Setting Up Quality of Service

Contents

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>8-4</td>
</tr>
<tr>
<td>Evaluating Traffic on Your Network</td>
<td>8-4</td>
</tr>
<tr>
<td>QoS Mechanisms on the ProCurve Secure Router</td>
<td>8-5</td>
</tr>
<tr>
<td>ToS Field</td>
<td>8-6</td>
</tr>
<tr>
<td>First In, First Out</td>
<td>8-10</td>
</tr>
<tr>
<td>WFQ</td>
<td>8-11</td>
</tr>
<tr>
<td>CBWFQ</td>
<td>8-11</td>
</tr>
<tr>
<td>LLQ</td>
<td>8-11</td>
</tr>
<tr>
<td>FRF.12</td>
<td>8-12</td>
</tr>
<tr>
<td>QoS Maps</td>
<td>8-12</td>
</tr>
<tr>
<td>Configuring WFQ</td>
<td>8-14</td>
</tr>
<tr>
<td>Overview</td>
<td>8-14</td>
</tr>
<tr>
<td>Conversations</td>
<td>8-14</td>
</tr>
<tr>
<td>Weight</td>
<td>8-15</td>
</tr>
<tr>
<td>Shortcomings</td>
<td>8-15</td>
</tr>
<tr>
<td>Packet Marking</td>
<td>8-16</td>
</tr>
<tr>
<td>Enabling WFQ</td>
<td>8-17</td>
</tr>
<tr>
<td>Setting the Queue Size</td>
<td>8-18</td>
</tr>
<tr>
<td>Configuring CBWFQ</td>
<td>8-19</td>
</tr>
<tr>
<td>Overview</td>
<td>8-19</td>
</tr>
<tr>
<td>Configuring Classes for CBWFQ</td>
<td>8-19</td>
</tr>
<tr>
<td>Creating a QoS Map Entry</td>
<td>8-20</td>
</tr>
<tr>
<td>Defining a Class</td>
<td>8-21</td>
</tr>
<tr>
<td>Allocating Bandwidth to a Class</td>
<td>8-26</td>
</tr>
<tr>
<td>Assigning the QoS Map to an Interface</td>
<td>8-28</td>
</tr>
<tr>
<td>Special Considerations for CBWFQ with Multilinks</td>
<td>8-29</td>
</tr>
<tr>
<td>CBWFQ Example Configuration</td>
<td>8-30</td>
</tr>
</tbody>
</table>
Setting Up Quality of Service

Contents

Configuring LLQ ................................................................. 8-32
  Overview ................................................................. 8-32
  Determining Bandwidth for the Queue ......................... 8-32
    Determining Bandwidth for VoIP .............................. 8-33
    Determining Bandwidth for Video Streaming ................. 8-36
  Placing Traffic in a Low-Latency Queue ....................... 8-37
    Creating a QoS Map Entry ........................................ 8-37
    Selecting the Traffic to Be Placed in the Low-Latency
    Queue ..................................................................... 8-37
    Setting the Bandwidth Guaranteed the Queue .............. 8-42
    Marking Low Latency Packets with a ToS Value ............ 8-44
    Assigning the QoS Map to an Interface ...................... 8-44

Marking Packets with a ToS value .................................... 8-45
  Creating a QoS Map Entry ............................................. 8-45
  Selecting the Traffic to Be Marked ............................... 8-46
  Setting the ToS Value ................................................ 8-50
  Assigning the QoS Map to an Interface ......................... 8-51
  Example Packet Marking Configuration ......................... 8-51

Configuring Rate Limiting for Frame Relay ....................... 8-53
  Overview ................................................................. 8-53
  Rate Limiting ........................................................... 8-53
    FRF.12 .................................................................... 8-53
  Configuring Rate Limiting .......................................... 8-54
    Setting the Committed Burst Rate ............................ 8-55
    Setting the Excessive Burst Rate .............................. 8-55
  Configuring Frame Relay Fragmentation ....................... 8-56
  Example Frame Relay QoS Configuration ....................... 8-57

Configuring QoS for Ethernet ........................................... 8-58
  Overview ................................................................. 8-58
  Rate Limiting ........................................................... 8-58
  Configuring Rate Limiting on an Ethernet Interface .......... 8-58
  Configuring QoS Policies on an Ethernet Interface ........... 8-59
Example: Configuring QoS for VoIP ........................................... 8-61
   Enabling Application-Level Gateways for Applications
   with Special Needs .............................................. 8-62
   Enabling SIP Services ........................................... 8-62
   Defining VoIP Traffic .......................................... 8-64
   Determining the Required Bandwidth .......................... 8-65
   Marking Signaling Traffic for Special Treatment .......... 8-66
   Configuring Frame Relay Rate Limiting ...................... 8-67
Monitoring QoS ......................................................... 8-68
   Viewing QoS Maps .............................................. 8-69
   Managing Queues ................................................ 8-70
Troubleshooting Common Configuration Problems .......... 8-71
   A Map Becoming Inactive .................................... 8-71
   An Ethernet Interface Refusing to Take
   a QoS-Policy .................................................. 8-72
Quick Start ............................................................. 8-72
   Configuring WFQ ............................................... 8-72
   Configuring CBWFQ ............................................. 8-73
   Configuring a Low-Latency Queue ............................ 8-75
   Marking Packets ............................................... 8-76
   Configuring Frame Relay Fragmentation .................... 8-77
   Configuring QoS on an Ethernet Interface ................. 8-78
Overview

Quality of service (QoS) protocols allow a router to distinguish different classes of traffic and serve each class according to its priority and needs.

Evaluating Traffic on Your Network

Several factors define the QoS that traffic receives, including:

- bandwidth
- delay
- number of dropped packets

Clearly, high bandwidth is better than low bandwidth, and low delay and low numbers of dropped packets are better than long delay and high numbers of dropped packets.

However, sometimes you must balance these factors. A link may have low bandwidth, but also low delay. You should understand which factors are most important for managing QoS for different kinds of traffic.

