay DE bit is set or not. To match frames that have the Frame Relay DE bit set, use the following command:

```
match fr-de
```

Configuring Classification using MQC

This topic explains how to use MQC to implement traffic classification.

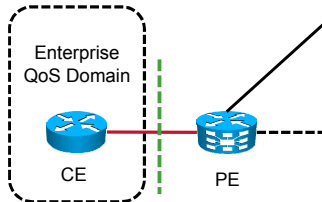
Configuring Classification Using MQC

Cisco IOS XR Software

```
ipv4 access-list Customer-Control permit ipv4 host 10.7.10.1 any precedence 6
ipv4 access-list Customer-Control permit ipv4 host 10.7.10.1 any precedence 7
ipv4 access-list Customer-Control permit ipv4 host 10.7.10.1 any dscp 48
ipv4 access-list Customer-Control permit ipv4 host 10.7.10.1 any dscp 56

ipv4 access-list Customer-Real-Time permit ipv4 host 10.7.10.1 any precedence 5
ipv4 access-list Customer-Real-Time permit ipv4 host 10.7.10.1 any dscp 46

class-map Customer-Control-in
  match access-group ipv4 Customer-Control
class-map Customer-Real-Time-in
  match access-group ipv4 Customer-Real-Time
```



Configuration of two classes on PE router:

- First is network control sourced from 10.7.10.1 IP address with certain IP precedence and DSCP values
- Second is real-time traffic with IP precedence 5 or DSCP 46 sourced from 10.7.10.1

© 2012 Cisco and/or its affiliates. All rights reserved. SPCORE v1.01-4.9

In the example, classification of traffic is configured using ACLs.

The customer is sending network control or real-time traffic using IP address 10.7.10.1.

ACL Customer-Control permits all traffic sourced from IP address 10.7.10.1 with IP precedence values of either 6 or 7, or with DSCP values of 48 or 56.

ACL Customer-Real-Time permits all traffic sourced from IP address 10.7.10.1 with an IP precedence value of 5, or with a DSCP value of 46.

These ACLs are used to classify packets within service classes, Customer-Control-in and Customer-Real-Time-in respectively. Classification of packets in this example is performed at the ingress, but it can be also performed at the egress.

Verifying Packet Classification

```
RP/0/RSP0/CPU0:PE7#show class-map list
!
1) ClassMap: Customer-Control-in      Type: qos
   Referenced by 0 Policymaps

2) ClassMap: Customer-Real-Time-in    Type: qos
   Referenced by 0 Policymaps

RP/0/RSP0/CPU0:PE7#show running-config class-map
!
class-map match-any Customer-Control-in
match access-group ipv4 Customer-Control
end-class-map
!
class-map match-any Customer-Real-Time-in
match access-group ipv4 Customer-Real-Time
end-class-map

RP/0/RSP0/CPU0:PE7#show access-lists ipv4 Customer-Control
!
ipv4 access-list Customer-Control
10 permit ipv4 host 10.7.10.1 any precedence internet
20 permit ipv4 host 10.7.10.1 any precedence network
30 permit ipv4 host 10.7.10.1 any dscp cs6
40 permit ipv4 host 10.7.10.1 any dscp cs7
```

- Verify class maps.
- Verify class map configuration.
- Verify access list used for class map.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4-10

Cisco IOS and IOS XE Software

The **show class-map** command lists all class maps with their match statements. This command can be issued from the EXEC or privileged EXEC mode. The **show class-map** command with a name of a class map displays the configuration of the selected class map.

In the figure, the **show class-map** command shows all the class maps that have been configured and which match statements are contained in the maps.

show class-map [*class-map-name*]

Cisco IOS XR Software

Verification of MQC classification is performed by using different commands. To view a list of configured service classes, use the following command:

show class-map list

To verify configured class maps and view match statements within class map commands, use the following command in privileged EXEC mode:

show running-config class-map

Using MQC for Class-Based Marking

This topic describes using MQC class-based marking.

Marking Options in MQC

- Class-based marking: static per-class marking of packets
- Used to mark inbound or outbound traffic
- Combined with any QoS feature on output
- Combined with policing on input
- Prerequisite for configuring class-based marking: IP Cisco Express Forwarding

Options for marking (set statements):

- IP precedence
- DSCP value
- QoS group number
- MPLS EXP bits
- 802.1Q or ISL CoS bits
- Frame Relay DE bit

© 2012 Cisco and/or its affiliates. All rights reserved.SPCORE v1.01—4-11

Marking packets or frames places information in the Layer 2 and Layer 3 headers of a packet so that the packet or frame can be identified and distinguished from other packets or frames.

MQC provides packet-marking capabilities using class-based marking. You can use class-based marking on the input or output of interfaces as part of a defined input or output service policy. On input, you can combine class-based marking with class-based policing, and on output, with any other class-based QoS feature.

Class-based marking supports these markers:

- IP precedence
- IP DSCP value
- QoS group
- MPLS EXP bits
- IEEE 802.1Q or ISL CoS or priority bits
- Frame Relay DE bit

Marking Options in MQC (Cont.)

- IP precedence: mark packets of class to specified IP precedence value

```
set precedence 5
```

- DSCP: mark packets of class to specified DSCP value

```
set dscp af31
```

- QoS group: mark packets of class to specified QoS group value

```
set qos-group
```

- MPLS EXP: mark packets of class to specified value of MPLS EXP bit

```
set mpls experimental topmost 5
```

- 802.1Q or ISL CoS: mark frames of class to specified CoS value

```
set cos 4
```

- Frame Relay DE: mark frames of class by setting Frame Relay DE bit

```
set fr-de
```

© 2012 Cisco and/or its affiliates. All rights reserved.SPCORE v1.01–4.12

IP Precedence

To set the precedence value in the packet header, use the **set precedence** command in policy map class configuration mode. The syntax of this command is as follows:

set precedence *precedence-value*

DSCP

The **set dscp** command cannot be used with the **set precedence** command to mark the same packet. The two values, DSCP and precedence, are mutually exclusive. A packet can have one value or the other, but not both.

To mark a packet by setting the DSCP value in the type of service (ToS) byte, use the **set dscp** command in QoS policy map class configuration mode.

set dscp {*dscp-value* | *from-field* [**table** *table-map-name*]}

Syntax Description

Parameter	Description
ip	(Optional) Specifies that the match is for IPv4 packets only. If not used, the match is on both IPv4 and IPv6 packets.
<i>ip-dscp-value</i>	A number from 0 to 63 that sets the DSCP value. The following keywords are examples of reserved keywords that can be specified instead of numeric values: EF (expedited forwarding) AF11 (assured forwarding class AF11) AF12 (assured forwarding class AF12)

Parameter	Description
<i>from-field</i>	Specific packet-marking category to be used to set the DSCP value of the packet. If you are using a table map for mapping and converting packet-marking values, this establishes the "map from" packet-marking category. Packet-marking category keywords are as follows: cos qos-group
table	(Optional) Used in conjunction with the <i>from-field</i> argument. Indicates that the values set in a specified table map will be used to set the DSCP value.
<i>table-map-name</i>	(Optional) Used in conjunction with the table keyword. The name of the table map used to specify the DSCP value. The name can be a maximum of 64 alphanumeric characters.

QoS Group

To set a QoS group identifier that can be used later to classify packets, use the **set qos-group** command in policy map class configuration mode. The syntax of this command is as follows:

set qos-group *value*

MPLS EXP

The **set mpls experimental** command has two options:

set mpls experimental topmost {*mpls-exp-value* | *qos-group* [*table* *table-map-name*]}

set mpls experimental imposition {*mpls-exp-value* | *qos-group* [*table* *table-map-name*]}

Note The new **set mpls experimental topmost** command is equivalent to the old **set mpls imposition** command.

These two commands, in combination with some new command switches, allow better control of MPLS EXP bits manipulation during label push, swap, and pop operations. These two commands allow you to use DiffServ tunneling modes.

CoS

To set the Layer 2 class of service (CoS) value of an outgoing packet, use the **set cos** command in policy map class configuration mode.

set cos {*cos-value* | *from-field* [**table** *table-map-name*]}

Arguments used in the *from-field* option have the same meaning as in the DSCP configuration command description.

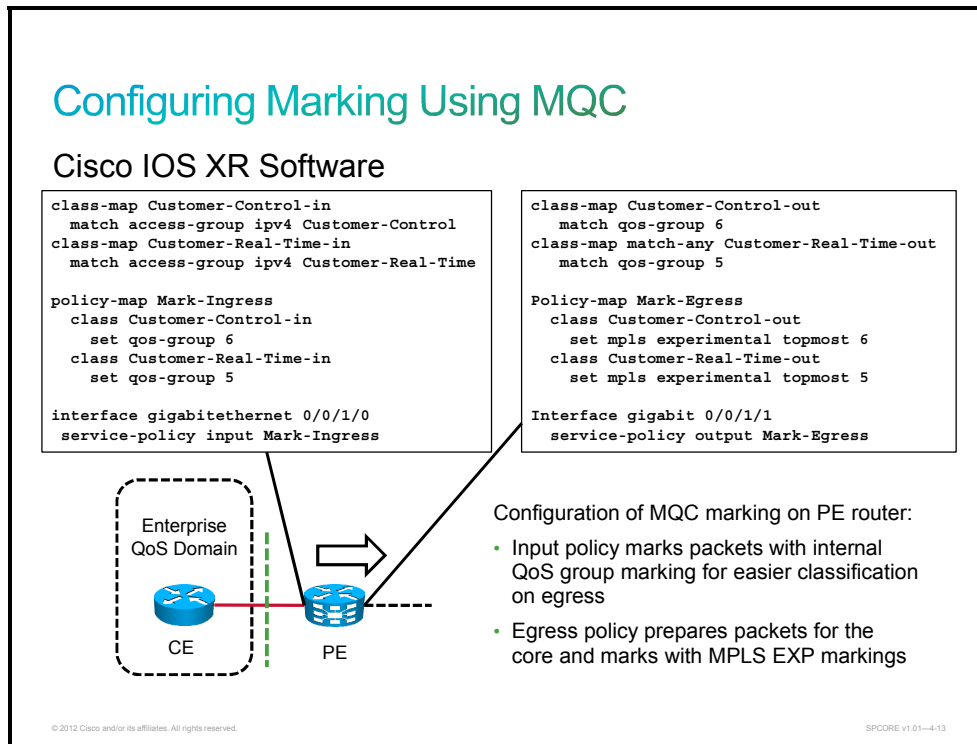
Frame Relay DE Bit

To change the DE bit setting in the address field of a Frame Relay frame to 1 for all traffic leaving an interface, use the **set fr-de** command in policy map class command. The syntax of this command is as follows:

set fr-de

Configuring Class-Based Marking using MQC

This topic explains how to use MQC to implement class-based marking.



When configuring class-based marking, you must complete these three configuration steps:

- Step 1** Create a class map.
- Step 2** Create a policy map.
- Step 3** Attach the policy map to an interface by using the **service-policy** command.

The syntax for the **class-map** command is as follows:

class-map {**match any** | **match all**} *class-map-name*

In the example, the input policy marks packets with internal QoS group marking for easier classification at egress. The output policy marks packets with MPLS EXP values, because QoS group markings have only local significance. This way, MPLS frames are prepared for the core, and have proper markings in the MPLS header.

Verifying Class-Based Marking

- Verification of configured policy map applied on the interface
- Verification of running configuration
- Verification of packet counters in policy map

```
P/0/RSP0/CPU0:PE7# show policy-map interface GigabitEthernet 0/0/0/0

GigabitEthernet0/0/0/0 input: Mark-Ingress

Class Customer-Control-in
  Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :              0/0              0
  Transmitted   : N/A
  Total Dropped : N/A
Class Customer-Real-Time-in
  Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :             10/1180             0
  Transmitted   : N/A
  Total Dropped : N/A
Class class-default
  Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :             38/3384             0
  Transmitted   : N/A
  Total Dropped : N/A
GigabitEthernet0/0/1/0 direction output: Service Policy not installed
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4-14

In Cisco IOS and IOS XE Software, the **show policy-map** command displays all classes for the service policy specified in the command line.

To display the configuration of all classes for a specified service policy map or all classes for all existing policy maps, use the **show policy-map EXEC** or privileged EXEC command.

The syntax for the **show policy-map** command is as follows:

show policy-map [*policy-map*]

In Cisco IOS, IOS XE, and IOS XR Software, the **show policy-map interface** command displays all service policies applied to the interface. In addition to the settings, marking parameters and statistics are displayed.

To display policy configuration information in Cisco IOS XR Software for all classes configured for all service policies on the specified interface, use the **show policy-map interface** command in EXEC mode.

show policy-map interface *type instance* [**input** | **output** [**member type instance**]]

Syntax Description

Parameter	Description
<i>type</i>	Interface type. For more information, use the question mark (?) online help function.
<i>instance</i>	Either a physical interface instance or a virtual interface instance as follows: - Physical interface instance: Naming notation is <i>rack/slot/module/port</i> and a slash between values is required as part of the notation. <i>rack</i>: Chassis number of the rack. <i>slot</i>: Physical slot number of the modular services card or line card. <i>module</i>: Module number. A physical layer interface module (PLIM) is always 0. <i>port</i>: Physical port number of the interface. Note: In references to a management Ethernet interface located on a route processor card, the physical slot number is alphanumeric (RP0 or RP1) and the module is CPU0. Example: interface MgmtEth0/RP1/CPU0/0 - Virtual interface instance: Number range varies depending on interface type. For more information about the syntax for the router, use the question mark (?) online help function.
input	(Optional) Attaches the specified policy map to the input interface.
output	(Optional) Attaches the specified policy map to the output interface.
member	(Optional) Specifies the interface of the bundle member.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- MQC classification options include classification based on source and destination parameters, classification based on internal markings, and classification based on packet markings.
- Use **show class-map** command to list all class maps with their match statements
- Marking can be configured on the ingress or egress interfaces.
- Use **show policy-map** command to display all classes for the service policy specified in the command line

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-15

Implementing Advanced QoS Techniques

Overview

Advanced quality of service (QoS) techniques include Network-Based Application Recognition (NBAR), QoS tunneling techniques, QoS policy propagation via Border Gateway Patrol (BGP), and hierarchical QoS.

NBAR, a feature in Cisco IOS Software, provides intelligent classification for the network infrastructure. NBAR is a classification engine that can recognize a wide variety of protocols and applications, including web-based applications and client and server applications that dynamically assign TCP or UDP port numbers.

The QoS for VPNs feature (QoS preclassify) provides a solution for ensuring that Cisco IOS QoS services operate in conjunction with tunneling and encryption on an interface.

QoS Policy Propagation via BGP (QPPB) allows an ISP to implement different QoS policies for different customers using the BGP routes of that customer.

MQC-based QoS tools are that these can be combined in a hierarchical fashion, meaning, MQC policies can contain other “nested” QoS policies within them. Such policy combinations are commonly referred to as hierarchical QoS (or HQoS) policies.

This lesson describes the operation of these advanced QoS techniques and how to configure them.

Objectives

Upon completing this lesson, you will be able to use NBAR for traffic classification, use QoS preclassification, and implement classification and marking in an interdomain network using QPPB. You will be able to meet these objectives:

- Describe using NBAR to discover network protocols and to classify packets
- Explain how to configure MQC Traffic Classification using the match protocol option
- Describe issues when implementing QoS with VPN and tunneling and the QoS Pre-Classify solution
- Explain how to configure QoS Pre-Classify
- Describe the QPPB classification mechanism
- Explain how to configure QPPB
- Describe a QoS implementation example using hierarchical QoS

Network-Based Application Recognition

This topic describes how to use NBAR to discover network protocols and classify packets.

NBAR Overview

- Available in Cisco IOS and IOS XE Software
- Solves problem of how to classify modern applications
- NBAR performs following functions:
 - Identification of application and protocols
 - Protocol discovery
 - Provides traffic statistics

Example: filter peer-to-peer applications

```
class-map match-any p2p          policy-map Filter-p2p
 match protocol kazaa2          class p2p
 match protocol edonkey         drop
 match protocol gnutella
 match protocol bittorrent      interface fastethernet 0/0
                                service-policy input Filter-p2p
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4.3

NBAR is a classification engine that recognizes and classifies a wide variety of protocols and applications, including web-based and other difficult-to-classify applications and protocols that use dynamic TCP/UDP port assignments.

When NBAR recognizes and classifies a protocol or application, the network can be configured to apply the appropriate QoS for that application or traffic with that protocol. The QoS is applied using the Modular QoS CLI, or MQC.

Examples of the QoS features that can be applied to the network traffic (using the MQC), after NBAR has recognized and classified the application or protocol, include the following:

- Class-based marking
- Class-based weighted fair queuing (CBWFQ)
- Low latency queuing (LLQ)
- Traffic policing
- Traffic shaping

NBAR includes a feature called Protocol Discovery that provides an easy way to discover application protocols that are operating on an interface. The Protocol Discovery feature discovers any protocol traffic supported by NBAR. You can apply Protocol Discovery to interfaces and use it to monitor both input and output traffic. Protocol Discovery maintains per-protocol statistics for enabled interfaces such as total number of input and output packets and bytes, and input and output bit rates.

You can load an external Packet Description Language Module (PDL) at run time to extend the NBAR list of recognized protocols. PDLs allow NBAR to recognize new protocols without requiring a new Cisco IOS image or a router reload.

NBAR introduces powerful application classification features into the network at a small-to-medium CPU overhead cost. The CPU utilization will vary based on factors such as the router processor speed and type, and the traffic rate.

NBAR gives you the ability to see the variety of protocols and the amount of traffic generated by each protocol. After gathering this information, NBAR allows you to organize traffic into classes.

NBAR Basic Requirements and Restrictions

- Cisco Express Forwarding must be enabled
- NBAR not supported on:
 - Fast EtherChannel
 - Interfaces where tunneling or encryption is used
- NBAR does not support the following:
 - More than 24 concurrent URLs
 - Non-IP traffic (MPLS-labeled packets not supported)
 - Fragmented packets
 - URL, host, or MIME classification with HTTPS
 - Traffic originated from or destined to the router running NBAR

© 2012 Cisco and/or its affiliates. All rights reserved.SPCORE v1.01--4.4

The following requirements and restrictions apply to NBAR:

- Before you configure NBAR, you must enable Cisco Express Forwarding.
- NBAR does not support the following:
 - More than 24 concurrent URLs, hosts, or Multipurpose Internet Mail Extension (MIME)-type matches.
 - Non-IP traffic.
 - Multiprotocol Label Switching (MPLS)-labeled packets. NBAR classifies IP packets only. You can, however, use NBAR to classify IP traffic before the traffic is handed over to MPLS.
 - Multicast and switching modes other than Cisco Express Forwarding.
 - Fragmented packets.
 - Pipelined persistent HTTP requests.
 - URL, host, or MIME classification with secure HTTP.
 - Asymmetric flows with stateful protocols.
 - Packets that originate from or that are destined to the router running NBAR.

- NBAR is not supported on the following logical interfaces:

- Fast EtherChannel
- Interfaces where tunneling or encryption is used

Note You cannot use NBAR to classify output traffic on a WAN link where tunneling or encryption is used. Therefore, you should configure NBAR on other interfaces on the router (such as a LAN link) to perform input classification before the traffic is switched to the WAN link for output. However, NBAR protocol discovery is supported on interfaces on which tunneling or encryption is used. You can enable protocol discovery directly on the tunnel or on the interface on which encryption is performed to gather key statistics about the various applications that are traversing the interface. The input statistics also show the total number of encrypted or tunneled packets received in addition to the per-protocol breakdowns.

NBAR Application Support

- Statically assigned TCP and UDP port numbers
- Non-TCP and non-UDP protocols
- Dynamically assigned TCP and UDP port numbers
- Deep packet inspection
- Differentiate about 100 protocols and applications

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-6

NBAR supports simpler configuration that is coupled with stateful recognition of flows. The simpler configuration means that a protocol analyzer capture does not need to be examined to calculate ports and details. Stateful recognition means smarter, deeper packet recognition. NBAR can be used to recognize and classify packets belonging to the following types of protocols and applications:

- **Applications that use statically assigned TCP and UDP port numbers:** These applications establish sessions to well-known TCP or UDP destination port numbers. Access control lists (ACLs) can also be used for classifying static port protocols. However, NBAR is easier to configure, and NBAR can provide classification statistics that are not available when ACLs are used.
- **Applications that use dynamically assigned TCP and UDP port numbers:** These applications use multiple sessions that use dynamic TCP or UDP port numbers. Typically, there is a control session to a well-known port number and the other sessions are established to destination port numbers negotiated through the control sessions. NBAR inspects the port number exchange through the control session. This kind of classification

requires stateful inspection—that is, the ability to inspect a protocol across multiple packets during packet classification.

- **Non-TCP and non-UDP IP protocols:** Some non-TCP and non-UDP IP protocols can be recognized by NBAR.

NBAR also has the capability to perform subport classification or classification that is based on deep-packet inspection. Deep-packet classification is classification that is performed at a finer level of granularity. For instance, if a packet is already classified as HTTP traffic, it may be further classified as HTTP traffic with a specific URL.

NBAR Application Support (Cont.)

List of applications varies depending on type and version of Cisco IOS Software

TCP and UDP Static Port Protocols				
BGP	IMAP	NNTP	RSVP	SNMP
BOOTP	IRC	Notes	SFTP	SOCKS
CU-SeeMe	Kerberos	Novadigm	SHTP	SQL Server
DHCP/DNS	L2TP	NTP	SIMAP	SSH
Finger	LDAP	PCAnywhere	SIRC	STELNET
Gopher	MS-PPTP	POP3	SLDAP	Syslog
HTTP	NetBIOS	Printer	SMTP	Telnet
HTTPS	NFS	RIP	SNMP	X Windows

TCP and UDP Stateful Protocols				Non-UDP and Non-TCP Protocols	
Citrix ICA	Gnutella	R-commands	StreamWorks	EGP	ICMP
Exchange	HTTP	RealAudio	SunRPC	EIGRP	IPINIP
FastTrack	Napster	RTP	TFTP	GRE	IPSec
FTP	Netshow	SQL*NET	VDOLive		

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.6

The tables list some of the NBAR-supported protocols available in Cisco IOS Software. The tables also provide information about the protocol type and the well-known port numbers (if applicable).

Non-TCP and Non-UDP NBAR-Supported Protocols

Protocol	Network Protocol	Protocol ID	Description
EGP	IP	8	Exterior Gateway Protocol
GRE	IP	47	Generic Routing Encapsulation
ICMP	IP	1	Internet Control Message Protocol
IPIP	IP	4	IP in IP
IPsec	IP	50, 51	IP Encapsulating Security Payload (ESP=50) and Authentication Header (AH=51)
EIGRP	IP	88	Enhanced Interior Gateway Routing Protocol
OSPF	IP	89	Open Shortest Path First

This table shows the IP protocols that are supported by NBAR.

TCP and UDP NBAR-Supported Protocols

Protocol	Network Protocol	Protocol ID	Description
AOL-messenger	TCP	5190, 443	AOL Instant Messenger chat messages
BGP	TCP/UDP	179	Border Gateway Protocol
Citrix ICA	TCP/UDP	TCP: 1494, 2512, 2513, 2598 UDP: 1604	Citrix ICA traffic
CU-SeeMe	TCP/UDP	TCP: 7648, 7649 UDP: 24032	Desktop video conferencing
DHCP/BOOTP	UDP	67, 68	Dynamic Host Configuration Protocol/ Bootstrap Protocol
DNS	TCP/UDP	53	Domain Name System
Doom	TCP/UDP	666	Doom
Exchange	TCP	135	MS-RPC for Exchange
FastTrack	TCP/UDP	Dynamically assigned	FastTrack peer-to-peer protocol
Finger	TCP	79	Finger user information protocol
FTP	TCP	Dynamically assigned, 20, 21	File Transfer Protocol
HTTP	TCP	80	Hypertext Transfer Protocol
HTTPS	TCP	443	Secure HTTP
IMAP	TCP/UDP	143, 220	Internet Message Access Protocol
IRC	TCP/UDP	194	Internet Relay Chat
Kazaa	TCP/UDP	Dynamically assigned	Kazaa
Kerberos	TCP/UDP	88, 749	Kerberos network authentication service
L2TP	UDP	1701	Layer 2 Tunneling Protocol
LDAP	TCP/UDP	389	Lightweight Directory Access Protocol
MSN-messenger	TCP	1863	MSN Messenger chat messages
NetShow	TCP/UDP	Dynamically assigned	Microsoft NetShow
NNTP	TCP/UDP	119	Network News Transfer Protocol
Notes	TCP/UDP	1352	Lotus Notes
Novadigm	TCP/UDP	3460-3465	Novadigm Enterprise Desktop Manager (EDM)
NTP	TCP/UDP	123	Network Time Protocol
PCAnywhere	TCP/UDP	TCP: 5631, 65301 UDP: 22, 5632	Symantec PCAnywhere

Protocol	Network Protocol	Protocol ID	Description
POP3	TCP/UDP	110	Post Office Protocol
RealAudio	TCP/UDP	Dynamically assigned	RealAudio Streaming Protocol
RSVP	UDP	1698,1699	Resource Reservation Protocol
RTSP	TCP/UDP	Dynamically assigned	Real Time Streaming Protocol
SFTP	TCP	990	Secure FTP
SIP	TCP/UDP	5060	Session Initiation Protocol
Skinny (SCCP)	TCP	2000, 2001, 2002	Skinny Client Control Protocol
Skype	TCP/UDP	Dynamically assigned	Peer-to-Peer VoIP Client Software
SMTP	TCP	25	Simple Mail Transfer Protocol
SNMP	TCP/UDP	161, 162	Simple Network Management Protocol
SOCKS	TCP	1080	Firewall security protocol
SQL*NET	TCP/UDP	1521	SQL*NET for Oracle
SSH	TCP	22	Secure Shell Protocol
SunRPC	TCP/UDP	Dynamically assigned	Sun Remote Procedure Call
Syslog	UDP	514	System logging utility
Telnet	TCP	23	Telnet protocol
TFTP	UDP	Static (69) with inspection	Trivial File Transfer Protocol
VDOLive	TCP/UDP	Static (7000) with inspection	VDOLive Streaming Video
Yahoo-messenger	TCP	5050, 5101	Yahoo Messenger chat messages
YouTube	TCP	Both static (80) and dynamically assigned	Online video-sharing website

Note For a complete list of NBAR-supported protocols (and details regarding protocol support with specific platforms and software versions), refer to the Classification section of the Cisco IOS Quality of Service Solutions Configuration Guide, Release 12.4 at <http://www.cisco.com>.

Protocol Discovery

- Analyzes application traffic patterns in real time
- Provides bidirectional, per-interface protocol statistics

Enabling NBAR protocol discovery on interface:

```
CE7(config-if)#ip nbar protocol-discovery
```

Monitoring traffic statistics with protocol discovery:

```
CE7#show ip nbar protocol-discovery stats packet-count top-n 3

GigabitEthernet0/0
Last clearing of "show ip nbar protocol-discovery" counters 00:06:02

  Protocol                Input          Output
  -----                -
  Packet Count            Packet Count
  -----                -
  --
  bgp                     34             34
  ospf                     0             42
  appletqc                 0              0
  unknown                  0             12
  Total                   34            88
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.7

NBAR includes a Protocol Discovery feature that provides an easy way to discover application protocols that are transiting an interface so that appropriate QoS features can be applied. The Protocol Discovery feature discovers any protocol traffic that is supported by NBAR.

Use the **ip nbar protocol-discovery** command in interface configuration mode (or VLAN configuration mode for Catalyst switches) to configure NBAR to keep traffic statistics for all protocols known to NBAR.

Use the **show ip nbar protocol-discovery** command to display statistics gathered by the NBAR Protocol Discovery feature. This command, by default, displays statistics for all interfaces on which protocol discovery is currently enabled.

The syntax for the **show ip nbar protocol-discovery** command in Cisco IOS Software Release 12.4 is as follows:

show ip nbar protocol-discovery [*interface type number*] [*stats {byte-count | bit-rate | packet-count | max-bit-rate}*] [*protocol protocol-name*] [*top-n number*]

Syntax Description

Parameter	Description
interface	(Optional) Specifies that protocol discovery statistics for the interface are to be displayed
<i>type</i>	Type of interface or subinterface whose policy configuration is to be displayed
<i>number</i>	Port, connector, VLAN, or interface card number
stats	(Optional) Specifies that the byte count, byte rate, or packet count is to be displayed
byte-count	(Optional) Specifies that the byte count is to be displayed
bit-rate	(Optional) Specifies that the bit rate is to be displayed
packet-count	(Optional) Specifies that the packet count is to be displayed

Parameter	Description
max-bit-rate	(Optional) Specifies that the maximum bit rate is to be displayed
protocol	(Optional) Specifies that statistics for a specific protocol are to be displayed
<i>protocol-name</i>	(Optional) User-specified protocol name for which the statistics are to be displayed
top-n	(Optional) Specifies that a top-n is to be displayed. A top-n is the number of most active NBAR-supported protocols, where n is the number of protocols to be displayed. For instance, if top-n 3 is entered, the three most active NBAR-supported protocols will be displayed.
<i>number</i>	(Optional) Specifies the number of most active NBAR-supported protocols to be displayed

NBAR HTTP Classification

- Static protocols configuration commonly recognized by port number (in this case by port 80):

```
Router(config-cmap)# match protocol http
```

- Mapping other than well-known port number to protocol (also mapping port 8080 port to HTTP):

```
Router(config)# ip nbar port-map http 80 8080
```

- Configuring deep packet inspection (subport classification) (matching host field in HTTP request):

```
Router(config-cmap)# match protocol http host *youtube.com*|*video.google.com*
```

© 2012 Cisco and/or its affiliates. All rights reserved.
SPCORE v1.01—4.8

The MQC uses traffic classes and traffic policies (policy maps) to apply QoS features to classes of traffic and applications recognized by NBAR. Configuring NBAR using the MQC involves defining a traffic class, configuring a traffic policy (policy map), and then attaching that traffic policy to the appropriate interface.

HTTP is often used on ports other than its well-known port, TCP port 80. In the example, the **ip nbar port-map** command is used to enable HTTP recognition on both TCP port 80 and TCP port 8080. One match statement in the class map that is called is used to match the HTTP protocol on ports 80 and 8080.

NBAR can classify application traffic by looking beyond the TCP and UDP port numbers of a packet. This capability is called subport classification. NBAR looks into the TCP or UDP payload itself and classifies packets based on content within the payload, such as transaction identifier or message type. Classification of HTTP traffic by URL, host, or MIME type is an example of subport classification.

The syntax for the **match protocol http** command in Cisco IOS Software Release 12.4 is as follows:

match protocol http [**url** *url-string* | **host** *hostname-string* | **mime** *MIME-type* | **c-header-field** *c-header-field-string* | **s-header-field** *s-header-field-string*]

Syntax Description

Parameter	Description
url	(Optional) Specifies matching by a URL
<i>url-string</i>	(Optional) User-specified URL of HTTP traffic to be matched
host	(Optional) Specifies matching by a hostname
<i>hostname-string</i>	(Optional) User-specified hostname to be matched
mime	(Optional) Specifies matching by a MIME text string
<i>MIME-type</i>	(Optional) User-specified MIME text string to be matched
c-header-field	(Optional) Specifies matching by a string in the header field in HTTP request messages
<i>c-header-field-string</i>	(Optional) User-specified text string within the HTTP request message to be matched
s-header-field	(Optional) Specifies matching by a string in the header field in HTTP response messages
<i>s-header-field-string</i>	(Optional) User-specified text within the HTTP response message to be matched

When matching by host, NBAR performs a regular expression match on the host field contents inside the HTTP packet and classifies all packets from that host.

To match the `www.anydomain.com` portion, use the hostname matching feature. The parameter specification strings can take the form of a regular expression with the options shown in the table.

Parameter	Description
*	Match zero or more characters in this position.
?	Match any one character in this position.
	Match one of a choice of characters.
()	Match one of a choice of characters in a range. For example, <code>cisco.(gif jpg)</code> matches either <code>cisco.gif</code> or <code>cisco.jpg</code> .
[]	Match any character in the range specified, or one of the special characters. For example, <code>[0-9]</code> is all of the digits. <code>[*]</code> is the <code>"**"</code> character and <code>[]</code> is the <code>"["</code> character.

The following example classifies, within class-map class1, packets based on any hostname containing the string *youtube.com* followed by or preceding zero or more characters:

```
class-map class1
  match protocol http host *cisco*
```

NBAR syntax is slightly different in Cisco IOS XE Software compared to Cisco IOS Software. For example, in the following, HTTP header fields are combined with a URL to classify traffic. In this example, traffic with a User-Agent field of CERN-LineMode/3.0 and a Server field of CERN/3.0, along with the URL www.cisco.com/routers, will be classified using NBAR:

```
class-map match-all c-http
  match protocol http user-agent "CERN-LineMode/3.0"
  match protocol http server "CERN/3.0"
  match protocol http url "www.cisco.com/routers"
```

Packet Description Language Module

- IOS Software recognizes more than 100 applications and protocols
- External PLDM can be loaded to extend the list of protocols
- Also used to enhance existing protocol recognition
- No new IOS version or reload required
- Currently available PLDMs:
 - BitTorrent, eDonkey2000, Kazaa2, Gnutella, WinMX, and Citrix ICA

Example: Load Citrix PLDM in Cisco IOS and IOS XE Software:

```
Router(config)# ip nbar pldm flash://citrix.pldm
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.0

New features are usually added to new versions of the Cisco IOS Software. NBAR is the first mechanism that supports dynamic upgrades without having to change the Cisco IOS Software version or restart a router. This is accomplished by loading one or more PDLMs onto a router. Adding PDLMs extends the functionality of NBAR by enabling NBAR to recognize additional protocols on your network.

A PDLM is a separate file available on <http://www.cisco.com>. You can load an external PDLM at run time to extend the NBAR list of recognized protocols. PDLMs allow NBAR to recognize new protocols without requiring a new Cisco IOS image or a router reload. PDLMs that are not embedded within Cisco IOS Software are referred to as non-native PDLMs. A native PDLM is a PDLM that is embedded within the Cisco IOS Software. You receive it automatically along with the Cisco IOS Software.

There are separate version numbers associated with the NBAR software and the Cisco IOS Software. These version numbers are used together to maintain the PDLM version.

- **PDLM version:** The version of the PDLM, either native or nonnative.
- **Cisco IOS NBAR software version:** The version of NBAR that resides with the Cisco IOS Software. You can display the Cisco IOS NBAR software version by executing the **show ip nbar version** command.

