QoS: Congestion Avoidance Configuration Guide, Cisco IOS XE Release 3S (ASR 1000)

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Congestion Avoidance Overview 1

Finding Feature Information 1

Weighted Random Early Detection 1

About Random Early Detection 1

How It Works 2

Packet Drop Probability 2

How TCP Handles Traffic Loss 3

How the Router Interacts with TCP 3

About WRED 4

Why Use WRED 4

How It Works 5

Average Queue Size 6

IPv6 QoS: MQC WRED-Based Drop 7

Finding Feature Information 7

Information About IPv6 QoS: MQC WRED-Based Drop 7

Implementation Strategy for QoS for IPv6 7

Congestion Avoidance for IPv6 Traffic 8

Additional References 8

Feature Information for IPv6 QoS: MQC WRED-Based Drop 9

Configuring Weighted Random Early Detection 11

Finding Feature Information 11

About Weighted Random Early Detection 11

How to Configure WRED 12

Enabling WRED 12

Changing WRED Parameters 12

Monitoring WRED 13

WRED Configuration Examples 13

Example WRED Configuration 13

Example Parameter-Setting WRED 14

Feature Information for Configuring Weighted Random Early Detection 15

Byte-Based Weighted Random Early Detection 17

Finding Feature Information 17

Restrictions for Byte-Based Weighted Random Early Detection 17

Information About Byte-Based Weighted Random Early Detection 18

Changes in functionality of WRED 18

Changes in Queue Limit and WRED Thresholds 18

How to Configure Byte-Based Weighted Random Early Detection 18

Configuring Byte-Based WRED 18

Configuring the Queue Depth and WRED Thresholds 20

Changing the Queue Depth and WRED Threshold Unit Modes 24

Verifying the Configuration for Byte-Based WRED 27

Configuration Examples for Byte-Based Weighted Random Early Detection 28

Example Configuring Byte-Based WRED 28

Additional References 29

Feature Information for Byte-Based Weighted Random Early Detection 30

WRED Explicit Congestion Notification 33

Finding Feature Information 33

Prerequisites for WRED-Explicit Congestion Notification 33

Information About WRED-Explicit Congestion Notification 33

WRED-Explicit Congestion Notification Feature Overview 33

How WRED Works 34

ECN Extends WRED Functionality 34

How Packets Are Treated When ECN Is Enabled 35

Benefits of WRED Explicit Congestion Notification 35

How to Configure WRED-Explicit Congestion Notification 36

Configuring Explicit Congestion Notification 36

Verifying the Explicit Congestion Notification Configuration 37

Configuration Examples for WRED-Explicit Congestion Notification 38

Example Enabling ECN 38

Example Verifying the ECN Configuration 39

Additional References 40

Feature Information for WRED Explicit Congestion Notification 41

QoS Time-Based Thresholds for WRED and Queue Limit 43

Finding Feature Information 43

Prerequisites for QoS Time-Based Thresholds for WRED and Queue Limit 43
Restrictions for QoS Time-Based Thresholds for WRED and Queue Limit 44
Information About QoS Time-Based Thresholds for WRED and Queue Limit 44
Benefits of QoS Time-Based Thresholds for WRED and Queue Limit 44
Setting Thresholds by Using WRED 44
Setting Thresholds by Using the queue-limit Command 45
random-detect Commands with the Milliseconds Keyword 45
Mixing Threshold Units of Measure 46
How to Configure QoS Time-Based Thresholds for WRED and Queue Limit 46
Enabling WRED and Using WRED to Specify Thresholds 46
Using the queue-limit Command to Specify the Thresholds 49
Attaching the Policy Map to an Interface in a QoS Time-Based Threshold for WRED Configuration 51
Verifying the QoS Time-Based Thresholds for WRED and Queue Limit Configuration 52
Troubleshooting Tips 53
Configuration Examples for QoS Time-Based Thresholds for WRED and Queue Limit 54
Example Using WRED to Set Thresholds 54
Example Using the queue-limit Command to Set Thresholds 54
Example Verifying the Configuration 55
Example WRED Threshold Configuration Sample Output 55
Example queue-limit command Threshold Configuration Sample Output 56
Additional References 57
Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit 58

Congestion Avoidance Overview

Congestion avoidance techniques monitor network traffic loads in an effort to anticipate and avoid congestion at common network bottlenecks. Congestion avoidance is achieved through packet dropping. Among the more commonly used congestion avoidance mechanisms is Random Early Detection (RED), which is optimum for high-speed transit networks. Cisco IOS XE Software includes an implementation of RED, called Weighted RED (WRED), that combines the capabilities of the RED algorithm with the IP Precedence feature. WRED, when configured, controls when the router drops packets.

- Finding Feature Information, page 1
- Weighted Random Early Detection, page 1

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Weighted Random Early Detection

WRED helps avoid the globalization problems that can occur. Global synchronization occurs as waves of congestion crest only to be followed by troughs during which the transmission link is not fully utilized. Global synchronization of TCP hosts, for example, can occur because packets are dropped all at once. Global synchronization manifests when multiple TCP hosts reduce their transmission rates in response to packet dropping and then increase their transmission rates once again when the congestion is reduced.

- About Random Early Detection, page 1
- About WRED, page 4

About Random Early Detection

The RED mechanism was proposed by Sally Floyd and Van Jacobson in the early 1990s to address network congestion in a responsive rather than reactive manner. Underlying the RED mechanism is the premise that most traffic runs on data transport implementations that are sensitive to loss and will temporarily slow down when some of their traffic is dropped. TCP, which responds appropriately--even robustly--to traffic drop by slowing down its traffic transmission, effectively allows the traffic-drop behavior of RED to work as a congestion-avoidance signalling mechanism.

TCP constitutes the most heavily used network transport. Given the ubiquitous presence of TCP, RED offers a widespread, effective congestion-avoidance mechanism.

In considering the usefulness of RED when robust transports such as TCP are pervasive, it is important to consider also the seriously negative implications of employing RED when a significant percentage of the traffic is not robust in response to packet loss. Neither Novell NetWare nor AppleTalk is appropriately robust in response to packet loss, therefore you should not use RED for them.

- How It Works, page 2
- Packet Drop Probability, page 2
- How TCP Handles Traffic Loss, page 3
- How the Router Interacts with TCP, page 3

How It Works

RED aims to control the average queue size by indicating to the end hosts when they should temporarily slow down transmission of packets.

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

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

Packet Drop Probability

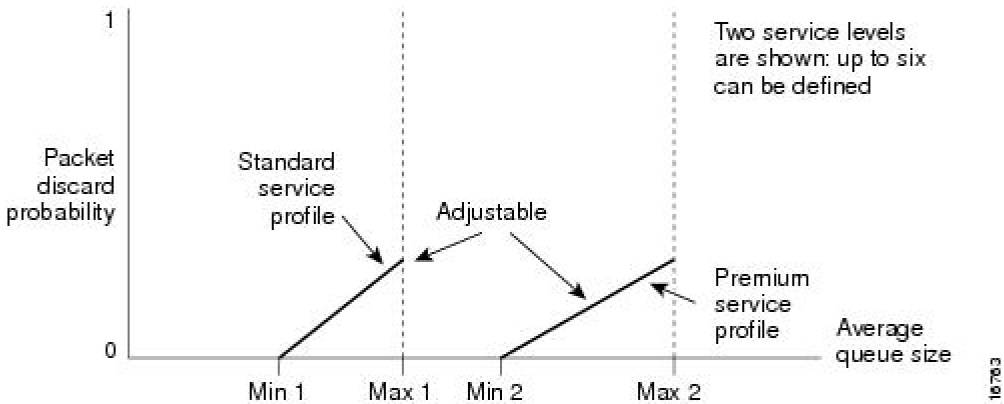
The packet drop probability is based on the minimum threshold, maximum threshold, and mark probability denominator.

When the average queue depth is 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.

The mark probability denominator is the fraction of packets dropped when the average queue depth is at the maximum threshold. For example, if the denominator is 512, one out of every 512 packets is dropped when the average queue is at the maximum threshold.

When the average queue size is above the maximum threshold, all packets are dropped. The figure below summarizes the packet drop probability.

Figure 1 RED Packet Drop Probability



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, and the transmission link will not be fully used.

The difference between the maximum threshold and the minimum threshold should be large enough to avoid global synchronization of TCP hosts (global synchronization of TCP hosts can occur as multiple TCP hosts reduce their transmission rates). If the difference between the maximum and minimum thresholds is too small, many packets may be dropped at once, resulting in global synchronization.

How TCP Handles Traffic Loss

Note The sections How TCP Handles Traffic Loss, page 3 and How the Router Interacts with TCP, page 3 contain detailed information that you need not read in order to use WRED or to have a general sense of the capabilities of RED. If you want to understand why problems of global synchronization occur in response to congestion and how RED addresses them, read these sections.