A router must handle several kinds of traffic, including:

- Data—Data packets tend to be relatively large. They can be fragmented and reconstructed without damage to their integrity. Data traffic can tolerate high latency and is bursty: packets can be queued and sent at various speeds without excessively degrading QoS. For data, bandwidth is usually more important than delay in determining the QoS. For example, more bandwidth helps FTP applications download files more quickly. However, less delay at any given moment does not perceptibly increase the QoS because the entire file must be downloaded before it can be used.
- Real-time traffic—Real-time traffic requires low latency; it cannot tolerate delays. Real-time traffic includes Voice over IP (VoIP) and interactive traffic such as Telnet. Such packets are typically small, and they cannot be fragmented. As well as requiring low latency, voice and other real-time traffic must have low jitter: that is, the delay for each packet transmitted should be similar so that the receiver does not hear a difference in conversation.
- High-priority traffic—Mission-critical traffic should be guaranteed a certain amount of bandwidth and should not be dropped when congestion occurs.
Setting Up Quality of Service

Overview

- Control plane traffic—The router always reserves bandwidth for control traffic. This traffic, such as Open Shortest Path First (OSPF) hellos and routing updates, must run on the interface and will always be transmitted no matter what queuing method the interface implements.

You should configure different QoS mechanisms depending on the type of traffic the router is serving. For example, you should configure low latency queues for VoIP and other low-latency or high-priority traffic.

QoS Mechanisms on the ProCurve Secure Router

The ProCurve Secure Router supports:
- packet marking in the Type of Service (ToS) field of the IP header
- weighted fair queuing (WFQ)
- class-based WFQ (CBWFQ)
- low-latency queuing (LLQ) (also called high-priority queuing)
- Frame Relay Fragmentation 12 (FRF.12)

QoS mechanisms on a router regulate the window of time between when a packet arrives on an interface and when the router forwards it. The router must decide:
- the queue in which to place the packet
- the queue to service first
- the bandwidth to allocate each queue
- the packets to drop when the link is congested

The router makes these decisions by mapping certain traffic classifications to a certain type of service. One of the ways to classify traffic is to mark packets’ ToS fields. It is important for you to understand how packet marking interacts with QoS mechanisms. You should consider two issues during the following discussion:
- the ToS values assigned to packets
- how devices actually handle packets marked with a specific ToS value

The first issue is addressed by the IP precedence and Differentiated Service (DiffServ) standards, which associate settings in the ToS field with certain types of service. IP precedence simply defines a packet’s relative priority. DiffServ values can also define standards of treatment for certain classes.
Setting Up Quality of Service
Overview

However, neither IP precedence nor DiffServ addresses the second issue: how a router actually provides differentiated service. You must configure other protocols to provide the service requested by the ToS value. You can configure the ProCurve Secure Router to:

- grant traffic with a higher IP precedence value relatively more bandwidth using WFQ
- grant a class of traffic (which can be defined by ToS value or other criteria) a certain amount of an interface's bandwidth
- place traffic into a first-served queue that is guaranteed a certain transmission rate (LLQ)

ToS Field

The IPv4 header includes an 8-bit ToS field, which allows you to mark traffic for special handling. Two standards define how the ToS field defines traffic: IP precedence, the original standard for using this field, and DiffServ, which was introduced in 1998 with the Request for Comments (RFC) 2474. (See http://www.ietf.org/rfc/rfc2474.txt for more information.)

**IP Precedence.** IP precedence includes two subfields—a three-bit precedence field used for prioritization and a four-bit subfield for the specific type of service. The remaining bit is unused.

You can mark seven priority traffic classes in the three-bit precedence field.

Packets for which this field is unmarked (IP precedence 0) receive routine handling. Traffic with an IP precedence value of 1 takes simple priority over routine traffic. Traffic with an IP precedence value of 2 has an immediate priority, a value of 3 has a flash priority, and a value of 4 has a flash override priority. The highest user-defined value is 5, which has critical priority.

Values 6 and 7 are reserved for Internet and network use, ensuring that routing updates and other network traffic receive a higher priority than user-generated traffic. Because routes must be accurate to ensure delivery of other traffic, routing traffic must receive priority treatment. Value 6 is designed for use between networks, and it should only be implemented by gateway devices. Value 7 is for private network use; organizations should determine themselves what type of service IP precedence 7 implies and for what types of traffic it should be assigned.
The four ToS bits within the ToS field each request a different type of service from forwarding nodes:

- a one in the first bit requests low delay
- a one in the second bit requests high throughput
- a one in the third bit requests high reliability
- a one in the fourth bit requests low cost

**Note**

As you see above, a one in two of the ToS bits refers to a high setting, but in the other two to a low setting. You will not be confused if you remember that a one ToS bit always requests faster, better service.

In practice, networks rarely use the ToS bits. However, several protocols have emerged to grant packets differentiated service according to the IP precedence setting alone. These protocols include WFQ and LLQ.

The ProCurve Secure Router can both mark packets with an IP precedence value and grant packets a differentiated service depending on a previously marked value. It can:

- mark packets with an IP precedence value so that the network to which they are forwarded will grant them a specific type of service
- read packets’ IP precedence value for WFQ
- read packets’ IP precedence value and assign them to a class for CBWFQ
- read packets’ IP precedence and assign packets with a certain priority to a low-latency queue

**DiffServ.** The DiffServ protocol redefines the ToS field as the Differentiated Services (DS) field. It combines the three IP precedence bits and three of the four under-used ToS bits into a six-bit Differentiated Service Code Point (DSCP).

The DSCP supports 63 values to IP precedence’s seven. (In both, a zero value refers to routine traffic for which a priority has not been set.)

The last two bits of the DS field are reserved for flow control; these are the congestion experienced bits.
The DSCP marks packets for a specific per-hop behavior (PHB). PHBs describe forwarding behavior. That is, standards for PHBs determine such issues as which packets should be forwarded first and which packets should be dropped during network congestion. DiffServ defines four types of PHBs:

- **Default PHB**—The Default PHB is for traffic with DSCP 0 (not set) or any undefined DSCP. If a packet is configured for the Default PHB, the router uses best-effort service to process and forward that packet.