Creating and Applying Custom Protocols

Goal: New custom applications that NBAR recognizes

Example: Create custom protocol with following properties:

- Source **TCP port 4567**
- Fifth byte of payload contains term **SALES**

```
Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567
```

Create class map that matches app_sales1 custom protocol:

```
Router(config)# class-map class1
Router(config-cmap)# match protocol app_sales1
```

Create policy and apply CBWFQ feature to class:

```
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 50
```

Apply service policy to interface:

```
Router(config)# interface ethernet 2/4
Router(config-if)# service-policy input policy1
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4-10

NBAR supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not currently support.

NBAR recognizes and classifies network traffic by protocol or application. You can extend the set of protocols and applications that NBAR recognizes by creating a custom protocol. Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify unsupported static port traffic. You define a custom protocol by using the keywords and arguments of the **ip nbar custom** command. However, after you define the custom protocol, you must create a traffic class and configure a traffic policy (policy map) to use the custom protocol when NBAR classifies traffic.

Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications, and allow NBAR to classify unsupported static port traffic.

To define a custom protocol, use the following command:

ip nbar custom *name* [*offset* [*format value*]] [*variable field-name field-length*]
[*source* | *destination*] [**tcp** | **udp**] [*range start end* | *port-number*]

Syntax Description

Parameter	Description
<i>name</i>	The name given to the custom protocol. This name is reflected wherever the name is used, including NBAR Protocol Discovery, the match protocol command, the ip nbar port-map command, and the NBAR Protocol Discovery MIB. The name must be no longer than 24 characters and can contain only lowercase letters (a-z), digits (0-9), and the underscore (_) character.
<i>offset</i>	(Optional) A digit representing the byte location for payload inspection. The offset function is based on the beginning of the payload directly after the TCP or UDP header.

Parameter	Description
<i>format value</i>	(Optional) Defines the format and length of the value that is being inspected in the packet payload. Current format options are ASCII, hex, and decimal. The length of the value is dependent on the chosen format. The length restrictions for each format are listed below: ASCII: Up to 16 characters can be searched. Regular expressions are not supported. Hex: Up to 4 bytes. Decimal: Up to 4 bytes.
variable <i>field-name</i> <i>field-length</i>	(Optional) When you enter the variable keyword, a specific portion of the custom protocol can be treated as an NBAR-supported protocol. For example, a specific portion of the custom protocol can be tracked using class-map statistics and can be matched using the class-map command. If you enter the variable keyword, you must define the following fields: <i>field-name</i> : Provides a name for the field to search in the payload. After you configure a custom protocol using a variable, you can use this field name with up to 24 different values per router configuration. <i>field-length</i> : Enters the field length in bytes. The field length can be up to 4 bytes, so you can enter 1, 2, 3, or 4 as the field-length value.
source destination	(Optional) Specifies the direction in which packets are inspected. If you do not specify source or destination, all packets traveling in either direction are monitored by NBAR.
tcp udp	(Optional) Specifies the TCP or the UDP implemented by the application.
range <i>start end</i>	(Optional) Specifies a range of ports that the custom application monitors. The <i>start</i> is the first port in the range, and the <i>end</i> is the last port in the range. One range of up to 1000 ports can be specified for each custom protocol.
<i>port-number</i>	(Optional) The port that the custom application monitors. Up to 16 individual ports can be specified as a single custom protocol.

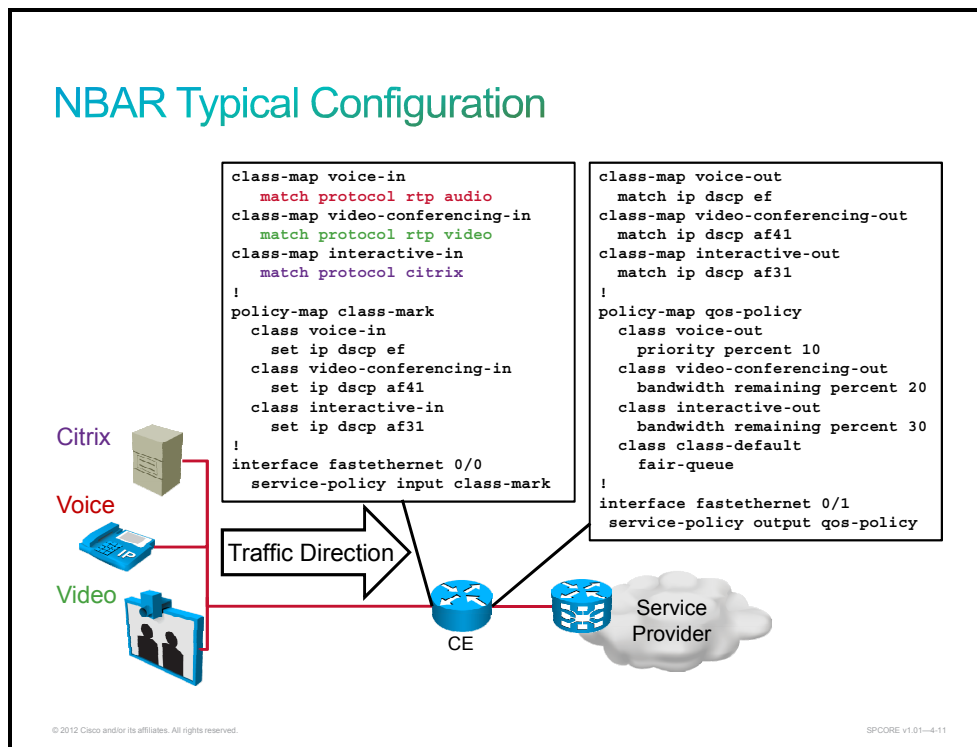
In the following example, the custom protocol app-sales1 will identify TCP packets that have a source port of 4567 and that contain the term “SALES” in the fifth byte of the payload:

```
Router(config)# ip nbar custom app-sales1 5 ascii SALES source
tcp 4567
```

To create traffic classes and policies that will be applied to an interface, we will use the standard functionality of the Modular QoS already defined.

Configuring MQC Traffic Classification Using NBAR (match protocol)

This topic explains how to configure MQC Traffic Classification using the match protocol option.



The example in the figure illustrates a simple classification of RTP sessions, both on the input interface and on the output interface of the router. On the input interface, three class maps have been created: voice-in, video-conferencing-in, and interactive-in. The voice-in class map will match the RTP audio protocol, the video-conferencing-in class map will match the RTP video protocol, and the interactive-in class map will match the Citrix protocol.

The policy map class-mark will then do the following:

- If the packet matches the voice-in class map, the packet differentiated services code point (DSCP) field will be set to Expedited Forwarding (EF). If the packet matches the videoconferencing-in class map, the packet DSCP field will be set to AF41. If the packet matches the interactive-in class map, the DSCP field will be set to AF31.
- The policy map class-mark is applied to the input interface, FastEthernet0/0.

On the output interface, three class maps have been created: voice-out, videoconferencing-out, and interactive-out. The voice-out class map will match the DSCP field for EF. The videoconferencing-out class map will match the DSCP field for AF41. The interactive-out class map will match the DSCP field for AF31.

In the figure, policy map qos-policy will then do the following:

- If the packet matches the class map voice-out, the LLQ priority bandwidth will be set to 10 percent of the interface bandwidth. If the packet matches the class map videoconferencing-out, the CBWFQ minimum-guaranteed bandwidth will be set to 20 percent of the interface bandwidth. If the packet matches the class map interactive-out, the CBWFQ will be set to 30 percent. All other packet flows will be classified as class-default, and fair queuing will be performed on them.
- The policy map class-mark is applied to the output interface, FastEthernet0/1.

NBAR Verification and Monitoring

Verify ports assigned to protocol:

```
CE7#show ip nbar port-map
port-map appleqtcp          udp 458
port-map appleqtcp          tcp 458
port-map bgp                 udp 179
port-map bgp                 tcp 179
port-map bittorrent          tcp 6969 6881 6882 6883 6884 6885 6886
6887 6888 6889
... <output omitted>
```

Monitor traffic statistics with protocol discovery:

```
CE7#show ip nbar protocol-discovery stats packet-count top-n 3

GigabitEthernet0/0
Last clearing of "show ip nbar protocol-discovery" counters 00:06:02

      Input      Output
      ----      -
Protocol      Packet Count      Packet Count
-----
--
bgp            34                34
ospf           0                42
appleqtcp      0                 0
unknown        0                12
Total          34                88
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4-12

Use the **show ip nbar protocol-discovery** command to display statistics gathered by the NBAR Protocol Discovery feature. This command, by default, displays statistics for all interfaces on which protocol discovery is currently enabled. The default output of this command includes, in the following order, input bit rate (in bits per second), input byte count, input packet count, and protocol name.

Protocol discovery can be used to monitor both input and output traffic and may be applied with or without a service policy enabled. NBAR Protocol Discovery gathers statistics for packets switched to output interfaces. These statistics are not necessarily for packets that exited the router on the output interfaces, because packets may have been dropped after switching for various reasons, including policing at the output interface, access lists, or queue drops. Syntax of this command is explained earlier in this lesson.

To display the current protocol-to-port mappings in use by NBAR, use the **show ip nbar port-map** privileged EXEC command.

show ip nbar port-map [*protocol-name*]

This command is used to display the current protocol-to-port mappings in use by NBAR. When the **ip nbar port-map** command has been used, the **show ip nbar port-map** command displays the ports assigned by the user to the protocol. If no **ip nbar port-map** command has been used, the **show ip nbar port-map** command displays the default ports. The *protocol-name* argument can also be used to limit the display to a specific protocol.

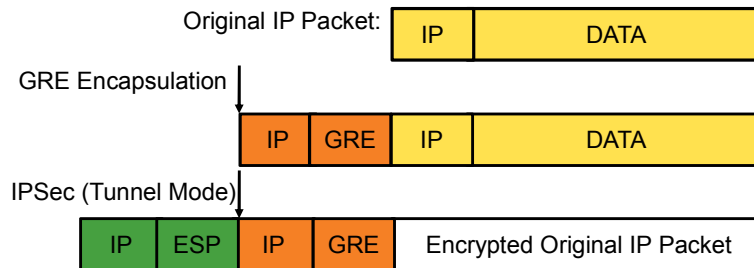
QoS Tunneling Techniques

This topic describes issues when implementing QoS with VPN and tunneling.

QoS and VPNs

- QoS features are unable to examine original IP headers when packets are encapsulated or encrypted
- Packets traveling across same tunnel have same IP headers
- Original (pre-tunnel) IP header may be encrypted

IP packet encapsulation with GRE and IPSec:

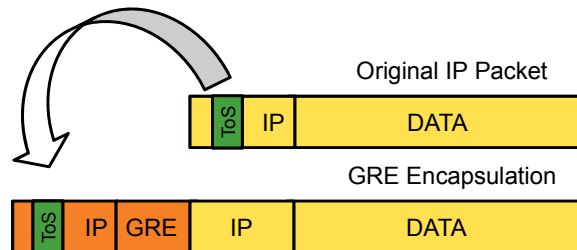


Attractive pricing is usually the driver behind deploying site-to-site IPSec VPNs as an alternative to private WAN technologies. Many of the same considerations required by private WANs need to be taken into account for IPSec VPN scenarios because they are usually deployed over the same Layer 2 WAN access media.

QoS classification is commonly based on the contents of packet headers. However, when an IP packet is encrypted, the IP header becomes unusable by QoS mechanisms that process the packet (post-encryption). Nevertheless, even if the packet is not encrypted but only encapsulated into a new header, QoS mechanisms only examine the last added IP header of the packet.

ToS Byte Preservation

- By default, ToS byte is copied to new header in any mode: AH, ESP, or GRE
- If packets are classified by ToS byte, no need for QoS preclassify
- Performed by tunneling mechanism



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4.14

For many QoS designs, classification is performed based on DSCP markings in the ToS byte of the IP packet header. As stated earlier, when an IP packet is encrypted, the IP header becomes unusable by QoS mechanisms that process the packet.

To overcome this predicament, the IPsec protocol standards have inherently provisioned the capability to preserve the ToS byte information of the original IP header, by copying it to the IP headers added by the tunneling and encryption process.

As shown in the figure, the original IP ToS byte values are copied initially to the IP header added by the GRE encapsulation. If another encapsulation such as IPsec is present, then these values are copied again to the IP header added by IPsec encryption.

QoS Preclassify

- Feature that allows packets to be classified before tunneling and encryption
- From the perspective of QoS preclassify, QoS policy may be applied on:
 - Physical interface
 - Tunnel interface
- Classification is performed based on pre-tunnel or post-tunnel header:

	QoS Preclassify Applied	QoS Preclassify Not Applied
QoS Policy on Physical interface	Pre-tunnel header classification	Post-tunnel header classification
QoS Policy on Tunnel interface	Pre-tunnel header classification (only for that tunnel)	Pre-tunnel header classification (only for that tunnel)

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4-15

QoS preclassify is a Cisco IOS and IOS XE feature that allows for packets to be classified on header parameters other than ToS byte values after encryption.

Because all original packet header fields are encrypted, including source or destination IP addresses, Layer 4 protocol, and source or destination port addresses, post-encryption QoS mechanisms cannot perform classification against criteria specified within any of these fields.

A solution to this constraint is to create a clone of the headers of the original packet before encryption. The crypto engine encrypts the original packet, and then the clone is associated with the newly encrypted packet and sent to the output interface. At the output interface, any QoS decisions based on header criteria, except for ToS byte values—which have been preserved—can be performed by matching on any or all of the five access-list tuple values of the clone. In this manner, advanced classification can be administered even on encrypted packets.

Typical use of the **qos pre-classify** command is denoted in the following ways:

- IP precedence or DSCP markings are already present in the ToS byte and that is all that will be needed to classify the traffic (as opposed to using source and destination IP addresses, source and destination port numbers, etc.). In this case, there is no need to use **qos pre-classify** because the pre-tunnel IP header is automatically copied to the post-tunnel IPsec or GRE header.
- If you want to classify traffic based on something other than IP precedence or DSCP markings (such as source and destination IP address, protocol, port number, etc.), then you must either:
 - Apply the service policy to the tunnel interface without **qos pre-classify** in order to use the pre-tunnel header—in this case, QoS policy is applied only for that tunnel.
 - Apply the service policy to the physical interface without using the **qos pre-classify** command, in order to classify traffic on the post-tunnel header.
 - Apply the service policy to the physical interface with the **qos pre-classify** command, in order to use the pre-tunnel header.

Configuring QoS Pre-Classify

This topic explains how to configure QoS Pre-Classify.

Configuring QoS Preclassify

- QoS preclassify can be configured on:

- GRE and IP/IP tunnels

```
Router(config)# interface tunnel0
Router(config-if)# qos pre-classify
```

- L2F and L2TP tunnels

```
Router(config)# interface virtual-template1
Router(config-if)# qos pre-classify
```

- IPSec tunnels

```
Router(config)# crypto map map1
Router(config-crypto-map)# qos pre-classify
```

- QoS preclassify feature is available in Cisco IOS and IOS XE Software.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--4.10

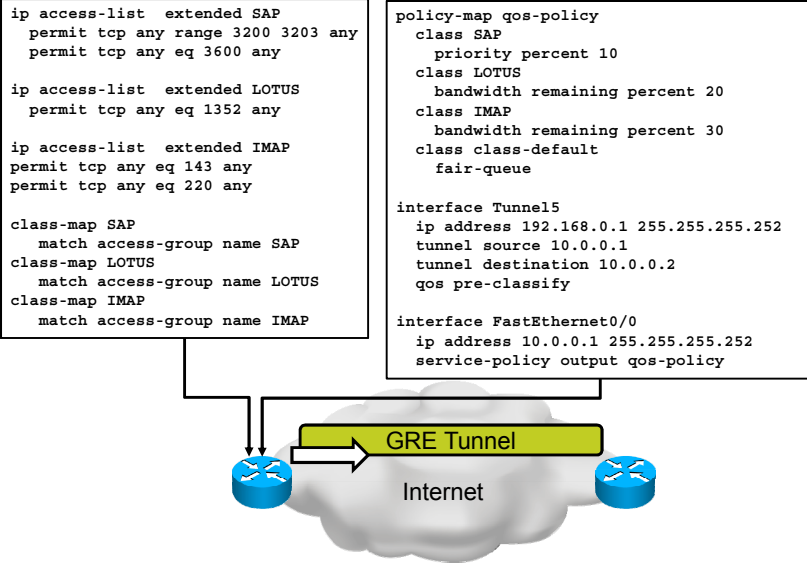
If QoS markings are applied before they enter the router, these markings will be automatically reflected into the GRE or IPSec header. Otherwise, if QoS markings are applied on the router itself, these markings will not be reflected into the GRE or IPSec header without the **qos pre-classify** command.

You can use the **qos pre-classify** Cisco IOS and IOS XE command to enable the QoS preclassification feature. Where you apply the command depends upon the type of VPN tunnel that you are using. For GRE tunnels, apply the command to a tunnel interface. For IPSec tunnels, apply the command to a crypto map. When configuring an IPSec encrypted IP GRE tunnel, apply the **qos pre-classify** command to both the tunnel interface and the crypto map. This command can be applied only to a tunnel interface, a crypto map, or a virtual template interface. Virtual template interfaces are used with Layer 2 Tunneling Protocol (L2TP) tunnels—when configuring L2TP tunnels, apply the command to a virtual-template interface.

QoS preclassify is supported for both GRE and IPSec, and is available for these platforms:

- Cisco 7100 Series VPN Routers and Cisco 7200 Series Routers (since Cisco IOS Software Release 12.1(5)T)
- Cisco 2600 and 3600 Series Routers (since Cisco IOS Software Release 12.2(2)T)
- Cisco ASR 1000 Series Routers (since Cisco IOS Software XE Release 2.1)

Configuring QoS Preclassify—Example



Assume a single site that is running multiple applications, in this case SAP (TCP ports 3200–3203 and also 3600), Lotus Notes (TCP port 1352), and IMAP (TCP ports 143 and 220).

Three extended access lists are configured to match those three classes of traffic: SAP, IMAP, and LOTUS. Those three access lists are used then to create three service classes that will be used in the policy “qos-policy”.

The QoS policy implements CBWFQ on three types of traffic, and is defined in such a way that it reserves 10 percent of interface bandwidth for SAP traffic, 20 percent of remaining interface bandwidth for LOTUS traffic, and 30 percent of remaining interface bandwidth for IMAP traffic.

GRE tunnel encapsulation is configured between two CE routers and FastEthernet 0/0 interface is configured as a source of the tunnel. The QoS policy is applied to the physical interface, and because you are using pre-tunnel packet header information other than the ToS byte for classification, the **qos pre-classify** command is necessary in this case.

QoS Preclassify Verification

To verify QoS preclassify, use one of two commands:

```
CE7#show interfaces tunnel 5
Tunnel5 is up, line protocol is up
Internet address is 192.168.95.81/30
Encapsulation TUNNEL, loopback not set
Tunnel source 192.168.107.71, destination
192.168.108.81
Tunnel protocol/transport GRE/IP
Input queue: 0/75/0/0 (size/max/drops/flushes);
Total output drops: 0
Queueing strategy: fifo (QoS pre-classification)
```

```
Router# show crypto map
Crypto Map "testtag" 10 ipsec-isakmp
Peer = 13.0.0.1
Extended IP access list 102
access-list 102 permit gre host 13.0.0.2 host 13.0.0.1
Current peer:13.0.0.1
Security association lifetime: 4608000 kilobytes/86400
seconds
PFS (Y/N): N
Transform sets={ proposal1,}
QoS pre-classification
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-4-18

To verify that the QoS for VPNs feature has been successfully enabled on an interface, use the **show interfaces** command. In the example, the line in the output that is marked red for emphasis verifies that the QoS for VPNs feature is successfully enabled.

To verify that the QoS for VPNs feature has been successfully enabled on a crypto map, use the **show crypto map** command. In the example, the line in the output that is marked red for emphasis verifies that the QoS for VPNs feature is successfully enabled.

show crypto map [*interface interface* | *tag map-name*]

Syntax Description

Parameter	Description
interface <i>interface</i>	(Optional) Displays only the crypto map set applied to the specified interface.
tag <i>map-name</i>	(Optional) Displays only the crypto map set with the specified <i>map-name</i> .

QoS Policy Propagation via BGP

This topic describes the QPPB classification mechanism.

QoS Policy Propagation via BGP

- Classification based on ACL not scalable
- QPPB allows marking of packets associated with BGP route
- Uses BGP attributes to associate marking information to IP networks
- QPPB can only mark and classify inbound packets

© 2012 Cisco and/or its affiliates. All rights reserved.

SPICORE v1.01-4-19

The QPPB feature allows packets to be classified based on access lists, BGP community lists, and BGP autonomous system (AS) paths. The supported classification policies include IP precedence setting and the ability to tag the packet with a QoS class identifier internal to the router. After a packet has been classified, you can use other QoS features to specify and enforce business policies to fit the business model.

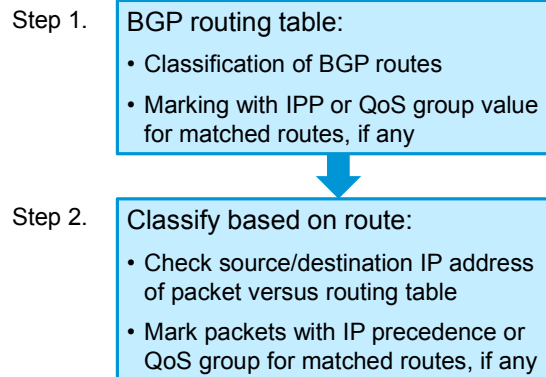
Commonly, classification of traffic would require an IP ACL for matching the packets, for all packets. For an ISP with many customers, however, classifying and marking packets based on referencing ACLs for a large number of packets may induce too much overhead traffic. Suppose that ISP 1 agrees to support the premium and best-effort customers of ISP 2, and ISP 2 agrees to support ISP 1 customers in a similar manner. The two ISPs would have to continually exchange information about which networks are premium and which are not, if they are using IP ACLs to classify the traffic. Additionally, when new customers are added, ISP 1 may be waiting on ISP 2 to update its QoS configuration before the desired level of service is offered to the new customer.

QPPB was created to overcome the issue of scalability of classifying based on ACLs, and the administrative problems of just listing the networks that need premium services. QPPB allows marking of packets based on an IP precedence or QoS group value associated with a BGP route. For instance, the BGP route for the Customer 1 network, Network A, could be given a BGP path attribute that both ISP 1 and ISP 2 agree should mean that this network receives better QoS service. Because BGP already advertises the routes, and the QoS policy is based on the networks described in the routes, QPPB marking can be done more efficiently than with the other classification and marking tools.

QPPB follows two steps: marking routes, and then marking packets based on the values marked on the routing entries. BGP routing information includes the network numbers used by the various customers, and other BGP path attributes. Because Cisco has worked hard over the years to streamline the process of table lookup in the routing table, to reduce per-packet processing for the forwarding process, QPPB can use this same efficient table lookup process to reduce classification and marking overhead.

QPPB Process

QPPB follows two steps:



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-20

There are two important points in the QPPB process:

- QPPB classifies BGP routes based on the attributes of the BGP routes, and marks BGP routes with an IP precedence or QoS group value.
- QPPB classifies packets based on the associated routing table entries, and marks the packets based on the marked values in the routing table entry.

QPPB allows routers to mark packets based on information contained in the routing table. Before packets can be marked, QPPB first must somehow associate a particular marked value with a particular route. QPPB, as the name implies, accomplishes this task using BGP. This first step can almost be considered as a separate classification and marking step by itself, because BGP routes are classified based on information that describes the route, and marked with some QoS value. The classification feature of QPPB can examine many of the BGP path attributes.

The two most useful BGP attributes for QPPB are the autonomous system number (AS number) sequence, referred to as the autonomous system path, and the community string. The autonomous system path contains the ordered list of AS numbers, representing the AS numbers between a router and the autonomous system of the network described in the route.

After QPPB has marked routes with IP precedence or QoS group values, the packet marking part must be performed. After the packets have been marked, traditional QoS tools can be used to perform queuing, congestion avoidance, policing, and so on, based on the marked value.

QPPB packet-marking logic flows as follows:

- Step 1** Process packets entering an interface.
- Step 2** Match the destination or source IP address of the packet to the routing table.
- Step 3** Mark the packet with the precedence or QoS group value shown in the routing table entry.

The three-step logic for QPPB packet marking follows the same general flow as the other classification and marking tools.

QoS and BGP Interaction

- QoS feature works independently of BGP routing
- BGP is used to propagate policies
- QoS feature works based on markings

© 2012 Cisco and/or its affiliates. All rights reserved.

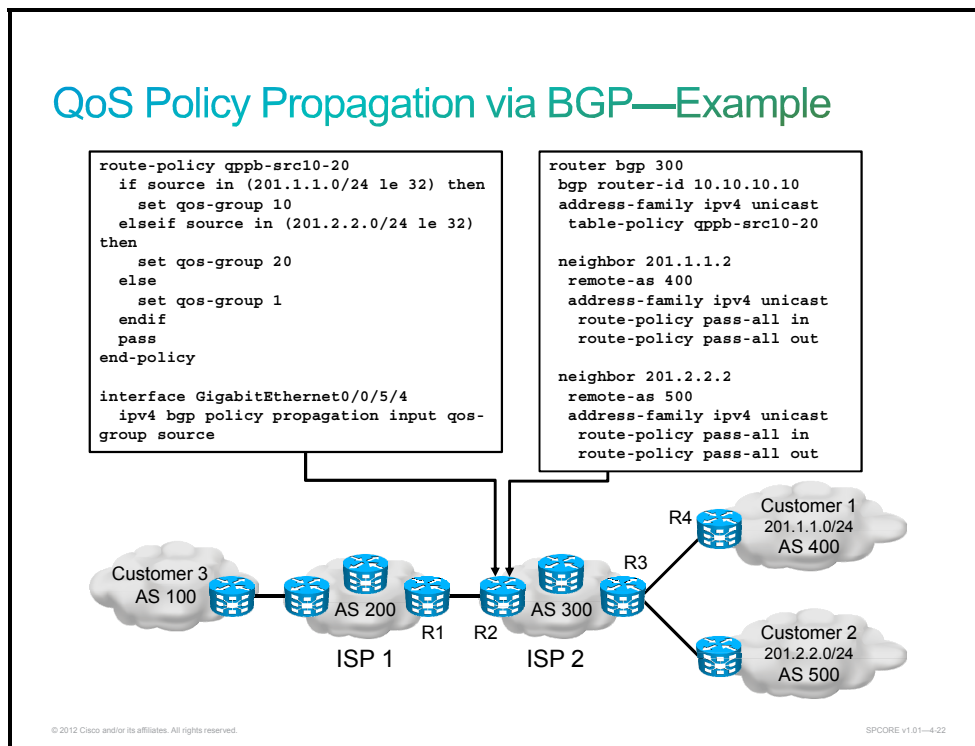
SPCORE v1.01-4-21

When using QPPB, the QoS feature works independently from BGP routing. BGP is only used to propagate the QoS policy.

In QPPB configurations, you specify whether to use IP precedence or the QoS group ID obtained from the source (input) address or destination (output) address entry in the routing table. You can specify either the input or output address.

Configuring QPPB

This topic explains how to configure QPPB.



QPPB allows for the marking of packets that have been sent to Customer 1, and for marking packets that have been sent by Customer 1.

For packets that Customer 1 has sent, going from right to left in the figure, QPPB on R2 can still mark the packets. These packets typically enter the ingress interface of R2, however, and the packets have source IP addresses in the network of Customer 1. To associate these packets with Network 1, QPPB examines the routing table entry that matches the source IP address of the packet. This match of the routing table is not used for packet forwarding—it is used only for finding the precedence or the QoS group value to set on the packet. In fact, the table lookup for destination addresses does not replace the normal table lookup for forwarding the packet, either. Because the routing table entry for network of Customer 1 has the QoS group set to 10, QPPB marks these packets with QoS group 10. In the same way, packets for the Customer 2 network have the QoS group set to 20.

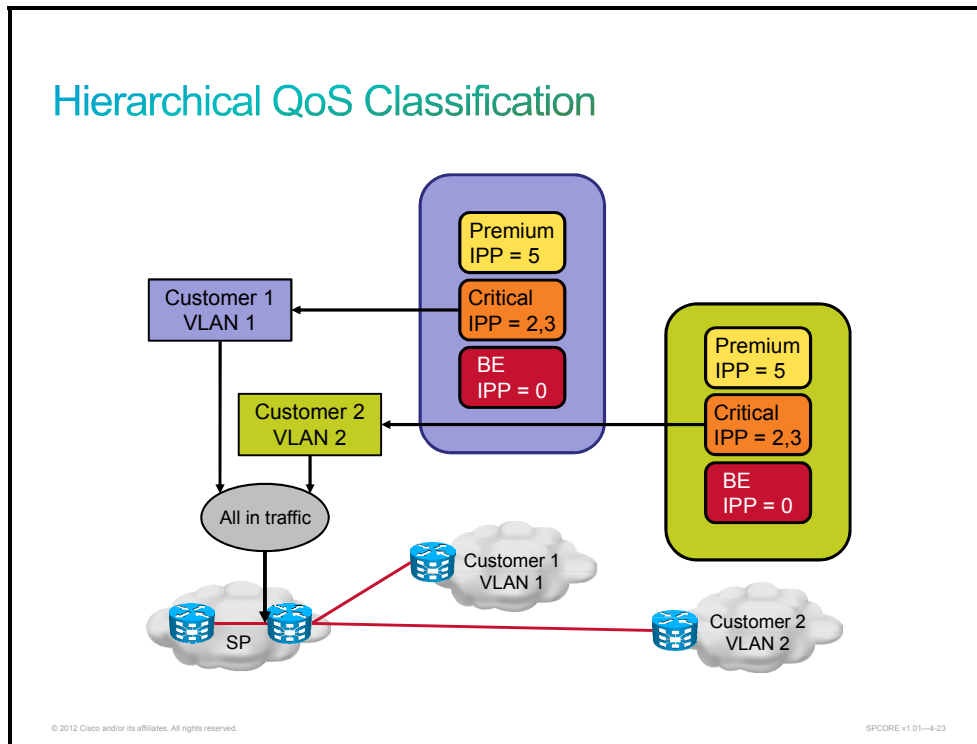
Syntax Description (IOS XR Software)

Parameter	Description
<code>route-policy name</code>	Enters route policy configuration mode and specifies the name of the route policy to be configured.
<code>set qos-group qos-group-value</code> <code>[discard-class discard-class-value]</code>	Sets the QoS group identifiers on IPv4 or MPLS packets. The set qos-group command is supported only on an ingress policy. Note The discard-class <i>discard-class-value</i> keyword and argument are only supported on the Cisco CRS-1 router.
<code>route-policy route-policy-name {in out}</code>	(Optional) Applies the specified policy to inbound IPv4 unicast routes.

Parameter	Description
<pre> ipv4 bgp policy propagation {input} {qos-group ip- precedence} {destination source} </pre>	Enables QPPB on an interface: input – Enables QPPB on the ingress IPv4 unicast interface. ip-precedence – Specifies that the QoS policy is based on the IP precedence. qos-group – Specifies that the QoS policy is based on the QoS group ID. destination – Specifies that the IP precedence bit or QoS group ID from the destination address entry is used in the route table. source – Specifies that the IP precedence bit or QoS group ID from the source address entry is used in the route table.

Hierarchical QoS

This topic describes a QoS implementation example using hierarchical QoS.



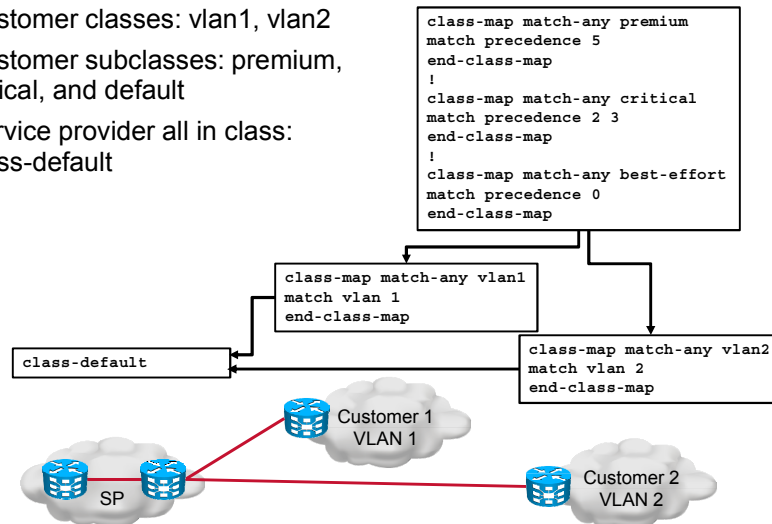
In next-generation network (NGN) service provider networks, various demands from customers are becoming more difficult to manage. Serving those requests must adhere to agreed-upon service-level agreements (SLAs) and deliver predictable levels of guaranteed bandwidth, delay, and packet loss for critical applications. Also, the service provider needs to differentiate between different classes of customers. Some customers have paid for a premium service level with demanding QoS parameters that must be met, and others only need basic service without special QoS requirements.

These different customers and their various applications use a common shared service provider network infrastructure. In this environment, service providers must have a way to ensure per-application and per-customer policies on the network. For example, if all premium traffic, such as voice, is set into one premium class, there is no way to differentiate which voice packets are passed from one customer to another. All voice traffic is considered as one class.

Hierarchical QoS solves this problem through multiple levels of classification and scheduling through QoS classes and policies. In this example, the service provider receives traffic from two customers over two VLANs. Traffic from each customer consists of different types of traffic (premium, critical, and best-effort). The service provider also needs to consider the capacity of the link between core routers, and set QoS policies accordingly for all consolidated traffic coming from all customers.

Hierarchical QoS Classification (Cont.)

- Customer classes: vlan1, vlan2
- Customer subclasses: premium, critical, and default
- Service provider all in class: class-default



In this example, hierarchical classification is performed. Premium, critical, and best-effort traffic is classified based on the IP precedence value in IP packets. Traffic from customers is classified based on the VLAN number, and consolidated traffic is classified by default by the class-default class. Classification configuration of these classes is shown in the example:

```
class-map match-any premium
match precedence 5
end-class-map
!
class-map match-any critical
match precedence 2 3
end-class-map
!
class-map match-any best-effort
match precedence 0
end-class-map
!
class-map match-any vlan1
match vlan 1
end-class-map
!
class-map match-any vlan2
match vlan 2
end-class-map
```

Hierarchical QoS Policies

Step 1.

```

policy-map child_policy
!
class premium
bandwidth percent 40
!
class critical
bandwidth percent 10
random-detect precedence 2 10 ms 100 ms
random-detect precedence 3 20 ms 200 ms
queue-limit 200 ms
!
class best-effort
bandwidth percent 20
queue-limit 200 ms
!
class class-default
!
end-policy-map

```

Step 2.

```

policy-map parent
class vlan1
service-policy child_policy
shape average percent 40
!
class vlan2
service-policy child_policy
shape average percent 40
!
end-policy-map

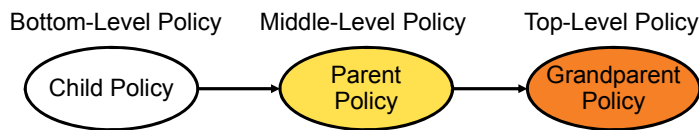
```

Step 3.