When the recipient of TCP traffic--called the receiver--receives a data segment, it checks the four octet (32-bit) sequence number of that segment against the number the receiver expected, which would indicate that the data segment was received in order. If the numbers match, the receiver delivers all of the data that it holds to the target application, then it updates the sequence number to reflect the next number in order, and finally it either immediately sends an acknowledgment (ACK) packet to the sender or it schedules an ACK to be sent 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 one marked with 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 include 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 resend the lost data segment.

When the sender receives an ACK, it makes this determination: It 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 indicates receipt of some data sent, the sender determines if new credit has been granted to allow it to send more data. When the ACK indicates receipt of none of the data sent 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 remit the first ACK, and that a second data segment was received out of order, forcing the receiver to remit 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 resends the segment. Then it adjusts its transmission rate to half of what is was before the drop was detected. This is the TCP back-off or slow-down behavior. Although this behavior is appropriately responsive to congestion, problems can arise when multiple TCP sessions are carried on concurrently with the same router and all TCP senders slow down transmission of packets at the same time.

How the Router Interacts with TCP

Note
The sections How TCP Handles Traffic Loss, page 3 and How the Router Interacts with TCP, page 3 contain detailed information that you need not read in order to use WRED or to have a general sense of the capabilities of RED. If you want to understand why problems of global synchronization occur in response to congestion and how RED addresses them, read these sections.

To see how the router interacts with TCP, we will look at an example. In this example, on average, the router receives traffic from one particular TCP stream every other, every 10th, and every 100th or 200th message in the interface in MAE-EAST or FIX-WEST. A router can handle multiple concurrent TCP sessions. Because network flows are additive, there is a high probability that when traffic exceeds the Transmit Queue Limit (TQL) at all, it will vastly exceed the limit. However, there is also a high probability that the excessive traffic depth is temporary and that traffic will not stay excessively deep except at points where traffic flows merge or at edge routers.

If the router drops all traffic that exceeds the TQL, many TCP sessions will 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 of global synchronization.

However, if the router drops no traffic, as is the case when queueing features such as fair queueing or priority queueing (PQ) are used, then the data is likely to be stored in main memory, drastically degrading router performance.

By directing one TCP session at a time to slow down, RED solves the problems described, allowing for full utilization of the bandwidth rather than utilization manifesting as crests and troughs of traffic.

About WRED

WRED combines the capabilities of the RED algorithm with the IP Precedence feature to provide for preferential traffic handling of higher priority packets. WRED can selectively discard lower priority traffic when the interface begins to get congested and provide differentiated performance characteristics for different classes of service.

You can configure WRED to ignore IP precedence when making drop decisions so that nonweighted RED behavior is achieved.

For interfaces configured to use the Resource Reservation Protocol (RSVP) feature, WRED chooses packets from other flows to drop rather than the RSVP flows. Also, IP Precedence governs which packets are dropped--traffic that is at a lower precedence has a higher drop rate and therefore is more likely to be throttled back.

WRED differs from other congestion avoidance techniques such as queueing strategies because it attempts to anticipate and avoid congestion rather than control congestion once it occurs.

- Why Use WRED, page 4
- How It Works, page 5
- Average Queue Size, page 6

Why Use WRED

WRED makes early detection of congestion possible and provides for multiple classes of traffic. It also protects against global synchronization. For these reasons, WRED is useful on any output interface where you expect congestion to occur.

However, WRED is usually used in the core routers of a network, rather than at the edge of the network. Edge routers assign IP precedences to packets as they enter the network. WRED uses these precedences to determine how to treat different types of traffic.

WRED provides separate thresholds and weights for different IP precedences, allowing you to provide different qualities of service in regard to packet dropping for different traffic types. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

WRED is also RSVP-aware, and it can provide the controlled-load QoS service of integrated service.

How It Works

By randomly dropping packets prior to periods of high congestion, WRED tells the packet source to decrease its transmission rate. If the packet source is using TCP, it will decrease its transmission rate until all the packets reach their destination, which indicates that the congestion is cleared.

WRED generally drops packets selectively based on IP precedence. Packets with a higher IP precedence are less likely to be dropped than packets with a lower precedence. Thus, the higher the priority of a packet, the higher the probability that the packet will be delivered.

WRED selectively drops packets when the output interface begins to show signs of congestion. By dropping some packets early rather than waiting until the queue is full, WRED avoids dropping large numbers of packets at once and minimizes the chances of global synchronization. Thus, WRED allows the transmission line to be used fully at all times.

In addition, WRED statistically drops more packets from large users than small. Therefore, traffic sources that generate the most traffic are more likely to be slowed down than traffic sources that generate little traffic.

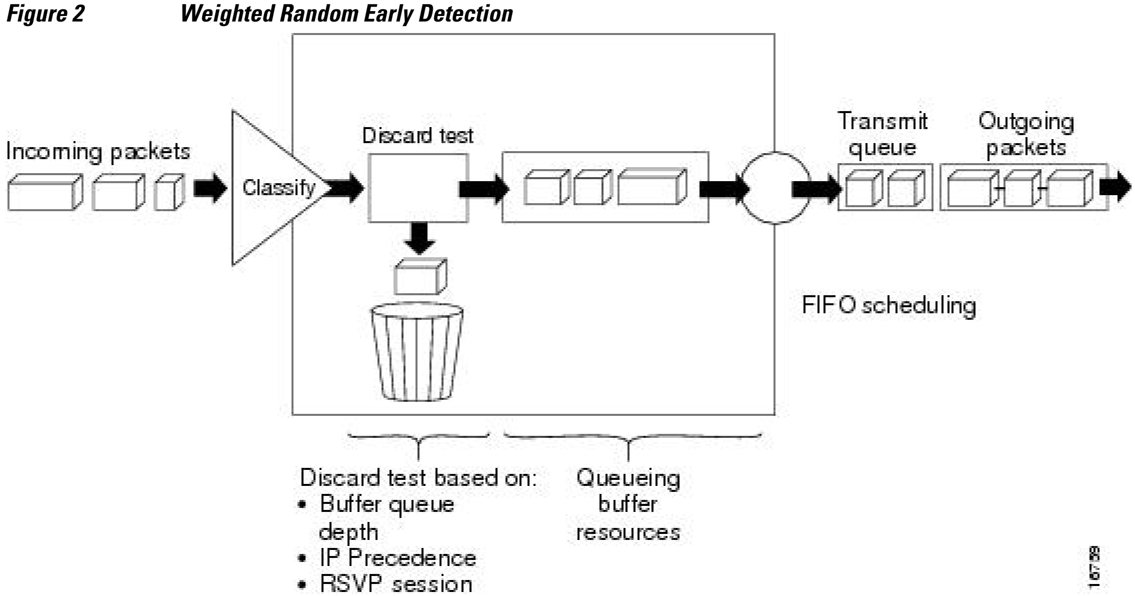
WRED helps to avoid the globalization problems. Global synchronization manifests when multiple TCP hosts reduce their transmission rates in response to packet dropping and then increase their transmission rates once again when the congestion is reduced.

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

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

The figure below illustrates how WRED works.

Figure 2 Weighted Random Early Detection



Average Queue Size

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

```
average = (old_average * (1-2 -n)) + (current_queue_size * 2
-n
)
```

where *n* is the exponential weight factor, a user-configurable value. The default value of the exponential weight factor is 4. It is recommended to use only the default value for the exponential weight factor. Change this value from the default value only if you have determined that your scenario would benefit from using a different value.

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

Note If the value of *n* gets too high, WRED will not react to congestion. Packets will be sent or dropped as if WRED were not in effect.

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

If the value of *n* gets too low, WRED will overreact to temporary traffic bursts and drop traffic unnecessarily.

IPv6 QoS: MQC WRED-Based Drop

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.

- Finding Feature Information, page 7
- Information About IPv6 QoS: MQC WRED-Based Drop, page 7
- Additional References, page 8
- Feature Information for IPv6 QoS: MQC WRED-Based Drop, page 9

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 QoS: MQC WRED-Based Drop

- Implementation Strategy for QoS for IPv6, page 7
- Congestion Avoidance for IPv6 Traffic, page 8

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queuing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks that are running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.

- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria that you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6 traffic, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**, **set precedence**, and **set dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.
- Work from the edge toward the core in applying QoS features.
- Build the policy to treat the traffic.
- Apply the policy.

Congestion Avoidance for IPv6 Traffic

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of class-based weighted fair queueing (CBWFQ). WRED supports class-based and flow-based queueing (using DSCP or precedence values).

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IPv6 commands	Cisco IOS IPv6 Command Reference
Cisco IOS IPv6 features	Cisco IOS IPv6 Feature Mapping
QoS Congestion Avoidance	“Congestion Avoidance Overview” module

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	IPv6 RFCs

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IPv6 QoS: MQC WRED-Based Drop

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1 Feature Information for IPv6 QoS: MQC WRED-Based Drop

Feature Name	Releases	Feature Information
IPv6 QoS: MQC WRED-Based Drop	Cisco IOS XE Release 2.1	WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.

Configuring Weighted Random Early Detection

This module describes the tasks for configuring Weighted Random Early Detection (WRED) on a router.