- **Class-Selector PHB**—The Class-Selector PHBs provide backward compatibility with IP precedence. In these PHBs, the last three bits in the DSCP are always set to zero, so only the first three bits (those that match the IP precedence bits) are significant for differentiating the eight classes. Network devices must grant each class the type of service given to the corresponding IP precedence value. Table 8-1 shows the DSCP for the Class-Selector PHB.

### Table 8-1. Class-selector PHBs

<table>
<thead>
<tr>
<th>DiffServ Value</th>
<th>DSCP</th>
<th>First 3 bits</th>
<th>IP Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000000</td>
<td>000</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>001000</td>
<td>001</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>010000</td>
<td>010</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>011000</td>
<td>011</td>
<td>3</td>
</tr>
<tr>
<td>32</td>
<td>100000</td>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>40</td>
<td>101000</td>
<td>101</td>
<td>5</td>
</tr>
<tr>
<td>48</td>
<td>110000</td>
<td>110</td>
<td>6</td>
</tr>
<tr>
<td>56</td>
<td>111000</td>
<td>111</td>
<td>7</td>
</tr>
</tbody>
</table>

- **Assured Forwarding PHB**—The Assured Forwarding PHBs allow you to create four traffic classes (AF1, AF2, AF3, and AF4) and assign different forwarding priorities to each. Values in the first three bits of the DSCP determine the AF class. Several DSCPs match each of the AF PHBs. Within each set of AF PHBs, different DSCPs can define different:
  - buffer space
  - bandwidth
  - drop precedence

Table 8-2 shows the DSCP for Assured Forwarding PHBs.
Setting Up Quality of Service

Overview

Table 8-2. Assured Forwarding PHB

<table>
<thead>
<tr>
<th>AF Class</th>
<th>Drop Precedence</th>
<th>DSCP</th>
<th>DiffServ Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF1</td>
<td>low</td>
<td>001010</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>001100</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>001110</td>
<td>14</td>
</tr>
<tr>
<td>AF2</td>
<td>low</td>
<td>010010</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>010100</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>010110</td>
<td>22</td>
</tr>
<tr>
<td>AF3</td>
<td>low</td>
<td>011010</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>011100</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>011110</td>
<td>30</td>
</tr>
<tr>
<td>AF4</td>
<td>low</td>
<td>100010</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>100100</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>100110</td>
<td>38</td>
</tr>
</tbody>
</table>

For example, you can define three subclasses with AF1. The third subclass would have a higher drop precedence that the first two. If the AF13 traffic exceeded its limits, the router would drop packets that matched this traffic class, rather than packets from AF12 or AF11. (Note that this means the AF1 class with DSCP 10 receives better service than AF13 with DSCP 14.)

You cannot configure drop precedence or buffer space on the ProCurve Secure Router (although you can mark packets for networks that do consider such specifications). You can configure bandwidth indirectly by assigning certain AF classes a specific proportion of an interface’s bandwidth with CBWFQ.

- Expedited Forwarding PHB—The Expedited Forwarding PHB ensures that the packet receives guaranteed bandwidth and the best level of service. This PHB ensures that the traffic has low latency, low jitter, and low loss.

A standard DSCP for the Expedited Forwarding PHB is 46.

On the ProCurve Secure Router, you can configure a low-latency queue for packets marked with the DSCP for the Expedited Forwarding PHB. You should reserve the Expedited Forwarding PHB for mission-critical applications. Using this PHB for a majority or all of the traffic defeats the purpose of QoS.
Setting Up Quality of Service
Overview

Only 13 DSCP values have actually been standardized. Individual network administrators define in more detail which set of DSCP values match to a specific PHB. This allows them to use DiffServ with the QoS policies already implemented in a network.

Remember, a PHB simply defines the standard of service that a node should provide for traffic marked for that PHB. DiffServ does not dictate how the node will actually grant such service. Such specifications are left to individual programmers and network administrators.

On the ProCurve Secure Router, you can mark packets with a DSCP, but you cannot define PHB. You can configure the ProCurve Secure Router to:

■ mark a packet with a DSCP for use in the network to which it will be forwarded
■ read a packet’s DSCP and map it to an IP precedence value to use for WFQ
■ read a packet’s DSCP value and assign it to a class for CBWFQ
■ read a packet’s DSCP value and assign it to a low-latency queue

First In, First Out

The most basic type of service for packets is First In, First Out (FIFO). A router gives all packets the same best-effort service, forwarding the first packet that arrives on an interface first. (See Figure 8-1.)

When a router uses switched processing to route packets, it forces a packet to wait in the queue until it has completed other CPU processes.

Routers can use fast caching to speed processing for packets that travel often-used routes. When a packet arrives on a fast-cache interface, the router interrupts its other processes to look up a route for the packet in the fast-cache table. This table contains the forwarding interfaces for the destinations of the most recently served packets. If the router does not find a match, it queues the packet as usual. If it does find a match, however, it forwards the packet immediately. Fast caching reduces delay and improves QoS. (Fast caching is enabled by default on the ProCurve Secure Router. For more information about configuring fast caching, see “Fast Caching” on page 11-12 in the Basic Management and Configuration Guide.)
Setting Up Quality of Service
Overview

Figure 8-1. First In, First Out

FIFO treats all packets in the same way. If you want the router to take packets’ ToS settings, or other criteria, into account when deciding how to treat them, you must implement a different queuing method.

WFQ

WFQ is one method for granting differentiated service to packets with various ToS values. When an interface uses WFQ, it classifies traffic flow into several conversations, or subqueues, according to source and destination IP addresses and protocol ports. The router then assigns each subqueue a weight according to its IP precedence value and a bandwidth relative to its weight.

CBWFQ

CBWFQ is an extension of WFQ that allows network administrators to define classes of conversation subqueues according to their own criteria. They can also allocate bandwidth to these classes manually. Instead of the router automatically assigning bandwidth to each subqueue based on relative IP precedence, administrators assign each class of subqueue an absolute or a relative amount of bandwidth.