```

policy-map grand-parent
class class-default
shape average 500 Mbps
service-policy parent
!
end-policy-map
!
interface GigabitEthernet0/0/0/9
service-policy output grand-parent

```



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-25

In this example, the grandparent policy is applied to the main Gigabit Ethernet interface. The grandparent policy limits all outbound traffic of the interface up to 500 Mb/s. The parent policy has class vlan1 and vlan2, and traffic in vlan1 or vlan2 is limited to 40 percent of 500 Mb/s. The child policy classifies traffic based on different services and allocates bandwidth for each class accordingly. This configuration is shown here:

```

policy-map grand-parent
class class-default
shape average 500 Mbps
service-policy parent
!
end-policy-map
!
policy-map parent
class vlan1
service-policy child_policy
shape average percent 40
!
class vlan2
service-policy child_policy
shape average percent 40
!
end-policy-map
!
policy-map child_policy
class premium
bandwidth percent 40
!
class critical

```

```
bandwidth percent 10
random-detect precedence 2 10 ms 100 ms
random-detect precedence 3 20 ms 200 ms
queue-limit 200 ms
!
class best-effort
bandwidth percent 20
queue-limit 200 ms
!
class class-default
!
end-policy-map
!
interface GigabitEthernet0/0/0/9
service-policy output grand-parent
```

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- NBAR is commonly used for classification and traffic statistics and identifies packets based on Layer 4 to 7 packet inspection
- Use the **show ip nbar protocol-discovery** command to display statistics gathered by the NBAR Protocol Discovery feature
- Encapsulated or encrypted packet headers are unreadable by QoS mechanisms. QoS preclassify allows packets to be classified based on information in headers other than ToS
- If QoS markings are applied on the router itself, these markings will not be reflected into the GRE or IPSec header without the **qos pre-classify** command
- The QPPB feature allows classifying packets based on ACL, BGP community lists, and BGP AS paths
- When using QPPB, the QoS feature works independently from BGP routing
- Hierarchical QoS enables per-subscriber and per-traffic class QoS classification and policies

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4-28

Module Summary

This topic summarizes the key points that were discussed in this module.

Module Summary

- Classifying packets into different classes is called classification, and marking packets is important to easily distinguish marked packets.
- The most common classification and marking options at the data link layer include CoS in the ISL or 802.1Q header and MPLS EXP bits. At the network layer, packets are typically classified based on source or destination IP or the ToS byte.
- When packets are encapsulated or encrypted, QoS mechanisms are unable to examine the original packet header. The QoS preclassify feature allows you to override this problem.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—4.1

Several Layer 2 classification and marking options exist depending on the technology, encapsulation, and transport protocol used. The most common classification and marking options at the data link layer include CoS in ISL or 802.1Q header, and Multiprotocol Label Switching (MPLS) experimental (EXP) bits.

At the network layer, IP packets are typically classified based on source or destination IP address, or the contents of the Type of Service (ToS) byte.

Quality of service (QoS) classification mechanisms are used to separate traffic and identify packets as belonging to a specific service class. The service class is the fundamental building block for separating traffic into different classes. After the packets are identified as belonging to a specific service class, QoS mechanisms such as policing, shaping, and queuing techniques can be applied to each service class to meet the specifications of the administrative policy.

Cisco IOS, IOS XE, and IOS XR Modular QoS CLI (MQC) classification with class maps is extremely flexible and can classify packets by using classification tools based on the following:

- Source and destination parameters
- Internal markings
- Packet markings

If packet header fields are encrypted—including source or destination IP addresses, Layer 4 protocol, and source or destination port addresses—the postencryption QoS mechanisms cannot perform classification against criteria specified within any of these fields. A solution to this constraint is the QoS preclassify feature, which creates a clone of the original packet headers before encryption and then uses the values in the clone to make QoS decisions at the output interface. Cisco QoS preclassify is a feature of Cisco IOS and IOS XE Software and is not supported in Cisco IOS XR Software.

Module Self-Check

Use the questions here to review what you learned in this module. The correct answers and solutions are found in the Module Self-Check Answer Key.

- Q1) What is the Cisco QoS baseline marking recommendation for call signaling traffic? (Source: Understanding Classification and Marking)
- A) AF31
 - B) EF
 - C) CS3
 - D) BE
- Q2) In which location should the administrator enforce the trust boundary? (Source: Understanding Classification and Marking)
- A) at the core of the network
 - B) as close as possible to the source of traffic flow
 - C) as close as possible to the destination of traffic flow
 - D) always at endpoint devices
- Q3) Which two options are internal markings? (Choose two.) (Source: Using Modular QoS CLI)
- A) DE bit
 - B) QoS group
 - C) source MAC address
 - D) Discard Class
- Q4) CoS markings are contained in which two of the following headers? (Choose two.) (Source: Using Modular QoS CLI)
- A) IP header
 - B) Frame Relay header
 - C) ISL header
 - D) 802.1Q header
 - E) MPLS header
- Q5) Which Cisco IOS command is used for traffic classification using NBAR? (Source: Implementing Advanced QoS Techniques)
- A) **match access-group name** *name*
 - B) **ip nbar protocol-discovery**
 - C) **ip nbar pldm** *pldm_file*
 - D) **match protocol** *protocol_name*
- Q6) In which two locations can the **qos pre-classify** command be applied for IPSec/GRE tunnels? (Choose two.) (Source: Implementing Advanced QoS Techniques)
- A) class map
 - B) crypto map
 - C) tunnel interface
 - D) physical interface

- Q7) Which statement about the ToS byte is true when IPSec or GRE tunnels are used? (Source: Implementing Advanced QoS Techniques)
- A) To copy the ToS byte from original to tunneled packet header, the QoS preclassify feature must be used.
 - B) The ToS byte is automatically copied from the original to the tunneled packet header by the tunneling mechanism.
 - C) The ToS byte is automatically copied from the tunnel to the original IP packet header only for incoming packets.
 - D) None of the above is true.
- Q8) Which Cisco IOS interface mode command enables bidirectional, per-interface protocol statistics? (Source: Implementing Advanced QoS Techniques)
-
- Q9) What has to be configured as a prerequisite before you configure NBAR to recognize HTTP requests? (Source: Implementing Advanced QoS Techniques)
- A) routing protocol
 - B) MPLS
 - C) IP Cisco Express Forwarding
 - D) IP HTTP server
- Q10) Which two markers does the QPPB feature support? (Choose two.) (Source: Implementing Advanced QoS Techniques)
- A) IP precedence
 - B) CoS
 - C) MPLS EXP
 - D) QoS group
- Q11) Which option can be used to specify QoS behavior at multiple policy levels? (Source: Implementing Advanced QoS Techniques)
- A) route maps or RPL
 - B) AutoQoS
 - C) hierarchical QoS
 - D) QoS CLI

Module Self-Check Answer Key

- Q1) C
- Q2) B
- Q3) B, D
- Q4) C, D
- Q5) D
- Q6) B, C
- Q7) B
- Q8) **ip nbar protocol-discovery**
- Q9) C
- Q10) A, D
- Q11) C

QoS Congestion Management and Avoidance

Overview

Congestion can occur in many different locations within a network and is the result of many factors, including oversubscription, insufficient packet buffers, traffic aggregation points, network transit points, and speed mismatches. Simply increasing link bandwidth is not adequate to solve the congestion issue, in most cases. Aggressive traffic can fill interface queues and starve more fragile flows such as voice and interactive traffic. The results can be devastating for delay-sensitive traffic types, making it difficult to meet the service-level requirements these applications require. Fortunately, there are many congestion management techniques available on Cisco platforms, which provide you with an effective means to manage software queues and to allocate the required bandwidth to specific applications when congestion exists.

When congestion occurs, some traffic is delayed or even dropped at the expense of other traffic. When drops occur, different problems may arise which can exacerbate the congestion, such as retransmissions and TCP global synchronization in TCP/IP networks. Network administrators can use congestion avoidance mechanisms to reduce the negative effects of congestion by penalizing the most aggressive traffic streams as software queues begin to fill.

This module examines the components of queuing systems and the different congestion management mechanisms available on Cisco routers. It further describes the problems with TCP congestion management and the benefits of deploying congestion avoidance mechanisms.

Module Objectives

Upon completing this module, you will be able to describe different Cisco QoS queuing mechanisms used to manage network congestion, as well as random early detection (RED) used to avoid congestion. This ability includes being able to meet these objectives:

- Define the operation of basic queuing algorithms
- Explain the problems that may result from the limitations of TCP congestion management mechanisms

Managing Congestion

Overview

Queuing algorithms are one of the primary ways to manage congestion in a network. Network devices handle an overflow of arriving traffic by using a queuing algorithm to sort traffic and determine a method of prioritizing the traffic onto an output link. Each queuing algorithm was designed to solve a specific network traffic problem and has a particular effect on network performance.

Class-based weighted fair queuing (CBWFQ) provides support for user-defined traffic classes. With CBWFQ, you define traffic classes based on match criteria. Packets satisfying the match criteria for a class constitute the traffic for that class. A queue is reserved for each class, and traffic belonging to a class is directed to the queue for that class. Low-latency queuing (LLQ) brings strict priority queuing to CBWFQ. Strict priority queuing allows delay-sensitive data such as voice to be dequeued and sent first, giving delay-sensitive data preferential treatment over other traffic.

This lesson describes several queuing algorithms and explains how to configure CBWFQ and LLQ.

Objectives

Upon completing this lesson, you will be able to define the operation of basic queuing algorithms. This ability includes being able to meet these objectives:

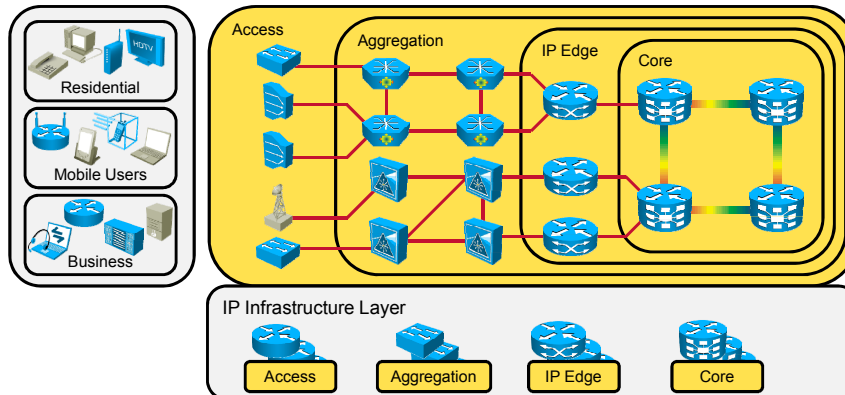
- Describe the need for congestion management queuing mechanisms
- Describe the FIFO queuing algorithm
- Describe the Priority queuing algorithm
- Describe the Round Robin queuing algorithm
- Describe the Weighted Round Robin queuing algorithm
- Describe the Deficit Round Robin queuing algorithm
- Describe the Modified Deficit Round Robin queuing algorithm
- Describe the different Cisco IOS and Cisco IOS XR Queue types
- Illustrate the high-level architecture of Cisco IOS XR routers
- Explain how to configure class-based weighted fair queuing
- Explain how to configure low latency queuing

Queuing Introduction

This topic describes the need for congestion management queuing mechanisms.

Congestion and Queuing in IP NGN

- Congestion can occur at any point in the network where there are points of speed mismatches or aggregation.
- Queuing manages congestion to provide bandwidth and delay guarantees.



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6.3

Congestion can occur in any layer of the IP Next-Generation Network (NGN) environment, where there are points of speed mismatches (for example, a Gigabit Ethernet link feeding a Fast Ethernet link), aggregation (for example, multiple Gigabit Ethernet links feeding an upstream Gigabit Ethernet), or confluence (the flowing together of two or more traffic streams).

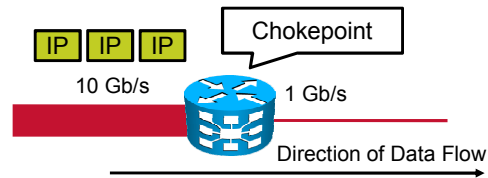
Congestion has undesired results for network performance because it causes tail drops. Tail drops occur when traffic cannot be enqueued, because the queue buffers are full.

Queuing algorithms are used to manage congestion. Many algorithms have been designed to serve different needs. A well-designed queuing algorithm will provide some bandwidth and delay guarantees to priority traffic.

Speed Mismatch and Aggregation

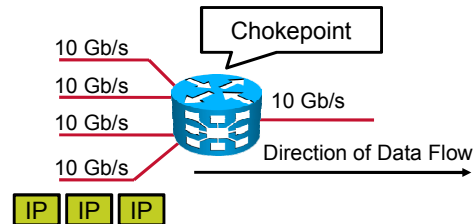
- Speed mismatch

- LAN to WAN
- LAN to LAN



- Aggregation

- More input than output links

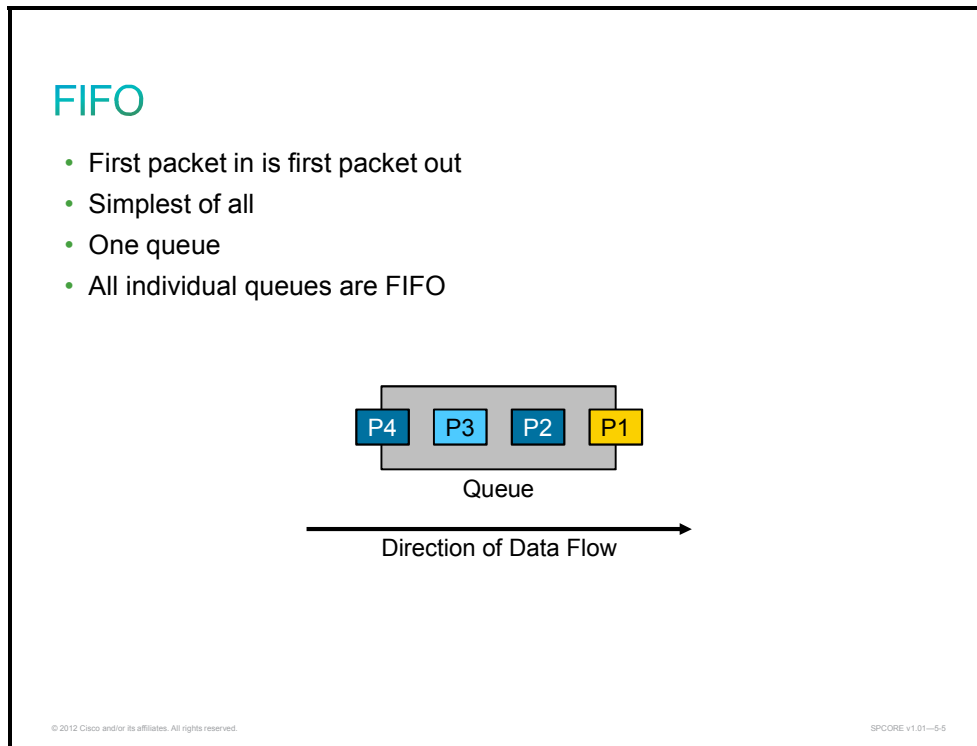


Speed mismatch is a common cause of congestion in a network. Speed mismatches can occur when traffic moves from a high-speed LAN environment (1000 Mb/s or higher) to lower-speed WAN links or in a LAN-to-LAN environment when, for example, a 10 Gb/s link feeds into a 1-Gb/s link.

Other typical places of congestion are aggregation points. In a LAN environment, congestion resulting from aggregation often occurs at the distribution layer of networks, where the different access layer devices feed traffic to the distribution-level devices.

FIFO Queuing

This topic describes the FIFO queuing algorithm.

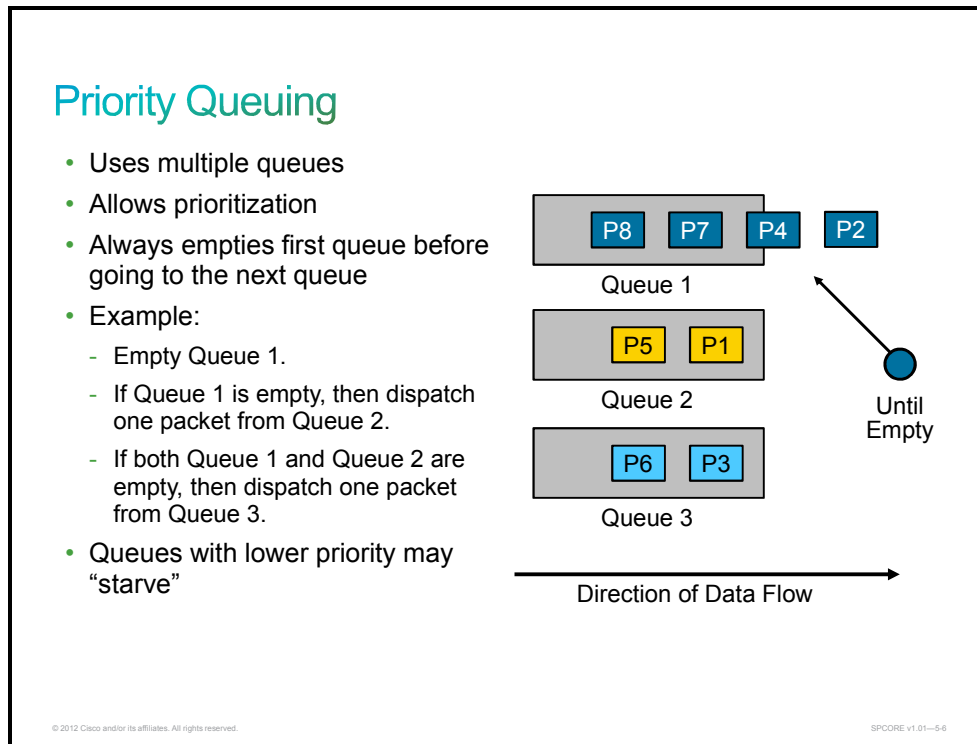


FIFO is the simplest queuing algorithm. Packets are placed into a single queue and serviced in the order they were received.

All individual queues are, in fact, FIFO queues. Other queuing methods rely upon FIFO as the congestion management mechanism for single queues, while using multiple queues to perform more advanced functions such as prioritization.

Priority Queuing

This topic describes the Priority queuing algorithm.



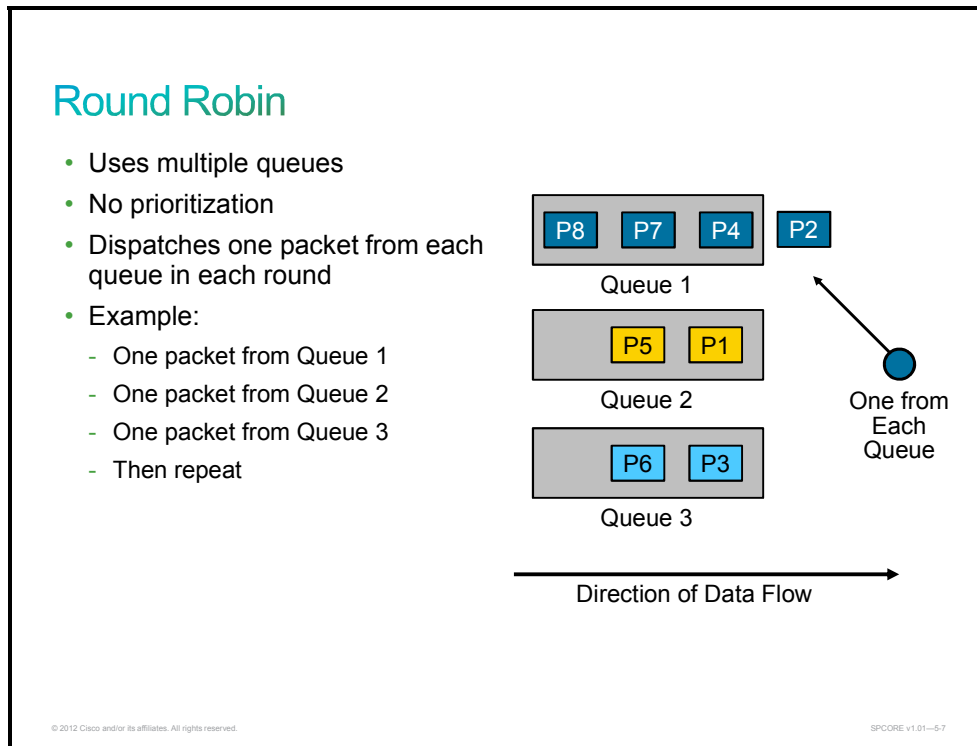
In priority queuing (PQ), each packet is assigned a priority and placed into a hierarchy of queues based on priority. When there are no more packets in the highest queue, the next lower queue is serviced. Packets are then dispatched from the next highest queue until either the queue is empty or another packet arrives for a higher-priority queue.

Packets will be dispatched from a lower queue only when all higher-priority queues are empty. If a packet arrives for a higher queue, the packet from the higher queue is dispatched before any packets in lower-level queues.

The problem with PQ is that queues with lower priority can “starve” if a steady stream of packets continues to arrive for a queue with a higher priority. Packets waiting in the lower-priority queues may never be dispatched.

Round Robin Queuing

This topic describes the Round Robin queuing algorithm.



With round-robin queuing, one packet is taken from each queue and then the process repeats.

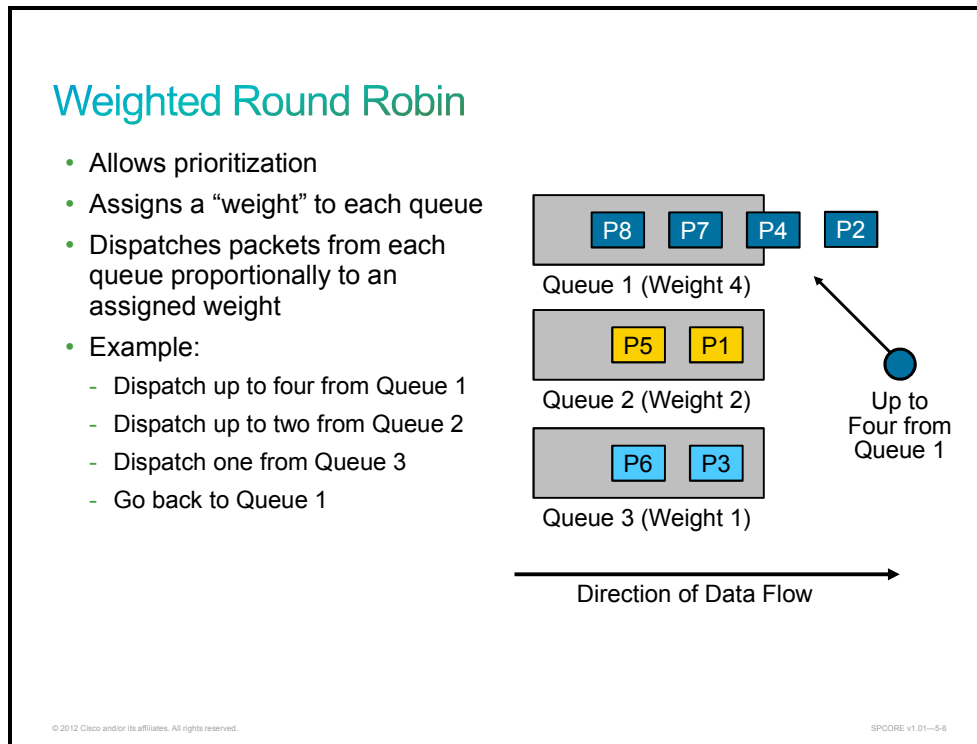
If all packets are the same size, all queues share the bandwidth equally. If packets being put into one queue are larger, that queue will receive a larger share of bandwidth.

No queue will starve with round robin because all queues receive an opportunity to dispatch a packet every round.

A limitation of round robin is the inability to prioritize traffic.

Weighted Round Robin Queuing

This topic describes the Weighted Round Robin queuing algorithm.



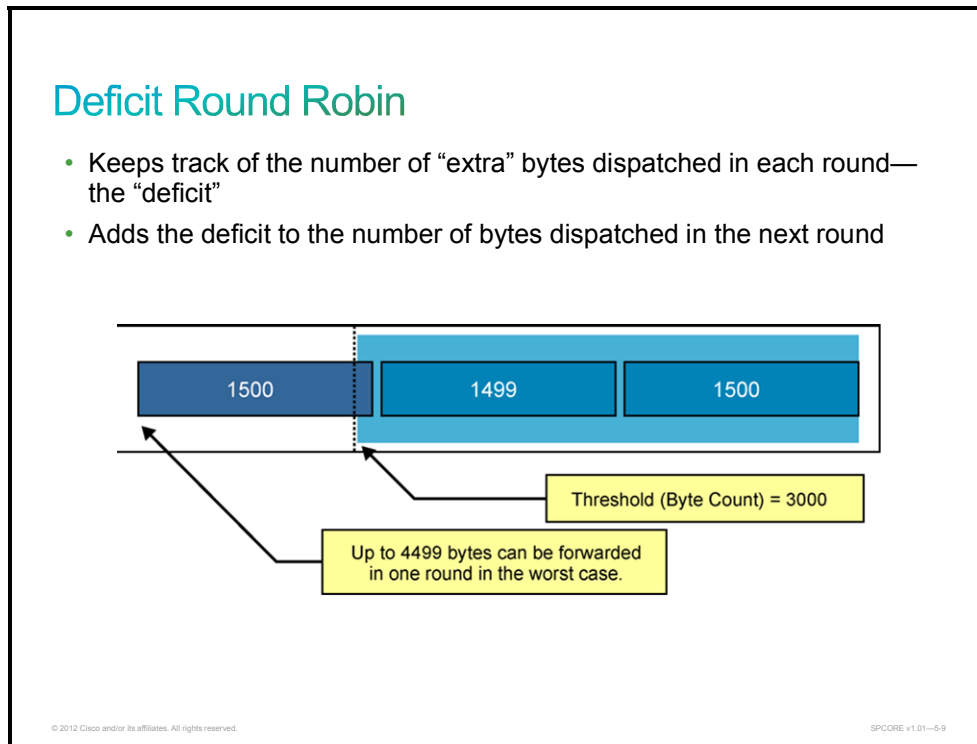
The weighted round robin (WRR) algorithm was developed to provide prioritization capabilities for round robin.

In WRR, packets are assigned a class (mission-critical, file transfer, and so on) and placed into the queue for that class of service. Packets are accessed round-robin style, but queues can be given priorities called “weights.” For example, in a single round, four packets from a high-priority class might be dispatched, followed by two from a middle-priority class, and then one from a low-priority class.

Some implementations of the WRR algorithm will dispatch a configurable number of bytes during each round.

Deficit Round Robin Queuing

This topic describes the Deficit Round Robin queuing algorithm.



The figure illustrates a drawback of WRR queuing. In this example, WRR is enabled on an interface with a maximum transmission unit (MTU) size of 1500 bytes. The byte count to be sent for the queue in each round is 3000 bytes (twice the MTU). The example shows how the router first sent two packets with a total size of 2999 bytes. Because this is still within the limit (3000), the router can send the next packet (MTU-sized). The result was that the queue received almost 50 percent more bandwidth in this round than it should have received. Clearly, the WRR algorithm does not allocate bandwidth accurately.

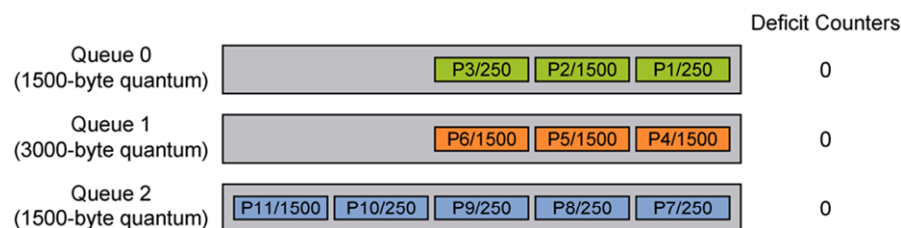
Deficit round robin (DRR) is an implementation of the WRR algorithm that was developed to resolve the inaccurate bandwidth allocation problem with WRR. Deficit round robin uses a deficit counter to track the number of “extra” bytes dispatched over the number of bytes that was to be configured to be dispatched each round. During the next round, the number of “extra” bytes—the deficit—is effectively subtracted from the configurable number of bytes that are dispatched.

Modified Deficit Round Robin Queuing

This topic describes the Modified Deficit Round Robin queuing algorithm.

Modified Deficit Round Robin

- Extends regular DRR by a low-latency queue serviced in:
 - Strict priority mode: Low-latency queue is serviced whenever it is not empty.
 - Alternate mode: MDRR alternately services the low-latency queue and any other configured queues.
- Each queue within MDRR is defined by:
 - Quantum value: Average number of bytes served in each round.
 - Deficit counter: Number of bytes a queue has transmitted in each round.



Modified deficit round robin (MDRR) is a class-based composite scheduling mechanism that allows for queueing of up to eight traffic classes. It operates in the same manner as CBWFQ, and allows definition of traffic classes based on customer match criteria (such as access lists). However, MDRR does not use the weighted fair queueing algorithm.

With MDRR configured in the queueing strategy, nonempty queues are served one after the other, in a round-robin fashion. Each time a queue is served, a fixed amount of data is dequeued. The algorithm then services the next queue. When a queue is served, MDDR keeps track of the number of bytes of data that were dequeued in excess of the configured value. In the next pass, when the queue is served again, less data is dequeued to compensate for the excess data that was served previously. As a result, the average amount of data dequeued per queue is close to the configured value.

Each queue within MDRR is defined by these two variables:

- **Quantum value:** Average number of bytes served in each round.
- **Deficit counter:** Number of bytes a queue has transmitted in each round. The counter is initialized to the quantum value.

Packets in a queue are served as long as the deficit counter is greater than zero. Each packet served decreases the deficit counter by a value equal to its length in bytes. A queue can no longer be served after the deficit counter becomes zero or negative. In each new round, the deficit counter for each nonempty queue is incremented by its quantum value. In general, the quantum size for a queue should not be smaller than the MTU of the interface to ensure that the scheduler always serves at least one packet from each nonempty queue.

Each MDRR queue can be given a relative weight, with one of the queues in the group defined as a priority queue. The weights assign relative bandwidth for each queue when the interface is congested. The MDRR algorithm dequeues data from each queue in a round-robin fashion if

there is data in the queue to be sent. During each cycle, a queue can dequeue a quantum based on its configured weight.

MDRR differs from regular DRR by adding a special low-latency queue that can be serviced in one of two modes:

- **Strict priority mode:** The low-latency queue is serviced whenever it is not empty. This provides the lowest delay possible for delay-sensitive traffic. The scheduler services only the current non-priority packet and then switches to the low-latency queue. The scheduler starts to service a non-priority queue only after the low-latency queue becomes completely empty. This mode can starve other queues, particularly if the matching flows are aggressive senders.
- **Alternate mode:** The MDRR scheduler alternatively services the low-latency queue and any other configured queues. Alternate mode can exercise less control over jitter and delay. If the MDRR scheduler starts to service frames from a data queue and then a voice packet arrives in the low-latency queue, the scheduler completely serves the non-priority queue until its deficit counter reaches zero. During this time, the low-latency queue is not serviced, and the packets are delayed. It is important to note that the priority queue in alternate priority mode is serviced more than once in a cycle, and thus takes more bandwidth than other queues with the same nominal weight. How much more is a function of how many queues are defined. For example, with three queues, the low latency queue is serviced twice as often as the other queues, and it sends twice its weight per cycle.

The figure shows three queues, each of which contains some packets that have been received and queued. For example, Queue 0 contains three packets: P1 (a 250-byte packet), P2 (a 1500-byte packet), and P3 (another 250-byte packet). Queue 0 is the low latency queue, and it is configured to operate in alternate mode. Each queue is assigned a quantum, as follows:

- Queue 0 has a quantum of 1500 bytes.
- Queue 1 has a quantum of 3000 bytes.
- Queue 2 has a quantum of 1500 bytes.

Cisco IOS and IOS XR Queue Types

This topic describes the different Cisco IOS and Cisco IOS XR Queue types.

Queue Types

- Hardware (Cisco IOS Software)
 - FIFO queuing
 - Configurable length
 - No reordering
- Software (Cisco IOS Software)
 - Congestion management for the hardware queues
 - Configurable scheduling method
 - Support for egress interfaces only
- Distributed (Cisco IOS XR Software)
 - ASIC-based
 - Ingress, fabric, and egress
 - Dynamic queue thresholds

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-11

Queuing on routers is necessary to accommodate bursts when the arrival rate of packets is greater than the departure rate, usually because of one of these two reasons:

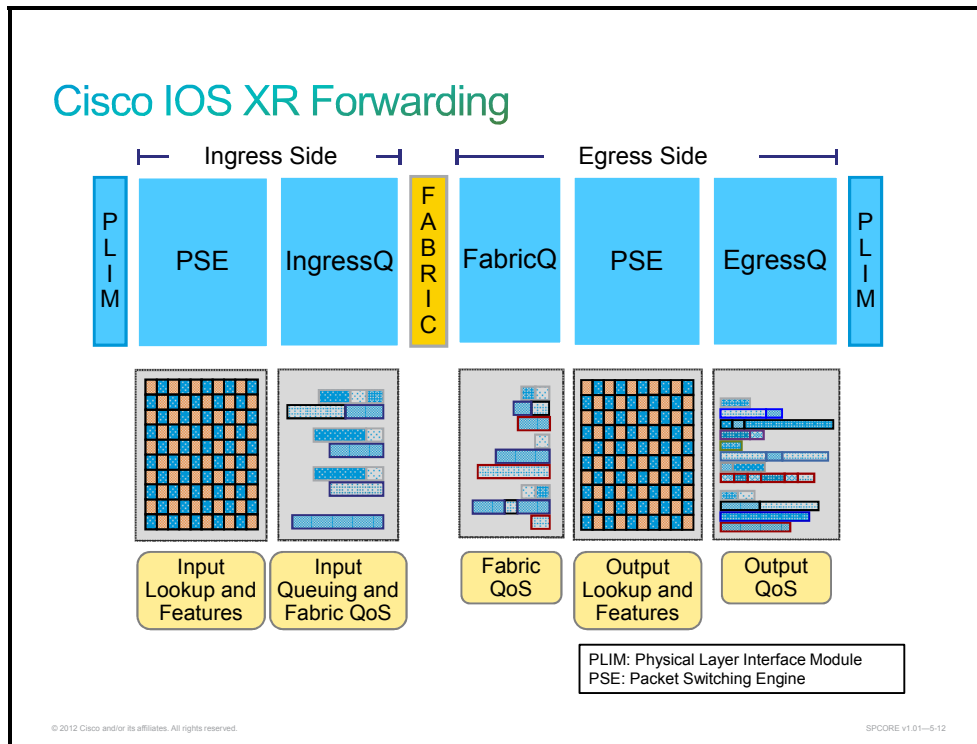
- The input interface is faster than the output interface.
- The output interface is receiving packets coming in from multiple other interfaces.

Queuing is implemented using various methods:

- **Hardware queue:** Uses the FIFO strategy, which is necessary for the interface drivers to transmit packets one by one. Depending on the platform, the hardware may have a configurable length. The packets in the hardware queue cannot be reordered. If the hardware queue is too long, it will contain a large number of packets scheduled in the FIFO fashion. A long FIFO hardware queue defeats the purpose of the QoS design, requiring a certain complex software queuing system.
- **Software queue:** Schedules packets into the hardware queue based on QoS requirements. Software queuing is implemented when the interface is congested, and the software queuing system is bypassed whenever there is room in the hardware queue. The software queue is, therefore, used only when data must wait to be placed into the hardware queue.
- **Distributed queuing:** Available on Cisco IOS XR Software. Distributed queuing extends the concepts of software and hardware queues by providing a distributed architecture consisting of interface modules, and router fabric. The queuing functions are supported using specialized ASICs. Queuing can be applied to ingress traffic on the input interface, to traffic traversing the fabric, and to egress traffic on the output interface. Each stage is configured separately. Cisco IOS XR uses the concept of dynamic queue thresholds to allocate the queuing space on demand.

Cisco IOS XR Forwarding Architecture

This topic illustrates the high-level architecture of Cisco IOS XR routers.

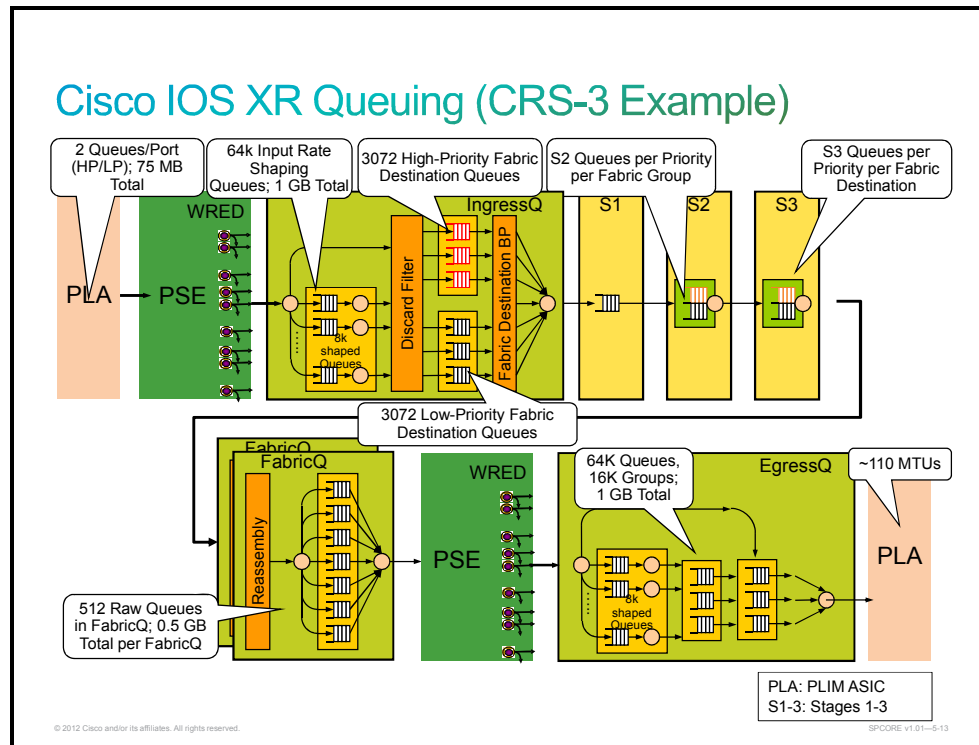


This figure illustrates the high-level architecture of Cisco IOS XR routers, such as Cisco Carrier Routing System 1 (CRS-1) and CRS-3. The three major building blocks are as follows:

- **Physical Layer Interface Modules (PLIMs):** Provide the interface circuitry
- **Packet Switching Engines (PSEs):** Responsible for packet lookup and packet processing
- **Fabric:** Provides the communication path between the line cards. It uses a three-stage, self-routed architecture, non-blocking switching, and fabric redundancy. Physically, the fabric is divided into eight planes over which the packets—broken into cells—are evenly distributed. Within the planes, the three fabric stages—S1, S2, and S3—dynamically route cells to their destination slots, where they are reassembled to form properly sequenced packets. The three stages of switching are:
 - Stage 1 (S1) is connected to the ingress line card, and delivers the cells across all stage 2 fabric cards.
 - Stage 2 (S2) supports multicast replication, and delivers the cells to the appropriate stage 3 fabric cards associated with the egress line card shelf.
 - Stage 3 (S3) is connected to the egress line card for delivery to the appropriate interface and subinterface.

Each of these three building blocks has its own queuing architecture. The PSE QoS features are separately configurable for ingress and egress direction.

Note This figure illustrates a generic version of the Cisco IOS XR forwarding architecture, although the hardware in the figure represents Cisco CRS-1 and CRS-3. The similar architecture is followed by the ASR 9000 Series Aggregation Services Routers. For example, the three-stage fabric architecture is not applicable to the ASR 9000 Series Aggregation Services Router.



This figure depicts the queuing architecture on the Cisco IOS XR platforms. Although the picture presents information pertaining to CRS-3, the concept applies to CRS-1. The components offer these queuing capabilities:

- The PLIM ASIC (PLA) embeds two queues per port. One queue is dedicated to high-priority traffic, the other to low-priority traffic. The total amount of PLA buffer space is 75 MB.
- The PSE has a total of 1 GB of memory space dedicated for traffic shaping. It is split into 64,000 individual shaping queues. In addition, it offers 3072 queues for high-priority traffic and 3072 queues for low-priority traffic.

Note Traffic shaping is explained in a later module.

- The fabric is capable of queuing in the second and third stage.
- The output PLA contains a hardware queue that can hold approximately 110 packets with the maximum MTU size.

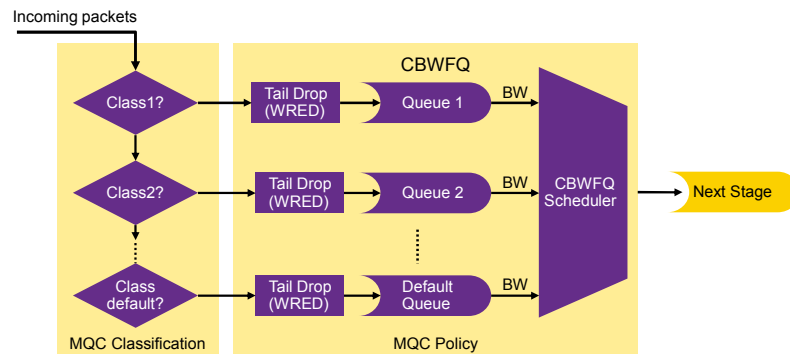
This architecture, including the two queues on input PLA, for high priority and low priority, provides complete, end-to-end packet prioritization. It is also known as high-priority propagation, which means that high-priority traffic always gets preference, even when competing with data from other ports and queues. This leads to lower latency and less jitter for priority traffic regardless of the congestion scenario.

Configuring CBWFQ

This topic describes how to configure class-based weighted fair queuing.

Class-Based Weighted Fair Queuing

- CBWFQ is a mechanism that is used to guarantee bandwidth to classes.
- CBWFQ supports user-defined traffic classes.
 - Classes are based on user-defined match criteria.
 - Packets satisfying the match criteria constitute the traffic for that class.
- A queue is reserved for each class.



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-14

CBWFQ provides support for user-defined traffic classes. With CBWFQ, you define the traffic classes based on match criteria. Packets satisfying the match criteria for a class constitute the traffic for that class. A queue is reserved for each class, and traffic belonging to a class is directed to that class queue.

After a class has been defined according to its match criteria, you can assign characteristics to it. To characterize a class, you assign the guaranteed bandwidth to it. The bandwidth assigned to a class is the minimum bandwidth allocated to the class during congestion.

CBWFQ Characteristics

- Each queue has a queue size:
 - Maximum number of packets that it can hold.
 - Maximum queue size is platform dependent.
 - Cisco IOS XR platforms use dynamic thresholds.
- Classification:
 - Uses class maps.
 - After classification, packet enqueued
 - If the queue limit has been reached, tail drop within each class

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—5-15

To characterize a class, you also specify the queue limit for that class, which is the maximum number of packets allowed to accumulate in the class queue. After a queue has reached its configured queue limit, enqueueing of additional packets to the class causes tail drop.

CBWFQ supports multiple class maps to classify traffic into its corresponding FIFO queues. Tail drop is the default dropping scheme of CBWFQ. You can use weighted random early detection (WRED) in combination with CBWFQ to prevent congestion of a class.

Note WRED is described in a later lesson.

The CBWFQ scheduler is used to guarantee bandwidth that is based on the configured weights.

CBWFQ Scheduling

- CBWFQ guarantees bandwidth according to weights assigned to traffic classes.
- Weights can be defined by specifying:
 - Bandwidth (in Kb/s, Mb/s, Gb/s)
 - Percentage of bandwidth (percentage of available interface bandwidth)
 - Percentage of remaining available bandwidth
 - One service policy cannot have mixed types of weights.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-18

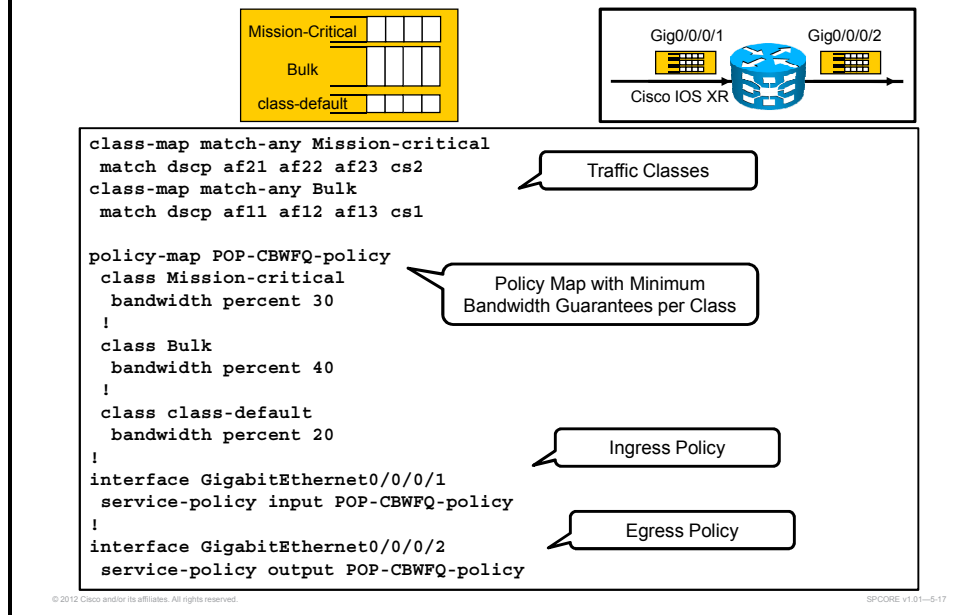
You can configure bandwidth guarantees by using one of these commands:

- The **bandwidth** command allocates a fixed amount of bandwidth by specifying the amount in kilobits, megabits, or gigabits per second.
- You can use the **bandwidth percent** command to allocate a percentage of the default or available bandwidth of an interface. The default bandwidth usually equals the maximum speed of an interface. The default value can be replaced by using the **bandwidth interface** command. It is recommended that the bandwidth reflect the real speed of the link. The value configured with the **bandwidth percent** command is the minimum guaranteed bandwidth allocated to the traffic class.
- You can use the **bandwidth remaining percent** command to define how any unallocated bandwidth should be apportioned. It is typically used in conjunction with the **bandwidth** configuration at the parent level in hierarchical policy maps. In such a combination, if the minimum bandwidth guarantees are met, the remaining bandwidth is shared in the ratio defined by the **bandwidth remaining** command in the class configuration in the policy map. The available bandwidth is equally distributed among those queuing classes that do not have the remaining bandwidth explicitly configured. The **bandwidth remaining** command does not offer any reserved bandwidth capacity.

A single service policy cannot mix the fixed bandwidth (in bits per second), **bandwidth percent**, and **bandwidth remaining** commands in the same level.

Note	On egress, the actual bandwidth of the interface is determined to be the Layer 2 capacity excluding cyclic redundancy check (CRC). These have to be included because they are applied per packet, and the system cannot predict how many packets of a particular packet size are being sent out.
-------------	--

Configuring CBWFQ



This figure illustrates a CBWFQ scenario on a Cisco IOS XR router. Two traffic classes have been defined (Mission-critical and Bulk) and configured to match respective DSCP values.

The policy map (POP-CBWFQ-policy) implements CBWFQ by allocating bandwidth guarantees of 30, 40, and 20 percent to the classes Mission-critical, Bulk, and class-default.

The CBWFQ structure is applied to two interfaces, in the input and output directions.

Note On a Cisco IOS-XR CBWFQ implementation, the algorithm used in dequeuing the packets from each CBWFQ queue is based on MDRR instead of WFQ.

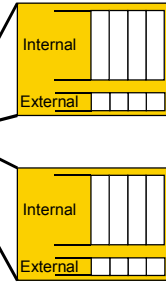
Configuring Hierarchical CBWFQ