- Finding Feature Information, page 11
- About Weighted Random Early Detection, page 11
- How to Configure WRED, page 12
- WRED Configuration Examples, page 13
- Feature Information for Configuring Weighted Random Early Detection, page 15

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

About Weighted Random Early Detection

Random Early Detection (RED) is a congestion avoidance mechanism that takes advantage of the congestion control mechanism of TCP. By randomly dropping packets prior to periods of high congestion, RED tells the packet source to decrease its transmission rate. WRED drops packets selectively based on IP precedence. Edge routers assign IP precedences to packets as they enter the network. (WRED is useful on any output interface where you expect to have congestion. However, WRED is usually used in the core routers of a network, rather than at the edge.) WRED uses these precedences to determine how it treats different types of traffic.

When a packet arrives, the following events occur:

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

Note WRED is useful with adaptive traffic such as TCP/IP. With TCP, dropped packets indicate congestion, so the packet source will reduce its transmission rate. With other protocols, packet sources may not respond or may resend dropped packets at the same rate. Thus, dropping packets does not decrease congestion. WRED treats non-IP traffic as precedence 0, the lowest precedence. Therefore, non-IP traffic is more likely to be dropped than IP traffic.

When you enable WRED with the **random-detect** interface configuration command, the parameters are set to their default values. The weight factor is 9. For all precedences, the mark probability denominator is 10, and maximum threshold is based on the output buffering capacity and the transmission speed for the interface.

The default minimum threshold depends on the precedence. The minimum threshold for IP Precedence 0 corresponds to half of the maximum threshold. The values for the remaining precedences fall between half the maximum threshold and the maximum threshold at evenly spaced intervals.

Note The default WRED parameter values are based on the best available data. We recommend that you do not change the parameters from their default values unless you have determined that your applications will benefit from the changed values.

How to Configure WRED

- Enabling WRED, page 12
- Changing WRED Parameters, page 12
- Monitoring WRED, page 13

Enabling WRED

Command	Purpose
`Router(config-if)#` **random-detect**	Enables WRED.

Changing WRED Parameters

Command	Purpose
`Router(config-if)#` **random-detect exponential-weighting-constant** *exponent*	Configures the weight factor used in calculating the average queue length.

Command	Purpose
Router(config-if)# **random-detect precedence** *precedence min-threshold max-threshold mark-prob-denominator*	Configures parameters for packets with a specific IP Precedence. The minimum threshold for IP Precedence 0 corresponds to half the maximum threshold for the interface. Repeat this command for each precedence. To configure RED, rather than WRED, use the same parameters for each precedence.

Monitoring WRED

Command	Purpose
Router# **show queue** *interface-type interface-number*	Displays the header information of the packets inside a queue.
Router# **show queueing interface** *interface-number* [**vc** [[*vpi*/] *vci*]]	Displays the WRED statistics of a specific virtual circuit (VC) on an interface.
Router# **show queueing random-detect**	Displays the queueing configuration for WRED.
Router# **show interfaces** [*type slot* | *port-adapter* | *port*]	Displays WRED configuration on an interface.

WRED Configuration Examples

Example WRED Configuration

The following example enables WRED with default parameter values:

```
interface Serial5/0
 description to qos1-75a
 ip address 200.200.14.250 255.255.255.252
 random-detect
```

Use the **show interfaces** command output to verify the configuration. Notice that the "Queueing strategy" report lists "random early detection (RED)."

```
Router# show interfaces serial 5/0
Serial5/0 is up, line protocol is up
  Hardware is M4T
  Description: to qos1-75a
  Internet address is 200.200.14.250/30
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 237/255
  Encapsulation HDLC, crc 16, loopback not set
  Keepalive not set
  Last input 00:00:15, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:05:08
  Input queue: 0/75/0 (size/max/drops); Total output drops: 1036
```

```
  Queueing strategy: random early detection(RED)
  5 minutes input rate 0 bits/sec, 2 packets/sec
  5 minutes output rate 119000 bits/sec, 126 packets/sec
     594 packets input, 37115 bytes, 0 no buffer
     Received 5 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     37525 packets output, 4428684 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Use the **show queue** command output to view the current contents of the interface queue. Notice that there is only a single queue into which packets from all IP precedences are placed after dropping has taken place. The output has been truncated to show only three of the five packets.

```
Router# show queue serial 5/0

Output queue for Serial5/0 is 5/0
Packet 1, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.4, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 128 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 2, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.5, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 160 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 3, linktype: ip, length: 118, flags: 0x280
  source: 190.1.3.6, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 192 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
```

Use the **show queueing** command output to view the current settings for each of the precedences. Also notice that the default minimum thresholds are spaced evenly between half and the entire maximum threshold. Thresholds are specified in terms of packet count.

```
Router# show queueing
Current random-detect configuration:
     Serial5/0
       Queueing strategy:random early detection (WRED)
       Exp-weight-constant:9 (1/512)
       Mean queue depth:28

       Class   Random       Tail    Minimum    Maximum     Mark
                 drop       drop  threshold  threshold  probability
         0        330          0         20         40     1/10
         1        267          0         22         40     1/10
         2        217          0         24         40     1/10
         3        156          0         26         40     1/10
         4         61          0         28         40     1/10
         5          6          0         31         40     1/10
         6          0          0         33         40     1/10
         7          0          0         35         40     1/10
        rsvp        0          0         37         40     1/10
```

Example Parameter-Setting WRED

The following example enables WRED on the interface and specifies parameters for the different IP precedences:

```
interface Hssi0/0/0
 description 45Mbps to R1
 ip address 10.200.14.250 255.255.255.252
 random-detect
 random-detect precedence 0 32 256 100
 random-detect precedence 1 64 256 100
```