LLQ

LLQ guarantees a set amount or a set percentage of bandwidth to certain types of traffic. LLQ also ensures that a router serves traffic in the low-latency queue first. (See Figure 8-2.) It is a better solution than WFQ for real-time traffic, such as VoIP, that cannot tolerate jitter or delays.
Setting Up Quality of Service

Overview

Figure 8-2. Low Latency Queuing

FRF.12

FRF.12 fragments large data frames so that a Frame Relay interface can forward each frame with less delay. This allows low latency frames, such as VoIP, more opportunities to be forwarded and minimizes delay.

For a more detailed discussion of each of these QoS protocols, see the sections in this chapter on configuring the protocols.

QoS Maps

A QoS map defines QoS policy for an interface on the ProCurve Secure Router. You use it for three functions:

- define a class for CBWFQ
- create and define the criteria for a low-latency queue
- mark packets with an IP precedence or DiffServ value

A QoS map is a list of sequenced entries. Each entry is defined by a name and a sequence number. The name of the map is the name of the map as a whole. The sequence number has two functions:

- It differentiates entries in the same QoS map—Maps with multiple entries allow you to implement a comprehensive QoS policy on an interface. For example, a single QoS map can establish several low-latency queues. It can also define multiple classes for the traffic sharing the remaining bandwidth using CBWFQ. Each entry defines one low-latency queue or one class.
It designates the order in which the ProCurve Secure Router matches traffic to these entries—the ProCurve Secure Router searches QoS entries with the lowest number first. Sequence numbers are only significant within the named map; QoS maps with different names can have entries with the same sequence number.

Each entry contains match commands and one or more actions. The match command determines the criteria for the class, low-latency queue, or marked packets.

This criteria can be based on:
- IP precedence or DiffServ value
- source and/or destination IP address and port (using an extended ACL)
- destination UTP protocol port
- bridged traffic

If you specify more than one match command for the QoS map, then traffic must match at least one of the criteria.

The action determines whether the router places matching traffic in a CBWFQ class or low-latency queue. The action can also mark the traffic with a ToS value.

After you create a QoS map, you must assign it to an interface to enable the class, low-latency queue, or packet marking.
Configuring WFQ

Overview

WFQ is one method for granting differentiated service to various types of traffic. It classifies traffic according to the source and destination IP addresses and protocol port, and allocates traffic bandwidth relative to IP precedence value. WFQ is best suited for granting high-priority traffic greater bandwidth. Because WFQ still queues all traffic, it is not best for VoIP and other real-time traffic that cannot tolerate delays.

Conversations

The outbound traffic on a point-to-point connection is the traffic flow. It consists of a queue of packets waiting for service. A router implementing WFQ classifies the traffic flow into several conversations. The router defines each conversation by creating a hash of the source or destination IP address, port number, and protocol type in packets’ IP headers; all packets with the same hash value are in the same conversation.

Using WFQ, the router then creates a number of individual subqueues, one for each flow or conversation (see Figure 8-3). The ProCurve Secure Router supports up to 256 conversation subqueues.

Figure 8-3. Weighted Fair Queuing
Weight

The router also assigns each conversation a weight based on the IP precedence value of its packets (see Figure 8-3). The rate at which that conversation gets serviced is proportional to the conversation's assigned weight, preventing high-weighted interactive traffic such as Telnet from being starved out by high-volume, lower-weighted traffic.

To determine how much bandwidth is allocated to a conversation, the ProCurve Secure Router compares its IP precedence value to the sum of all IP precedence values for conversations on the interface (adding one to these values so that routine traffic is not entirely starved out). WFQ uses the following formula:

\[
\text{Weight} = \frac{(\text{IP precedence value} + 1)}{ \text{sum of all (IP precedence values + 1)}}
\]

On the ProCurve Secure Router, WFQ is enabled by default on all WAN interfaces with E1 bandwidth or less. You must set the threshold, or number of packets, allowed in a queue. (By default, the threshold is 64 packets.)

Shortcomings

A closer examination of the formula WFQ uses to allocate bandwidth reveals how WFQ becomes less useful as an interface supports more conversations.

First, examine a situation in which WFQ functions well. Most traffic is routine and the interface supports few subqueues. Network control traffic is given IP precedence 7 and traffic to a server is given IP precedence 4. The PPP 1 interface supports 5 queues, three with a precedence of 0, one with 4, and one with 7. When needed, traffic to the server is guaranteed over one-fourth of the bandwidth and network control traffic is guaranteed half. Traffic with a higher precedence receives relatively more bandwidth. (See Table 8-3.)

Table 8-3. WFQ Example 1

<table>
<thead>
<tr>
<th>Subqueue</th>
<th>Precedence</th>
<th>Precedence + 1</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>.0625</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>.0625</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>.0625</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>.312</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>8</td>
<td>.5</td>
</tr>
<tr>
<td>Total</td>
<td>—</td>
<td>16</td>
<td>1</td>
</tr>
</tbody>
</table>
Now, consider an interface that handles more conversations at once—for example, 100 routine subqueues, 5 subqueues with a precedence of 3, and 2 queues for VoIP traffic with a precedence of 5. Even though VoIP traffic receives relatively more bandwidth than any individual routine subqueue, routine traffic altogether consumes 75 percent of the bandwidth. Neither VoIP queue is guaranteed even 5 percent of the total bandwidth. (See Table 8-4.) In addition, even though some subqueues receive relatively more bandwidth, all traffic must wait in queues. This level of service is inadequate for real-time traffic such as VoIP, which requires low latency and jitter.