```
class-map match-any External
match access-group ipv4 External-nets
!
class-map match-any Internal
match access-group ipv4 Internal-nets
!
policy-map cbwfq-child
class Internal
bandwidth remaining percent 80
!
class External
bandwidth remaining percent 20
!
policy-map cbwfq-parent
class Mission-critical
service-policy cbwfq-child
bandwidth percent 30
!
class Bulk
service-policy cbwfq-child
bandwidth percent 40
!
interface GigabitEthernet0/0/0/1
service-policy output cbwfq-parent
```

Bandwidth Remaining
on Child Level

Mission-Critical
Bulk
class-default

Bandwidth Guarantee and Child
Service Policy on Parent Level



This figure illustrates a hierarchical queuing scenario that consists of two scheduling levels. On the parent level, the two classes (Mission-critical and Bulk) have been allocated minimum bandwidth guarantees of 30 and 40 percent, respectively. Each class has the *cbwfq-child* policy applied to it, which divides the bandwidth to two sub-classes (Internal and External) in the ratio 80:20.

This scenario illustrates the use of the **bandwidth percent** and **bandwidth remaining** commands. The **bandwidth percent** command sets the bandwidth guarantees on the parent level, while the **bandwidth remaining** command defines how the bandwidth should be apportioned to the child classes.

In this case, the policy is applied to the output interface, but could also be configured for the ingress direction.

Monitoring CBWFQ

```
RP/0/RSP0/CPU0:POP# show policy-map interface GigabitEthernet 0/0/0/2
GigabitEthernet0/0/0/2 direction input: Service Policy not installed

GigabitEthernet0/0/0/2 output: POP-CBWFQ-policy

Class Bulk
Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :      1320/1343760      152
  Transmitted   :      1319/1342742      152
  Total Dropped :      0/0      0
Queueing statistics
  Queue ID      : 266
  High watermark (Unknown)
  Inst-queue-len (packets)      : 2
  Avg-queue-len  (Unknown)
  Taildropped (packets/bytes)    : 0/0
  Queue (conform)      :      457/465226      53
  Queue (exceed)      :      862/877516      99
  RED random drops (packets/bytes) : 0/0
<to be continued>
```

Class Bulk Statistics

Conform Counter

Exceed does not mean drop.

The **show policy-map interface** command displays the configuration of all classes configured for all service policies on the specified interface. This includes the queuing statistics for each traffic class defined in the policy map.

In this first section of the command output, you see the statistics for the Bulk class..

The queuing statistics include current queue length in packets, tail-drop counters, and conform and exceed queue statistics. The conform and exceed counters are related to the committed information rate (CIR) and peak information rate (PIR) value. These correspond to the “guaranteed” bandwidth for the queue, and the “maximum” bandwidth for the queue. Even if the QoS policy does not explicitly set these values, the system chooses them for internal processing. The “conform” counter in **show policy-map** is the number of packets or bytes that were transmitted within the CIR value, and the “exceed” value is the number of packets or bytes that were transmitted within the PIR value.

Note The “exceed” in this case does NOT equate to a packet drop, but rather a packet that is above the CIR rate on that queue.

Monitoring CBWFQ (Cont.)

```
Class Mission-critical
Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :      45127/876107310      3433
  Transmitted  :      8234122/104028      8732
  Total Dropped :      0/0      0
Queueing statistics
  Queue ID      : 267
  High watermark (Unknown)
  Inst-queue-len (packets)      : 127
  Avg-queue-len (Unknown)
  Taildropped (packets/bytes)    : 34/98765
  Queue (conform)      :      874545/563736658      10
  Queue (exceed)      :      9877/7267370      22
  RED random drops (packets/bytes) : 0/0

Class class-default
Classification statistics      (packets/bytes)      (rate - kbps)
  Matched      :      127/107310      33
  Transmitted  :      122/104028      32
  Total Dropped :      0/0      0
Queueing statistics
  Queue ID      : 268
  High watermark (Unknown)
  Inst-queue-len (packets)      : 10
  Avg-queue-len (Unknown)
  Taildropped (packets/bytes)    : 0/0
  Queue (conform)      :      45/36658      10
  Queue (exceed)      :      77/67370      22
  RED random drops (packets/bytes) : 0/0
```

Class Mission-Critical Counters

class-default Statistics

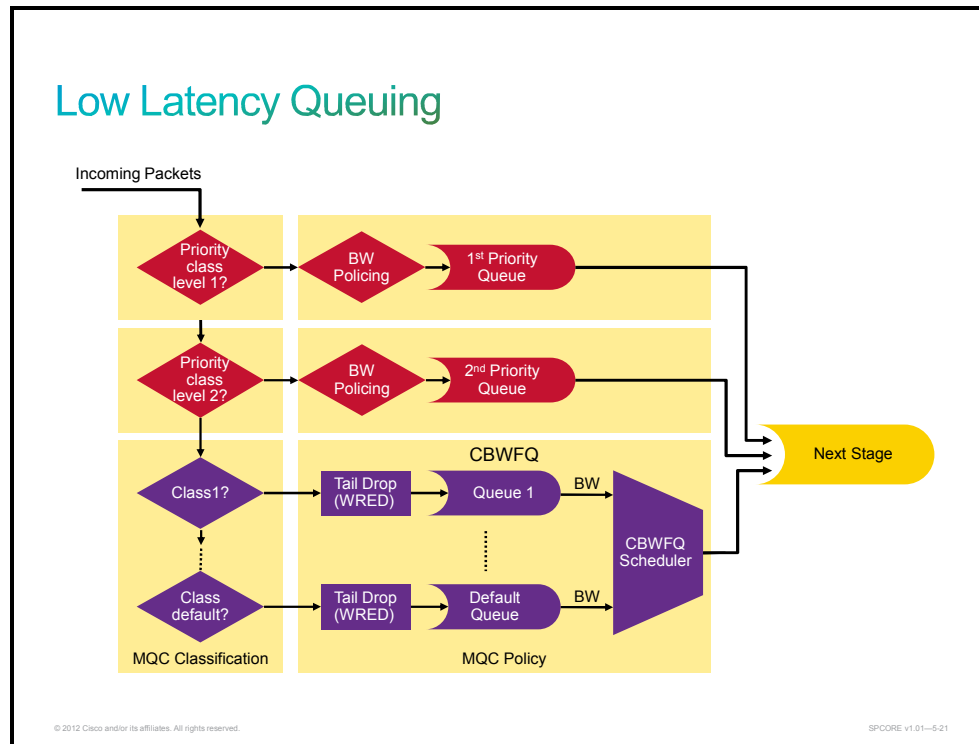
© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-20

In this section of the command output, you see the statistics for the remaining two classes (Mission-critical and class-default).

Configuring LLQ

This topic describes how to configure low latency queuing.



The LLQ feature brings strict priority queuing to CBWFQ. Strict priority queuing allows delay-sensitive data such as voice to be dequeued and sent first (before packets in other queues are dequeued), giving delay-sensitive data preferential treatment over other traffic.

For CBWFQ, the weight for a packet belonging to a specific class is derived from the bandwidth that you assigned to the class when you configured it. This scheme poses problems for voice traffic, which is largely intolerant of delay, especially variation in delay. For voice traffic, variations in delay introduce irregularities of transmission that are heard as jitter.

The LLQ feature provides strict priority queuing, reducing jitter in voice conversations. Configured by the **priority** command, LLQ enables use of a single, strict priority queue within CBWFQ at the class level, allowing you to direct traffic belonging to a class to the CBWFQ strict priority queue. To enqueue class traffic to the strict priority queue, you configure the **priority** command for the class after you specify the named class within a policy map. Classes to which the priority command is applied are considered priority classes. Within a policy map, you can give one or more classes priority status. When multiple classes within a single policy map are configured as priority classes, all traffic from these classes is enqueued to the same single strict priority queue.

If LLQ is used within the CBWFQ system, it creates an additional priority queue in the CBWFQ system, which is serviced by a strict priority scheduler. Any class of traffic can therefore be attached to a service policy, which uses priority scheduling, and hence can be prioritized over other classes.

Cisco IOS XR Software uses two priority queues: level 1 and level 2. Level 1 has a higher priority than level 2. Any number of classes can be assigned to a priority queue.

LLQ Benefits

- High-priority classes are guaranteed:
 - Low-latency propagation of packets
 - Bandwidth
- Consistent configuration and operation across all media types
- Entrance criteria to a class can be defined by any classifier:
 - Not limited to UDP ports as with IP RTP priority
 - Defines trust boundary to ensure simple classification and entry to a queue

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01—5-22

The LLQ priority scheduler guarantees both low-latency propagation of packets and bandwidth to high-priority classes. Low latency is achieved by expediting traffic using a priority scheduler. Bandwidth is also guaranteed by the nature of priority scheduling, but is policed to a user-configurable value. The strict PQ scheme allows delay-sensitive data such as voice to be dequeued and sent first—that is, before packets in other queues are dequeued. Delay-sensitive data is given preferential treatment over other traffic.

Because you can configure the priority status for a class within CBWFQ, you are not limited to UDP port numbers to stipulate priority flows, unlike IP Real-Time Transport Protocol (IP RTP). Instead, all of the valid match criteria used to specify traffic for a class now apply to priority traffic.

Policing of priority queues also prevents the priority scheduler from monopolizing the CBWFQ scheduler and starving non-priority classes, as legacy PQ does. By configuring the maximum amount of bandwidth allocated for packets belonging to a class, you can avoid starving nonpriority traffic.

Configuring LLQ

```
policy-map llq-policy
class Voice-internal
  priority level 1
  police rate percent 5
  queue-limit 20 ms
!
class Bulk
  bandwidth percent 60
  queue-limit 50 ms
!
class Voice-external
  priority level 1
  police rate percent 10
!
class Video
  priority level 2
  police rate percent 20
!
interface GigabitEthernet0/0/0/1
  service-policy input llq-policy
!
interface GigabitEthernet0/0/0/2
  service-policy output llq-policy
```

Priority queue has precedence but requires a throttle (policing)

Default maximum threshold for priority queues is 10 ms

CBWFQ queue with minimum bandwidth guarantee

Default maximum threshold for regular queues is 100 ms

All traffic with same priority level directed to the same queue

Level 2 has lower priority than level 1

Ingress LLQ

Egress LLQ

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-5-23

This scenario illustrates how to configure LLQ on a Cisco IOS XR platform. The policy map (llq-policy) defines special handling for four traffic classes (Voice-internal, Bulk, Voice-external, and Video). The traffic classifications are not shown in this example. Three of the four classes are declared as priority traffic. Two classes (Voice-internal and Voice-external) are assigned to the priority level 1 queue. The video class is assigned to the lower-precedence level 2 priority queue. In Cisco IOS XR Software, each priority class must have a policing statement that limits the amount of traffic forwarded within that class and thus prevents starving of other classes. In Cisco IOS and IOS XE Software, the priority classes implicitly police the priority bandwidth.

Note Policing is discussed in a later module.

The **queue-limit** command has been configured in some classes to change the default maximum threshold per queue. The default maximum threshold for priority queues is 10 ms. Default maximum threshold for regular queues is 100 ms.

The LLQ structure has been applied to two interfaces, respectively in the input and output direction.

Monitoring LLQ

```
RP/0/RSP0/CPU0:PE7#show policy-map interface gigabit 0/0/0/2
GigabitEthernet0/0/0/2 direction input: Service Policy not installed