```
random-detect precedence 2 96 256 100
random-detect precedence 3 120 256 100
random-detect precedence 4 140 256 100
random-detect precedence 5 170 256 100
random-detect precedence 6 290 256 100
random-detect precedence 7 210 256 100
random-detect precedence rsvp 230 256 100
```

Feature Information for Configuring Weighted Random Early Detection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2 Feature Information for Configuring Weighted Random Early Detection

Feature Name	Releases	Feature Information
Class-Based Weighted Fair Queueing (CBWFQ) and Weighted Random Early Detection (WRED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers. **Note** For information about CBWFQ, see the "Configuring Weighted Fair Queueing" module.
Random Early Detection (RED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
Weighted Random Early Detection	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
Weighted RED (WRED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.

Byte-Based Weighted Random Early Detection

This module explains how to enable byte-based Weighted Random Early Detection (WRED), and set byte-based queue limits and WRED thresholds.

- Finding Feature Information, page 17
- Restrictions for Byte-Based Weighted Random Early Detection, page 17
- Information About Byte-Based Weighted Random Early Detection, page 18
- How to Configure Byte-Based Weighted Random Early Detection, page 18
- Configuration Examples for Byte-Based Weighted Random Early Detection, page 28
- Additional References, page 29
- Feature Information for Byte-Based Weighted Random Early Detection, page 30

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Byte-Based Weighted Random Early Detection

- WRED is only useful when the bulk of the traffic is TCP/IP traffic. With TCP, dropped packets indicate congestion, so the packet source will reduce its transmission rate. With other protocols, packet sources may not respond or may resend dropped packets at the same rate. Thus, dropping packets does not decrease congestion.
- You cannot configure byte-based WRED on a class in which the queue-limit is configured in milliseconds or packets.

Information About Byte-Based Weighted Random Early Detection

- Changes in functionality of WRED, page 18
- Changes in Queue Limit and WRED Thresholds, page 18

Changes in functionality of WRED

This feature extends the functionality of WRED. In previous releases, you specified the WRED actions based on the number of packets. With the byte-based WRED, you can specify WRED actions based on the number of bytes.

Changes in Queue Limit and WRED Thresholds

In Cisco IOS XE Release 2.4, the Cisco ASR 1000 Series Aggregation Services Routers support the addition of bytes as a unit of configuration for both queue limits and WRED thresholds. Therefore, as of this release, packet-based and byte-based limits are configurable, with some restrictions.

How to Configure Byte-Based Weighted Random Early Detection

- Configuring Byte-Based WRED, page 18
- Configuring the Queue Depth and WRED Thresholds, page 20
- Changing the Queue Depth and WRED Threshold Unit Modes, page 24
- Verifying the Configuration for Byte-Based WRED, page 27

Configuring Byte-Based WRED

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-map-name*
4. **match ip precedence** ip-precedence-value
5. exit
6. **policy-map** *policy-name*
7. **class** *class-name*
8. **random-detect**
9. **random-detect precedence** *precedence min-threshold* **bytes** *max-threshold* **bytes** *mark-prob-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-map-name* **Example:** `Router(config)# class-map c1`	Specifies the user-defined name of the traffic class.
Step 4	**match ip precedence** ip-precedence-value **Example:** `Router(config-cmap)# match ip precedence 1`	Specifies up to eight IP Precedence values used as match criteria.
Step 5	exit **Example:** `Router(config-cmap)# exit`	Exits from class-map configuration mode.
Step 6	**policy-map** *policy-name* **Example:** `Router(config)# policy-map p1`	Specifies the name of the traffic policy to configure.
Step 7	**class** *class-name* **Example:** `Router(config-pmap)# class c1`	Specifies the name of a predefined traffic class, which was configured with the **class-map** command, used to classify traffic to the traffic policy.

	Command or Action	Purpose
Step 8	**random-detect** **Example:** `Router(config-pmap-c)# random-detect`	Enables WRED.
Step 9	**random-detect precedence** *precedence min-threshold* **bytes** *max-threshold* **bytes** *mark-prob-denominator* **Example:** **Example:** `Router(config-pmap-c)# random-detect precedence 1 2000 bytes 3000 bytes 200`	Configures the parameters for bytes with a specific IP precedence.

Configuring the Queue Depth and WRED Thresholds

Be sure that your configuration satisfies the following conditions when configuring the queue depth and WRED thresholds:

- When configuring byte-based mode, the queue limit must be configured prior to the WRED threshold and before the service policy is applied.
- When setting the queue depth and WRED thresholds in an enhanced QoS policies aggregation configuration, the limits are supported only for the default class at a subinterface policy map and for any classes at the main interface policy map.

Note Consider the following restrictions when you configure the queue depth and WRED thresholds:

- Do not configure the queue limit unit before you configure a queueing feature for a traffic class.
- If you do not configure a queue limit, then the default mode is packets.
- When you configure WRED thresholds, the following restrictions apply:
    - The WRED threshold must use the same unit as the queue limit. For example, if the queue limit is in packets, then the WRED thresholds also must be in packets.
    - If you do not configure a queue limit in bytes, then the default mode is packets and you must also configure the WRED threshold in packets.
    - The queue limit size must be greater than the WRED threshold.
- The unit modes for either the queue limit or WRED thresholds cannot be changed dynamically after a service policy is applied.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. *qos-queueing-feature*
6. **queue-limit** *queue-limit-size* [**bytes** | **packets**]
7. **random-detect** [**dscp-based** | **prec-based**]
8. Do one of the following:
   - **random-detect dscp** dscp-value {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
   -
   -
   - **random-detect precedence** precedence {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} max-probability-denominator

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface`	Specifies the name of the traffic policy that you want to configure or modify and enters policy-map configuration mode.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class AF1`	Specifies the name of the traffic class and enters policy-map class configuration mode.

	Command or Action	Purpose
Step 5	*qos-queueing-feature* **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 90`	Enters a QoS configuration command. Some of the queueing features that are currently supported are **bandwidth**, **priority**, and **shape**. **Note** Multiple QoS queueing commands can be entered at this step. However, due to dependencies between the queue limit and WRED thresholds, you should configure WRED after you configure the queue limit.
Step 6	**queue-limit** *queue-limit-size* [**bytes** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 547500 bytes`	Specifies the maximum number (from 1 to 8192000) of bytes or packets that the queue can hold for this class.
Step 7	**random-detect** [**dscp-based** | **prec-based**] **Example:** `Router(config-pmap-c)# random-detect dscp-based`	Enables WRED in either DSCP-based mode or precedence-based mode.
Step 8	Do one of the following: • **random-detect dscp** dscp-value {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **random-detect precedence** precedence {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} max-probability-denominator **Example:** `Router(config-pmap-c)# random-detect precedence 8 750000 bytes 750000 bytes`	Configures WRED parameters for a particular DSCP value or IP precedence. **Note** Use the *min-threshold max-threshold* arguments without the **bytes** keyword to configure packet-based thresholds, when the **queue-limit** unit is also packets (the default). Alternatively, use these arguments with the **bytes** keyword when the **queue-limit** unit is configured in bytes.

Examples

Correct Configuration

Invalid Configuration

Correct Configuration

Invalid Configuration

The following examples show both correct and invalid configurations to demonstrate some of the restrictions.

The following example shows the correct usage of setting the queue limit in bytes mode after the **bandwidth remaining ratio** queueing feature has been configured for a traffic class:

```
class AF1
    bandwidth remaining ratio 90
        queue-limit 750000 bytes
```

The following example shows an invalid configuration for the queue limit in bytes mode before the **bandwidth remaining ratio** queueing feature has been configured for a traffic class:

```
class AF1
    queue-limit 750000 bytes
        bandwidth remaining ratio 90
```

The following example shows the correct usage of setting the queue limit in bytes mode after the **bandwidth remaining ratio** queueing feature has been configured for a traffic class, followed by the setting of the thresholds for WRED in compatible byte mode:

```
class AF1
    bandwidth remaining ratio 90
        queue-limit 750000 bytes
            random-detect dscp-based
                random-detect dscp 8 750000 bytes 750000 bytes
```

This example shows an invalid configuration of the WRED threshold in bytes without any queue limit configuration, which therefore defaults to a packet-based queue depth. Therefore, the WRED threshold must also be in packets:

```
class AF1
    bandwidth remaining ratio 90
    random-detect dscp-based
    random-detect dscp 8 750000 bytes 750000 bytes
```

Changing the Queue Depth and WRED Threshold Unit Modes

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no service-policy output** *policy-map-name*
5. **exit**
6. **policy-map** *policy-map-name*
7. **class** *class-name*
8. **queue-limit** *queue-limit-size* [**bytes** | **packets**]
9. Do one of the following:
   - **no random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
   -
   -
   - **no random-detect precedence** *precedence* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator*
10. Do one of the following:
    - **random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
    -
    -
    - **random-detect precedence** *precedence* {*min-threshold max-threshold*| *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Router(config)# policy-map main-interface`	Specifies the interface where you want to remove a service policy, and enters interface configuration mode.
Step 4	**no service-policy output** *policy-map-name* **Example:** `Router(config-if)# no service-policy output main-interface-policy`	Removes a service policy applied to the specified interface.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns you to global configuration mode.
Step 6	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface-policy`	Specifies the name of the Traffic policy that you want to modify and enters policy-map configuration mode.
Step 7	**class** *class-name* **Example:** `Router(config-pmap)# class AF1`	Specifies the name of the traffic class and enters policy-map class configuration mode.
Step 8	**queue-limit** *queue-limit-size* [**bytes** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 5000 packets`	Specifies the maximum number (from 1 to 8192000) of bytes or packets that the queue can hold for this class.

	Command or Action	Purpose
Step 9	Do one of the following: • **no random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **no random-detect precedence** *precedence* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator* **Example:** `Router(config-pmap-c)# no random-detect dscp 8 750000 bytes 750000 bytes`	Removes the previously configured WRED parameters for a particular DSCP value or IP precedence.
Step 10	Do one of the following: • **random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **random-detect precedence** *precedence* {*min-threshold max-threshold*| *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator* **Example:** `Router(config-pmap-c)# random-detect dscp 8 4000 4000`	Configures WRED parameters for a particular DSCP value or IP precedence. **Note** Use the *min-threshold max-threshold* arguments without the **bytes** keyword to configure packet-based thresholds, when the queue-limit unit is also packets (the default). Alternatively, use these arguments with the **bytes** keyword when the queue-limit unit is configured in bytes.

Examples

The following example shows how to change the queue depth and WRED thresholds to packet-based values once a service policy has been applied to an interface:

```
interface GigabitEthernet1/2/0
no service-policy output main-interface-policy
end
policy-map main-interface-policy
class AF1
    queue-limit 5000 packets
    no random-detect dscp 8 750000 bytes 750000 bytes
    random-detect dscp 8 4000 4000
```

Verifying the Configuration for Byte-Based WRED

SUMMARY STEPS

1. **show policy-map**
2. The **show policy-map interface** command shows output for an interface that is configured for byte-based WRED.

DETAILED STEPS

Step 1 **show policy-map**

The **show policy-map** command shows the output for a service policy called pol1 that is configured for byte-based WRED.

Example:

```
Router# show policy-map
Policy Map pol1
  Class class c1
Bandwidth 10 (%)
exponential weight 9
        class   min-threshold(bytes)   max-threshold(bytes)  mark-probability
        -------------------------------------------------------------------

        0       -                       -                     1/10
        1       20000                   30000                 1/10
        2       -                       -                     1/10
        3       -                       -                     1/10
        4       -                       -                     1/10
        5       -                       -                     1/10
        6       -                       -                     1/10
        7       -                       -                     1/10
        rsvp    -                       -                     1/10
```

Step 2 The **show policy-map interface** command shows output for an interface that is configured for byte-based WRED.

Example:

```
Router# show policy-map interface
serial3/1
Service-policy output: pol
Class-map: silver (match-all)
366 packets, 87840 bytes
30 second offered rate 15000 bps, drop rate 300 bps
Match: ip precedence 1
Queueing
Output Queue: Conversation 266
Bandwidth 10 (%)
(pkts matched/bytes matched) 363/87120
depth/total drops/no-buffer drops) 147/38/0
exponential weight: 9
mean queue depth: 25920
class     Transmitted       Random drop      Tail drop     Minimum Maximum Mark
          pkts/bytes        pkts/bytes       pkts/bytes    thresh  thresh  prob
                                                           (bytes)  (bytes)
0             0/0               0/0              0/0       20000    40000  1/10
1           328/78720          38/9120           0/0       22000    40000  1/10
2             0/0               0/0              0/0       24000    40000  1/10
```

```
3            0/0              0/0              0/0      26000    40000  1/10
4            0/0              0/0              0/0      28000    40000  1/10
```

Configuration Examples for Byte-Based Weighted Random Early Detection

Example Configuring Byte-Based WRED

The following example shows a service policy called wred-policy that sets up byte-based WRED for a class called prec2 and for the default class. The policy is then applied to Fast Ethernet interface 0/0/1.