Table 8-4. WFQ Example 2

<table>
<thead>
<tr>
<th>Subqueue</th>
<th>Precedence</th>
<th>Precedence + 1</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>.0076</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>.0076</td>
</tr>
<tr>
<td>...100</td>
<td>0</td>
<td>1</td>
<td>.0076</td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>4</td>
<td>.030</td>
</tr>
<tr>
<td>...105</td>
<td>3</td>
<td>4</td>
<td>.030</td>
</tr>
<tr>
<td>106</td>
<td>5</td>
<td>6</td>
<td>.045</td>
</tr>
<tr>
<td>107</td>
<td>5</td>
<td>6</td>
<td>.045</td>
</tr>
<tr>
<td>Total</td>
<td>—</td>
<td>132</td>
<td>1</td>
</tr>
</tbody>
</table>

Packet Marking

WFQ allocates bandwidth to conversation subqueues according to the IP precedence value in the IP headers of packets in the subqueue. The higher the value, the greater the bandwidth the queue is given. In order for WFQ to function, therefore, packets must somehow be marked with this value.

Packets can be marked:
- by an application or other device outside the router (typically, packet marking is most effective when it is implemented near the edge)
- by the ProCurve Secure Router

The router can also recognize DiffServ values, but it does not grant differentiated service for each DiffServ value. Instead, it maps several DiffServ values to a single IP precedence value and then treats the traffic as if it were marked with that value. (See Table 8-5.)
Table 8-5. Mapping DiffServ to IP Precedence

<table>
<thead>
<tr>
<th>DiffServ</th>
<th>IP Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-7</td>
<td>0</td>
</tr>
<tr>
<td>8-15</td>
<td>1</td>
</tr>
<tr>
<td>16-23</td>
<td>2</td>
</tr>
<tr>
<td>24-31</td>
<td>3</td>
</tr>
<tr>
<td>32-39</td>
<td>4</td>
</tr>
<tr>
<td>40-47</td>
<td>5</td>
</tr>
<tr>
<td>48-55</td>
<td>6</td>
</tr>
<tr>
<td>56-63</td>
<td>7</td>
</tr>
</tbody>
</table>

If applications and devices outside the router will handle all packet marking, you only need to enable WFQ and set a threshold level for subqueues.

If you want the router itself to mark packets with an IP precedence or DiffServ value, you must configure a QoS map to do so. You would then apply this map to a WAN interface. The router will mark matching outgoing packets with the value you set. To learn how to configure the ProCurve Secure Router to mark packets, see “Marking Packets with a ToS value” on page 8-45.

Enabling WFQ

WFQ is automatically enabled on all interfaces with E1 bandwidth or less. You enable WFQ on the logical, rather than the physical, interface. For Frame Relay, the interface, rather than the subinterface, handles queuing; however, you enable WFQ on Asynchronous Transfer Mode (ATM) subinterfaces.

To enable or disable WFQ, move to the configuration mode context for that interface and enter:

Syntax: `[no] fair-queue [threshold values]`

If you disable WFQ, the ProCurve Secure Router will use FIFO queuing for that interface.
Specifying the threshold when you enable WFQ is optional. The threshold determines the maximum number of packets the interface can hold in each conversation subqueue. When the queue reaches this limit, the ProCurve Secure Router discards any subsequent packets it receives. You can specify a threshold from 16 to 512 packets. For example:

ProCurve(config-fr 1)# fair-queue 256

The default threshold is 64.

Setting the Queue Size

You can also specify how many packets an interface can hold in all conversation subqueues together. Enter:

**Syntax:** hold-queue <packets> out

You can set the limit between 16 and 1000. The default number of packets that WFQ interfaces can hold is 400.

The ProCurve Secure Router also uses this setting with interfaces that implement FIFO queuing. The hold queue size is the maximum number of packets in the interface’s single queue and, so, the limit for the interface. The default number of packets that FIFO can hold is 200.
Configuring CBWFQ

Overview

CBWFQ is an extension of WFQ that allows you to tailor a QoS policy to your organization’s needs. With CBWFQ, you control:

- how traffic is divided into conversation subqueues
- how much bandwidth is allocated to each subqueue

You exercise this control by defining classes. For each class, you specify the traffic matching criterion and set a minimum guaranteed bandwidth. Each interface implementing CBWFQ supports up to four classes.

WFQ automatically classifies traffic into conversations according to source and destination IP address, port number, and protocol type. With CBWFQ, you manually configure how traffic is classified. You define a class according to IP header fields, and the interface places all traffic that fits that definition into the same subqueue.

WFQ only looks at IP precedence to determine the bandwidth to allocate each subqueue. With CBWFQ, you can specify the bandwidth for a class’s subqueue as an absolute value or as a percentage. The bandwidth is the minimum guaranteed to the class; it may burst above this value. You can reserve up to 75 percent of the interface’s bandwidth for all CBWFQ classes together.

Traffic that does not fall within a defined class is divided into subqueues using typical WFQ, and is allocated its share of whatever remains of the connection’s bandwidth. Because classes may burst above their guaranteed bandwidth, other traffic may be starved out of the connection.

Configuring Classes for CBWFQ

When you configure CBWFQ on the ProCurve Secure Router, you must configure:

- the criterion for a class
- the absolute or relative bandwidth allocated to each class

On the ProCurve Secure Router, you define classes for CBWFQ in a QoS map entry.
To configure CBWFQ, you must complete these steps:

1. Create a QoS map entry.
2. Define a class. You can define classes according to:
   - ToS value
   - IP header fields—source and destination IP address, port, and protocol
   - destination UDP protocol port
   - bridged protocol
3. Allocate bandwidth to the class.
4. Create QoS map entries with the same name and different sequence numbers to configure multiple classes. Repeat steps 2 and 3.
5. Apply the QoS map to a WAN interface.

Creating a QoS Map Entry

To create a QoS map, enter the following command from the global configuration mode context:

**Syntax:** qos map <mapname> <sequence number>

The mapname is alphanumeric and case-sensitive. Valid sequence numbers range from 0 to 65,535.

If you are using LLQ and CBWFQ on the same interface, the map entries for the low-latency queues and the CBWFQ classes should use the same name. For example, you could use map entries 0 and 1 for low-latency queues and configure map entry 2 to define a CBWFQ class:

```
ProCurve(config)# qos map QoSMap 2
```

**Note**

The router matches packets to lower-numbered entries first. If you configure classes that might match the same traffic, you should assign the entry for the more specific definition a lower sequence number.