GigabitEthernet0/0/0/2 output: llq-policy

Class Voice-internal
  Classification statistics          (packets/bytes)          (rate - kbps)
    Matched                        :      1320/1343760          152
    Transmitted                    :      1319/1342742          152
    Total Dropped                  :           0/0              0
  Queueing statistics
    Queue ID                       :      266
    High watermark (Unknown)       :
    Inst-queue-len (packets)       :      2
    Avg-queue-len (Unknown)       :
    Taildropped(packets/bytes)     :      0/0
    Queue (conform)                :      457/465226          53
    Queue (exceed)                 :           2/516           7
    RED random drops(packets/bytes):      0/0
<output omitted>
```

Statistics for priority queue presented as for any other queue

Remaining queues (Bulk, Voice-external, Video and class-default) omitted

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--9-24

The **show policy-map interface** command is used to verify LLQ operations in the same way that it provides CBWFQ-related information. The command displays the queuing statistics for each traffic class defined in the respective policy map.

This output shows the counters for the Voice-internal class. The output for the remaining queues (Bulk, Voice-external, Video and class-default) has been omitted.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- Congestion can occur at any point in the network but particularly at points of speed mismatches and traffic aggregation
- FIFO is the simplest queuing algorithm
- In priority queuing (PQ), each packet is assigned a priority and placed into a hierarchy of queues based on priority
- With round-robin queuing, one packet is taken from each queue and then the process repeats
- The weighted round robin (WRR) algorithm was developed to provide prioritization capabilities for round robin
- Deficit round robin (DRR) resolves the inaccurate bandwidth allocation problem with WRR
- Modified deficit round robin (MDRR) is a class-based composite scheduling mechanism that allows for queueing of up to eight traffic classes

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-25

Summary (Cont.)

- Distributed queuing, available on Cisco IOS XR Software, extends the concepts of software and hardware queues by providing a distributed architecture consisting of interface modules and router fabric.
- Queuing architecture on Cisco IOS XR platforms provide complete, end-to-end packet prioritization
- CBWFQ assigns minimum bandwidth guarantees to traffic classes.
- LLQ combines priority queuing with minimum bandwidth guarantees for nonpriority queues.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-26

Implementing Congestion Avoidance

Overview

TCP supports traffic management mechanisms such as slow start and fast retransmit. When congestion occurs, tail-dropping the TCP traffic can cause TCP global synchronization, resulting in poor bandwidth use. This lesson describes how TCP manages the traffic flow between two hosts, and the effects of tail-dropping on TCP traffic.

Congestion avoidance techniques monitor network traffic loads in an effort to anticipate and avoid congestion at common network bottleneck points. Congestion avoidance is achieved through packet dropping using a more complex dropping technique than simple tail drop.

This lesson describes the weighted random early detection (WRED) congestion avoidance, which is the Cisco implementation of random early detection (RED).

Objectives

Upon completing this lesson, you will be able to explain the problems that may result from the limitations of TCP congestion management mechanisms. This ability includes being able to meet these objectives:

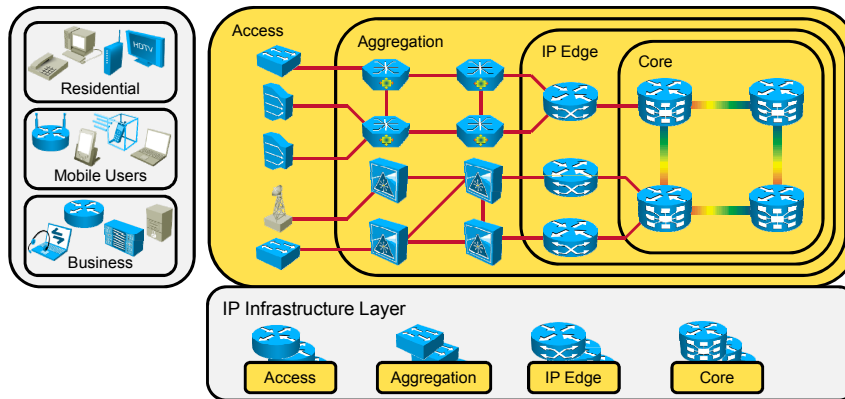
- Explain the need for congestion avoidance mechanisms
- Describe the TCP congestion management mechanisms
- Describe the TCP Global Synchronization problem caused by Tail Drop
- Describe the Random Early Detection Congestion Avoidance mechanism
- Describe how to configure RED and WRED using MQC

Congestion Avoidance Introduction

This topic explains the need for congestion avoidance mechanisms.

Congestion Avoidance in IP NGN

- Congestion avoidance used in all IP NGN layers
- Tail drop has undesired results
- Techniques to prevent congestion



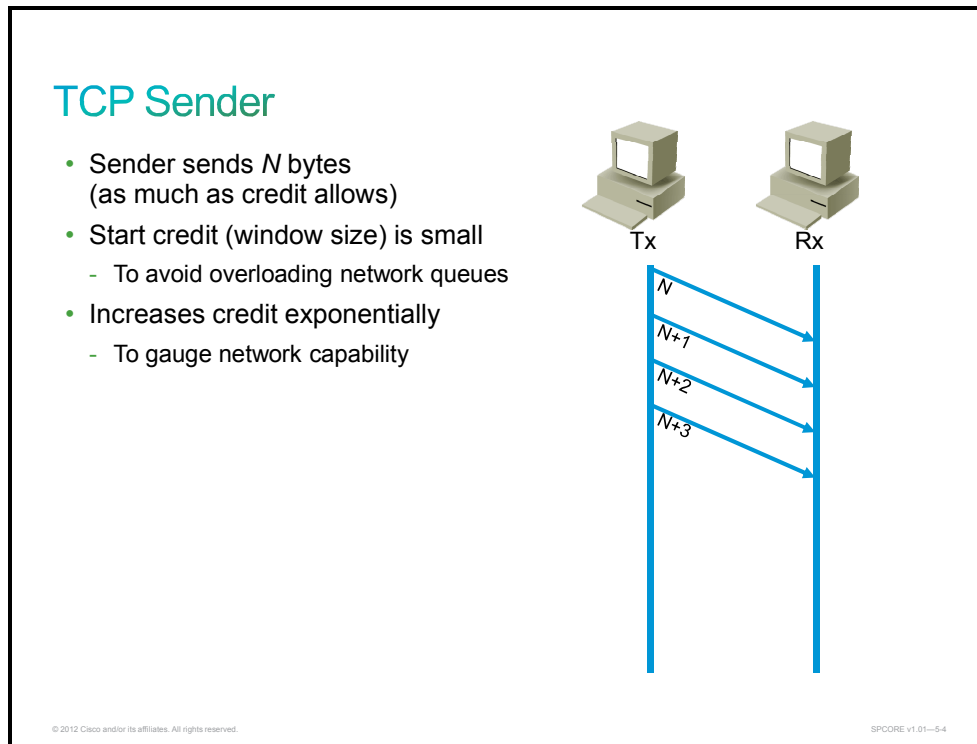
© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6.3

Congestion can occur in any layer of the IP Next-Generation Network (NGN) environment. Congestion has undesired results for network performance because it causes tail drops. Tail drops occur when traffic cannot be enqueued, because the queue buffers are full. There are techniques to prevent congestions. The most common methods, Random Early Detection (RED) and Weighted Random Early Detection (WRED), are supported on Cisco routers. These mechanisms are discussed in this lesson.

TCP Congestion Management

This topic describes the TCP congestion management mechanisms.



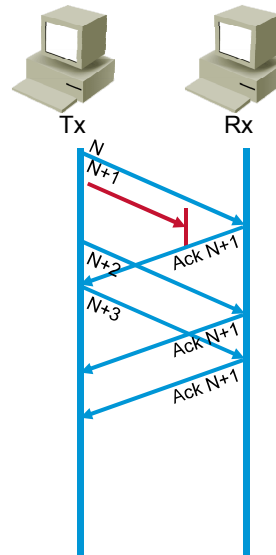
Before any data is transmitted using TCP, a connection must first be established between the transmitting and receiving hosts. When the connection is initially established, the two hosts must agree on certain parameters that will be used during the communication session. One of the parameters that must be decided is called the window size, or how many data bytes to transmit at a time. Initially, TCP sends a small number of data bytes, and then exponentially increases the number sent. For example, a TCP session originating from host A begins with a window size of 1 and therefore sends one packet. When host A receives a positive acknowledgment (ACK) from the receiver, Host A increases its window size to 2. Host A then sends two packets, receives a positive ACK, and increases its window size to 4, and so on.

Note TCP tracks window size by byte count. For the purposes of illustration, N is used.

In traditional TCP, the maximum window size is 64 KB (65,535 bytes). Extensions to TCP, specified in RFC 1323, allow for tuning TCP by extending the maximum TCP window size to 2^{30} bytes. TCP extensions for high performance, although supported on most operating systems, may not be supported on your system.

TCP Receiver

- Receiver schedules an ACK on receipt of next message.
- TCP acknowledges the next segment it expects to receive, not the last segment it received.
- In the example, N+1 is blocked, so the receiver keeps acknowledging N+1 (the next segment it expects to receive).



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6.6

When the receiver receives a data segment, the receiver checks that data segment sequence number (byte count). If the data received fills in the next sequence of numbers expected, the receiver indicates that the data segment was received in order. The receiver then delivers all the data that it holds to the target application, and updates the sequence number to reflect the next byte number in expected order.

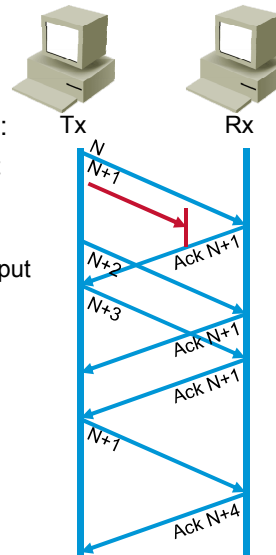
When this process is complete, the receiver performs one of these actions:

- Immediately transmits an ACK to the sender
- Schedules an ACK to be transmitted to the sender after a short delay

The ACK notifies the sender that the receiver received all data segments up to but not including the byte number in the new sequence number. Receivers usually try to send an ACK in response to alternating data segments they receive. They send the ACK because, for many applications, if the receiver waits out a small delay, it can efficiently piggyback its reply acknowledgment on a normal response to the sender. However, when the receiver receives a data segment out of order, it immediately responds with an ACK to direct the sender to retransmit the lost data segment.

TCP Slow Start

- If ACK acknowledges something:
 - Updates credit and sends
- If not, presumes it indicates a lost packet:
 - Sends first unacknowledged message right away
 - Halves current credit (slows down)
 - Increases slowly to gauge network throughput



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-5-6

When the sender receives an ACK, the sender determines if any data is outstanding:

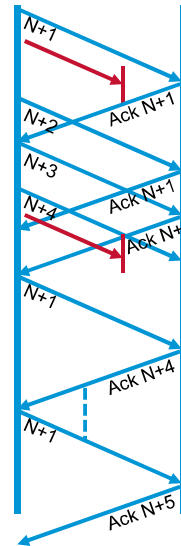
- If no data is outstanding, the sender determines that the ACK is a keepalive, meant to keep the line active, and it does nothing.
- If data is outstanding, the sender determines whether the ACK indicates that the receiver has received some or none of the data.
 - If the ACK acknowledges receipt of some data sent, the sender determines if new credit has been granted to allow it to send more data.
 - When the ACK acknowledges receipt of none of the sent data and there is outstanding data, the sender interprets the ACK to be a repeatedly sent ACK. This condition indicates that some data was received out of order, forcing the receiver to retransmit the first ACK, and that a second data segment was received out of order, forcing the receiver to retransmit the second ACK. In most cases, the receiver would receive two segments out of order, because one of the data segments had been dropped.

When a TCP sender detects a dropped data segment, it retransmits the segment. Then the sender slows its transmission rate so that the rate is half of what it was before the drop was detected. This is known as the TCP slow-start mechanism.

In the figure, a station transmits three packets to the receiving station. Unfortunately, the first packet is dropped somewhere in the network. Therefore the receiver sends an ACK 1 to request the missing packet. Because the transmitter does not know if the ACK was just a duplicate ACK, it will wait for three ACK 1 packets from the receiver. Upon receipt of the third ACK, the missing packet, packet 1, is resent to the receiver. The receiver now sends an ACK 4 indicating that it has already received packets 2 and 3 and is ready for the next packet.

Multiple Drops in TCP

- If multiple drops occur in the same session:
 - Current TCPs wait for timeout.
 - Selective acknowledge may be a workaround.
 - New “fast retransmit” phase takes several round-trip times to recover.



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6.7

Although the TCP slow-start behavior is appropriately responsive to congestion, problems can arise when multiple TCP sessions are concurrently carried on the same router and all TCP senders slow down transmission of packets at the same time.

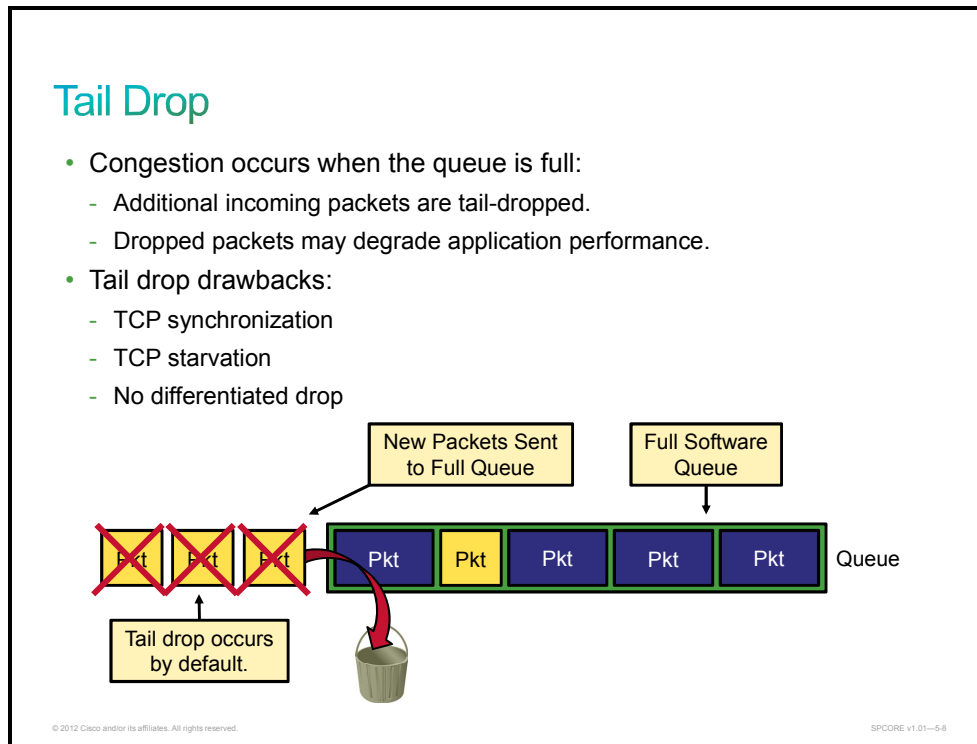
If a TCP sender does not receive acknowledgement for sent segments, it cannot wait indefinitely before it assumes that the data segment that was sent never arrived at the receiver. TCP senders maintain the retransmission timer to trigger a segment retransmission. The retransmission timer can impact TCP performance. If the retransmission timer is too short, duplicate data will be sent into the network unnecessarily. If the retransmission timer is too long, the sender will wait (remain idle) for too long, slowing down the flow of data.

The selective acknowledgment (SACK) mechanism, as proposed in RFC 2018, can improve the time it takes for the sender to recover from multiple packet losses, because noncontiguous blocks of data can be acknowledged, and the sender only has to retransmit data that is actually lost. SACK is used to convey extended acknowledgement information from the receiver to the sender to inform the sender of noncontiguous blocks of data that have been received. Using the example in the figure, instead of sending back an ACK $N + 1$, the receiver can send a SACK $N + 1$ and also indicate back to the sender that $N + 3$ has been correctly received with the SACK option.

In standard TCP implementations, a TCP sender can only discover that a single packet has been lost each round-trip time (RTT), causing poor TCP performance when multiple packets are lost. The sender must receive three duplicate ACK packets before it realizes that a packet has been lost. As a result of receiving the third ACK, the sender will immediately send the segment referred to by the ACK. This TCP behavior is called fast retransmit.

Tail Drop and TCP Global Synchronization

This topic describes the TCP Global Synchronization problem caused by Tail Drop.



When an interface on a router cannot transmit a packet immediately, the packet is queued. Packets are then taken out of the queue and eventually transmitted on the interface.

If the arrival rate of packets to the output interface exceeds the router capability to buffer and forward traffic, the queues increase to their maximum length and the interface becomes congested. Tail drop is the default queuing response to congestion. Tail drop treats all traffic equally and does not differentiate between classes of service. Applications may suffer performance degradation due to packet loss caused by tail drop. When the output queue is full and tail drop is in effect, all packets trying to enter at the tail of the queue are dropped until the congestion is eliminated and the queue is no longer full.

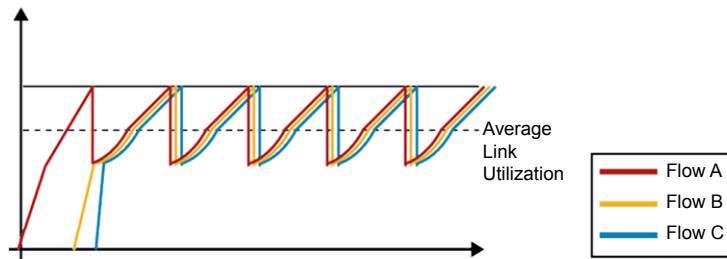
The simple tail-drop scheme does not work well in environments with a large number of TCP flows or in environments in which selective dropping is required. Administrators should understand the network interaction between TCP stack intelligence and dropping in order to implement a more efficient and fair dropping scheme, especially in service provider environments.

Tail drop has the following shortcomings:

- When congestion occurs, dropping affects most of the TCP sessions, which simultaneously back off and then restart again. This causes inefficient link utilization at the congestion point (TCP global synchronization).
- TCP starvation, in which all buffers are temporarily seized by aggressive flows, and normal TCP flows experience buffer starvation.
- There is no differentiated drop mechanism, and therefore premium traffic is dropped in the same way as best-effort traffic.

TCP Synchronization

- Multiple TCP sessions start at different times.
- TCP window sizes are increased.
- Tail drops cause many packets of many sessions to be dropped at the same time.
- TCP sessions restart at the same time (synchronized).



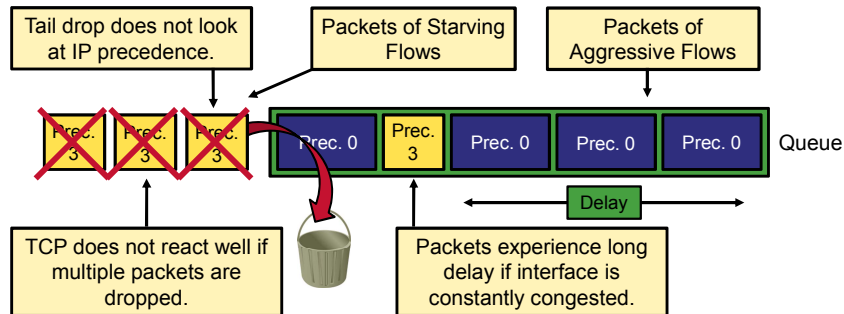
A router can handle multiple concurrent TCP sessions. It is likely that when traffic exceeds the queue limit, it exceeds this limit due to the bursty nature of packet networks. However, there is also a high probability that excessive traffic depth caused by packet bursts are temporary and that traffic does not stay excessively deep except at points where traffic flows merge, or at edge routers.

If the receiving router drops all traffic that exceeds the queue limit, as is done with tail drop by default, many TCP sessions simultaneously go into slow start. Consequently, traffic temporarily slows down to the extreme and then all flows slow-start again. This activity creates a condition called global synchronization.

Global synchronization occurs as waves of congestion crest only to be followed by troughs during which the transmission link is not fully used. Global synchronization of TCP hosts can occur because packets are dropped all at once, and when multiple TCP hosts reduce their transmission rates in response to packet dropping. When congestion is reduced, their transmission rates are increased.

Delay, Jitter, and Starvation

- Constant high buffer usage (long queue) causes delay.
- More aggressive flows can cause other flows to starve.
- No differentiated dropping occurs.



During periods of congestion, packets are queued up to the full queue length, which also causes increased delay for packets that are already in the queue. In addition, queuing introduces unequal delays for packets of the same flow, thus producing jitter.

Another TCP-related phenomenon that reduces optimal throughput of network applications is TCP starvation. When multiple flows are established over a router, some of these flows may be much more aggressive than other flows. For instance, when a file transfer application TCP transmit window increases, the TCP session can send a number of large packets to its destination. The packets immediately fill the queue on the router, and other, less aggressive flows can be starved because there is no differentiated treatment indicating which packets should be dropped. As a result, these less aggressive flows are tail-dropped at the output interface.

Based on the knowledge of TCP behavior during periods of congestion, you can conclude that tail drop is not the optimal mechanism for congestion avoidance and therefore should not be used. Instead, more intelligent congestion avoidance mechanisms should be used that slow down traffic before actual congestion occurs.

Random Early Detection (RED) Introduction

This topic describes the Random Early Detection Congestion Avoidance mechanism.

Random Early Detection

- Tail drop can be avoided if congestion is prevented.
- RED:
 - Mechanism that randomly drops packets before a queue is full.
 - Increases drop rate as the average queue size increases.
- RED results:
 - TCP sessions slow down to the approximate rate of output-link bandwidth.
 - Average queue size is small (much less than the maximum queue size).
 - TCP sessions are desynchronized by random drops.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-11

Random early detection (RED) is a dropping mechanism that randomly drops packets before a queue is full. The dropping strategy is based primarily on the average queue length—that is, when the average size of the queue increases, RED will be more likely to drop an incoming packet than when the average queue length is shorter.

Because RED drops packets randomly, it has no per-flow intelligence. The rationale is that an aggressive flow will represent most of the arriving traffic, which means it is likely that RED will drop a packet of an aggressive session. In other words, RED punishes more aggressive sessions with higher statistical probability and is, therefore, able to somewhat selectively slow down the most significant cause of congestion. Directing one TCP session at a time to slow down allows for full utilization of the bandwidth, rather than utilization that manifests itself as crests and troughs of traffic.

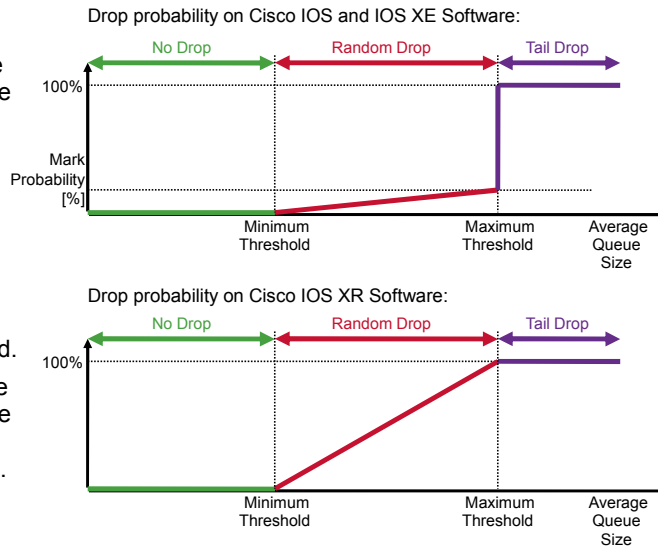
As a result of implementing RED, the problem of TCP global synchronization is much less likely to occur, and TCP can utilize link bandwidth more efficiently. In RED implementations, the average queue size also decreases significantly, as the possibility of the queue filling up is reduced. This is because of very aggressive dropping in the event of traffic bursts, when the queue is already quite full.

RED distributes losses over time and normally maintains a low queue depth while absorbing traffic spikes. RED can also utilize markers, such as differentiated services code point (DSCP), to establish different drop profiles for different classes of traffic. This is referred to as weighted random early detection (WRED).

RED Profiles

RED modes:

- **No drop:** When the average queue size is between 0 and the minimum threshold.
- **Random drop:** When the average queue size is between the minimum and the maximum threshold.
- **Tail drop:** When the average queue size is at maximum threshold or above.



A RED traffic profile is used to determine the packet-dropping strategy and is based on the average queue length. The probability of a packet being dropped is based on two thresholds contained within the RED profile:

- **Minimum threshold:** When the average queue length is equal or above the minimum threshold, RED starts dropping packets. The rate of packet drop increases linearly as the average queue size increases, until the average queue size reaches the maximum threshold.
- **Maximum threshold:** When the average queue size is above the maximum threshold, all packets are dropped.

Cisco IOS and IOS XE Software use one additional parameter—mark probability denominator. This is the fraction of packets that are dropped when the average queue depth is at the maximum threshold. For example, if the denominator is 20, one out of every 20 packets is dropped when the average queue is at the maximum threshold. In Cisco IOS XR Software, this value is set to 1.

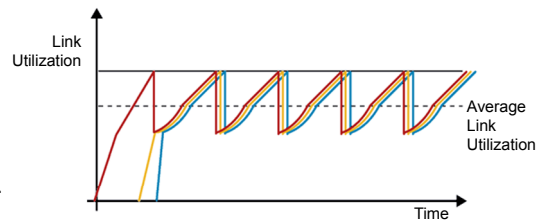
The minimum threshold value should be set high enough to maximize the link utilization. If the minimum threshold is too low, packets may be dropped unnecessarily. The difference between the maximum threshold and the minimum threshold should be large enough to avoid global synchronization. If the difference is too small, many packets may be dropped at once, resulting in global synchronization.

Based on the average queue size, RED has three dropping modes:

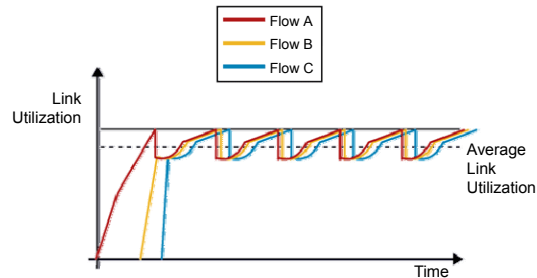
- When the average queue size is between 0 and the configured minimum threshold, no drops occur and all packets are queued.
- When the average queue size is between the configured minimum threshold and the configured maximum threshold, random drops occur, which is linearly proportional to the mark probability denominator and the average queue length.
- When the average queue size is at or higher than the maximum threshold, RED performs full (tail) drop in the queue. This is unlikely, as RED should slow down TCP traffic ahead of congestion. If a lot of non-TCP traffic is present, RED cannot effectively drop traffic to reduce congestion, and tail drops are likely to occur.

TCP Traffic with RED

- Without RED:
 - TCP synchronization prevents average link utilization close to the link bandwidth.
 - Tail drops cause TCP sessions to go into slow-start.



- With RED:
 - Average link utilization is much closer to link bandwidth.
 - Random drops cause TCP sessions to reduce window sizes.



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-13

The first figure shows TCP throughput behavior compared to link bandwidth in a congested network scenario where the tail-drop mechanism is in use on a router. The global synchronization phenomenon causes all sessions to slow down when congestion occurs. All sessions are penalized when tail drop is used because it drops packets with no discrimination between individual flows. When all sessions slow down, congestion on the router interface is removed and all TCP sessions restart their transmission at roughly the same time. Again, the router interface quickly becomes congested, causing tail drop. As a result, all TCP sessions back off again. This behavior cycles constantly, resulting in a link that is generally underutilized.

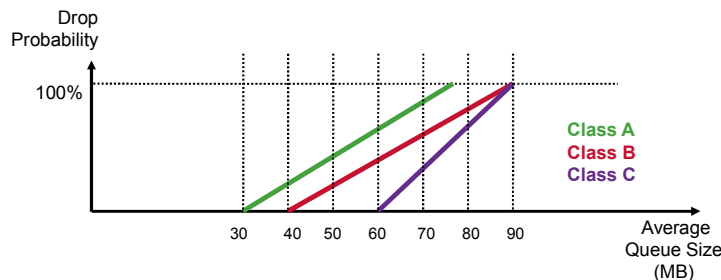
The second figure shows TCP throughput behavior compared to link bandwidth in a congested network scenario in which RED has been configured on a router. RED randomly drops packets, influencing a small number of sessions at a time, before the interface reaches congestion. Overall throughput of sessions is increased, as well as average link utilization. Global synchronization is very unlikely to occur, due to selective, but random, dropping of adaptive traffic.

Configuring WRED

This topic describes how to configure RED and WRED using MQC.

Weighted Random Early Detection

- WRED can use multiple different RED profiles.
 - Drops less important packets more aggressively than important packets.
- Each profile is identified by:
 - Minimum threshold
 - Maximum threshold
 - Maximum drop probability (Cisco IOS and IOS XE Software only)



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-5.14

WRED performs differentiated packet dropping based on packet markers, such as DSCP. As with RED, WRED monitors the average queue length in the router and determines when to begin discarding packets based on the length of the interface queue. When the average queue length is greater than the user-specified minimum threshold, WRED begins to randomly drop packets (both TCP and UDP packets) with a certain probability. If the average length of the queue becomes larger than the maximum threshold, WRED reverts to a tail-drop packet discard strategy.

WRED can selectively discard lower-priority traffic when the interface becomes congested, and can provide differentiated performance characteristics for different classes of service.

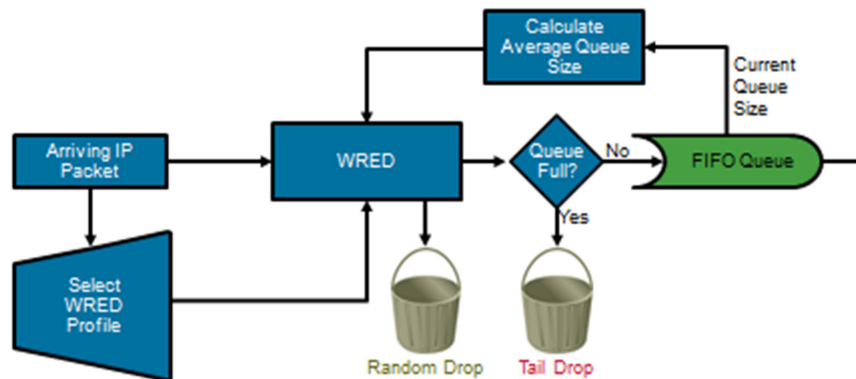
WRED is only useful when the bulk of the traffic is TCP traffic. With TCP, dropped packets indicate congestion, so the packet source reduces its transmission rate. With other protocols, packet sources might not respond or might resend dropped packets at the same rate, and so dropping packets might not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

WRED is more often used in the core than in the edge and the access network. Access and edge routers mark packets. WRED uses these assigned values to determine how to treat different types of traffic.

WRED is not recommended for voice and video. WRED will not throttle back voice traffic because voice traffic is UDP-based. The network itself should be designed not to lose voice packets because lost voice packets result in reduced voice quality.

The figure illustrates differentiated RED profiles, for classes A, B, and C. Each class has the minimum and maximum thresholds set to specific values.

WRED Operation



The traffic profile defines the minimum threshold and maximum threshold. Cisco IOS and IOS XE Software also use the mark probability denominator.

When a packet arrives at the output queue, a packet marker is used to select the correct WRED profile for the packet. The packet is then passed to WRED for processing. Based on the selected traffic profile and the average queue length, WRED calculates the probability for dropping the current packet and either drops the packet or passes it to the queue.

If the queue is already full, the packet is tail-dropped. Otherwise, the packet will eventually be transmitted. If the average queue length is greater than the minimum threshold but less than the maximum threshold, based on the drop probability, WRED will either queue the packet or perform a random drop.

Class-Based RED and WRED

- Class-based implementation on Cisco IOS, IOS XE, and IOS XR Software
- WRED profile selection is based on (all Cisco routers):
 - IP precedence (8 profiles)
 - DSCP (64 profiles)
 - Discard class (8 profiles)
- Additional WRED profile selection on Cisco IOS XR Software:
 - MPLS EXP (8 profiles)
 - Discard eligibility indicator (2 profiles)
 - CoS (8 profiles)
- RED and WRED can be applied in Cisco IOS XR Software:
 - Interface input and output
 - Layer 2 subinterfaces
 - Layer 2 and Layer 3 main interfaces

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-18

RED and WRED are implemented on Cisco routers using class-based QoS CLI. It is typically combined with CBWFQ, and less commonly with LLQ.

WRED profile selection options differ depending on the platform in use. All Cisco routers (Cisco IOS, IOS XE, and IOS XR Software) support selection based on the following:

- IP precedence (8 profiles)
- DSCP (64 profiles)
- Discard class (8 profiles)

Cisco IOS XR Software offers these additional WRED profile selection options:

- Multiprotocol Label Switching experimental bits (MPLS EXP) (8 profiles)
- Discard eligibility indicator (2 profiles)
- Class of service (CoS) (8 profiles)

In Cisco IOS XR Software, you can apply the RED or WRED functionality on the following:

- Interface input and output
- Layer 2 subinterfaces
- Layer 2 and Layer 3 main interfaces

RED and WRED in Cisco IOS, IOS XE, and IOS XR Software

	Cisco IOS XR	Cisco IOS and IOS XE
Enabled by default?	No	No
Enable RED with default thresholds	random-detect default	N/A
Set custom RED thresholds	random-detect min max	N/A
Enable WRED with default curves	N/A	random-detect [precedence-based / DSCP-based / discard-class-based]
Configure WRED curve	random-detect marker marker-value min max	random-detect marker marker-value min max mark-prob-denominator
Directionality	Interface input/output	Interface output
Threshold units	Packets; bytes (kilo-, mega-, giga-); milli- or microseconds	Packets, bytes, milliseconds

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-17

This table summarizes the main implementation differences between Cisco IOS XR and IOS and IOS-XE Software.

RED and WRED are not enabled by default. On IOS XR devices, you have three configuration options for each class in a policy map:

- Enable RED using default minimum and maximum threshold values. This is done with the **random-detect default** command. This mode does not provide any differentiation between packets with various markers belonging to the class and thus acts as RED in that class.
- Configure RED explicit minimum and maximum thresholds using the **random-detect min-threshold max-threshold** command.
- Configure RED profiles for each packet marker within the traffic class. Defining multiple curves effectively enables WRED for the traffic class.

In Cisco IOS and IOS XE Software, you do not enable RED for a traffic class. You can only enable WRED by choosing the markers used for selecting the curves: DSCP, IP precedence, or discard class. You can also define custom curves and thus override the default values.

In Cisco IOS XR Software, you can apply the feature in either the input or output direction. Cisco IOS and IOS XE Software support only output. You can define thresholds using a wide range of units: packets, bytes (including megabytes and gigabytes on Cisco IOS XR Software) or milliseconds (and microseconds on Cisco IOS XR Software).

Configuring RED and WRED

```
policy-map POP1-policy
class Bulk
  random-detect default
  bandwidth percent 40
!
class Mission-critical
  bandwidth percent 40
  random-detect dscp af21 300 kbytes 500 kbytes
  random-detect dscp af22 250 kbytes 500 kbytes
  random-detect dscp cs2 200 kbytes 500 kbytes
!
class Top-priority
  priority level 1
  police rate percent 10
  random-detect 400 kbytes 500 kbytes
!
interface GigabitEthernet0/0/0/1
service-policy input POP1-policy
!
interface GigabitEthernet0/0/0/2
service-policy output POP1-policy
```

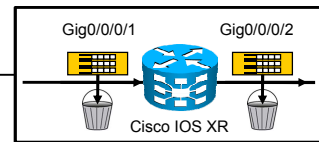
RED with Default
Thresholds in a
CBWFQ Queue

Custom WRED Curves
in a CBWFQ Queue

Custom RED Thresholds in a LLQ

Ingress RED/WRED

Egress RED/WRED



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-5-18

This example illustrates a WRED scenario on a Cisco IOS XR router installed in a point of presence (POP). The policy, POP1-policy, consists of four classes: Bulk, Mission-critical, Top-priority, and class-default (not shown in this configuration). The Top-priority class is defined as priority and has an LLQ. The remaining two classes have bandwidth guarantees using the CBWFQ principle.

The first class, Bulk, has been configured for RED, using default minimum and maximum thresholds. The minimum threshold falls within the range of 0 to 1,073,741,823 bytes, and can be configured in other units. The range of the maximum threshold is the value of the minimum threshold argument or 23 (whichever is larger) to 1,073,741,823.

The second class, Mission-critical, defines RED curves for three different DSCP values, and is thus enabled for WRED using the explicit profiles.

The third class, Top-priority, uses a RED configuration with non-default thresholds. Priority queues are rarely configured for RED. This example uses RED to illustrate the capabilities of Cisco IOS XR.

The WRED-enabled policy is finally applied to two interfaces, in the input and output direction. Support for input and output WRED on Cisco IOS XR Software results from the distributed QoS ASICs and buffer capabilities, on both ingress and egress line cards.

Monitoring WRED

```
RP/0/RSP0/CPU0:POP-PE#show policy-map interface gigabitEthernet 0/0/0/2

GigabitEthernet0/0/0/1 output: POP1-policy

Class Bulk
  Classification statistics          (packets/bytes)          (rate - kbps)
    Matched      :                962/1367076          87
    Transmitted   :                852/1265616          80
    Total Dropped :                102/89316            5
  Queueing statistics
    Queue ID      : 266
    High watermark (Unknown)
    Inst-queue-len (packets)      : 18
    Avg-queue-len  (Unknown)
    Taildropped (packets/bytes)    : 0/0
    Queue(conform) :                317/469866          30
    Queue(exceed)  :                535/795750          50
    RED random drops (packets/bytes) : 102/89316
  WRED profile for Default WRED Curve
    RED Transmitted (packets/bytes) : N/A
    RED random drops (packets/bytes) : 102/89316
    RED maxthreshold drops (packets/bytes) : N/A
<to be continued>
```

RED with Default Thresholds for the CBWFQ Class Bulk

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-19

The **show policy-map interface** command displays the configuration of all classes configured for all service policies on the specified interface. This includes all WRED parameters implementing the drop policy on the specified interface.

In the first part of the output, you see the RED statistics for the first class, Bulk, configured for RED with default thresholds.

Monitoring WRED (Cont.)

```
...
Class Mission-critical
  Classification statistics          (packets/bytes)          (rate - kbps)
    Matched      :                468/665064          101
    Transmitted   :                460/654180          110
    Total Dropped :                 7/9366             2
  Queueing statistics
    Queue ID      : 266
    High watermark (Unknown)
    Inst-queue-len (packets)      : 19
    Avg-queue-len  (Unknown)
    Taildropped (packets/bytes)    : 0/0
    Queue(conform) :                170/242940          34
    Queue(exceed)  :                290/411240          76
    RED random drops (packets/bytes) : 7/9366
  WRED profile for WRED Curve 1
    RED Transmitted (packets/bytes) : N/A
    RED random drops (packets/bytes) : 7/9366
    RED maxthreshold drops (packets/bytes) : N/A
  WRED profile for WRED Curve 2
    RED Transmitted (packets/bytes) : N/A
    RED random drops (packets/bytes) : 254/7398234
    RED maxthreshold drops (packets/bytes) : N/A
  WRED profile for WRED Curve 3
    RED Transmitted (packets/bytes) : N/A
    RED random drops (packets/bytes) : 26536/83956920
    RED maxthreshold drops (packets/bytes) : N/A
<to be continued>
```

Three Custom WRED Curves for the CBWFQ Class Mission-critical

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-20

In the second part of the output, you see the WRED statistics for the second class, Mission-critical, configured with three explicit RED curves. This class is thus configured for WRED.

Monitoring WRED (Cont.)

```
...
Class Top-priority
  Classification statistics          (packets/bytes)    (rate - kbps)
    Matched      :          962/1367076          87
    Transmitted  :          852/1265616          80
    Total Dropped :          102/89316           5
  Policing statistics          (packets/bytes)    (rate - kbps)
    Policed(conform) :          734/834859          78
    Policed(exceed)  :          54/38475           5
    Policed(violate) :           0/0              0
    Policed and dropped :           0/0
  Queueing statistics
    Queue ID      : 226
    High watermark (Unknown)
    Inst-queue-len (packets) : 18
    Avg-queue-len  (Unknown)
    Taildropped(packets/bytes) : 0/0
    Queue(conform) :          317/469866          30
    Queue(exceed)  :          535/795750          50
    RED random drops(packets/bytes) : 102/89316