```
policy wred-policy
 class prec2
   bandwidth 1000
   random-detect
   random-detect precedence 2 100 bytes 200 bytes 10
class class-default
   random-detect
   random-detect precedence 4 150 bytes 300 bytes 15
   random-detect precedence 6 200 bytes 400 bytes 5
interface fastethernet0/0/1
  service-policy output wred-policy
```

The following example shows the byte-based WRED results for the service policy attached to Ethernet interface 0/0/1.

```
Router# show policy-map interface
 Ethernet0/0/1
Service-policy output: wred-policy (1177)
Class-map: prec2 (match-all) (1178/10)
0 packets, 0 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: ip precedence 2  (1179)
Queueing
queue limit 62500 bytes
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts queued/bytes queued) 0/0
bandwidth 1000 (kbps)
Exp-weight-constant: 9 (1/512)
Mean queue depth: 0 bytes
class     Transmitted       Random drop      Tail drop Minimum        Maximum     Mark
          pkts/bytes        pkts/bytes       pkts/bytes thresh         thresh     prob
                                                        bytes          bytes
0             0/0               0/0              0/0     15625          31250     1/10
1             0/0               0/0              0/0     17578          31250     1/10
2             0/0               0/0              0/0       100            200     1/10
3             0/0               0/0              0/0     21484          31250     1/10
4             0/0               0/0              0/0     23437          31250     1/10
5             0/0               0/0              0/0     25390          31250     1/10
6             0/0               0/0              0/0     27343          31250     1/10
7             0/0               0/0              0/0     29296          31250     1/10
Class-map: class-default (match-any) (1182/0)
0 packets, 0 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: any  (1183)
0 packets, 0 bytes
5 minute rate 0 bps
queue limit 562500 bytes
```

```
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts queued/bytes queued) 0/0
Exp-weight-constant: 9 (1/512)
Mean queue depth: 0 bytes
class     Transmitted       Random drop      Tail drop Minimum       Maximum     Mark
          pkts/bytes        pkts/bytes       pkts/bytes thresh        thresh     prob
                                                         bytes        bytes
0             0/0               0/0              0/0     140625       281250     1/10
1             0/0               0/0              0/0     158203       281250     1/10
2             0/0               0/0              0/0     175781       281250     1/10
3             0/0               0/0              0/0     193359       281250     1/10
4             0/0               0/0              0/0        150          300     1/15
5             0/0               0/0              0/0     228515       281250     1/10
6             0/0               0/0              0/0        200          400     1/5
7             0/0               0/0              0/0     263671       281250     1/10
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS Commands	*Cisco IOS Quality of Service Solutions Command Reference*
Modular QoS CLI	Modular Quality of Service Command-Line Interface module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Byte-Based Weighted Random Early Detection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3 Feature Information for Byte-Based Weighted Random Early Detection

Feature Name	Releases	Feature Information
Byte-Based Weighted Random Early Detection	Cisco IOS XE Release 2.4	The Byte-Based Weighted Random Early Detection feature extends the functionality of WRED. In previous releases, you specified the WRED actions based on the number of packets. With the byte-based WRED, you can specify WRED actions based on the number of bytes. This feature was introduced on Cisco ASR 1000 Series Routers. The following commands were introduced or modified: **random-detect**, **random-detect precedence**, **show policy-map, show policy-map interface**.

WRED Explicit Congestion Notification

- Finding Feature Information, page 33
- Prerequisites for WRED-Explicit Congestion Notification, page 33
- Information About WRED-Explicit Congestion Notification, page 33
- How to Configure WRED-Explicit Congestion Notification, page 36
- Configuration Examples for WRED-Explicit Congestion Notification, page 38
- Additional References, page 40
- Feature Information for WRED Explicit Congestion Notification, page 41

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for WRED-Explicit Congestion Notification

ECN must be configured through the Modular Quality of Service Command-Line Interface (MQC). For more information about the MQC, see the "Applying QoS Features Using the MQC" module.

Information About WRED-Explicit Congestion Notification

- WRED-Explicit Congestion Notification Feature Overview, page 33
- How WRED Works, page 34
- ECN Extends WRED Functionality, page 34
- Benefits of WRED Explicit Congestion Notification, page 35

WRED-Explicit Congestion Notification Feature Overview

Currently, the congestion control and avoidance algorithms for Transmission Control Protocol (TCP) are based on the idea that packet loss is an appropriate indication of congestion on networks transmitting data using the best-effort service model. When a network uses the best-effort service model, the network delivers data if it can, without any assurance of reliability, delay bounds, or throughput. However, these algorithms and the best-effort service model are not suited to applications that are sensitive to delay or packet loss (for instance, interactive traffic including Telnet, web-browsing, and transfer of audio and video data). Weighted Random Early Detection (WRED), and by extension, Explicit Congestion Notification (ECN), helps to solve this problem.

RFC 3168, *The Addition ofExplicit Congestion Notification (ECN) to IP*, states that with the addition of active queue management (for example, WRED) to the Internet infrastructure, routers are no longer limited to packet loss as an indication of congestion.

How WRED Works

WRED makes early detection of congestion possible and provides a means for handling multiple classes of traffic. WRED can selectively discard lower priority traffic when the router begins to experience congestion and provide differentiated performance characteristics for different classes of service. It also protects against global synchronization. Global synchronization occurs as waves of congestion crest, only to be followed by periods of time during which the transmission link is not used to capacity. For these reasons, WRED is useful on any output interface or router where congestion is expected to occur.

WRED is implemented at the core routers of a network. Edge routers assign IP precedences to packets as the packets enter the network. With WRED, core routers then use these precedences to determine how to treat different types of traffic. WRED provides separate thresholds and weights for different IP precedences, enabling the network to provide different qualities of service, in regard to packet dropping, for different types of traffic. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

For more information about WRED, refer to the "Congestion Avoidance Overview" module.

ECN Extends WRED Functionality

WRED drops packets, based on the average queue length exceeding a specific threshold value, to indicate congestion. ECN is an extension to WRED in that ECN marks packets instead of dropping them when the average queue length exceeds a specific threshold value. When configured with the WRED -- Explicit Congestion Notification feature, routers and end hosts would use this marking as a signal that the network is congested and slow down sending packets.

As stated in RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP*,implementing ECN requires an ECN-specific field that has two bits--the ECN-capable Transport (ECT) bit and the CE (Congestion Experienced) bit--in the IP header. The ECT bit and the CE bit can be used to make four ECN field combinations of 00 to 11. The first number is the ECT bit and the second number is the CE bit. The table below lists each of the ECT and CE bit combination settings in the ECN field and what the combinations indicate.

Table 4 ECN Bit Setting

ECT Bit	CE Bit	Combination Indicates
0	0	Not ECN-capable
0	1	Endpoints of the transport protocol are ECN-capable

ECT Bit	CE Bit	Combination Indicates
1	0	Endpoints of the transport protocol are ECN-capable
1	1	Congestion experienced

The ECN field combination 00 indicates that a packet is not using ECN.

The ECN field combinations 01 and 10--called ECT(1) and ECT(0), respectively--are set by the data sender to indicate that the endpoints of the transport protocol are ECN-capable. Routers treat these two field combinations identically. Data senders can use either one or both of these two combinations. For more information about these two field combinations, and the implications of using one over the other, refer to RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP*.

The ECN field combination 11 indicates congestion to the endpoints. Packets arriving a full queue of a router will be dropped.

- How Packets Are Treated When ECN Is Enabled, page 35

How Packets Are Treated When ECN Is Enabled

- If the number of packets in the queue is below the minimum threshold, packets are transmitted. This happens whether or not ECN is enabled, and this treatment is identical to the treatment a packet receives when WRED only is being used on the network.
- If the number of packets in the queue is between the minimum threshold and the maximum threshold, one of the following three scenarios can occur:
    - If the ECN field on the packet indicates that the endpoints are ECN-capable (that is, the ECT bit is set to 1 and the CE bit is set to 0, or the ECT bit is set to 0 and the CE bit is set to 1)--and the WRED algorithm determines that the packet should have been dropped based on the drop probability--the ECT and CE bits for the packet are changed to 1, and the packet is transmitted. This happens because ECN is enabled and the packet gets marked instead of dropped.
    - If the ECN field on the packet indicates that neither endpoint is ECN-capable (that is, the ECT bit is set to 0 and the CE bit is set to 0), the packet may be dropped based on the WRED drop probability. This is the identical treatment that a packet receives when WRED is enabled without ECN configured on the router.
    - If the ECN field on the packet indicates that the network is experiencing congestion (that is, both the ECT bit and the CE bit are set to 1), the packet is transmitted. No further marking is required.
- If the number of packets in the queue is above the minimum threshold, packets are dropped based on the drop probability. This is the identical treatment a packet receives when WRED is enabled without ECN configured on the router.