Using different kinds of criteria to define classes on the same interface can also complicate matters. For example, a packet might fall into a class of traffic from subnet 192.168.3.0 and into a class of traffic with IP precedence 4. You should either use the same kind of criteria for all the classes or take care to assign a lower sequence number to the entry you want to take precedence in defining traffic.
Defining a Class

You define a class by matching the QoS map entry to packets that meet certain criteria.

**Table 8-6. QoS Map Criteria**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Match Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToS value—IP precedence</td>
<td><code>match precedence &lt;0-7&gt;</code></td>
</tr>
<tr>
<td>ToS value—DiffServ</td>
<td><code>match dscp &lt;0-63&gt;</code></td>
</tr>
<tr>
<td>IP header—source or destination IP address and protocol port</td>
<td><code>match list &lt;ACL listname&gt;</code></td>
</tr>
<tr>
<td>destination UDP protocol port</td>
<td><code>match ip rtp &lt;first port number&gt; [last port number] [all]</code></td>
</tr>
<tr>
<td>bridged traffic</td>
<td><code>match protocol bridge [netbeui]</code></td>
</tr>
</tbody>
</table>

Each QoS map entry can use only one set of criteria to match traffic. To match another group of traffic, you must configure another entry.

Enter one of the **match** commands shown in Table 8-6 to select traffic. Different options for the **match** command will be discussed separately in the following sections.

**Classifying Traffic According to a ToS Value.** WFQ allocates relatively more bandwidth to traffic with higher IP precedence (or DiffServ values). Simple WFQ must use this formula:

\[
\text{(IP precedence value +1)/sum of all (IP precedence values +1)}
\]

With CBWFQ, you can control how much bandwidth the router assigns to a class defined by a ToS value.

You would create an entry for each ToS value your system uses and then enter this command:

**Syntax:** `match [dscp <value> | precedence <value>]`

Valid DiffServ (DSCP) values are from 0 to 63; valid IP precedence values are from 0 to 7. For example:

ProCurve(config-qos-map)# match precedence 5
**Note**

This ToS value is set by an application or device before the packet arrives on the interface. Although the router can mark traffic with ToS values, these values are generally used in the network to which the router forwards the packet.

DiffServ defines four classes of AF PHB, each class receiving successively better service. (The first subclass in an AF class receives better treatment because it has a lower drop precedence.) You could configure four classes to match four AF PHB and allocate each successive class relatively more bandwidth. (See Table 8-7 for the DSCP for AF PHB.)

**Table 8-7. Example of Assured Forwarding PHB**

<table>
<thead>
<tr>
<th>AF Class</th>
<th>Drop Precedence</th>
<th>DSCP</th>
<th>DiffServ Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF1</td>
<td>low</td>
<td>001010</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>001100</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>001110</td>
<td>14</td>
</tr>
<tr>
<td>AF2</td>
<td>low</td>
<td>010010</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>010100</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>010110</td>
<td>22</td>
</tr>
<tr>
<td>AF3</td>
<td>low</td>
<td>011010</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>011100</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>011110</td>
<td>30</td>
</tr>
<tr>
<td>AF4</td>
<td>low</td>
<td>100010</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>100100</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>100110</td>
<td>38</td>
</tr>
</tbody>
</table>
You would enter these commands to match classes to the four simple AF PHBs:

ProCurve(config)# qos map Class 11
ProCurve(config-qos-map)# match dscp 10
ProCurve(config)# qos map Class 12
ProCurve(config-qos-map)# match dscp 18
ProCurve(config)# qos map Class 13
ProCurve(config-qos-map)# match dscp 26
ProCurve(config)# qos map Class 14
ProCurve(config-qos-map)# match dscp 34

This example only shows how to define what traffic is placed in the class. When actually configuring CBWFQ, you would set the bandwidth for the class at the same time. See “Allocating Bandwidth to a Class” on page 8-26.

Classifying Traffic According to IP Header Fields. You can classify packets according to the fields in their IP headers—that is, according to their source and destination IP addresses, port number, and protocol. This method of classifying traffic mimics simple WFQ. However, you can place traffic to and from an entire range of addresses in the same class. One of the shortcomings of WFQ is that the more subqueues an interface supports, the less that interface can grant higher priority subqueues significantly greater bandwidth. Dividing traffic into a small number of classes alleviates this problem.

You classify traffic in this way by matching the QoS map entry to an extended access control list (ACL). The ACL actually selects the traffic. An extended ACL can define traffic according to its source and destination IP address, as well as a variety of fields in the IP, TCP, or UDP headers.

To classify traffic:

1. Configure an ACL
   a. Create an extended ACL—QoS maps can only use extended ACLs.
   b. Add any necessary deny entries to the ACL.
   c. Add permit entries for the source and/or destination addresses of traffic in the class.
2. Match the QoS map entry to the ACL.

**Configuring an ACL.** Create an ACL by entering this command from the global configuration mode context:

**Syntax:** ip access-list extended <listname>
For example:

ProCurve(config)# ip access-list extended ClassSelector

ACLs exclude all traffic that you do not explicitly permit, so you may not need to enter any deny statements. However, you will often permit an entire range of addresses. If you want to deny a host or hosts within this range, you must explicitly deny those hosts. You must enter the deny statements first because the router processes ACL entries in order and stops processing them as soon as it finds a match.

You use this command to select traffic in the ACL:

**Syntax:**

```
[deny | permit] ip [any | host <source A.B.C.D> | <source A.B.C.D> <wildcard bits>] [any | host <destination A.B.C.D> | <destination A.B.C.D> <wildcard bits>]
```

Very often, you will want an ACL to select an entire range of addresses or subnets. ACLs on the ProCurve Secure Router use wildcard bits (which operate on reverse logic from subnet masks) to select a range of addresses.