    WRED profile for Default WRED Curve
    RED Transmitted (packets/bytes) : N/A
    RED random drops (packets/bytes) : 102/89316
    RED maxthreshold drops (packets/bytes) : N/A
Class class-default
...
```

RED with Custom
Thresholds for the
LLQ Class Top-priority

In the third part of the output, you see the RED statistics for the third class, Top-priority, configured for RED with manually configured thresholds.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- Congestion has undesired results for network performance because it causes tail drops
- When the TCP receiver receives a data segment, the receiver checks that data segment sequence number
- Tail drop causes TCP synchronization, starvation, and delay.
- RED is a mechanism that randomly drops packets before a queue is full, preventing congestion and avoiding tail drop.
- WRED profiles define the minimum and maximum threshold. The **show policy-map interface** command displays the QoS configuration and statistics, including WRED.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-22

Module Summary

This topic summarizes the key points that were discussed in this module.

Module Summary

- The two most common ways to manage congestion are CBWFQ and LLQ. Cisco IOS XR platforms support CBWFQ and LLQ on input and output interfaces.
- Using RED or WRED prevents congestion by randomly dropping packets. Cisco IOS XR platforms support WRED on input and output interfaces.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--5-1

Effective congestion management is the key to quality of service (QoS) in IP Next-Generation Network (NGN) environments. Low-latency traffic such as voice and video must be constantly moved to high-priority queues in order to ensure reasonable quality.

Cisco routers offer a variety of queuing algorithms to provide effective congestion management. Class-based weighted fair queuing (CBWFQ) guarantees a minimum service level to the defined traffic classes. Low latency queuing (LLQ) is specifically designed to provide the highest QoS to high-priority traffic, such as voice and video.

Cisco IOS XR routers support queuing on ingress and egress interfaces and within the fabric. The three methods are individually configured.

Congestion management is an area of concern for all networks that require a differentiated treatment of packet flows. Active queue management mechanisms address the limitations of relying solely on TCP congestion management techniques, which simply wait for queues to overflow and then drop packets to signal that congestion has occurred.

Congestion avoidance mechanisms such as random early detection (RED) and weighted RED (WRED) allow for specific packet flows to be selectively penalized and slowed by applying a traffic profile. Traffic flows are matched against this profile and transmitted or dropped, depending upon the average length of the interface output queue. In addition, RED and WRED are extremely effective at preventing global synchronization of many TCP traffic flows.

References

For additional information, refer to these resources:

- To learn more about congestion management on Cisco IOS XR Software, refer to *Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software* at this URL:
http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r3.6/qos/configuration/guide/qc36cong.html
- To learn more about fabric QoS on Cisco IOS XR Software, refer to *Configuring Fabric Quality of Service Policies and Classes on Cisco IOS XR Software* at this URL:
http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r3.6/qos/configuration/guide/qc36fab.html

Module Self-Check

Use the questions here to review what you learned in this module. The correct answers and solutions are found in the Module Self-Check Answer Key.

- Q1) What happens when the highest-priority queue becomes congested in a priority queuing algorithm? (Source: Managing Congestion)
- A) All the other queues starve.
 - B) Tail dropping focuses on the highest-priority queue.
 - C) Other queues are served on a round-robin basis.
 - D) Packets in the highest-priority queue are moved to a lower-priority queue.
- Q2) If the hardware queue is *not* full, how will the next packet be serviced by the software queue? (Source: Managing Congestion)
- A) software queue will be bypassed
 - B) software queue will enqueue the packet
 - C) software queue will expedite the packet
 - D) software queue will only meter the packet
- Q3) How does WFQ implement tail dropping? (Source: Managing Congestion)
- A) drops the last packet to arrive
 - B) drops all nonvoice packets first
 - C) drops the lowest-priority packets first
 - D) drops packets from the most aggressive flows
- Q4) Which option is the default dropping scheme for CBWFQ? (Source: Managing Congestion)
- A) RED
 - B) WRED
 - C) tail drop
 - D) class-based policing
- Q5) What does LLQ bring to CBWFQ? (Source: Managing Congestion)
- A) strict priority scheduling
 - B) alternate priority scheduling
 - C) nonpoliced queues for low-latency traffic
 - D) special voice traffic classification and dispatch
- Q6) Which type of traffic should you limit the use of the **priority** command to? (Source: Managing Congestion)
- A) critical data traffic
 - B) voice traffic
 - C) bursty traffic
 - D) video and teleconferencing ABR traffic

- Q7) What are two ways in which TCP manages congestion? (Choose two.) (Source: Implementing Congestion Avoidance)
- A) TCP uses tail drop on queues that have reached their queue limit.
 - B) TCP uses dropped packets as an indication that congestion has occurred.
 - C) TCP uses variable window sizes to reduce and increase the rates at which packets are sent.
 - D) TCP measures the average size of device queues and drops packets, linearly increasing the amount of dropped packets with the size of the queue.
- Q8) Two stations (A and B) are communicating using TCP. Station A has negotiated a TCP window size of 5, and sends five packets to station B. Station A receives three ACK messages from station B indicating ACK 3.
- Which two options best describe the status of the communication between A and B? (Choose two.) (Source: Implementing Congestion Avoidance)
- A) Station B is acknowledging receipt of packets 1, 2, and 3, but has lost packets 4 and 5.
 - B) Station A initiates a fast retransmit and immediately sends packet 3 to B.
 - C) Station B has not received packet 3.
 - D) Station B has received packets 1, 2, and 3, but not packet 4. It cannot be determined where packet 5 was received at B until packet 4 has been sent.
 - E) Station A will send packets 4 and 5 to station B upon receipt of the station B ACK.
- Q9) What are three important limitations of using a tail-drop mechanism to manage queue congestion? (Choose three.) (Source: Implementing Congestion Avoidance)
- A) Tail drop can cause many flows to synchronize, lowering overall link utilization.
 - B) Tail drop can cause starvation of fragile flows.
 - C) Tail drop increases the amount of packet buffer memory required, because queues must be full before congestion management becomes active.
 - D) Tail drop results in variable delays, which can interfere with delay-sensitive traffic flows.
- Q10) What are three advantages of active congestion management using RED? (Choose three.) (Source: Implementing Congestion Avoidance)
- A) RED uses selective packet discard to eliminate global synchronization of TCP flows.
 - B) RED avoids congestion by ensuring that interface queues never become full.
 - C) RED increases the overall utilization of links.
 - D) RED uses selective packet discard to penalize aggressive flows.
- Q11) What are the three traffic drop modes in RED? (Choose three.) (Source: Implementing Congestion Avoidance)
- A) no drop
 - B) full drop
 - C) random drop
 - D) deferred drop
- Q12) Is RED enabled by default in Cisco IOS XR Software? (Source: Implementing Congestion Avoidance)
- A) yes
 - B) no

Module Self-Check Answer Key

Q1)	A
Q2)	A
Q3)	D
Q4)	C
Q5)	A
Q6)	B
Q7)	B, C
Q8)	B, C
Q9)	A, B, D
Q10)	A, C, D
Q11)	A, B, C
Q12)	B

QoS Traffic Policing and Shaping

Overview

Traffic policing and traffic shaping are two quality of service (QoS) techniques that can be used to limit the amount of bandwidth that a specific application can use on a link.

Traffic policing and shaping are of interest especially to ISPs. The high-cost, high-traffic networks are the major assets, and as such, are the focus of all attention. Service providers often use traffic policing and shaping as a method to optimize the use of their network, sometimes by intelligently shaping or policing traffic according to importance.

This module describes the operations of traffic policing and traffic shaping, and how these techniques can be used to rate-limit traffic.

Module Objectives

Upon completing this module, you will be able to describe the concepts of traffic policing and shaping, including token bucket, dual token bucket, and dual-rate policing. This ability includes being able to meet these objectives:

- Use traffic policing and traffic shaping to condition traffic
- Configure class-based policing to rate-limit traffic
- Configure class-based shaping to rate-limit traffic

Understanding Traffic Policing and Shaping

Overview

You can use traffic policing to control the maximum rate of traffic sent or received on an interface. Traffic policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network. You can use traffic shaping to control the traffic going out an interface in order to match its flow to the speed of the remote target interface, and to ensure that the traffic conforms to policies that have been put into place for it. Traffic policing and traffic shaping differ in the way they respond to traffic violations. Policing typically drops traffic, while shaping typically queues excess traffic. This lesson describes the traffic-policing and traffic-shaping quality of service (QoS) mechanisms that are used to limit the available bandwidth to traffic classes. Because both traffic policing and traffic shaping use the token bucket metering mechanism, this lesson also describes how a token bucket works.

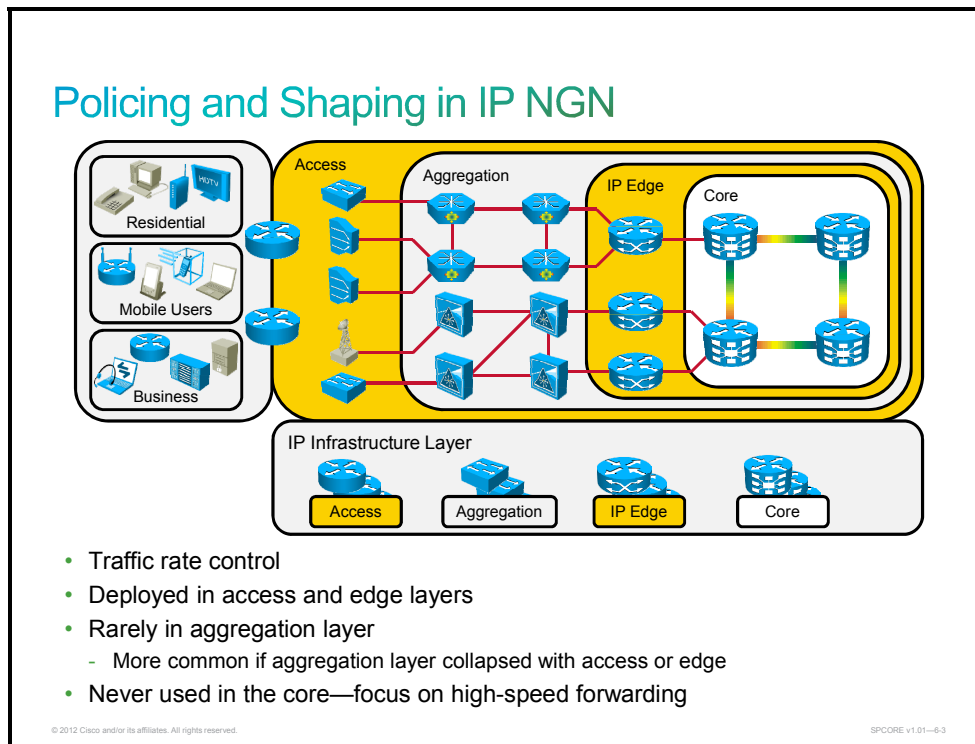
Objective

Upon completing this lesson, you will be able to explain how to use traffic policing and traffic shaping to condition traffic. This ability includes being able to meet this objective:

- Describe the purpose of traffic conditioning using traffic policing and traffic shaping
- Compare traffic policing vs shaping
- Describe the different token bucket implementations used in traffic policing
- Describe the token bucket implementation used in traffic shaping
- Describe where Traffic Policing and Shaping are typically deployed in Service Provider IP NGN
- Describe the use of Traffic Conditioning Mechanisms for Cisco Telepresence traffic

Traffic Policing and Shaping

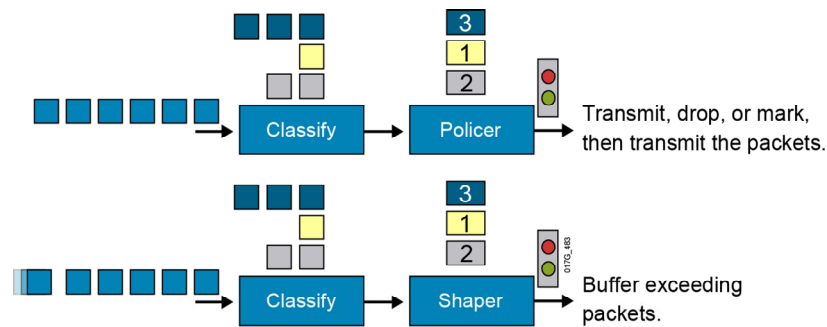
This topic describes the purpose of traffic conditioning using traffic policing and traffic shaping.



Both traffic shaping and policing mechanisms are traffic-conditioning mechanisms that are used in a network to control the traffic rate. Both mechanisms use classification so that they can differentiate traffic. They both measure the rate of traffic and compare that rate to the configured traffic-shaping or traffic-policing policy.

Traffic shaping and policing are deployed in the access and IP edge layers of the next-generation networks (NGNs). Traffic rate control implementations in the aggregation layer are rare, and exist mainly in situations where the aggregation layer is collapsed with the access or IP edge. Traffic policing and shaping are never used in the core because the main purpose of the core is high-speed forwarding through a highly available core infrastructure.

Traffic Policing and Shaping Overview



- These mechanisms must classify packets before policing or shaping the traffic rate.
- Traffic shaping queues excess packets to stay within the desired traffic rate.
- Traffic policing typically drops or marks excess traffic to stay within a traffic rate limit.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-4

The difference between traffic shaping and policing can be described in terms of their implementation:

- Traffic shaping buffers excessive traffic so that the traffic stays within the desired rate. With traffic shaping, traffic bursts are smoothed out by queuing the excess traffic to produce a steadier flow of data. Reducing traffic bursts helps reduce congestion in the network.
- Traffic policing drops excess traffic in order to control traffic flow within specified rate limits. Traffic policing does not introduce any delay to traffic that conforms to traffic policies. Traffic policing can cause more TCP retransmissions, because traffic in excess of specified limits is dropped.

Traffic-policing mechanisms such as class-based policing have marking capabilities in addition to rate-limiting capabilities. Instead of dropping the excess traffic, traffic policing can alternatively mark and then send the excess traffic. This allows the excess traffic to be re-marked with a lower priority before the excess traffic is sent out. Traffic shapers, on the other hand, do not re-mark traffic—they only delay excess traffic bursts to conform to a specified rate.

Policing and Shaping Use Cases

- Use policing to:
 - Limit access to resources when high-speed access is used but not desired (sub-rate access)
 - Limit the traffic rate of certain applications or traffic classes
 - Mark down (recolor) exceeding traffic at Layer 2 or Layer 3
- Use shaping to:
 - Prevent and manage congestion in networks, where asymmetric bandwidths are used along the traffic path
 - Regulate the sending traffic rate to match the subscribed (committed) rate

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6.6

Traffic policing is typically used to satisfy one of these requirements:

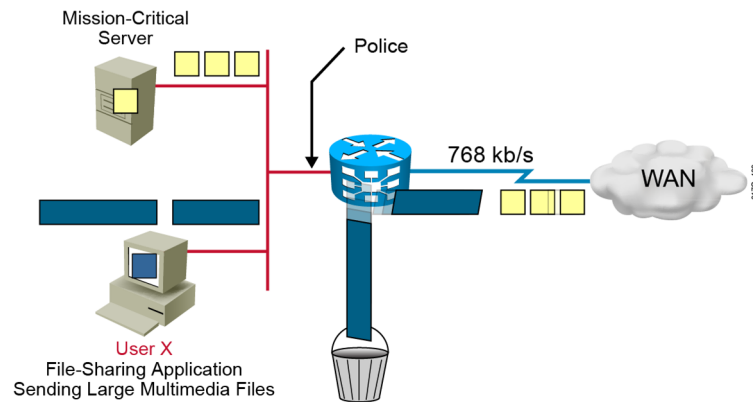
- Limiting the access rate on an interface when high-speed physical infrastructure is used in transport. Rate limiting is typically used by service providers to offer customers sub-rate access. For example, a customer may have a 1-Gb/s connection to the service provider but pay only for a 100-Mb/s access rate. The service provider can rate-limit the customer traffic to 100 Mb/s.
- Engineering bandwidth so that traffic rates of certain applications or classes of traffic follow a specified traffic rate policy—for example, rate-limiting traffic from file-sharing applications to 64 kb/s maximum.
- Re-marking excess traffic with a lower priority at Layer 2 and Layer 3, or both, before sending the excess traffic out. Cisco class-based traffic policing can be configured to mark packets at both Layer 2 and Layer 3. For example, excess traffic can be re-marked to a lower differentiated services code point (DSCP) value and also have the Frame Relay discard eligible (DE) bit set before the packet is sent out.

Traffic shaping, on the other hand, is commonly used for the following:

- To prevent and manage congestion in networks where asymmetric bandwidths are used along the traffic path. If shaping is not used, buffering can occur at the slow (usually the remote) end, which can lead to queuing, causing delays, and overflow, causing drops.
- To prevent dropping of noncompliant traffic by the service provider by avoiding bursts above the subscribed (committed) rate. This allows the customer to keep local control of traffic regulation.

Traffic Policing Example

- Rate-limit file-sharing application traffic to 1 Mb/s.
- Do not rate-limit traffic from the mission-critical server.



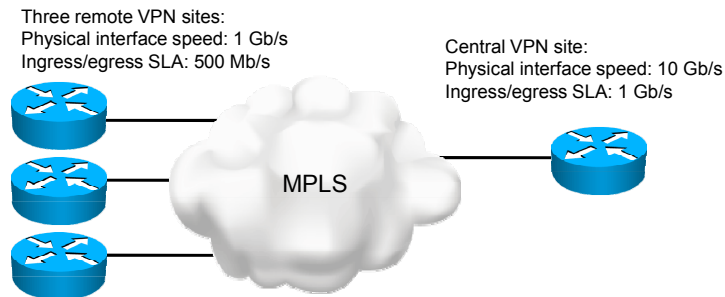
© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6.6

You can use traffic policing to divide the shared resource (the upstream WAN link) between many flows. In this example, the router internal LAN interface has an input traffic-policing policy applied to it, in which the mission-critical server traffic rate is not rate-limited, but the User X file-sharing application traffic is rate-limited to 1 Mb/s. All file-sharing application traffic from User X that exceeds the rate limit of 1 Mb/s will be dropped.

Traffic Shaping Example

- Central to remote site **speed mismatch**
- Remote to central site **oversubscription**
- Both situations result in buffering and in delayed or dropped packets



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6.7

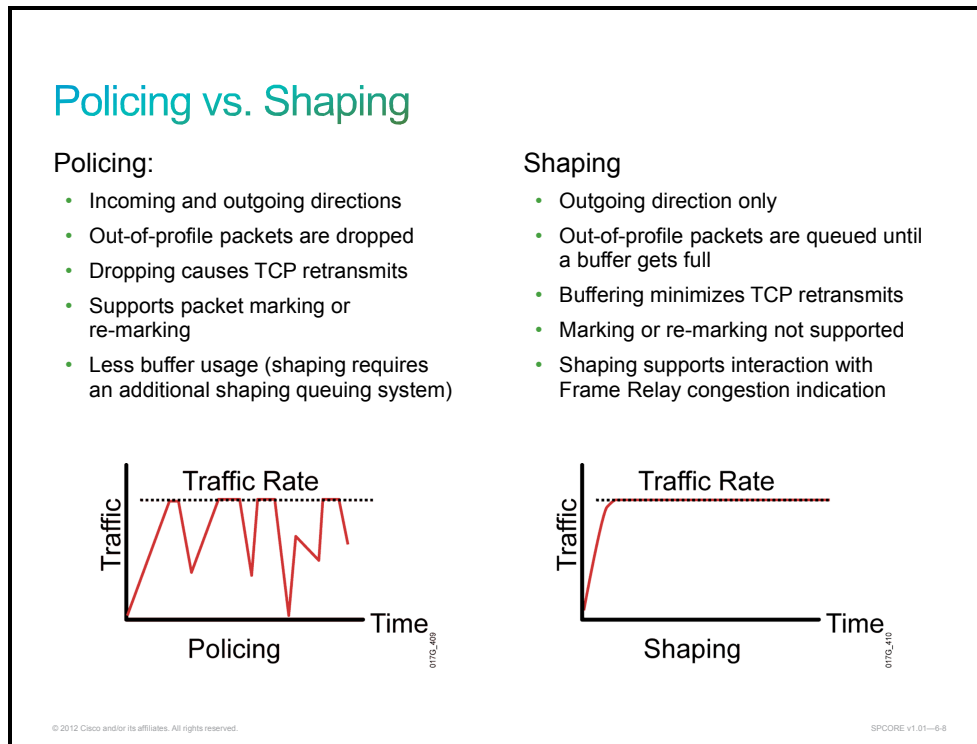
Traffic-shaping tools limit the transmit rate from a source by queuing the excess traffic. This limit is typically a value lower than the line rate of the transmitting interface. Traffic shaping can be used to account for speed mismatches that are common in nonbroadcast multiaccess (NBMA) networks or VPNs consisting of multiple sites.

In the figure, these two types of speed mismatches are shown:

- The central site can have a higher-speed link than the remote site. You can deploy traffic shaping at the central-site router to shape the traffic rate out of the central-site router to match the link speed of the remote site. For example, the central router can shape the outgoing traffic rate going to a specific remote site to 500 Mb/s to match that remote-site ingress service level agreement (SLA). At each remote-site router, traffic shaping is also implemented to shape the remote-site outgoing traffic rate to 500 Mb/s to match the committed information rate (CIR).
- The aggregate link speed of all the remote sites can be higher than the central-site SLA, thereby over-subscribing the central-site SLA. In this case, you can configure the remote-site routers for traffic shaping to avoid oversubscription at the central site. For example, you can configure the bottom two remote-site routers to shape the outgoing traffic rate to 250 Mb/s to avoid the central-site router from being oversubscribed.

Comparing Traffic Policing vs. Shaping

This topic compares traffic policing vs. shaping.



Shaping queues excess traffic by holding packets inside a shaping queue. Use traffic shaping to shape the outbound traffic flow when the outbound traffic rate is higher than a configured shape rate. Traffic shaping smoothes traffic by storing traffic above the configured rate in a shaping queue. Therefore, shaping increases buffer utilization on a router and causes unpredictable packet delays.

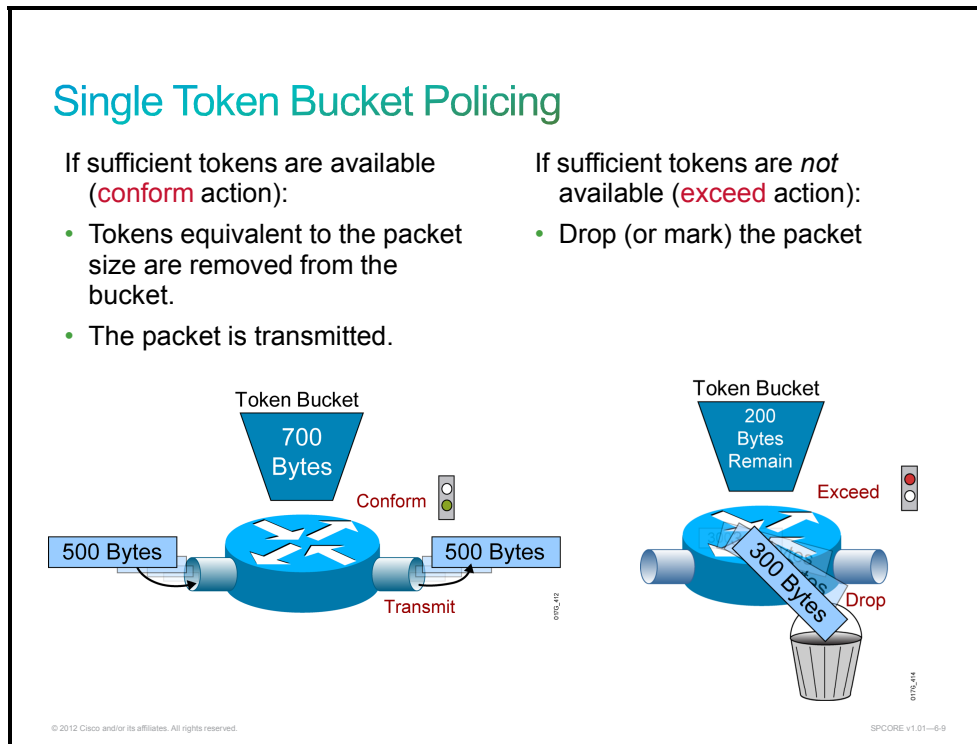
You can apply policing to either the inbound or outbound direction, while you can apply shaping only in the outbound direction. Policing drops nonconforming traffic instead of queuing the traffic like shaping. Policing also supports marking of traffic. Traffic policing is more efficient in terms of memory utilization than traffic shaping because no additional queuing of packets is needed.

Both traffic policing and traffic shaping ensure that traffic does not exceed a bandwidth limit, but each mechanism has different impacts on the traffic:

- Policing drops packets more often, generally causing more retransmissions of connection-oriented protocols such as TCP.
- Shaping adds variable delay to traffic, possibly causing jitter.

Traffic Policing Token Bucket Implementations

This topic describes the different token bucket implementations used in traffic policing.



The token bucket is a mathematical model that is used by routers and switches to regulate traffic flow. The model has two basic components:

- **Tokens:** Each token represents permission to send a fixed number of bits into the network. Tokens are put into a token bucket at a certain rate.
- **Token bucket:** A token bucket has the capacity to hold a specified number of tokens. Each incoming packet, if forwarded, takes tokens from the bucket, representing the packet size. If the bucket fills to capacity, newly arriving tokens are discarded. Discarded tokens are not available to future packets. If there are not enough tokens in the token bucket to send the packet, the traffic conditioning mechanisms may take these actions:
 - Wait for enough tokens to accumulate in the bucket (traffic shaping)
 - Discard the packet (traffic policing)

Using a single token bucket model, the measured traffic rate can conform or exceed the specified traffic rate. The measured traffic rate is conforming if there are enough tokens in the single token bucket to transmit the traffic. The measured traffic rate is exceeding if there are not enough tokens in the single token bucket to transmit the traffic.

The figure shows a single token bucket traffic-policing implementation. The current capacity of tokens accumulated in the token bucket is 700 bytes. When a 500-byte packet arrives at the interface, its size is compared to the token bucket capacity (in bytes). The 500-byte packet conforms to the rate limit (500 bytes < 700 bytes). The packet is forwarded, and 500 bytes worth of tokens are taken out of the token bucket, leaving 200 bytes worth of tokens for the next packet.

When the next 300-byte packet arrives immediately after the first packet, and no new tokens have been added to the bucket (which is done periodically), the packet exceeds the rate limit. The current packet size (300 bytes) is greater than the current capacity of the token bucket (200

bytes), and the exceed action is performed. The exceed action can be to drop the packet, or to re-mark the packet and then transmit it out.

Example: Token Bucket as a Coin Bank

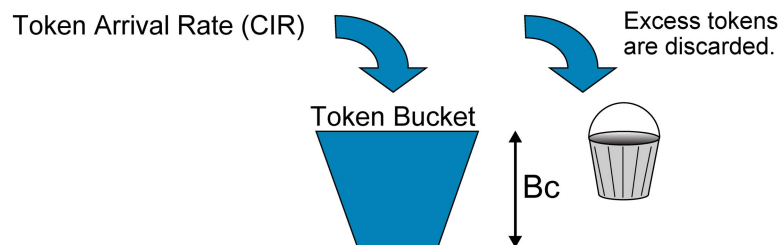
Think of a token bucket as a coin bank. Every day you can insert a coin into the bank (the token bucket). At any given time, you can only spend what you have saved up in the bank. On the average, if your saving rate is \$1 per day, your long-term average spending rate will be \$1 per day if you constantly spend what you saved. However, if you do not spend any money on a given day, you can build up your savings in the bank to the maximum that the bank can hold. For example, if the size of the bank is limited to \$5, and if you save and do not spend for five straight days, the bank will contain \$5. When the bank fills to its capacity, you will not be able to put any more money in it. Then, at any time, you can spend up to \$5 (bursting above the long-term average rate of \$1 per day).

Using this example, having \$2 in the bank and trying to spend \$1 is considered conforming, because you are not spending more than you have saved.

Having \$2 in the bank and trying to spend \$3 is considered exceeding, because you are trying to spend more than you have saved.

Single Token Bucket Class-Based Policing

- Bc is the normal burst size.
- Tc is the time interval.
- CIR is the committed information rate.
- $CIR = Bc / Tc$



Token bucket operations rely on parameters such as the CIR, the normal burst size (Bc), and the committed time interval (Tc). The mathematical relationship between CIR, Bc, and Tc is as follows:

$$CIR (bps) = Bc (bits) / Tc (sec)$$

With traffic policing, new tokens are added into the token bucket based on the interpacket arrival rate and the CIR. Every time a packet is policed, new tokens are added back into the token bucket. The number of tokens added back into the token bucket is calculated as follows:

$$(Current\ packet\ arrival\ time - previous\ packet\ arrival\ time) * CIR$$

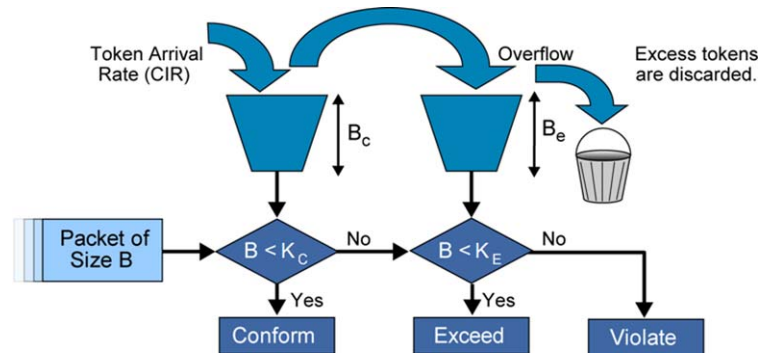
An amount (Bc) of tokens is forwarded without constraint in every time interval (Tc). For example, if 8,000,000 bits (Bc) worth of tokens are placed in the bucket every 250 milliseconds (Tc), the router can steadily transmit 8,000,000 bits every 250 milliseconds if traffic constantly arrives at the router.

$$CIR (normal\ burst\ rate) = 8000000\ bits\ (Bc) / 0.25\ seconds\ (Tc) = 32\ Mbps$$

Without any excess bursting capability, if the token bucket fills to capacity (Bc of tokens), the token bucket will overflow and newly arriving tokens will be discarded. Using the example, in which the CIR is 32 Mb/s (Bc = 8,000,000 bits and Tc = 0.25 seconds), the maximum traffic rate can never exceed a hard rate limit of 32 Mb/s.

Dual Token Bucket (Single Rate) Class-Based Policing

- B_e : Excess burst size
- K_c : Tokens available in B_c bucket
- K_e : Tokens available in B_e bucket
- The return value is **conform** or **exceed** or **violate**.



You can configure class-based traffic policing to support excess bursting capability. With excess bursting, after the first token bucket is filled to B_c , extra (excess) tokens can be accumulated in a second token bucket. Excess burst (B_e) is the maximum amount of excess traffic over and above B_c that can be sent during the time interval *after* a period of inactivity. With a single rate-metering mechanism, the second token bucket with a maximum size of B_e fills at the same rate (CIR) as the first token bucket. If the second token bucket fills up to capacity, no more tokens can be accumulated and the excess tokens are discarded.

When using a dual token bucket model, the measured traffic rate can be as follows:

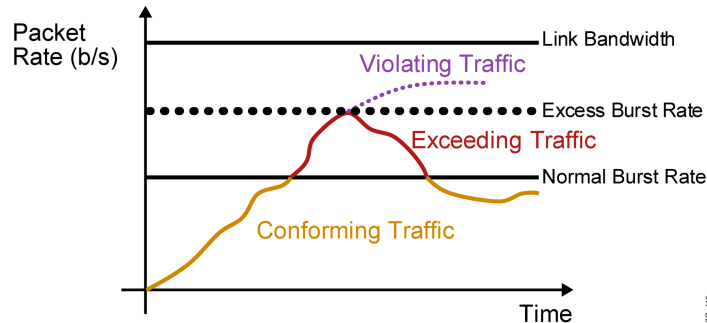
- **Conforming:** There are enough tokens in the first token bucket with a maximum size of B_c .
- **Exceeding:** There are not enough tokens in the first token bucket, but there are enough tokens in the second token bucket with a maximum size of B_e .
- **Violating:** There are not enough tokens in the first or second token bucket.

With dual token bucket traffic policing, the typical actions performed are sending all conforming traffic, re-marking (to a lower priority), sending all exceeding traffic, and dropping all violating traffic. The main benefit of using a dual token bucket method is the ability to distinguish between traffic that exceeds the B_c but not the B_e . This enables a different policy to be applied to packets in the B_e category.

To use the coin bank example, think of the CIR as the savings rate (\$1 per day). B_c is how much you can save in the first coin bank (\$5). T_c is the interval at which you put money into the coin bank (one day). B_e is how much you can save in the second coin bank once the first bank is filled up. If $B_e = \$5$, then you can spend up to a maximum of \$10 ($B_c + B_e$) once both banks are filled up.

Dual Token Bucket (Single Rate) Class-Based Policing (Cont.)

- Traffic is **conforming**, **exceeding**, or **violating**



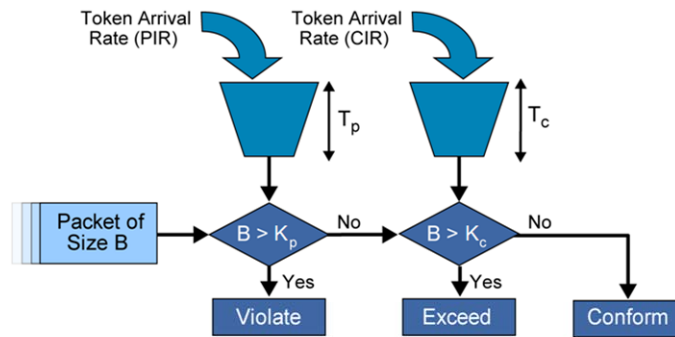
© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-12

Using a dual token bucket model allows traffic exceeding the normal burst rate (CIR) to be metered as exceeding, and traffic that exceeds the excess burst rate to be metered as violating traffic. Different actions can then be applied to the conforming, exceeding, and violating traffic.

Dual-Rate Class-Based Policing

- K_c : Tokens available in CIR bucket
- K_p : Tokens available in PIR bucket
- Enforce traffic policing according to two separate rates:
 - Committed information rate
 - Peak information rate



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-13

With dual-rate metering, traffic rate can be enforced according to two separate rates: CIR and peak information rate (PIR). Before this feature was available, you could meter traffic using a single rate based on the CIR with single or dual buckets. Dual-rate metering supports a higher level of bandwidth management and supports a sustained excess rate based on the PIR.

With dual-rate metering, the PIR token bucket fills at a rate based on the packet arrival rate, and the configured PIR and the CIR token bucket fills at a rate based on the packet arrival rate and the configured CIR.

When a packet arrives, the PIR token bucket is first checked to see if there are enough tokens in the PIR token bucket to send the packet. The violating condition occurs if there are not enough tokens in the PIR token bucket to transmit the packet. If there are enough tokens in the PIR token bucket to send the packet, then the CIR token bucket is checked. The exceeding condition occurs if there are enough tokens in the PIR token bucket to transmit the packet but not enough tokens in the CIR token bucket to transmit the packet. The conforming condition occurs if there are enough tokens in the CIR bucket to transmit the packet.

Dual-rate metering is often configured on interfaces at the edge of a network to police the rate of traffic entering or leaving the network. In the most common configurations, traffic that conforms is sent and traffic that exceeds is sent with a decreased priority, and traffic that violates is dropped. Users can change these configuration options to suit their network needs.

Dual-Rate Class-Based Policing (Cont.)

Dual-rate policer marks packets as conforming, exceeding, or violating a specified rate.

- If $(B > K_p)$, the packet is marked as violating the specified rate.
- If $(B > K_c)$, the packet is marked as exceeding the specified rate, and the PIR token bucket is updated:
 - $K_p = K_p - B$.
- If the packet is marked as conforming to the specified rate, both token buckets are updated:
 - $K_p = K_p - B$
 - $K_c = K_c - B$

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-14

In addition to rate limiting, traffic policing using dual-rate metering allows marking of traffic according to whether the packet conforms, exceeds, or violates a specified rate.

The token bucket algorithm provides users with three different actions for each packet: a conform action, an exceed action, and an optional violate action. Traffic entering the interface with dual-rate policing configured is placed into one of these categories. Within these three categories, users can decide packet treatments. For example, a user may configure a policing policy as follows:

- Conforming packets are transmitted. Packets that exceed may be transmitted with a decreased priority, while packets that violate are dropped.
- The violating condition occurs if there are not enough tokens in the PIR bucket to transmit the packet.
- The exceeding condition occurs if there are enough tokens in the PIR bucket to transmit the packet but not enough tokens in the CIR bucket to transmit the packet. In this case, the packet can be transmitted and the PIR bucket is updated to $K_p - B$ remaining tokens, where K_p is the size of the PIR bucket and B is the size of the packet to be transmitted.

The conforming condition occurs if there are enough tokens in the CIR bucket to transmit the packet. In this case, the packets are transmitted and both buckets (K_c and K_p) are decremented to $K_p - B$ and to $K_c - B$, respectively, where K_c is the size of the CIR bucket, K_p is the size of the PIR bucket, and B is the size of the packet to be transmitted.

Example: Dual-Rate Token Bucket as a Coin Bank

Using a dual-rate token bucket is like using two coin banks, each with a different savings rate. However, you can take out money from only one of the banks at a time.

For example, you can save \$10 per day into the first coin bank (PIR = peak spending rate = \$10 per day) and then at the same time, you can save \$5 per day in the second bank (CIR = normal average spending rate = \$5 per day). However, the maximum amount you can spend is \$10 per day, not \$15 per day, because you can take out money from only one bank at a time.

In this example, after one day of savings, your first coin bank (PIR bucket) will contain \$10 and your second coin bank (CIR bucket) will contain \$5. The three different spending cases are examined here to show how dual-rate metering operates, using the coin bank example:

- **Case 1:** If you try to spend \$11 at once, then you are violating ($K_p < B$) your peak spending rate of \$10 per day. In this case, you will not be allowed to spend the \$11 because \$11 is greater than the \$10 you have in the first coin bank (PIR bucket). Remember, you can only take out money from one of the banks at a time.
- **Case 2:** If you try to spend \$9 at once, then you are exceeding ($K_p > B > K_c$) your normal average spending rate of \$5 per day. In this case, you will be allowed to spend the \$9 and just the first coin bank (PIR bucket) will be decremented to $\$10 - \$9$, or \$1.

After spending \$9, the maximum amount that you can continue to spend on that day is decremented to \$1.
- **Case 3:** If you try to spend \$4, then you are conforming ($K_p > B$ and $K_c > B$) to your normal average spending rate of \$5 per day. In this case, you will be allowed to spend the \$4, and both coin banks (PIR and CIR bucket) will be updated.

The first coin bank (PIR bucket) will be updated to $\$10 - \$4 = \$6$, and the second bank (CIR bucket) will be updated to $\$5 - \$4 = \$1$.

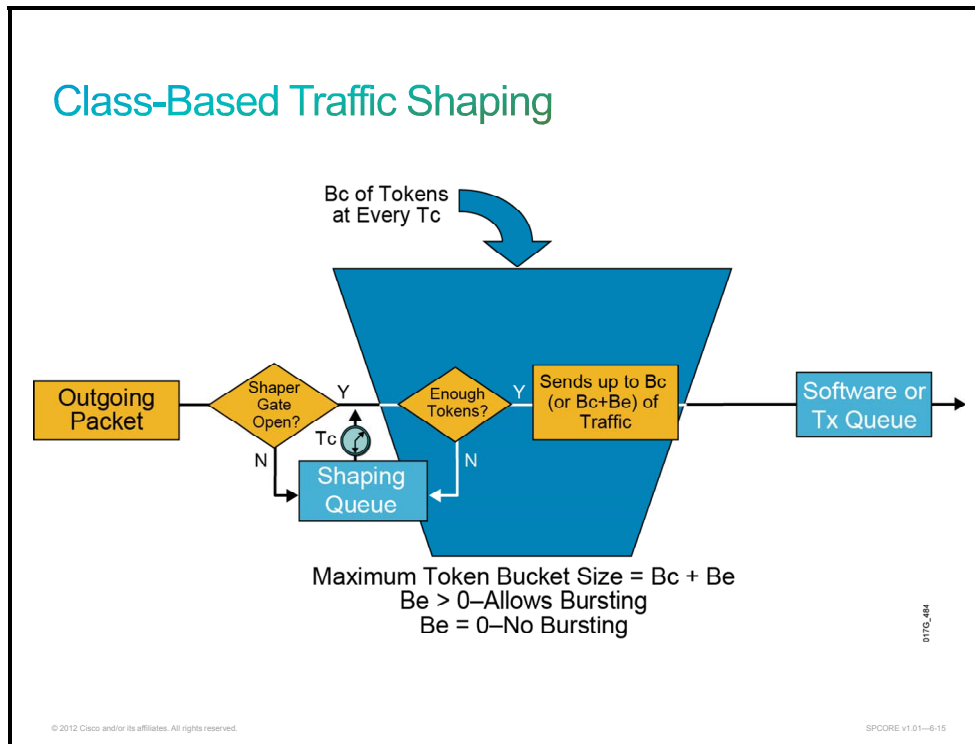
Both coin banks are updated because after spending \$4, the maximum amount you can continue to spend on that day is decremented to \$6, and the normal spending rate for that same day is decremented to \$1.

Therefore, after spending \$4, the following will occur:

- If you spend \$7 on that same day, then you will be violating your peak spending rate for that day. In this case, you will not be allowed to spend the \$7 because \$7 is greater than the \$6 that you have in the first coin bank (PIR bucket).
- If you spend \$5 on that same day, then you will be exceeding your normal average spending rate for that day. In this case, you will be allowed to spend the \$5 and the first coin bank (PIR bucket) will be decremented to $\$6 - \$5$, or \$1.
- If you spend \$0.50 on that same day, then you will be conforming to your normal average spending rate for that day. In this case, you will be allowed to spend the \$0.50, and both coin banks (PIR and CIR bucket) will be updated. The first coin bank (PIR bucket) will be updated to $\$6 - \$0.50 = \$5.50$, and the second coin bank (CIR bucket) will be updated to $\$1 - \$0.50 = \$0.50$.

Traffic Shaping Token Bucket Implementation

This topic describes the token bucket implementation used in traffic shaping.



Class-based traffic shaping only applies for outbound traffic.

Class-based traffic shaping uses the basic token bucket mechanism, in which Bc of tokens are added at every Tc time interval. The maximum size of the token bucket is Bc + Be. You can think of the traffic shaper operation like opening and closing of a transmit gate at every Tc interval. If the shaper gate is opened, the shaper checks to see if there are enough tokens in the token bucket to send the packet. If there are enough tokens, the packet is immediately forwarded. If there are *not* enough tokens, the packet is queued in the shaping queue until the next Tc interval. If the gate is closed, the packet is queued behind other packets in the shaping queue.

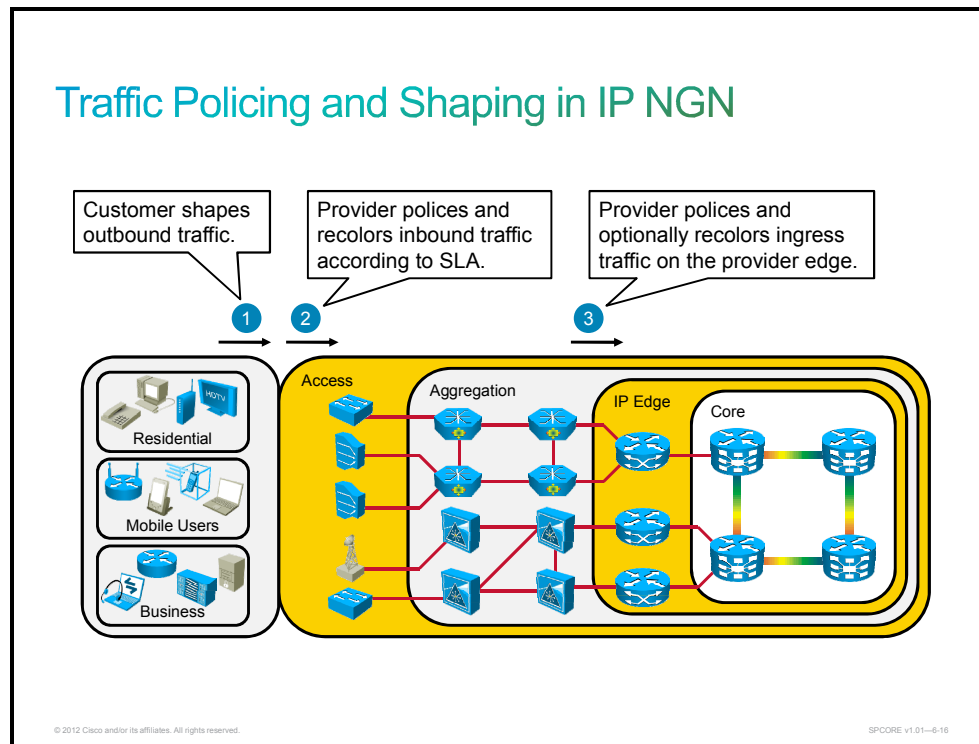
For example, on a 128 kb/s link, if the CIR is 96 kb/s, the Bc is 12 KB, the Be is 0, and the Tc = 0.125 seconds, then during each Tc (125 ms) interval, the traffic shaper gate opens and up to 12 KB can be sent. To send 12 KB over a 128-kb/s line will only take 91.25 ms. Therefore the router will, on average, be sending at three-quarters of the line rate (128 kb/s * 3/4 = 96 kb/s).

For example, on a 1 Gb/s link, if the CIR is 100 Mb/s, the Bc is 12.5 MB, the Be is 0, and the Tc = 0.125 seconds, then during each Tc (125 ms) interval, the traffic shaper gate opens and up to 12.5 MB can be sent. To send 12.5 Mb over a 1 Gb/s line will only take 12.5 ms. Therefore the router will, on average, be sending at 10% of the transmission capacity (1 Gb/s * 10% = 100 Mb/s, or 125 ms * 10% = 12.5 ms).

Traffic shaping also includes the ability to send more than Bc of traffic in some time intervals after a period of inactivity. This extra number of bits in excess to the Bc is called Be.

Traffic Policing and Shaping in IP NGN

This topic describes where Traffic Policing and Shaping are typically deployed in Service Provider IP NGN.



In an NGN network, traffic shaping is most commonly implemented at the customer site, and the traffic policing is deployed in the service provider network.

The customer shapes outbound traffic going to the service provider network to make sure that it conforms to the contractual rate and as such is not dropped by the service provider. Traffic shaping is more interesting to customers than policing, because they often prefer to delay the excess traffic rather than to drop it.

The service provider polices inbound customer traffic according to the SLA. This protects the network resources from unaccounted excess traffic. This policing is deployed close to the customer access point, typically in the access layer.

Depending on the environment, the service provider may police traffic at the IP edge, before it is forwarded into the core. This policing protects the resources in the core from excess traffic. The difference between policing at the access and the edge layers is that intra-point-of-presence (POP) traffic—traffic originated from and destined to systems attached to the same POP—never makes it to the IP edge provider edge (PE) router, and therefore is not subject to the rate-limiting at the edge.

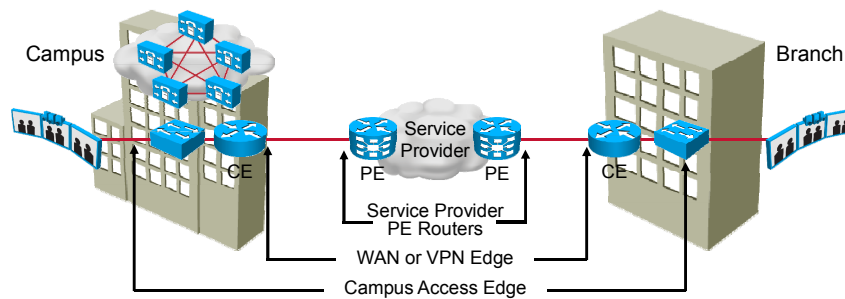
Sometimes, the service provider may also shape traffic going to the customer. This technique prevents oversubscription of the access link bandwidth and is typically implemented only if specifically requested by the customer.

Traffic Policing and Shaping with Cisco Telepresence

This topic describes the use of Traffic Conditioning Mechanisms for Cisco Telepresence traffic.

Traffic Policing and Shaping with Cisco TelePresence

- Policing Cisco TelePresence traffic should generally be avoided whenever possible. Exceptions include the following:
 - At the WAN or VPN edge
 - At the service provider PE routers, in the ingress direction
 - At the campus access edge
- It is recommended to avoid shaping Cisco TelePresence flows unless absolutely necessary.



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-17

Although some exceptions exist, policing Cisco TelePresence traffic should generally be avoided whenever possible. Cisco TelePresence is highly sensitive to drops (with a 0.05 percent packet loss target), so policing its traffic rates could be extremely detrimental to its flows and could ultimately ruin the high level of user experience that it is intended to deliver.

However, there are three places where Cisco TelePresence traffic may be legitimately policed.

- **At the WAN or VPN edge:** The first place where Cisco TelePresence traffic may be legitimately policed automatically occurs if Cisco TelePresence is assigned to a low-latency queue at the WAN or VPN edge. This is because any traffic that is assigned to a low-latency queue is automatically policed by an implicit policer set to the exact value as the LLQ rate. For example, if Cisco TelePresence is assigned an LLQ of 15 Mb/s, it is also implicitly policed by the LLQ algorithm to exactly 15 Mb/s, and any excess traffic is dropped.
- **At the service provider PE routers, in the ingress direction:** The second most common place that Cisco TelePresence is likely to be policed in the network is at the service provider PE routers, in the ingress direction. Service providers must police traffic classes, especially real-time traffic classes, to enforce service contracts and prevent possible oversubscription on their networks and thus ensure service-level agreements.
- **At the campus access edge:** The third (and optional) place where policing Cisco TelePresence may prove beneficial in the network is at the campus access edge. You can deploy access-edge policers for security purposes to mitigate the damage caused by the potential abuse of trusted switch ports.

It is recommended to avoid shaping Cisco TelePresence flows unless absolutely necessary because of the QoS objective of shapers themselves. Specifically, the role of shapers is to delay traffic bursts above a certain rate and to smooth out flows to fall within contracted rates. Sometimes this is done to ensure that traffic rates are within the CIR of a carrier. Other times, shaping is performed to protect other data classes from a bursty class.

Shapers temporarily buffer traffic bursts above a given rate, and therefore introduce jitter as well as absolute delay. Because Cisco TelePresence is so sensitive to delay and especially jitter, shaping is not recommended for Cisco TelePresence flows.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- Traffic shaping and policing are deployed in the access and IP edge layers of the next-generation networks (NGNs)
- Traffic shaping queues excess packets to stay within the contractual rate, while traffic policing typically drops excess traffic to stay within the limit
- Token bucket operations rely on parameters such as the CIR, the normal burst size (Bc), and the committed time interval (Tc)
- Class-based traffic shaping only applies for outbound traffic
- In an NGN network, traffic shaping is most commonly implemented at the customer site, and the traffic policing is deployed in the service provider network
- Policing Cisco TelePresence traffic should generally be avoided whenever possible

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-18

Implementing Traffic Policing

Overview

Traffic policing is implemented on Cisco IOS XR, IOS XE, and IOS routers using the Modular QoS CLI (MQC).

Cisco IOS XR routers introduced a new feature, Local Packet Transport Service (LPTS), that provides software architecture to deliver locally destined traffic to the correct node on the router and provides security against overwhelming the router resources with excessive traffic.

This lesson describes the configuration tasks that are used to implement class-based traffic policing to rate-limit certain traffic classes. It also explains how to configure the LPTS feature.

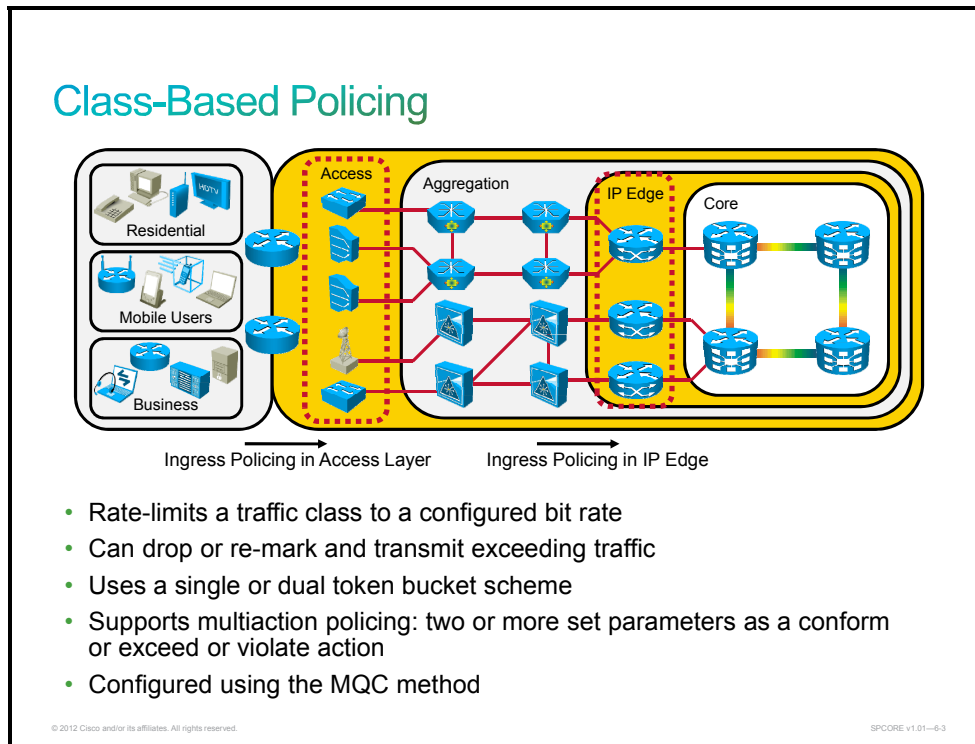
Objectives

Upon completing this lesson, you will be able to implement class-based policing. You will be able to meet these objectives:

- Describe class-based policing
- Explain a Single-Rate, Single Token Bucket Policing Configuration
- Explain a Single-Rate, Dual Token Bucket Policing Configuration
- Explain a Multiaction Policing Configuration
- Explain a Dual Rate Policing Configuration
- Explain a Percentage Based Policing Configuration
- Explains a Hierarchical Policing Configuration
- Describe the show command used to Monitoring Class-Based Policing operations
- Explain a Cisco Access Switch Policing Configuration
- Explain a Cisco Access Switch Aggregate Policer Configuration
- Describe LPTS, a feature available on Cisco IOS XR routers

Class-Based Policing

This topic describes class-based policing.



The class-based policing feature performs these functions:

- Limits the input or output transmission rate of a class of traffic that is based on user-defined criteria
- Marks packets by setting different Layer 2 or Layer 3 markers, or both

The two most common places for deploying policing in IP next generation networks are the access and the IP edge layer. Although policing can be applied in both directions, inbound and outbound, service providers often use this method to limit the amount of traffic allowed into the network and align its profile with the respective SLAs.

You can implement class-based policing using a single or double token bucket method as the metering mechanism. When the violate action option is not specified in the **police** MQC command, the single token bucket algorithm is engaged. When the violate action option is specified in the **police** MQC command, the dual token bucket algorithm is engaged.

A dual token bucket algorithm allows traffic to do the following:

- Conform to the rate limit when the traffic is within the average bit rate
- Exceed the rate limit when the traffic exceeds the average bit rate but does not exceed the allowed excess burst
- Violate the rate limit when the traffic exceeds both the average rate and the excess burst

Depending on whether the current packet conforms with, exceeds, or violates the rate limit, one or more actions can be taken, such as transmit, drop, or set a specific value in the packet.

Multiaction policing is a mechanism that can apply more than one action to a packet; for example, setting the DSCP as well as the quality of service (QoS) group on the exceeding packets.

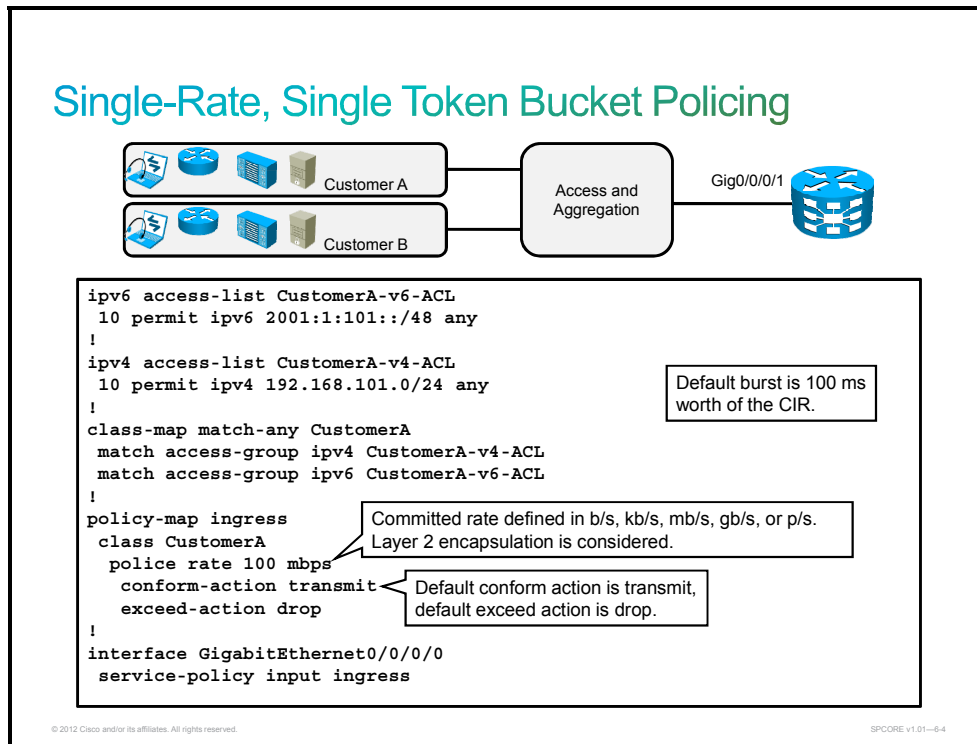
Class-based policing also supports single- or dual-rate metering. With the dual-rate policer, traffic policing can be enforced according to two separate rates: committed information rate (CIR) and peak information rate (PIR).

Cisco class-based policing mechanisms conform to these two differentiated services (DiffServ) RFCs:

- **RFC 2697, “*A Single Rate Three Color Marker*”:** The single-rate three-color marker meters an IP packet stream and marks its packets to one of three states: conform, exceed, or violate. Marking is based on a CIR and two associated burst sizes, a committed burst (Bc) size and an excess burst (Be) size. A packet is marked conform if it does not exceed the Bc, marked exceed if it does exceed the Bc but not the Be, and marked violate otherwise.
- **RFC 2698, “*A Two Rate Three Color Marker*”:** The two-rate three-color marker meters an IP packet stream and marks its packets to one of three states: conform, exceed, or violate. A packet is marked violate if it exceeds the PIR. Otherwise a packet is marked either exceed or conform, depending on whether it exceeds or does not exceed the CIR. This process is useful, for example, for ingress policing of a service where a peak rate needs to be enforced separately from a committed rate.

Single-Rate, Single Token Bucket Policing Configuration

This topic explains a Single-Rate, Single Token Bucket Policing Configuration.



The class-based policing configuration example shows two configured traffic classes that are based on the traffic source IP address. Traffic originated by customer A is policed to a fixed bandwidth with no excess burst capability using a single token bucket. Conforming traffic is sent as is, and exceeding traffic is dropped. In this case, the traffic from customer A is policed to a rate of 100 Mb/s. The committed rate can be defined as a value in bits per second, kilobits per second, megabits per second, gigabits per second, or packets per second.

Configured values take into account the Layer 2 encapsulation applied to traffic. This applies to both ingress and egress policing. For Ethernet transmission, the encapsulation is considered to be 14 bytes, whereas for IEEE 802.1Q, the encapsulation is 18 bytes.

A similar policy, not shown here, can be applied to traffic from Customer B, by adding another class in the ingress policy map.

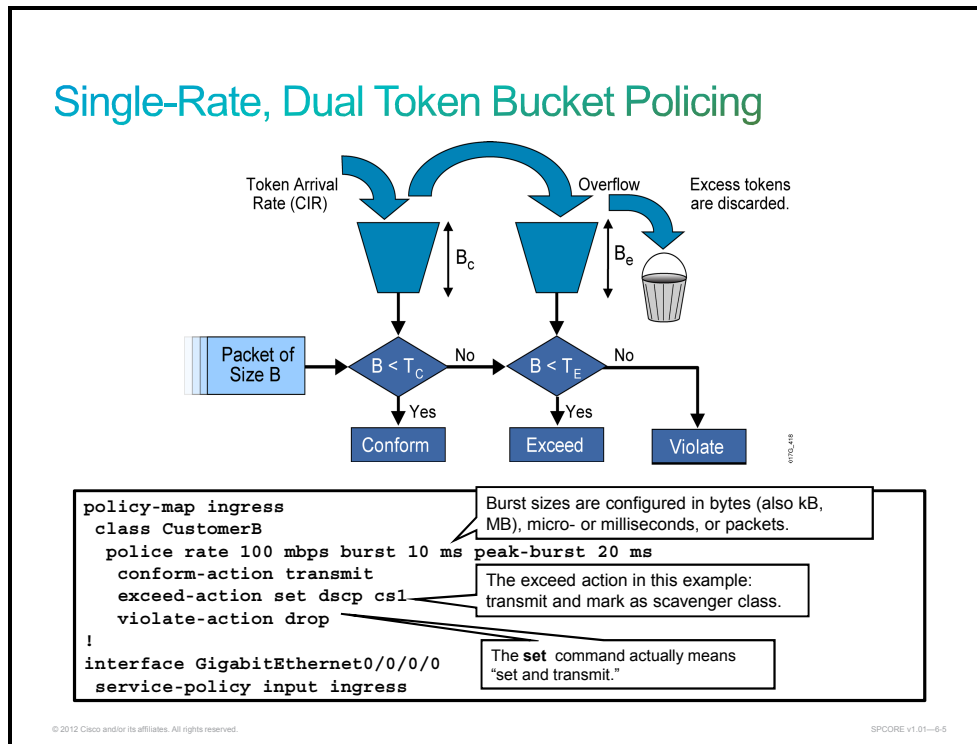
Because the violate action is not specified, this example will use a single token bucket scheme and no excess bursting will be allowed.

In this example, the burst value is not specifically configured. Therefore it is automatically set to the default value of 100 ms worth of the CIR value. For example, if a CIR value of 1,000,000 kb/s is entered, the burst value is calculated to be 12,500,000 bytes. However, the maximum burst value supported is 2,097,120 bytes.

The default conform action is transmit, the default exceed action is drop. If no action is configured, the default action is taken.

Single-Rate, Dual Token Bucket Policing Configuration

This topic explains a Single-Rate, Dual Token Bucket Policing Configuration.



The class-based policing configuration example assumes that the traffic class CustomerB is defined using any of the discussed methods, such as by matching the traffic using IPv4 or IPv6 access groups, upstream MAC address, DSCP values, or any other parameters.

Traffic from the customer B is policed to a fixed bandwidth with excess burst capability using a dual token bucket, by configuring a violate action. Conforming traffic will be sent as is, and exceeding traffic will be marked as scavenger traffic using the DSCP cs1 value, and transmitted. All violating traffic will be dropped.

If at least one set action is defined per category, the transmit action is implicit. You do not have to explicitly configure it. In this example, the exceed action is to set the DSCP, and implicitly, transmit the traffic.

In this example, because the violate action is specified, a dual token bucket scheme with excess bursting will be used.

Both the committed burst and the peak burst are explicitly set. Cisco IOS XR enables you to configure the bursts using their size (in bytes, kilobytes, or megabytes), duration (micro- or milliseconds), or packet count.

The burst size and duration are directly related. The burst size is equal to the burst duration multiplied with the link speed.

When you define custom burst sizes, for optimum performance use these formulas to determine the burst values:

$$B_c = CIR \text{ b/s} * (1 \text{ byte} / 8 \text{ bits}) * 1.5 \text{ seconds}$$

$$B_e = 2 * B_c$$

For example, if CIR = 2,000,000 b/s, the calculated burst value is $2,000,000 * (1/8) * 1.5 = 375,000$ bytes. Set the peak-burst value according to the formula $\text{peak-burst} = 2 * \text{burst}$.

Multiaction Policing Configuration

This topic explains a Multiaction Policing Configuration.

Multiaction Policing

Transmit and set a maximum of two:

DSCP	Precedence	FR-DE	Discard-Class
CoS	QoS Group	DEI	MPLS EXP

IOS XR Software