Benefits of WRED Explicit Congestion Notification

Improved Method for Congestion Avoidance

This feature provides an improved method for congestion avoidance by allowing the network to mark packets for transmission later, rather than dropping them from the queue. Marking the packets for transmission later accommodates applications that are sensitive to delay or packet loss and provides improved throughput and application performance.

Enhanced Queue Management

Currently, dropped packets indicate that a queue is full and that the network is experiencing congestion. When a network experiences congestion, this feature allows networks to mark the IP header of a packet with a CE bit. This marking, in turn, triggers the appropriate congestion avoidance mechanism and allows the network to better manage the data queues. With this feature, ECN-capable routers and end hosts can respond to congestion before a queue overflows and packets are dropped, providing enhanced queue management.

How to Configure WRED-Explicit Congestion Notification

- Configuring Explicit Congestion Notification, page 36
- Verifying the Explicit Congestion Notification Configuration, page 37

Configuring Explicit Congestion Notification

To configure ECN, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**}
5. **bandwidth** {*bandwidth-kbps* | **percent** *percent*
6. **random-detect**
7. **random-detect ecn**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy. Enters QoS policy-map configuration mode. • Enter the name of the policy map.
Step 4	**class** {*class-name*| **class-default**} **Example:** `Router(config-pmap)# class class-default`	Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Enters policy-map-class configuration mode. • Enter the name of the class or enter the **class-default**keyword.

Step 5	**bandwidth** {*bandwidth-kbps* | **percent** *percent*
	
	Example:
	
	`Router(config-pmap-c)#`
	
	`bandwidth percent 35`
		Specifies or modifies the bandwidth (either in kbps or a percentage) allocated for a class belonging to a policy map.
		
		• Enter the bandwidth in kilobytes per second or enter the bandwidth percentage.

Step 6	**random-detect**
	
	Example:
	
	`Router(config-pmap-c)#`
	
	`random-detect`
		Enables WRED or distributed WRED (dWRED).

Step 7	**random-detect ecn**
	
	Example:
	
	`Router(config-pmap-c)#`
	
	`random-detect ecn`
		Enables ECN.

Step 8	**end**
	
	Example:
	
	`Router(config-pmap-c)#`
	
	`end`
		(Optional) Exits policy-map class configuration mode.

Verifying the Explicit Congestion Notification Configuration

To verify the ECN configuration, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **show policy-map**
3. **show policy-map interface**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map** **Example:** `Router# show policy-map`	If ECN is enabled, displays ECN marking information for a specified policy map.
Step 3	**show policy-map interface** **Example:** `Router# show policy-map interface`	If ECN is enabled, displays ECN marking information for a specified interface.

Step 4	**end**
	
	Example:
	
	`Router#`
	`end`
		(Optional) Exits privileged EXEC mode.

Configuration Examples for WRED-Explicit Congestion Notification

Example Enabling ECN

The following example enables ECN in the policy map called pol1:

```
Router(config)# policy-map pol1
 Router(config-pmap)# class class-default
```

```
Router(config-pmap-c)# bandwidth per 70
Router(config-pmap-c)# random-detect
Router(config-pmap-c)# random-detect ecn
```

Example Verifying the ECN Configuration

The following is sample output from the **show policy-map** command. The words "explicit congestion notification" (along with the ECN marking information) included in the output indicate that ECN has been enabled.

```
Router# show policy-map
   Policy Map pol1
     Class class-default
       Weighted Fair Queueing
             Bandwidth 70 (%)
             exponential weight 9
             explicit congestion notification
             class    min-threshold    max-threshold    mark-probability
             ----------------------------------------------------------
             ----------------------------------------------------------

             0        -                -                1/10
             1        -                -                1/10
             2        -                -                1/10
             3        -                -                1/10
             4        -                -                1/10
             5        -                -                1/10
             6        -                -                1/10
             7        -                -                1/10
             rsvp     -                -                1/10
```