You can also select certain types of traffic (for example, HTTP or Telnet) by specifying a protocol such as TCP or UDP and then indicating the source or destination port after the address:

**Syntax:**

```
[deny | permit] <protocol> [any | host <A.B.C.D> | <A.B.C.D> <wildcard bits>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>] [any | host <A.B.C.D> | <A.B.C.D> <wildcard bits>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>]
```

For example:

ProCurve(config-ext-nacl)# permit tcp host 192.168.4.1 eq telnet any

The **eq** keyword selects a single port and the **range** keyword allows you to enter a range of ports. You can specify the port by number, or for well-known protocols, by keyword. Use the ? help command to get a complete list of keywords. For example:

ProCurve(config-ext-nacl)# permit tcp any ?
Figure 8-4. Classifying Network Traffic

In Figure 8-4, Network 1 at site A transmits mission-critical data to network 4 at site B. Host 26 on network 4 is a local DHCP server; it does not need to receive this critical data. To select the traffic for the class, you would enter:

ProCurve(config)# ip access-list extended ClassSelector
ProCurve(config-ext-nacl)# deny ip any host 192.168.4.26
ProCurve(config-ext-nacl)# permit ip 192.168.1.0 0.0.0.255 192.168.4.0 0.0.0.255

You could configure another ACL that will be used to define a class for Web traffic:

ProCurve(config)# ip access-list extended WebTraffic
ProCurve(config-ext-nacl)# permit tcp any any eq www

For more information about configuring ACLs, see Chapter 5: Applying Access Control to Router Interfaces.

Matching a QoS Map Entry to an ACL. Move to the configuration mode context for the QoS map entry you have created. Then enter this command:

**Syntax:** match list <ACL listname>

For example:

ProCurve(config)# qos map Class 15
ProCurve(config-qos-map)# match list ClassSelector

Classifying Traffic According to Destination UDP Port. Different applications require different levels of service. You can group similar applications together into a class according to their destination UDP port and then grant that class a certain portion of the interface’s bandwidth.
You use this command:

**Syntax:** match ip rtp <first port number> <last port number> [all]

The `match ip rtp` command configures the router to match all UDP packets destined to even port numbers in the specified range. (Typically, servers listen for user traffic on even ports.) If you want to match traffic to both even and odd ports, you must add the `all` keyword.

You can use this command to define a CBWFQ class; however, this command selects real-time traffic, for which you should generally configure a low-latency queue. See “Placing Traffic in a Low-Latency Queue” on page 8-37.

**Classifying Bridged Traffic.** You can configure one or more interfaces on a the ProCurve Secure Router to act as a bridge. In effect, the router extends a LAN throughout two or more remote sites. Traffic between hosts at each local site can obviously travel faster than that between hosts at different sites. Local hosts are not only physically closer, but they can also take advantage of higher-speed Ethernet connections.

Often, an interface will bridge all traffic. However, a Frame Relay interface may carry one subinterface that routes traffic and one that bridges traffic. You can define bridged traffic as a class and set the maximum bandwidth that class is guaranteed.

To place all bridged traffic in a class, enter:

```plaintext
ProCurve(config-qos-map)# match protocol bridge
```

Instead of placing all bridged traffic in a class, you can place only NetBIOS Extended User Interface (NetBEUI) traffic. NetBEUI allows hosts to communicate within the LAN. You can define such traffic as a class of its own. For example:

```plaintext
ProCurve(config)# qos map Class 12
ProCurve(config-qos-map)# match protocol bridge netbeui
```

**Allocating Bandwidth to a Class**

You can allocate bandwidth for classes with absolute or relative values. For example, you define three classes on an interface with a 2 Mbps connection. You could allocate 500 Kbps to one class, 250 to another, and 200 to the last. Or you could allocate 25 percent of the bandwidth to one class, 12 percent to another, and 10 percent to the last. The ProCurve Secure Router allows you to reserve up to 75 percent of an interface’s bandwidth for all classes together.
If you have configured one or more low-latency queues on the interface, you might want to divide the remaining bandwidth rather than the total bandwidth. This option eases the configuration process; you do not have to figure out how much bandwidth must be reserved for the low-latency queues.

You assign a class its bandwidth from the configuration mode for the QoS map entry that defines it. You must specify bandwidth in the same way (absolute, percentage, or remaining percentage) for each class in the QoS map.

To specify the maximum bandwidth guaranteed to the queue, move to the QoS map entry for the class and enter:

**Syntax:** bandwidth [<Kbps> | percent <percentage> | remaining percent <percentage>]

For example, to set the bandwidth as an absolute value, enter:

```
ProCurve(config-qos-map)# bandwidth 500
```

To specify bandwidth as a percentage of total bandwidth, use the **percent** keyword:

```
ProCurve(config-qos-map)# bandwidth percent 25
```

The **percent** keyword calculates bandwidth from the total available bandwidth on an interface. The total available bandwidth is the access rate for Point-to-Point Protocol (PPP) and High-level Data Link Control (HDLC) interfaces and for ATM subinterfaces. The total available bandwidth is the rate-limited bandwidth for Ethernet and Frame Relay interfaces. However, you can only allocate up to 75 percent of the available bandwidth to queues.

To specify bandwidth as a percentage of the bandwidth not allocated to low-latency queues, use the **remaining percent** keyword. The **remaining percent** keyword calculates bandwidth from the amount remaining after the bandwidth guaranteed to low latency queues has been subtracted from the available bandwidth. Unlike commands using the **percent** keyword, this command does not subtract bandwidth from the bandwidth available for the low-latency queues.

Percentages must be whole values from 1 to 100, inclusive.
The bandwidth available for queues on a ProCurve Secure Router is 75 percent of an interface’s access rate or rate-limited rate. The Secure Router OS will deactivate a QoS map when you assign it to an interface that does not have enough bandwidth available to grant the guaranteed rate.

Other traffic can use the remaining 25 percent of the bandwidth, although this traffic may also be starved out by traffic in a class bursting past its guaranteed level.

The router automatically provides bandwidth for control traffic such as routing updates; control traffic takes priority over all classes.