```
policy-map ingress
class CustomerC
  police rate 100 mbps burst 10 ms peak-burst 20 ms
  conform-action set dscp af11
  conform-action set mpls experimental topmost 4
  exceed-action set dscp cs1
  violate-action drop
```

Option to set **dscp, cos, precedence, discard-class, mpls exp topmost, mpls exp imposition, dei, fr-de.**

Maximum of Two Actions per Category

IOS and IOS-XE Software

```
policy-map ingress
class CustomerC
  police rate 100000000 bps burst 1250000 bytes peak-burst 2500000 bytes
  conform-action set-dscp-transmit af11
  conform-action set-mpls-exp-topmost-transmit 4
  exceed-action set-dscp-transmit cs1
  violate-action drop
```

Drop cannot be combined with any other action.

© 2012 Cisco and/or its affiliates. All rights reserved. SPCORE v1.01--6.6

This class-based policing configuration is an example of a multiaction class-based policing.

In this case, the traffic from customer C is policed to 100 Mb/s. All conforming traffic will be marked with the DSCP value of AF11, the top-most experimental field in the Multiprotocol Label Switching (MPLS) header will be set to 4, and the traffic will be transmitted. All exceeding traffic will be marked as scavenger class (DSCP set to CS1) and transmitted. All violating traffic will be dropped.

You can configure a maximum of two actions per category.

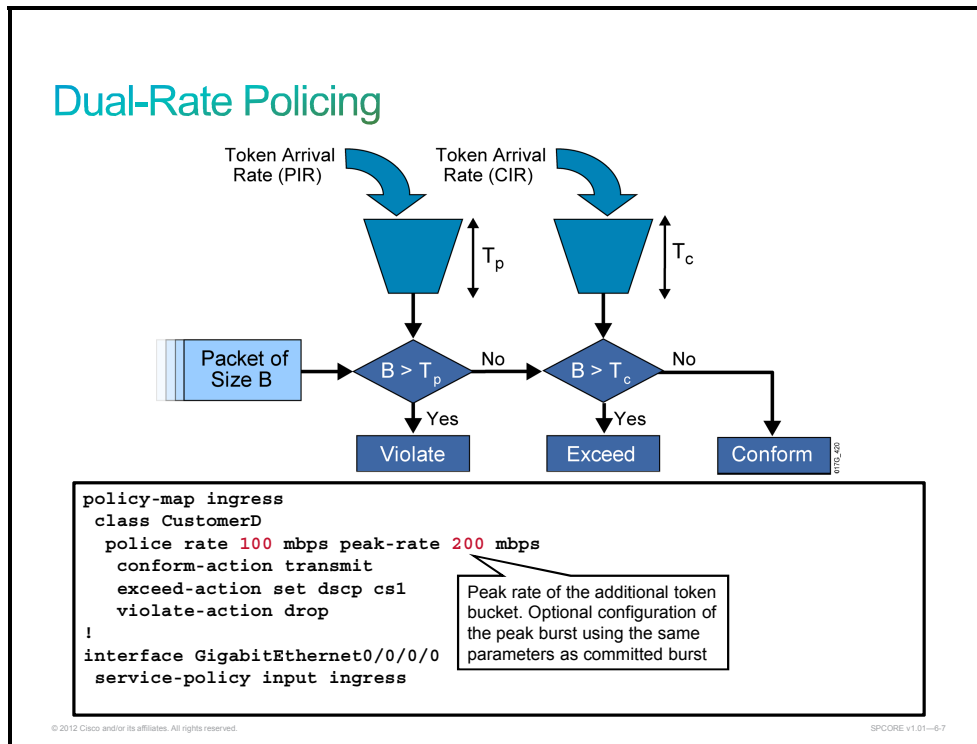
Depending on whether the current packet conforms with, exceeds, or violates the rate limit, one or more actions can be taken by class-based policing:

- **Transmit:** The packet is transmitted.
- **Drop:** The packet is dropped.
- **Set IP precedence or DSCP value:** The IP precedence or differentiated services code point (DSCP) bits in the packet header are rewritten. The packet is then transmitted. This action can be used to either color (set precedence) or recolor (modify existing packet precedence) the packet.
- **Set QoS group and transmit:** The QoS group is set and the packet is forwarded. Because the QoS group is only significant within the local router (that is, the QoS group is not transmitted outside the router), the QoS group setting is used in later QoS mechanisms, such as class-based weighted fair queuing (CBWFQ), and performed in the same router on an outgoing interface.
- **Set MPLS experimental (EXP) bits:** The MPLS EXP bits are set. You can set the experimental bits in the top-most, or in the imposed MPLS header. The packet is then transmitted. These are usually used to signal QoS parameters in an MPLS cloud.

- **Set Frame Relay discard eligible (DE) bit:** The Frame Relay DE bit is set in the Layer 2 (Frame Relay) header and the packet is transmitted. This setting can be used to mark excessive or violating traffic (which should be dropped with preference on Layer 2 switches) at the edge of a Frame Relay network.
- **Set discard class:** The discard class is an integer from 0 to 7. You can mark IP or MPLS packets with this identifier. Like the QoS group identifiers, the discard class has only local significance on a node.
- **Set drop eligible indicator (DEI):** This parameter is present in 802.1ad and 802.1ah frames. The value of the DEI bit can be 0 or 1, with 1 signifying a higher drop probability.
- **Set Layer 2 class of service (CoS):** The IEEE 802.1Q CoS value ranges from 0 to 7. The optional inner keyword specifies the inner CoS in, for example, a queue-in-queue (QinQ) configuration.

Dual Rate Policing Configuration

This topic explains a Dual Rate Policing Configuration.



With dual-rate policing, traffic policing can be enforced according to two separate rates: CIR and PIR. The use of these two rates can be specified, along with their corresponding values, by using two keywords, **rate** and **peak-rate**, in the **police** command. The peak rate is configured using the same options as the committed rate, as a value in bits per second, kilobits per second, megabits per second, gigabits per second, or packets per second. The configuration approach for the committed rate and the peak rate is independent from one another.

The policer uses an incremental step size of 64 kb/s. The configured value is rounded down to the nearest 64 kb/s. The value shown in the output of the running-configuration shows the configured value as entered by the user. A police rate minimum of 8 p/s and a granularity of 8 p/s is supported.

The **burst** and **peak-burst** keywords and their associated arguments (*conform-burst* and *peak-burst*, respectively) are optional. If the bursts are not configured, the operating system computes the default values. Like the conform burst, the peak burst can be explicitly defined by the number of bytes, the time duration, or number of packets. The configuration approach for the burst and peak burst is independent from one another.

Percentage Based Policing Configuration

This topic explains a Percentage Based Policing Configuration.

Percentage-Based Policing

IOS XR Software

```
policy-map ingress
class CustomerE
  police rate percent 10 peak-rate percent 20
  conform-action transmit
  exceed-action set dscp cs1
  violate-action drop
!
```

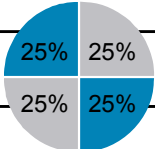
Percent values of the committed and peak rates configured independently of one another

```
interface GigabitEthernet0/0/0/0
  service-policy input ingress
```

IOS and IOS-XE Software

```
policy-map ingress
class CustomerE
  police rate percent 10 peak-rate percent 20
  conform-action transmit
  exceed-action set-dscp-transmit cs1
  violate-action drop
!
```

```
interface GigabitEthernet0/1
  service-policy input ingress
```



The percentage-based policing feature provides the ability to configure traffic policing based on a percentage of bandwidth available on the interface. Configuring traffic policing and traffic shaping in this manner enables the use of the same policy map for multiple interfaces with differing amounts of bandwidth.

Without this feature, traffic policing would have to be configured on the basis of a user-specified amount of bandwidth available on the interface. Policy maps would be configured on the basis of that specific amount of bandwidth, and separate policy maps would be required for each interface with a different bandwidth.

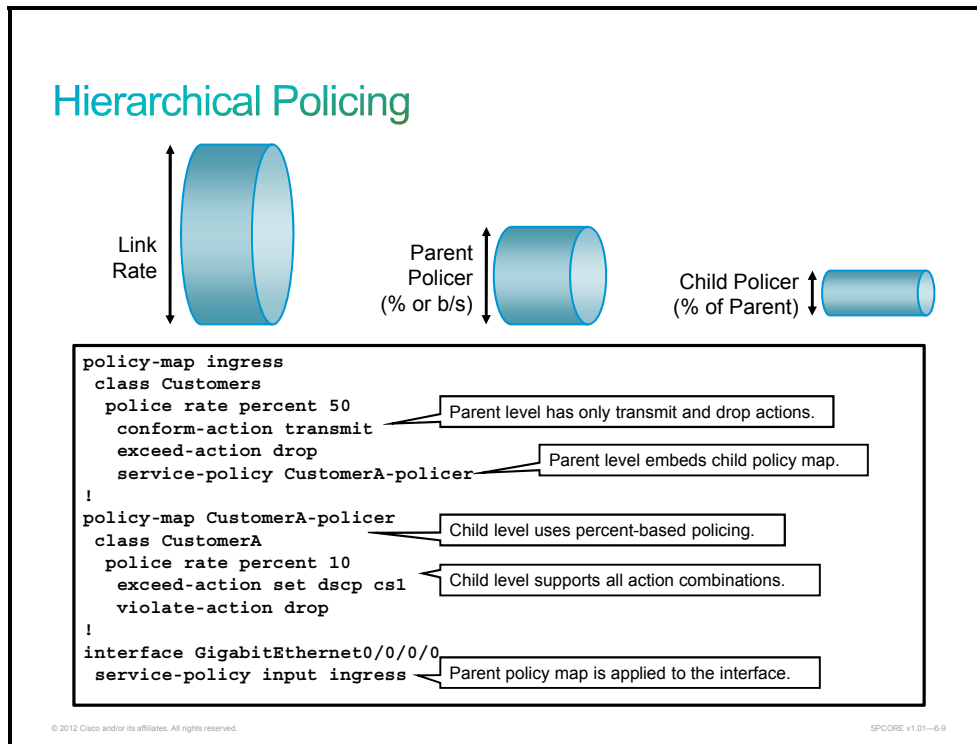
A mixed configuration model is also permitted. In other words, you can configure one rate using the absolute value, and the other using the percent figure. It is, however, strongly discouraged, as it overly complicates the configuration.

The **percent** keyword has this significance:

- For a one-level policy, the **percent** keyword specifies the CIR as a percentage of the link rate. For example, the command `police rate percent 35` configures the CIR as 35 percent of the link rate.
- For a two-level policy, in the parent policy, the **percent** keyword specifies the parent CIR as a percentage of the link rate. In the child policy, the **percent** keyword specifies the child CIR as a percentage of the maximum policing or shaping rate of the parent. If traffic policing or shaping is not configured on the parent, the parent inherits the interface policing or shaping rate. Two-level policies are described next.

Hierarchical Policing Configuration

This topic explains a Hierarchical Policing Configuration.



In hierarchical policing, the routers use a two-level policy map. The parent and child policies have class maps containing policing statements.

The parent-level policer can be configured using percent or absolutely defined rate values. The parent level can perform only transmit and drop actions. It includes one or more child policies that are executed for conforming traffic.

The child level must use a policing rate that defined as a percentage and supports any combination of actions. The order of the actions within the hierarchical policy map is from child to parent. This is with the exception of the queuing action (shape), which is discussed in a different lesson.

Hierarchical policing supports both ingress and egress service policy direction.

Monitoring Class-Based Policing Operations

This topic describes the show command used to Monitoring Class-Based Policing operations.

Monitoring Class-Based Policing

```
RP/0/RSP0/CPU0:PE1# show policy-map interface gigabit 0/0/0/0

GigabitEthernet0/0/0/0 input: ingress

Class CustomerA
  Classification statistics      (packets/bytes)      (rate - kbps)
    Matched                    :      1633/192694             48
    Transmitted                 : N/A
    Total Dropped               : N/A
  Policing statistics          (packets/bytes)      (rate - kbps)
    Policed(conform)            :      1303/153754             39
    Policed(exceed)             :      315/37170              9
    Policed(violate)            :      15/1770                0
    Policed and dropped         :      15/1770
Class class-default
  Classification statistics      (packets/bytes)      (rate - kbps)
    Matched                    :           0/0                0
    Transmitted                 : N/A
    Total Dropped               : N/A
GigabitEthernet0/0/0/0 direction output: Service Policy not installed
```

Other command options are **list**, **target**, **type**, **pmap-name**, and **shared-policy-instance**.

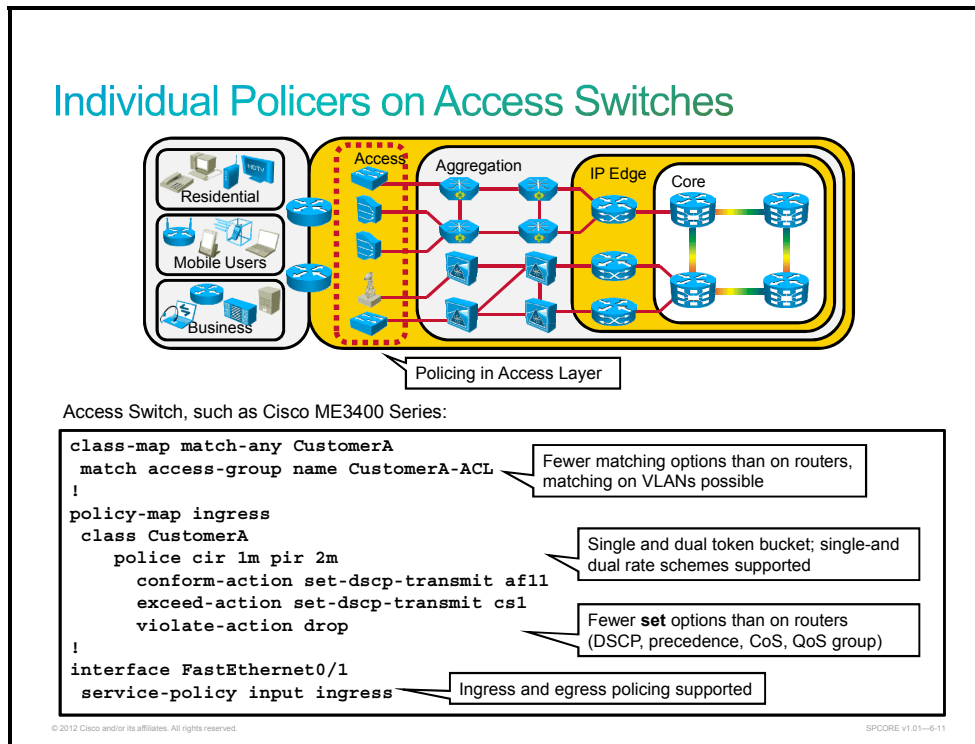
Conform, exceed, and violate counts

You can verify the policing operations using the **show policy-map** command. This command has many options, such as **interface**, **list**, **target**, **type**, **pmap-name**, and **shared-policy-instance**. This figure presents the output of the **show policy-map** combined with the interface to which the service policy has been applied.

The output shows that the ingress policy map has been assigned in the input direction, and that no policy is installed for output traffic. The command provides statistics on the conforming, exceeding, and violating traffic, including the packet and byte counts and resulting rates.

Cisco Access Switches Policing Configuration

This topic explains a Cisco Access Switch Policing Configuration.



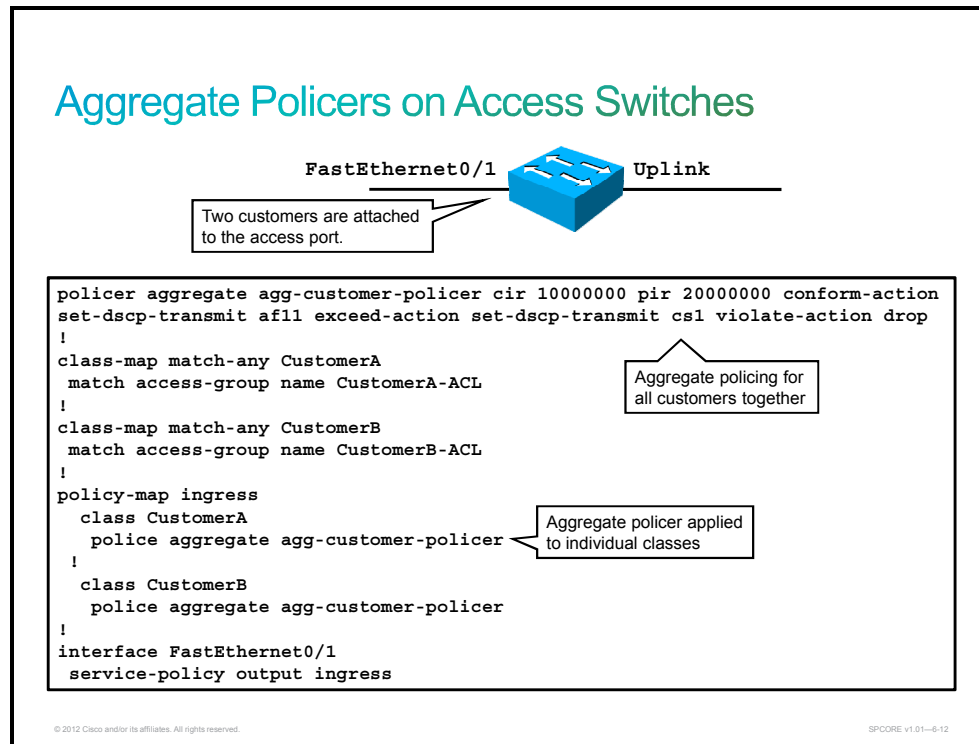
Cisco access switches, such as the Cisco ME3400 Series, provide traffic policing features that can be deployed in the access layer. They support two types of class-based traffic policing: individual policers and aggregate policers.

This figure presents a scenario with individual policers. The configuration resembles the approach taken on Cisco routers, especially the Cisco IOS and IOS XE platforms. Class maps define traffic, policy maps apply actions to traffic classes, and the service policy applies the policy to an interface, either to input or output traffic.

Access switches offer fewer options for matching traffic and setting packet markers than Cisco routers do. Although the exact set depends on the platform, IOS release and image type, you may match based on CoS, DSCP, precedence, IP access groups, and VLANs. The policers support single and dual token buckets, as well as single-and dual-rate schemes. Hierarchical policing is not supported. The switches offer fewer **set** options than Cisco routers and include markers, such as DSCP, IP precedence, CoS, and QoS group.

Cisco Access Switches Aggregate Policer Configuration

This topic explains a Cisco Access Switch Aggregate Policer Configuration.



An aggregate policer differs from an individual policer because it is shared by multiple traffic classes within a policy map.

You can use the **policer aggregate** global configuration command to set a policer for all traffic received on a physical interface. When you configure an aggregate policer, you can configure multiple conform and exceed actions, as well as specific burst sizes. If you do not specify burst size (Bc), the system calculates an appropriate burst size value. The calculated value is appropriate for most applications. These policing parameters are applied to all traffic classes shared by the aggregate policer.

Aggregate policing applies only to input policy maps.

This example illustrates how to configure multiple conform and exceed actions simultaneously for an aggregate policer as parameters in the **policer aggregate** global configuration command. After you configure the aggregate policer, you create a policy map and an associated class map, associate the policy map with the aggregate policer, and apply the service policy to a port. After you configure the policy map and policing actions, attach the policy to an ingress port by using the **service-policy** interface configuration command.

Note Only one policy map can use any specific aggregate policer. Aggregate policing cannot be used to aggregate traffic streams across multiple interfaces. It can be used only to aggregate traffic streams across multiple classes in a policy map attached to an interface and aggregate streams across VLANs on a port in a per-port, per-VLAN policy map.

Local Packet Transport Services

This topic describes LPTS, a feature available on Cisco IOS XR routers.

Local Packet Transport Services Overview

- Software architecture to deliver locally destined traffic to the correct control plane process
- Security against overwhelming the router resources with excessive traffic
- Policing flows of locally destined traffic to a sustainable value
- Available only in Cisco IOS XR Software, not IOS Software
- Complex forwarding decisions
 - Ingress line card identifies the destination stack
 - DRP
 - NSR may force replication to active/standby RPs
- Components:
 - Port arbitrator
 - Flow managers
 - Tables such as the IFIB that route packets to the correct route processor or line card
- LPTS is automatically enabled with default policing values.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-13

LPTS provides software architecture to deliver locally destined traffic to the correct control plane process on the router and provides security against overwhelming the router resources with excessive traffic. LPTS achieves security by policing flows of locally destined traffic to a value that can be easily sustained by the CPU capabilities of the platform.

LPTS can be thought of as a security measure for an IOS XR router by taking preemptive measures for traffic flows destined to the router. LPTS is an IOS XR feature and is not available in existing Cisco IOS Software releases.

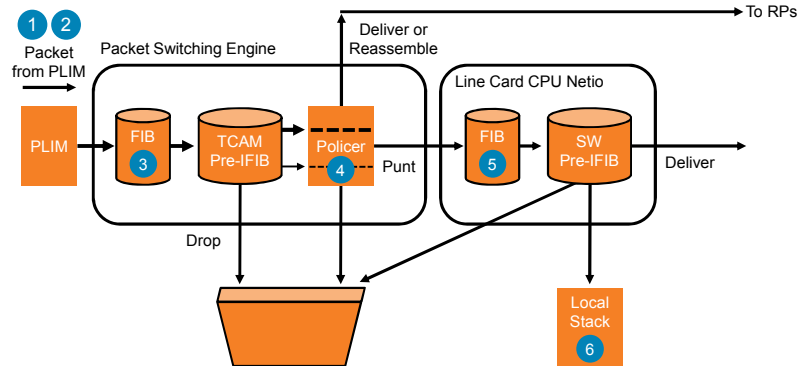
Cisco IOS XR Software runs on platforms with a distributed architecture where the control plane and the forwarding planes are decoupled from one another. A Cisco IOS XR router may deliver different traffic types to different nodes within the router. With support for distributed route processors (DRPs), a line card receiving a control plane packet makes complex decisions to identify the node to which the packet should be delivered. Furthermore, nonstop routing (NSR) might require a control packet be replicated both to an active and a standby RP.

LPTS uses two components to accomplish this task: the port arbitrator and flow managers. The port arbitrator and flow managers are processes that maintain the tables that describe packet flows for a logical router, known as the internal forwarding information base (IFIB). The IFIB is used to route received packets to the correct RP or line card for processing.

LPTS is automatically enabled and does not require custom configuration. LPTS policing can be tuned to reflect specific requirements. Even without explicit rate-limiting configuration, LPTS **show** commands are provided for monitoring the activity of LPTS flow managers and the port arbitrator.

LPTS Process

1. PLIM receives the frame.
2. PLIM extracts the Layer 3 packet and passes it to the forwarding ASIC.
3. The FIB lookup determines if the packet is destined to a local node.
4. The LPTS port arbitrator and flow manager populate the pIFIB table.
5. The pIFIB lookup returns a match and assigns an FGID.
6. The FGID helps deliver the packet to the destination stack.



The figure represents the LPTS operation on the CRS platform. A similar approach exists on other Cisco IOS XR Software platforms. LPTS uses this process to identify local packets and deliver them to the appropriate stack:

1. The Physical Layer Interface Module (PLIM) receives the frame.
2. On receiving the packet and performing the necessary Layer 1 and 2 checks, the PLIM extracts the Layer 3 packet and passes it to the forwarding ASIC (or the Packet Switching Engine [PSE], as it is commonly called).
3. The Layer 3 forwarding engine does a forwarding information base (FIB) lookup and determines whether the packet is a locally destined for_us packet.
4. The LPTS infrastructure maintains tables in the ternary content addressable memory (TCAM) of the line card and also on the RP for handling the for_us packets. The table on the RP, called the IFIB, is a detailed list of all possible flows of traffic types that can be destined to the router. A smaller table called the pre-IFIB, a subset of IFIB, exists on the line card. The pIFIB lists flows of critical traffic. These tables are populated by a set of processes known as an LPTS port arbitrator (lpts_pa) and LPTS flow manager (lpts_fm). A process called pifibm_server runs on the line card and is responsible for programming hardware for the policing values for different flows. To qualify for a match in the pIFIB, the incoming packet must exactly match the pIFIB table entry in a single lookup.
5. If the pIFIB lookup returns a full match, the packet then is assigned a fabric group identifier (FGID) allocated by the lpts_pa process. The FGID serves as an identifier that helps a packet traverse the path through the various ASICs on the switch fabric, to be delivered to the FabricQ ASIC on the destination node. From there, the packet finds its way to the primary/standby RP, DRP, or the line card CPU. The destination node could also be an RP, a DRP, or the line card CPU of the line card on which the packet was received. If a line card pIFIB entry results in a partial match, the incoming packet is referred to the IFIB maintained on the RP.
6. The CPU on the RP, DRP, and line card run the software processes that decapsulate the packets and deliver them to the correct stack.

LPTS Local Traffic Types

Received Traffic Type	Processed in Packet Switching Engine	Processed by Line Card CPU	Processed by Route Processor
Transit packets, IP options	LPTS policed	X	-
Transit packets, IP option Router Alert	LPTS policed	X	X
Packets that require ARP resolution	LPTS policed	X	-
ICMP	LPTS policed	X	-
Management traffic (SSH, SNMP, XML)	LPTS policed	-	X
Management traffic (NetFlow, Cisco Discovery Protocol)	LPTS policed	X	-
Routing (BGP, OSPF, ISIS, etc.)	LPTS policed	-	X
Multicast control traffic (PIM, HSRP, etc.)	LPTS policed	-	X
First packet of multicast stream	LPTS policed	X	-
Broadcasts	LPTS policed	X	X
Traffic needing fragmentation	LPTS policed	X	-
MPLS traffic needing fragmentation	LPTS policed	X	-
Layer 2 packets (keepalives and similar)	LPTS policed	X	-

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-15

Although routers are generally used for forwarding packets, there are scenarios in which the traffic may be locally destined. The LPTS mechanism must identify the locally destined traffic. It may fall into these categories:

- All IPv4, IPv6, and MPLS traffic related to routing protocols or the control plane such as MPLS Label Distribution Protocol (LDP) or Resource Reservation Protocol (RSVP). The control plane computations for protocols are done on the RP. Therefore, whenever routing or MPLS control plane traffic is received on a line card interface, it needs to be delivered to the RP of the router.
- MPLS packets with the Router Alert label
- IPv4, IPv6, or MPLS packets with a Time to Live (TTL) value of less than 2
- IPv4 or IPv6 packets with options
- IP packets requiring fragmentation or reassembly
- Layer 2 keepalives
- Address Resolution Protocol (ARP) packets
- Internet Control Message Protocol (ICMP) message generation and response

The table provides a list of the various locally destined traffic types, along with an indication about LPTS handling, and whether the traffic is processed by the line card CPU, the RP, or both.

Monitoring LPTS Policing: Default State