The following is sample output from the **show policy-map interface**command. The words "explicit congestion notification" included in the output indicate that ECN has been enabled.

```
Router# show policy-map interface
 Serial4/1
 Serial4/1
  Service-policy output:policy_ecn
        Class-map:prec1 (match-all)
          1000 packets, 125000 bytes
          30 second offered rate 14000 bps, drop rate 5000 bps
          Match:ip precedence 1
          Weighted Fair Queueing
            Output Queue:Conversation 42
            Bandwidth 20 (%)
            Bandwidth 100 (kbps)
            (pkts matched/bytes matched) 989/123625
        (depth/total drops/no-buffer drops) 0/455/0
             exponential weight:9
             explicit congestion notification
             mean queue depth:0
     class   Transmitted  Random drop  Tail drop   Minimum     Maximum     Mark
             pkts/bytes   pkts/bytes    pkts/bytes threshold   threshold   probability
       0       0/0          0/0          0/0          20          40        1/10
       1     545/68125      0/0          0/0          22          40        1/10
       2       0/0          0/0          0/0          24          40        1/10
       3       0/0          0/0          0/0          26          40        1/10
       4       0/0          0/0          0/0          28          40        1/10
       5       0/0          0/0          0/0          30          40        1/10
       6       0/0          0/0          0/0          32          40        1/10
       7       0/0          0/0          0/0          34          40        1/10
     rsvp      0/0          0/0          0/0          36          40        1/10
     class   ECN Mark
            pkts/bytes
       0     0/0
       1    43/5375
       2     0/0
       3     0/0
       4     0/0
```

```
   5      0/0
   6      0/0
   7      0/0
rsvp      0/0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
MQC	"Applying QoS Features Using the MQC" module
Congestion avoidance concepts	"Congestion Avoidance Overview" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2309	*Internet Performance Recommendation*
RFC 2884	*Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks*
RFC 3168	*The Addition of Explicit Congestion Notification (ECN) to IP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for WRED Explicit Congestion Notification

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 5 Feature Information for WRED Explicit Congestion Notification

Feature Name	Software Releases	Feature Configuration Information
WRED Explicit Congestion Notification	Cisco IOS XE Release 2.1	Currently, the congestion control and avoidance algorithms for Transmission Control Protocol (TCP) are based on the idea that packet loss is an appropriate indication of congestion on networks transmitting data using the best-effort service model. When a network uses the best-effort service model, the network delivers data if it can, without any assurance of reliability, delay bounds, or throughput. However, these algorithms and the best-effort service model are not suited to applications that are sensitive to delay or packet loss (for instance, interactive traffic including Telnet, web-browsing, and transfer of audio and video data). Weighted Random Early Detection (WRED), and by extension, Explicit Congestion Notification (ECN), helps to solve this problem. The following commands were introduced or modified: **random-detect ecn**, **show policy-map**, **show policy-map interface**.

QoS Time-Based Thresholds for WRED and Queue Limit

The QoS Time-Based Thresholds for WRED and Queue Limit feature allows you to specify the Weighted Random Early Detection (WRED) minimum and maximum thresholds or the queue limit threshold in milliseconds (ms). Previously, these thresholds could only be specified in packets or bytes. Now, all three units of measure are available. Once the threshold limits are configured in a policy map, the policy map can be used on multiple interfaces, including those with different amounts of bandwidth.

- Finding Feature Information, page 43
- Prerequisites for QoS Time-Based Thresholds for WRED and Queue Limit, page 43
- Restrictions for QoS Time-Based Thresholds for WRED and Queue Limit, page 44
- Information About QoS Time-Based Thresholds for WRED and Queue Limit, page 44
- How to Configure QoS Time-Based Thresholds for WRED and Queue Limit, page 46
- Configuration Examples for QoS Time-Based Thresholds for WRED and Queue Limit, page 54
- Additional References, page 57
- Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit, page 58

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for QoS Time-Based Thresholds for WRED and Queue Limit

Before configuring this feature, a traffic class must be configured and a policy map must exist. To create the traffic class (specifying the appropriate match criteria) and the policy map, use the modular quality of service (QoS) command-line interface (MQC).

Restrictions for QoS Time-Based Thresholds for WRED and Queue Limit

This feature allows you to specify either the WRED thresholds or the queue limit threshold in packets (the default unit of measure), bytes, or milliseconds (ms). However, these units cannot be mixed. That is, the unit of measure in the *same* class, in the *same* policy map, cannot be mixed. For example, if you specify the minimum threshold for a particular class in milliseconds, the maximum threshold for that class must also be in milliseconds.

Information About QoS Time-Based Thresholds for WRED and Queue Limit

- Benefits of QoS Time-Based Thresholds for WRED and Queue Limit, page 44
- Setting Thresholds by Using WRED, page 44
- Setting Thresholds by Using the queue-limit Command, page 45
- random-detect Commands with the Milliseconds Keyword, page 45
- Mixing Threshold Units of Measure, page 46

Benefits of QoS Time-Based Thresholds for WRED and Queue Limit

Queue Limit Thresholds Specified in Additional Units of Measure

Previously, the WRED thresholds and the queue limit thresholds could only be specified in packets or bytes. With this feature, the thresholds can be specified either in packets, bytes or milliseconds. These additional units of measure provide more flexibility and allow you to fine-tune your configuration.

Policy Maps Can be Reused as Needed on Multiple Interfaces

The WRED and queue limit thresholds are specified and configured in policy maps. Once the threshold limits are configured in a policy map, the policy map can be used on multiple interfaces, including those with different amounts of bandwidth. This is especially useful when the bandwidth for a class on given interface is being specified as a percentage of the total bandwidth available.

Setting Thresholds by Using WRED

WRED is a congestion avoidance mechanism. WRED combines the capabilities of the Random Early Detection (RED) algorithm with the IP precedence feature to provide for preferential traffic handling of higher priority packets. WRED can selectively discard lower priority traffic when the interface begins to get congested and provide differentiated performance characteristics for different classes of service.

WRED differs from other congestion avoidance techniques such as queueing strategies because it attempts to anticipate and avoid congestion rather than control congestion once it occurs.

WRED is enabled by using the **random-detect**command. Then the minimum threshold, maximum threshold, and mark probability denominator can be set to determine the treatment that packets receive by using the appropriate command. For example, the **random-detect precedence** command can be used to determine the thresholds for a specific IP precedence.

Setting Thresholds by Using the queue-limit Command

The **queue-limit**command allows you to specify or modify the maximum number of packets the queue can hold (that is, the threshold) for a class policy configured in a policy map. Packets belonging to a class are subject to the guaranteed bandwidth allocation and the queue limits that characterize the traffic class. With the **queue-limit** command, the threshold is the aggregate threshold for the entire class.

After a queue has reached its configured queue limit, enqueuing of additional packets to the traffic class causes tail drop or WRED (if configured) to take effect, depending on how the policy map is configured. (Tail drop is a means of avoiding congestion that treats all traffic equally and does not differentiate between classes of service.)

Queues fill during periods of congestion. When the output queue is full and tail drop is in effect, packets are dropped until the congestion is eliminated and the queue is no longer full).

Tail drop is used for distributed class-based weighted fair queueing (DCBWFQ) traffic classes unless you explicitly configure a service policy to use WRED to drop packets as a means of avoiding congestion. Note that if you use WRED instead of tail drop for one or more traffic classes making up a service policy, you must ensure that WRED is not configured for the interface to which you attach that service policy.

random-detect Commands with the Milliseconds Keyword

This feature allows you to specify the WRED minimum and maximum thresholds in milliseconds (ms). You can specify the threshold in milliseconds by using the **ms** keyword available with the **random-detect** commands listed in the table below.

Table 6 random-detect Commands with the Milliseconds (ms) Keyword

Command	Description
random-detect clp	Configures the WRED parameters for a particular cell loss priority (CLP) value, or a particular CLP value for a class policy in a policy map.
random-detect cos	Configures the WRED parameters for a particular class of service (CoS) value, or a particular CoS value for a class policy in a policy map.
random-detect discard-class	Configures the WRED parameters for a particular discard-class, or a particular discard-class for a class policy in a policy map.
random-detect dscp	Configures the WRED parameters for a particular differentiated services code point (DSCP) value, or a particular DSCP value for a class policy in a policy map.
random-detect precedence	Configures WRED parameters for a particular IP precedence, or a particular IP precedence for a class policy in a policy map.

Mixing Threshold Units of Measure

With this feature, the thresholds can be specified in packets (the default unit of measure), bytes, or milliseconds (ms). For instance, with WRED, you can specify the minimum threshold and the maximum threshold in packets, bytes, or milliseconds. However, the units cannot be mixed. For example, if you specify the minimum threshold in milliseconds, the maximum threshold must also be specified in milliseconds.

How to Configure QoS Time-Based Thresholds for WRED and Queue Limit

- Enabling WRED and Using WRED to Specify Thresholds, page 46
- Using the queue-limit Command to Specify the Thresholds, page 49
- Attaching the Policy Map to an Interface in a QoS Time-Based Threshold for WRED Configuration, page 51
- Verifying the QoS Time-Based Thresholds for WRED and Queue Limit Configuration, page 52

Enabling WRED and Using WRED to Specify Thresholds

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-name*
4. **class** {*class-name***class-default**}
5. To continue with the configuration, you must either specify a bandwidth (Enabling WRED and Using WRED to Specify Thresholds, page 46) or enable traffic shaping (Enabling WRED and Using WRED to Specify Thresholds, page 46). Choose one or the other.
6. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
7. **shape** [**average** | **peak**] *mean-rate* [*burst-size*] [*excess-burst-size*]
8. **random-detect**
9. **random-detect precedence** {*precedence* | **rsvp**} *min-threshold* {**bytes**| **ms**| **packets**} *max-threshold*{**bytes** | **ms**| **packets**} [*mark-probability-denominator*]
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-name*
	
	Example:
	
	`Router(config)#`
	
	`policy-map`
	
	`policy1`
		Specifies the name of the policy map to be created. Enters policy-map configuration mode.
		
		• Enter policy map name.

Step 4	**class** {*class-name***class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	To continue with the configuration, you must either specify a bandwidth (Enabling WRED and Using WRED to Specify Thresholds, page 46) or enable traffic shaping (Enabling WRED and Using WRED to Specify Thresholds, page 46). Choose one or the other.	
Step 6	**bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 40`	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the bandwidth to be set or modified.

	Command or Action	Purpose
Step 7	**shape** [**average** | **peak**] *mean-rate* [*burst-size*] [*excess-burst-size*] **Example:** `Router(config-pmap-c)# shape average 51200`	(Optional) Enables either average or peak rate traffic shaping. • Specify either average or peak traffic shaping.
Step 8	**random-detect** **Example:** `Router(config-pmap-c)# random-detect`	Enables WRED or distributed WRED (DWRED).
Step 9	**random-detect precedence** {*precedence* | **rsvp**} *min-threshold* {**bytes**| **ms**| **packets**} *max-threshold*{**bytes** | **ms**| **packets**} [*mark-probability-denominator*] **Example:** `Router(config-pmap-c)# random-detect precedence 2 512 ms 1020 ms`	Configures WRED and DWRED parameters for a particular IP precedence. • Specify the IP precedence or RSVP value, and thresholds, as needed. **Note** In this example, the WRED parameters were specified for traffic with a specific IP precedence value. Other values can be specified with other **random-detect**commands. For a list of the other **random-detect** commands, see Enabling WRED and Using WRED to Specify Thresholds, page 46.
Step 10	**exit** **Example:** `Router(config-pmap-c)# exit`	(Optional) Exits policy-map class configuration mode.

Using the queue-limit Command to Specify the Thresholds

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-name*
4. **class** {*class-name* **class-default**}
5. To continue with the configuration, you must either specify a bandwidth (Using the queue-limit Command to Specify the Thresholds, page 49) or enable traffic shaping (Using the queue-limit Command to Specify the Thresholds, page 49). Choose one or the other.
6. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
7.
8. **shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]]
9. **queue-limit** *number-of-packets* [**bytes** | **ms** | **packets**]
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-name*
	
	Example:
	`Router(config)#`
	`policy-map`
	`policy1`
		Specifies the name of the policy map to be created. Enters policy-map configuration mode.
		• Enter policy map name.

	Command or Action	Purpose
Step 4	**class** {*class-name* **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	To continue with the configuration, you must either specify a bandwidth (Using the queue-limit Command to Specify the Thresholds, page 49) or enable traffic shaping (Using the queue-limit Command to Specify the Thresholds, page 49). Choose one or the other.	
Step 6	**bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 40`	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the bandwidth to be set or modified.
Step 7		
Step 8	**shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]] **Example:** `Router(config-pmap-c)# shape average 51200`	(Optional) Enables either average or peak rate traffic shaping. • Specifies either average or peak traffic shaping.
Step 9	**queue-limit** *number-of-packets* [**bytes** | **ms** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 200 ms`	Specifies or modifies the maximum number of packets the queue can hold (that is, the queue limit) for a class configured in a policy map. • Enter the queue limit. The unit of measure can be bytes, milliseconds, or packets.
Step 10	**exit** **Example:** `Router(config-pmap-c)# exit`	(Optional) Exits policy-map class configuration mode.

Attaching the Policy Map to an Interface in a QoS Time-Based Threshold for WRED Configuration

Note Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi* / *vci* [**ilmi** | **qsaal** | **smds**]
5. **service-policy** {**input**| **output**} *policy-map-name*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**interface** *type number*
	
	Example:
	`Router(config)#`
	`interface serial4/0`
		Configures an interface (or subinterface) type and enters interface configuration mode.
		• Enter the interface type number.

Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi** | **qsaal** | **smds**] **Example:** `Router(config-if)# pvc cisco 0/16 ilmi`	(Optional) Creates or assigns a name to an ATM PVC and specifies the encapsulation type on an ATM PVC. Enters ATM VC configuration mode. **Note** This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, skip this step and proceed with Step 5 .

	Command or Action	Purpose

Step 5	**service-policy** {**input**| **output**} *policy-map-name*
	
	Example:
	`Router(config-if)#`
	`service-policy output policy1`
	
	Example:
		Specifies the name of the policy map to be attached to the input *or* output direction of the interface.
		
		Note Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached varies according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
		
		• Enter the policy map name.

Step 6	**exit** **Example:** `Router(config-if)# exit`	(Optional) Exits interface configuration mode.

Verifying the QoS Time-Based Thresholds for WRED and Queue Limit Configuration

SUMMARY STEPS

1. **enable**
2. **show policy-map** [*policy-map*]
3. and/or
4. **show policy-map interface** *interface-name*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show policy-map** [*policy-map*] **Example:** `Router# show policy-map policy1`	Displays all information about a class map, including the match criterion. • Enter class map name.
Step 3	and/or	

Step 4	**show policy-map interface** *interface-name*
	
	Example:
	
	`Router#`
	`show policy-map interface serial4/0`
		Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		
		• Enter the interface name.

Step 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

• Troubleshooting Tips, page 53

Troubleshooting Tips

The commands in the "Verifying the Configuration" section allow you to verify that you achieved the intended configuration and that the feature is functioning correctly. If, after using the **show** commands listed above, you find that the configuration is not correct or the feature is not functioning as expected, perform these operations:

If the configuration is not the one you intended, complete the following steps:

1. Use the **show running-config** command and analyze the output of the command.
2. If the policy map does not appear in the output of the **show running-config** command, enable the **logging console** command.
3. Attach the policy map to the interface again.

If the packets are not being matched correctly (for example, the packet counters are not incrementing correctly), complete the following procedures:

1. Run the **show policy-map**command and analyze the output of the command.
2. Run the **show running-config** command and analyze the output of the command.
3. Use the **show policy-map interface** command and analyze the output of the command. Check the the following findings:

    a. If a policy map applies queueing, and the packets are matching the correct class, but you see unexpected results, compare the number of the packets in the queue with the number of the packets matched.

If the interface is congested, and only a small number of the packets are being matched, check the tuning of the transmission (tx) ring, and evaluate whether the queueing is happening on the tx ring. To do this, use the **show controllers** command, and look at the value of the tx count in the output of the command..

Configuration Examples for QoS Time-Based Thresholds for WRED and Queue Limit

Example Using WRED to Set Thresholds

In the following example, WRED has been configured in the policy map called "policy1". In this WRED configuration, the bandwidth has been specified as a percentage (80%), and the minimum and maximum thresholds for IP precedence 2 are set to 512 milliseconds and 1020 milliseconds, respectively.

```
Router> enable
Router# configure terminal
Router(config)#

policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 80
Router(config-pmap-c)# random-detect
Router(config-pmap-c)# random-detect precedence 2 512 ms 1020 ms
Router(config-pmap-c)# exit

Router(config-pmap)# exit

Router(config)# interface s4/0
Router(config-if)#

service-policy output policy1
Router(config-if)# end
```

Example Using the queue-limit Command to Set Thresholds

In the following example, a policy map called "policy2" has been configured. The policy2 policy map contains a class called "class1." The bandwidth for this class has been specified as a percentage (80%) and the **queue-limit** command has been used to set the threshold to 200 milliseconds.

```
Router> enable
Router# configure terminal
Router(config)#

policy-map policy2
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 80
Router(config-pmap-c)# queue-limit 200 ms
Router(config-pmap-c)# exit

Router(config-pmap)# exit

Router(config)# interface s4/0
Router(config-if)#

service-policy output policy1
Router(config-if)# end
```

Example Verifying the Configuration

To verify that this feature is configured correctly, use either the **show policy-map**command or the **show policy-map interface** command.

This section contains two sets of sample output from the **show policy-map interface** command and the **show policy-map** command--one set showing the output when WRED is used to configure the feature, one set showing the output when the **queue-limit**command is used to configure the feature.

Example WRED Threshold Configuration Sample Output

The following is sample output of the **show policy-map** command when WRED has been used to specify the thresholds. The words "time-based wred" indicates that the thresholds have been specified in milliseconds (ms).

```
Router# show policy-map
  Policy Map policy1
    Class class1
      bandwidth 80 (%)
       time-based wred, exponential weight 9
      class    min-threshold    max-threshold    mark-probability
      ----------------------------------------------------------
      0       -                -                1/10
      1       -                -                1/10
      2       512              1024             1/10
      3       -                -                1/10
      4       -                -                1/10
      5       -                -                1/10
      6       -                -                1/10
      7       -                -                1/10
```

The following is sample output of the **show policy-map interface** command when WRED has been used to specify the thresholds.

```
Router# show policy-map interface Ethernet2/0
 Ethernet2/0
Service-policy output: policy1 (1100)
    Class-map: class1 (match-all) (1101/1)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol ftp (1102)
      Queueing
      queue limit 16 ms/ 16000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
      bandwidth 80.00% (%) (8000 kbps)
        Exp-weight-constant: 9 (1/512)
        Mean queue depth: 0 ms/ 0 bytes
        class    Transmitted  Random drop   Tail drop    Minimum       Maximum       Mark
                 pkts/bytes   pkts/bytes    pkts/bytes   thresh        thresh        prob
                                                         ms/bytes      ms/bytes
        0        0/0          0/0           0/0           4/4000        8/8000       1/10
        1        0/0          0/0           0/0           4/4500        8/8000       1/10
        2        0/0          0/0           0/0         512/512000   1024/1024000    1/10
        3        0/0          0/0           0/0           5/5500        8/8000       1/10
        4        0/0          0/0           0/0           6/6000        8/8000       1/10
        5        0/0          0/0           0/0           6/6500        8/8000       1/10
        6        0/0          0/0           0/0           7/7000        8/8000       1/10
        7        0/0          0/0           0/0           7/7500        8/8000       1/10
    Class-map: class-default (match-any) (1105/0)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any  (1106)
        0 packets, 0 bytes
        5 minute rate 0 bps
```

```
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
```

Formula for Converting the Threshold from Milliseconds to Bytes

When converting the threshold from milliseconds to bytes, the following formula is used:

milliseconds * (bandwidth configured for the class) / 8 = total number of bytes

For this example, the following numbers would be used in the formula:

512 ms * 8000 kbps / 8 = 512000 bytes

Note Class1 has a bandwidth of 8000 kbps.

Example queue-limit command Threshold Configuration Sample Output

The following is sample output of the **show policy-map** command when the **queue-limit** command has been used to specify the thresholds in milliseconds.

```
Router# show policy-map
  Policy Map policy1
    Class class1
      bandwidth 80 (%)
      queue-limit 200 ms
```

The following is sample output from the **show policy-map interface** command when the **queue-limit** command has been used to specify the thresholds.

```
Router# show policy-map interface
 Ethernet2/0
  Service-policy output: policy1 (1070)
    Class-map: class1 (match-all) (1071/1)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol ftp (1072)
      Queueing
      queue limit 200 ms/ 200000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
      bandwidth 80.00% (%) (8000 kbps)
    Class-map: class-default (match-any) (1075/0)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
Match: any  (1076)
        0 packets, 0 bytes
        5 minute rate 0 bps

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
```

Formula for Converting the Threshold from Milliseconds to Bytes

When converting the threshold from milliseconds to bytes, the following formula is used:

milliseconds * (bandwidth configured for the class)/ 8 = total number of bytes

For this example, the following numbers would be used in the formula:

200 ms * 8000 kbps / 8 = 200000 bytes

Note Class1 has a bandwidth of 8000 kbps.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of service (QoS) commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Congestion avoidance mechanisms, including tail drop, RED and WRED	*Cisco IOS Quality of Service Solutions Configuration Guide*
Congestion management mechanisms, including CBWFQ, and DCBWFQ	*Cisco IOS Quality of Service Solutions Configuration Guide*
Byte-Based WRED	Byte-Based Weight Random Early Detection module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7 Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit

Feature Name	Releases	Feature Information
QoS Time-Based Thresholds for WRED and Queue Limit	Cisco IOS XE Release 3.2S	The QoS Time-Based Thresholds for WRED and Queue Limit feature allows you to specify the Weighted Random Early Detection (WRED) minimum and maximum thresholds or the queue limit threshold in milliseconds (ms). The following commands are introduced or modified: **queue-limit, random-detect precedence, show policy-map, show policy-map interface.**