```
RP/0/RSP0/CPU0:Pl# show lpts flows brief
+ - Additional delivery destination; L - Local interest; P - In Pre-IFIB
```

L3	L4	VRF-ID	Interface	Location	LP Local-Address,Port	Remote-Address,Port
IPv4	ICMP	*	any	(drop)	LP any,ECHO	any
IPv4	ICMP	*	any	(drop)	LP any,TSTAMP	any
IPv4	ICMP	*	any	(drop)	LP any,MASKREQ	any
IPv4	UDP	*	any	(drop)	LP any	any
IPv4	UDP	*	any	0/0/CPU0	P any 128.9.0.0/16	
IPv6	ICMP6	*	any	(drop)	LP any,ECHOREQ	any
IPv6	ICMP6	*	any	(drop)	LP any,NDTRSLCT	any

<output truncated>

```
RP/0/RSP0/CPU0:PE1# show lpts pifib brief
* - Any VRF; I - Local Interest;
X - Drop; R - Reassemble;
```

Type	VRF-ID	L4	Interface	Deliver	Local-Address,Port	Remote-Address,Port
ISIS	default	-	Lo0	0/RSP0/CPU0	-	-
ISIS	*	-	any	0/RSP0/CPU0	-	-
IPv4_frag	*	any	any	R	any	any
IPv4	default	IGMP	any	0/RSP0/CPU0	any	any
IPv4	default	TCP	any	0/RSP0/CPU0	10.0.1.1,65438	10.0.2.1,179
IPv4	default	TCP	any	0/RSP0/CPU0	10.0.1.1,646	10.0.2.1,31759
IPv4	default	TCP	any	0/RSP0/CPU0	any,179	10.0.2.1
IPv4	default	TCP	any	0/RSP0/CPU0	any,646	10.0.2.1

<output truncated>

Other options include **ifib**, **port-arbitrator**, and VRF-related information

Since LPTS is enabled by default, you can verify its operations without any prior configuration. There is a wide range of **show lpts** commands. This figure presents two examples.

The **show lpts flows** command is used to display LPTS flows, which are aggregations of identical binding requests from multiple clients and are used to program the LPTS IFIB and pIFIB.

The **show lpts pifib** command with the **brief** keyword performs the following functions:

- Displays entries of all or part of a pIFIB
- Displays a short description of each entry in the LPTS pIFIB, optionally displaying packet counts for each entry

These statistics are used only for packets that are processed by a line card, RP, or DRP.

Configuring LPTS Policing

```
RP/0/RSP0/CPU0:PE1(config)#
lpts pifib hardware police
flow fragment rate 1000
flow icmp local rate 500
flow icmp application rate 1000
flow icmp default rate 1000
flow ssh default rate 200
flow http known rate 150
flow http default rate 300
!
lpts pifib hardware police location 0/0/CPU0
```

Traffic Type Example	Description	Default Policer (p/s)
Fragment	Fragmented packets	2500
ICMP local	ICMP packets with local interest	1500
ICMP application	ICMP packets with interest to applications	1500
ICMP default	Other ICMP packets	1500
SSH default	Packets from new or newly established SSH sessions	300
HTTP known	Packets from known HTTP sessions	200
HTTP default	Packets from new or newly established HTTP sessions	400

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-17

This figure illustrates how to configure LPTS hardware policing. The configuration involves two steps:

- Step 1** Fine-tune the policing parameters. This setting is performed in LPTS pIFIB hardware police configuration mode. You can change the default rate-limit thresholds for the defined traffic types. This example shows how to modify the rate limits for several flows. The table provides a description of the flows and the default rate-limit values. For a description of all traffic flows, search the Cisco.com documentation.
- Step 2** Set the hardware policing node. The **lpts pifib hardware police location *node-id*** command specifies the designated node. The *node-id* argument is entered in the rack/slot/module notation.

Monitoring LPTS Policing

```
RP/0/RSP0/CPU0:P1# show lpts pifib hardware police location 0/0/CPU0
```

```
Node 0/0/CPU0:
```

```
Burst = 100ms for all flow types
```

Dropped packets
caused by exceeded
packet rate

FlowType	Policer	Type	Cur. Rate	Def. Rate	Accepted	Dropped
unconfigured-default	100	Static	2500	2500	0	0
Fragment	101	Global	2000	2500	556801	145
OSPF-mc-known	102	Static	2000	2000	102270	0
OSPF-mc-default	103	Static	1500	1500	227991	0
OSPF-uc-known	104	Static	2000	2000	20	0
OSPF-uc-default	105	Static	1000	1000	16	0
ISIS-known	143	Static	2000	2000	0	0
ISIS-default	144	Static	1500	1500	809515	0
BFD-known	150	Static	9600	9600	0	0
BFD-default	160	Static	9600	9600	0	0
BGP-known	106	Static	2500	2500	0	0
BGP-cfg-peer	107	Static	2000	2000	17704	0
BGP-default	108	Static	1500	1500	152783	0
PIM-mcast	109	Static	2000	2000	14497	0
PIM-ucast	110	Static	1500	1500	10	0
PIM-mcast	109	Static	2000	2000	14503	0
PIM-ucast	110	Static	1500	1500	10	0
IGMP	111	Static	500	500	8	0
ICMP-local	112	Global	500	1500	61042	13
ICMP-app	152	Global	1000	1500	9802	0

```
<output truncated>
```

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-18

The **show lpts pifib hardware police location** command is very useful for LPTS monitoring. It provides statistics on conforming and exceeding packets.

Following the fine-tuning configuration, where the rate limits for selected traffic types were lowered from their default values, this example provides information about packets that have been accepted and dropped using the new values. In this example, a number of fragmented packets and ICMP packets with local significance have been discarded.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- The two most common places for deploying policing in IP next generation networks are the access and the IP edge layer
- If violate action is not specified, this example will use a single token bucket scheme and no excess bursting will be allowed
- The burst size and duration are directly related. The burst size is equal to the burst duration multiplied with the link speed
- You can configure a maximum of two actions per category
- With dual-rate policing, traffic policing can be enforced according to two separate rates: CIR and PIR
- The percentage-based policing feature provides the ability to configure traffic policing based on a percentage of bandwidth available on the interface

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-19

Summary (Cont.)

- In hierarchical policing, the routers use a two-level policy map. The parent and child policies have class maps containing policing statements
- You can verify the policing operations using the **show policy-map** command
- Cisco access switches, such as the Cisco ME3400 Series, provide traffic policing features that can be deployed in the access layer
- An aggregate policer differs from an individual policer because it is shared by multiple traffic classes within a policy map
- LPTS can be thought of as a security measure for an IOS XR router by taking preemptive measures for traffic flows destined to the router

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-20

Implementing Traffic Shaping

Overview

Traffic shaping is implemented on Cisco IOS XR, IOS XE, and IOS routers using the Modular QoS CLI (MQC).

Traffic shaping allows you to control outgoing traffic on an interface to match the speed of transmission to the speed of the remote interface, and to ensure that the traffic conforms to administrative quality of service (QoS) policies. You can shape traffic adhering to a particular profile to meet downstream requirements, thereby eliminating bottlenecks due to data rate mismatches.

This lesson describes the tasks that are used to configure class-based traffic shaping in order to rate-limit certain traffic classes.

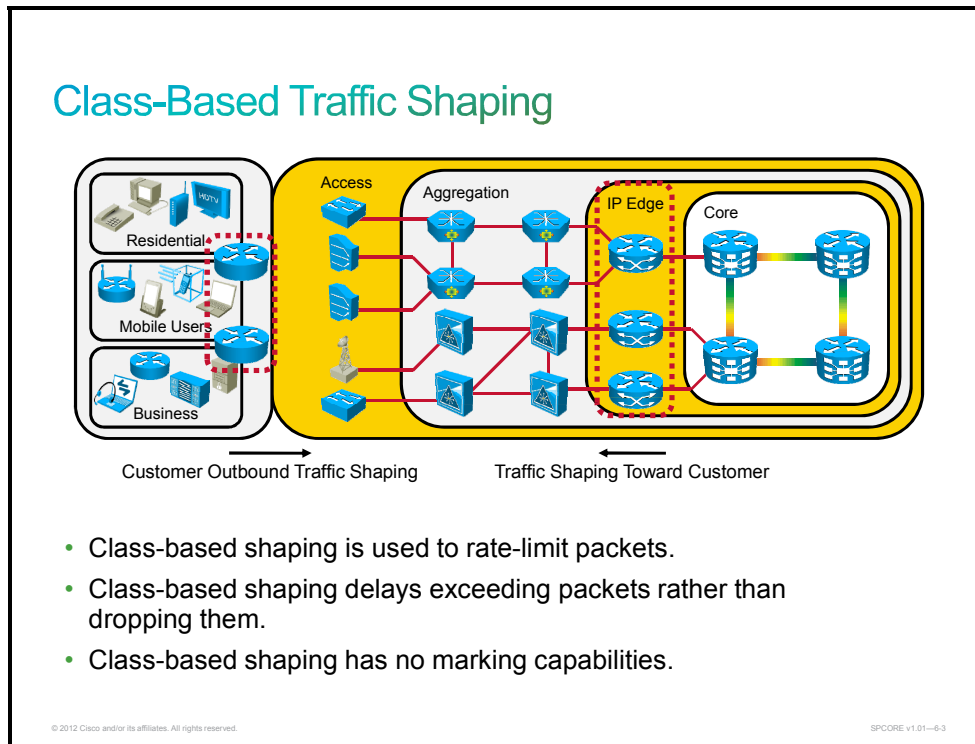
Objectives

Upon completing this lesson, you will be able to implement class-based shaping to rate-limit traffic. You will be able to meet this objective:

- Describe class-based shaping
- Explain a Single-Level Shaping Configuration
- Explain a Hierarchical Shaping Configuration
- Describe the show command used to Monitoring Class-Based Shaping operations

Class-Based Shaping

This topic describes class-based shaping.



Traffic shaping allows you to control the traffic going out from an interface in order to match its transmission speed to the speed of the remote target interface and to ensure that the traffic conforms to policies contracted for it.

Traffic shaping is typically deployed in two scenarios:

- On customer edge (CE) devices, on the links toward the service provider, to limit the outbound traffic to the contractual limits. This prevents dropping in the service provider network.
- In the provider edge (PE), on the links toward the customers, to throttle the traffic destined to a customer. This prevents tail drops on slow access links.

You can shape traffic adhering to a particular profile to meet downstream requirements, thereby eliminating bottlenecks in topologies with traffic-rate mismatches or oversubscriptions. Class-based shaping has these properties:

- Class-based shaping is configured via the MQC.
- Class-based shaping has no packet-marking capabilities.
- Class-based shaping works by queuing exceeding packets until the packets conform to the configured shaped rate.
- Class-based shaping can also be used in hierarchical policies.

Shaping to Average Rate

- Shaping to the **average** rate
 - Forwarding at the configured average rate
 - Allowed bursting up to Be when there are extra tokens available
 - Most common method
 - Supported on Cisco IOS XR, Cisco IOS, and Cisco IOS XE routers
- Shaping to the **peak** rate
 - Forwarding at the peak rate of up to $Bc + Be$ at every Tc
 - Rarely used
 - Not supported on Cisco IOS XR Software

© 2012 Cisco and/or its affiliates. All rights reserved.

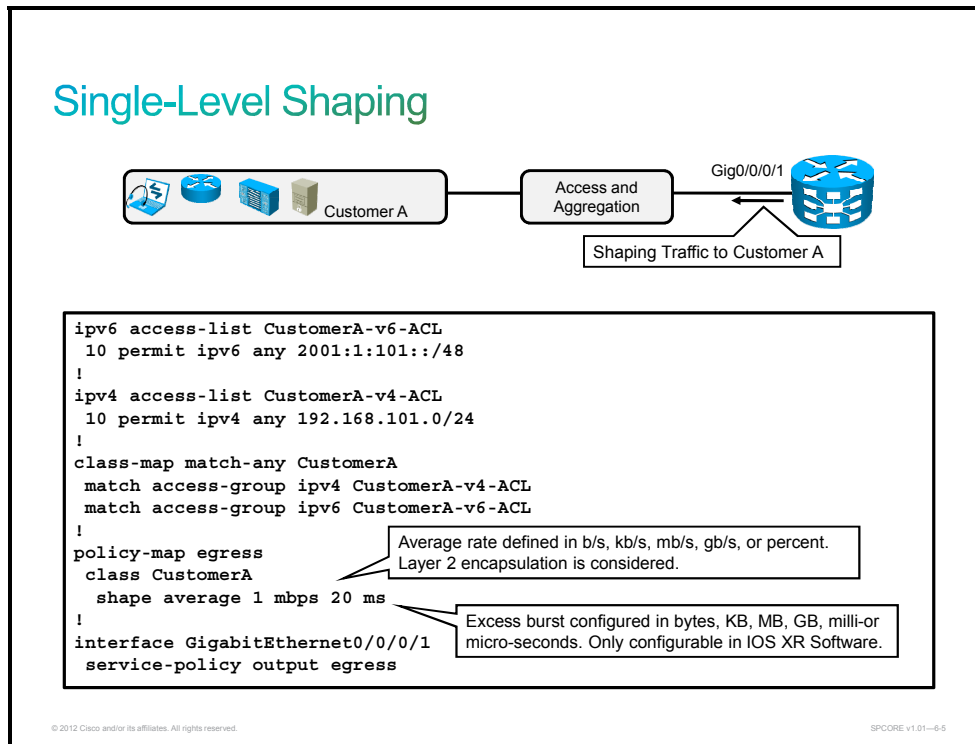
SPCORE v1.01-6-4

The main method of class-based shaping configuration is based on the configured average rate. Shaping to the average rate forwards up to a committed burst (Bc) of traffic at every committed time window (Tc) interval, with additional bursting capability when enough tokens are accumulated in the bucket. An amount equal to Bc worth of tokens is added to the token bucket at every Tc time interval. After the token bucket is emptied, additional bursting cannot occur until tokens are allowed to accumulate, which can occur only during periods of silence or when the transmit rate is lower than the average rate. After a period of low traffic activity, up to Bc + excess burst (Be) of traffic can be sent. Shaping to the average rate is the most common approach. It is supported on Cisco IOS XR, Cisco IOS, and Cisco IOS XE routers.

A rarely used method is shaping to the peak rate. Shaping to the peak rate forwards up to Bc + Be of traffic at every Tc time interval. An amount equal to Bc + Be worth of tokens are added to the token bucket at every Tc time interval. Shaping to the peak rate sends traffic at the peak rate, which is defined as the average rate multiplied by $(1 + Be/Bc)$. Sending packets at the peak rate may result in dropping in the WAN cloud during network congestion. Shaping to the peak rate is recommended only when the network has additional available bandwidth beyond the committed information rate (CIR) and applications can tolerate occasional packet drops. Shaping to the peak rate is not supported on Cisco IOS XR routers.

Single-Level Shaping Configuration

This topic explains a Single-Level Shaping Configuration.



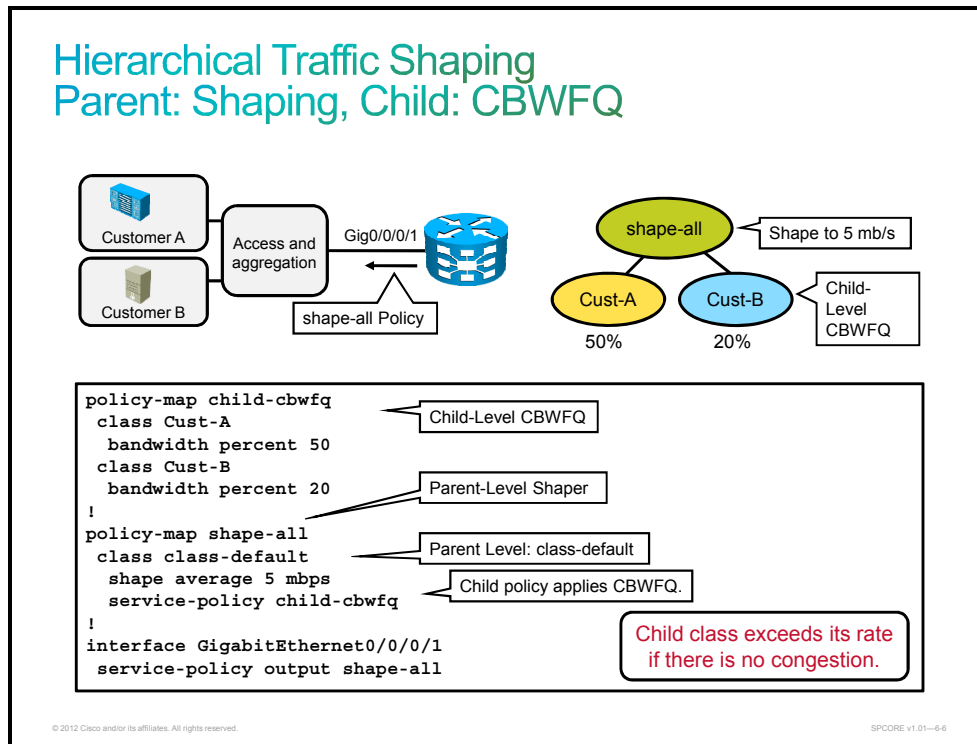
The **shape average** command in policy-map class configuration mode is used to shape traffic to the indicated average bit rate. The bit rate can be specified in bits per second, kilobits per second, megabits per second, gigabits per second, or percent. The configured traffic rate includes the Layer 2 encapsulation.

The optional setting of the excess burst allows you to modify the excess burst, configured in bytes, kilobytes, megabytes, gigabytes, milliseconds, or microseconds, or leave it at the default value computed by IOS XR Software. The option to specify the excess burst is available in Cisco IOS XR Software only.

This figure illustrates a scenario in which traffic toward the customer is shaped in the provider edge (PE). Access control lists (ACLs) specify the traffic going to that customer, and the class map uses the ACLs for matching. The policy map shapes the customer traffic class for the average rate of 1 Mb/s with an excess burst duration of 20 ms. The policy is applied to the customer-facing interface in the output direction.

Hierarchical Shaping Configuration

This topic explains a Hierarchical Shaping Configuration.



Class-based shaping can be used in combination with class-based weighted fair queuing (CBWFQ). The shape rate provides a maximum rate limit for the traffic class, while the bandwidth statement within CBWFQ provides a minimum bandwidth guarantee.

The example uses hierarchical policy maps and configures CBWFQ inside the class-based shaping. The parent policy is the shape-all policy. This parent policy references the child policy child-cbwfq.

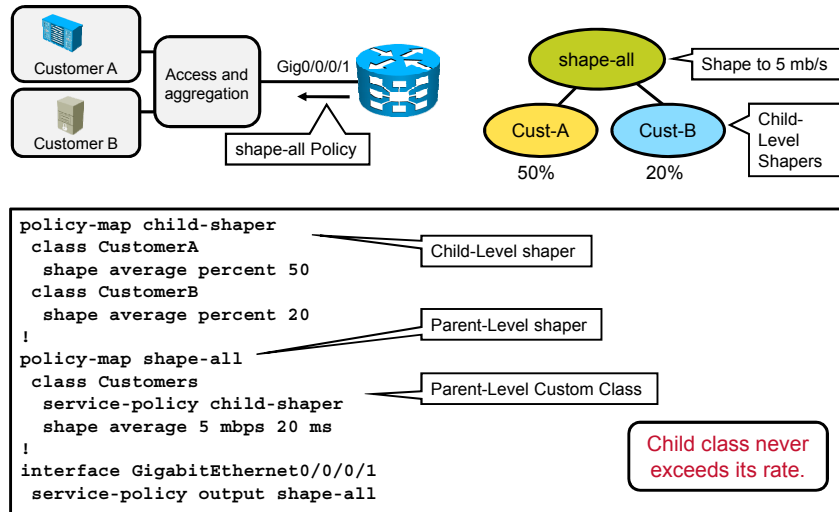
The parent policy map specifies an average shape rate of 5 Mb/s for all the traffic (matched by the class-default class) and assigns the child-cbwfq service policy as the child policy. The shaped traffic is further subdivided between two customer classes:

- **Cust-A:** Minimum guarantee equal to 50 percent of the shaped bandwidth
- **Cust-B:** Minimum guarantee equal to 20 percent of the shaped bandwidth

The shaped rate provides an upper bandwidth limit while the bandwidth statement within CBWFQ provides a minimum bandwidth guarantee. The bandwidth guarantees do not prevent each customer from consuming more bandwidth, within the shaped rate, if there is congestion. In this example, if only Customer A traffic is available on the link, it will take up to 5 Mb/s of the average rate.

You could also combine CBWFQ on the child level with policing on the parent level. In that case, the superclass traffic would not be queued but instead would be dropped when the configured parent rate is exceeded.

Hierarchical Traffic Shaping Parent: Shaping, Child: Shaping



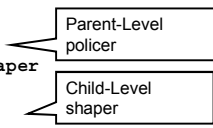
This figure presents another hierarchical traffic shaping scenario. Traffic shaping is configured on both the parent and the child level.

The difference from the previous scenario is in the handling of the child classes. They are shaped to the individual average rate and can never exceed it, even if other subclasses are not competing for the parent bandwidth. Thus, while CBWFQ on the child level allowed the class to consume more bandwidth if it was available, shaping on the child level strictly confines the class to the average rate.

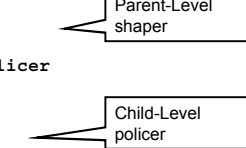
You may have CBWFQ and shaping configured simultaneously on the same level. In that case, ensure that the shape percentage value is always greater than the percentage value for bandwidth. In other words, the bandwidth guarantee must be lower than the average rate.

Hierarchical Traffic Shaping and Policing on Parent or Child Level

```
policy-map police-all
class Customers
  police rate 10 mbps
  service-policy child-shaper
!
policy-map child-shaper
class Customer
  shape average percent 10
!
interface GigabitEthernet0/0/0/1
  service-policy output police-all
```



```
policy-map shape-all
class Customers
  shape average 5 mbps
  service-policy child-policer
!
policy-map child-policer
class Customer
  police rate percent 10
!
interface GigabitEthernet0/0/0/1
  service-policy output shape-all
```



© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6.8

Hierarchical traffic shaping can also be combined with traffic policing.

The first example illustrates policing on the parent level and shaping on the child level.

Generally, you would have more than one child per class. The aggregate traffic is policed to 10 Mb/s. The traffic that is destined to the specific customer is shaped to 10 percent of the policer bandwidth, or 1 Mb/s.

The second example illustrates shaping on the parent level and policing on the child level. The aggregate traffic is shaped to 5 Mb/s. The traffic that is destined to the customer is policed to 10 percent of the shaper bandwidth, or 0.5 Mb/s.

Although these two approaches are less common than hierarchical traffic shaping with CBWFQ, they show that other types of hierarchical policies are possible.

Monitoring Class-Based Shaping Operations

This topic describes the show command used to Monitoring Class-Based Shaping operations.

Monitoring Traffic Shaping

```
RP/0/RSP0/CPU0:PE# show policy-map interface gigabitEthernet 0/0/0/2
GigabitEthernet0/0/0/2 output: egress

Class Customer
  Classification statistics      (packets/bytes)      (rate - kbps)
  Matched                      :      200/283600              0
  Transmitted                  :      200/283600              0
  Total Dropped                :           0/0                0
  Queueing statistics
    Queue ID                   : 266
    High watermark (Unknown)   : 0
    Inst-queue-len (packets)   : 0
    Avg-queue-len (Unknown)    : 0
    Taildropped(packets/bytes) : 3/3785
    Queue(conform)             : 42/59556
    Queue(exceed)              : 158/224044
    RED random drops(packets/bytes) : 0/0

Class class-default
  Classification statistics      (packets/bytes)      (rate - kbps)
  Matched                      :      58/5484              0
  Transmitted                  :      58/5484              0
<output truncated>
```

Diagram illustrating the output of the `show policy-map interface` command for monitoring traffic shaping. The output is divided into two sections: **Class Customer** and **Class class-default**. Each section displays classification statistics (Matched, Transmitted, Total Dropped) and queueing statistics (Queue ID, High watermark, Inst-queue-len, Avg-queue-len, Taildropped, Queue(conform), Queue(exceed), RED random drops). The output is truncated at the bottom. Annotations highlight specific values: **Tail Drop Counter** points to the Queue ID (266), **Traffic Within the Shaped Rate** points to the Queue(conform) value (42/59556), and **Traffic Above the Shaped Rate** points to the Queue(exceed) value (158/224044).

The **show policy-map interface** command displays all service policies that are applied to the interface.

The output provides statistics for a single-level traffic shaper that is applied to the GigabitEthernet 0/0/0/2 on a Cisco IOS XR platform. The command syntax on Cisco IOS and IOS XE routers is the same but the output differs slightly.

On IOS XR devices, every hardware queue has a configured CIR and peak information rate (PIR) value. These correspond to the guaranteed bandwidth for the queue, and the maximum bandwidth (shape rate) for the queue. These two rates correspond to the conform and exceed queue counters in the **show policy-map interface** command. The conform queue counter is the number of packets or bytes that were transmitted within the CIR value, and the exceed value is the number of packets or bytes that were transmitted within the PIR value. The exceed value in this case does not equate to a packet drop, but rather a packet that is above the CIR rate on that queue.

The other relevant value when monitoring traffic shaping is the tail-dropped packets or bytes counter. It indicates how much traffic could not be delayed due to buffer restrictions and was dropped.

Summary

This topic summarizes the key points that were discussed in this lesson.

Summary

- Traffic shaping is typically deployed on CE devices on the links toward the SP and on the PE devices, on the links toward the customers
- The **shape average** command in policy-map class configuration mode is used to shape traffic to the indicated average bit rate
- Class-based shaping can be used in combination with class-based weighted fair queuing (CBWFQ)
- The **show policy-map interface** command displays all service policies that are applied to the interface

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01-6-10

Module Summary

This topic summarizes the key points that were discussed in this module.

Module Summary

- Traffic shaping queues excess packets to stay within the shape rate. Traffic policing typically drops excess traffic to stay within the rate limit. Alternatively, it can re-mark, then send excess traffic. Both traffic shaping and policing measure the traffic rate using a token bucket mathematical model.
- Class-based policing features include drop or re-mark and transmit exceeding traffic, single or dual token bucket, single- or dual-rate policing, and multi-action policing.
- Class-based shaping can be combined with CBWFQ or policing in a hierarchical traffic-shaping structure.

© 2012 Cisco and/or its affiliates. All rights reserved.

SPCORE v1.01--6-1

Rate-limiting is the ability to prevent excess traffic from entering or leaving the network. Rate-limiting is required because of speed mismatches, oversubscriptions, and subrate access and to prevent unwanted traffic from causing network congestion.

To measure the traffic rate, a token bucket is used. Parameters that define the operations of a token bucket include the committed information rate (CIR), normal burst size (Bc), excess burst size (Be), and committed rate measurement interval (Tc).

Both traffic policing and traffic shaping are quality of service (QoS) mechanisms that are used to rate-limit a traffic class. Traffic policing operates by dropping excess traffic, while traffic shaping delays excess traffic with the aid of queuing.

In Cisco IOS Software, the most current rate-limiting mechanisms are class-based policing and class-based shaping. Both of these rate-limiting mechanisms are configured using the Modular QoS CLI (MQC).

Traffic policing and shaping is of special interest to ISPs. The high-cost, high-traffic networks are the major assets and are the focus of all attention. Service providers often use traffic policing and shaping as a method to optimize the use of their network, sometimes by intelligently shaping or policing traffic according to importance.

References

For additional information, refer to this resource:

- To learn more about configuring class-based policing and shaping on Cisco IOS XR Software, refer to *Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software* at this URL:
http://www.cisco.com/en/US/docs/ios_xr_sw/iosxr_r3.2/qos/configuration/guide/qc32cong.html

Module Self-Check

Use the questions here to review what you learned in this module. The correct answers and solutions are found in the Module Self-Check Answer Key.

- Q1) Which option describes a major difference between traffic policing versus traffic shaping? (Source: Understanding Traffic Policing and Shaping)
- A) Traffic policing drops excess traffic, while traffic shaping delays excess traffic by queuing it.
 - B) Traffic policing is applied only in the outbound direction, while traffic shaping can be applied in both the inbound and outbound directions.
 - C) Traffic policing is *not* available on access switches such as the Cisco ME 3400 Series, while traffic shaping *is* available on such devices.
 - D) Traffic policing requires policing queues to buffer excess traffic, while traffic shaping does not require any queues to buffer excess traffic.
- Q2) Which mathematical model is used by traffic policing mechanisms to meter traffic? (Source: Understanding Traffic Policing and Shaping)
- A) token bucket
 - B) RED
 - C) FIFO metering
 - D) Predictor or Stacker
- Q3) When configuring single-rate class-based policing, which configuration parameter is used to enable a dual token bucket? (Source: Implementing Traffic Policing)
- A) configuring a violate action
 - B) configuring an exceed action
 - C) configuring the PIR in addition to the CIR
 - D) configuring Be
- Q4) What is the main advantage of using multiaction policing? (Source: Implementing Traffic Policing)
- A) to distinguish between exceeding and violating traffic
 - B) to distinguish between conforming and exceeding traffic
 - C) to allow the setting of Layer 2 and Layer 3 QoS markers at the same time
 - D) to allow marking of the traffic before transmission
- Q5) You must enable LPTS hardware policing to protect the router. (Source: Implementing Traffic Policing)
- A) true
 - B) false
- Q6) Which two statements are true when class-based shaping is used in conjunction with CBWFQ? (Choose two.) (Source: Implementing Traffic Shaping)
- A) The **bandwidth** command defines the minimum guaranteed bandwidth for the traffic class.
 - B) The **bandwidth** command defines the maximum guaranteed bandwidth for the traffic class.
 - C) A child class can use all parent bandwidth if there is no congestion.
 - D) A child class can use only the dedicated bandwidth.

- Q7) What are two configuration options when configuring class-based traffic shaping on Cisco IOS XR Software? (Choose two.) (Source: Implementing Traffic Shaping)
- A) excess burst
 - B) single or dual token bucket
 - C) shape average
 - D) single or multiaction traffic shaping
 - E) single- or dual-rate traffic shaping

Module Self-Check Answer Key

Q1)	A
Q2)	A
Q3)	A
Q4)	C
Q5)	B
Q6)	A, C
Q7)	A, C