Traffic that does not fit into one of the classes you have defined is served with typical WFQ. It is divided into conversation subqueues according to source and destination IP addresses and port, and is allocated a portion of the remaining bandwidth based on its IP precedence value.

**Specifying Bandwidth by Remaining Percent Versus Percent.** For example, you limit an Ethernet interface’s rate to 10 Mbps. You guarantee at least 4 Mbps to low-latency queues. You then assign one class 25 percent of the remaining bandwidth and another class 15 percent of the remaining bandwidth. Subtracting 4 Mbps from 10 Mbps leaves 6 Mbps. The first class receives 1.5 Mbps and the second, 900 Kbps. The bandwidth required for the map is 6.4 Mbps. The bandwidth available for queues is 75 percent of the rate limited bandwidth, or 7.5 Mbps. The map can become active.

Now, consider how much bandwidth the classes would receive if you configured the QoS map using the `percent` keyword rather than the `remaining percent` keyword. The first class would receive 25 percent of 10 Mbps, or 2.5 Mbps. The second class would receive 15 percent of 10 Mbps, or 1.5 Mbps. With the low-latency queues, the bandwidth required for the map is 8 Mbps. Because queues can only consume up to 7.5 Mbps on the Ethernet interface, the router would force the map to become inactive. You would have to reconfigure the QoS map.

**Assigning the QoS Map to an Interface**

You must create a separate QoS map entry for each class you want to define, giving each entry the same name but a different sequence number. You can define up to four classes. You can also implement low-latency queues on the same interface. Simply create an entry for these queues in the same QoS map. (See “Configuring LLQ” on page 8-32.)
Next, apply the QoS map to the logical interface for the connection on which you want to enable CBWFQ. Move to the interface configuration mode context and enter:

**Syntax:** qos-policy out <mapname>

For example:

```
ProCurve(config)# interface frame-relay 1
ProCurve(config-fr 1)# qos-policy out Class
```

### Special Considerations for CBWFQ with Multilinks

Multilink protocols such as Multilink PPP (MLPPP) and Multilink Frame Relay (MLFR) increase the total bandwidth of a connection. Although the bundle of carrier lines acts as a single logical connection, each carrier line is physically separate, and you should remember this as you allocate the interface's bandwidth. Carrier lines may go down and alter the bandwidth actually available.

For example, an MLPPP connection with two T1 lines provides 3.0 Mbps of bandwidth. You can allocate up to 75 percent of this bandwidth, or 2.25 Mbps, to the interface’s classes. You could allocate 300 Kbps (10 percent) to Class 1, 600 Kbps (20 percent) to Class 2, 600 Kbps to Class 3, and 750 Kbps (25 percent) to Class 4. However, if one of the lines fails, the connection will only have 1.5 Mbps of bandwidth to provide the 2.25 guaranteed. If Class 3 and Class 4 are already consuming their full minimum bandwidth (1.35 Mbps), traffic from Class 2 will not be able to receive its guaranteed level of service.

You should consider allocating bandwidth to the multilink connection as if it had one carrier line less than the total. This is particularly true when the multilink is designed more to provide redundancy than to increase a connection’s bandwidth.

**Note**

Even when you assign bandwidth to classes as a percentage, the router assigns it as an absolute value of the bandwidth normally available on the interface. This means that when one or more lines in a multilink bundle goes down, the router does not automatically readjust the bandwidth allocated to various classes.
CBWFQ Example Configuration

In Figure 8-5, Site A includes two networks that connect to the Internet. It also connects to remote Site B through a virtual private network (VPN). Your organization does not want Internet traffic to starve out traffic to the remote site. You can configure CBWFQ to reserve at least 25 percent of the bandwidth for Network 1 to access the remote site and 20 percent for Network 2. Site A also includes the company Web server. Company policy dictates that 15 percent of the bandwidth must be reserved for traffic from the server.

You would implement this policy as follows:

1. Configure the ACLs to select VPN traffic:
   a. Match traffic from Network 1 and 2 to Site B:
      
      ProCurve(config)# ip access-list extended Network1
      ProCurve(config-ext-nacl)# permit ip 192.168.1.0 0.0.0.255 192.168.16.0 0.0.15.255
      ProCurve(config)# ip access-list extended Network2
      ProCurve(config-ext-nacl)# permit ip 192.168.2.0 0.0.0.255 192.168.16.0 0.0.15.255
   b. Match traffic from the Web server:
      
      ProCurve(config)# ip access-list extended WebTrafficOut
      ProCurve(config-ext-nacl)# permit tcp host 192.168.1.26 eq www any
2. Match the ACLs to the classes and set the bandwidth for each:
   a. First, define the class for traffic from the Web server. Set the entry number lower than that for the class for Network 1 traffic so that the router does not inadvertently match traffic from the server to the wrong class:

```
ProCurve(config)# qos map Class 10
ProCurve(config-qos-map)# match list WebTraffic Out
ProCurve(config-qos-map)# bandwidth percent 15
```

b. Define the classes for VPN traffic from the local networks to the remote sites:

```
ProCurve(config)# qos map Class 11
ProCurve(config-qos-map)# match list Network1
ProCurve(config-qos-map)# bandwidth percent 25
ProCurve(config)# qos map Class 12
ProCurve(config-qos-map)# match list Network2
ProCurve(config-qos-map)# bandwidth percent 20
```

**Note**

The QoS map entries for the classes started at 10 to leave room for low-latency queues. For example, employees at Sites A and B might use VoIP phones. Because voice traffic needs particularly low delay service, you could configure a low-latency queue for such traffic in map entry Class 1. Assigning this entry a lower sequence number prevents voice packets that match other entries from being placed in the wrong queue.

3. Assign the QoS map to the PPP interface that connects to the Internet:

```
ProCurve(config)# interface ppp 1
ProCurve(config-ppp 1)# qos-policy out Class
```
Configuring LLQ

Overview

LLQ is a method for guaranteeing a set amount of bandwidth to certain traffic and reducing this traffic’s latency. You should use LLQ for voice and other real-time applications that involve traffic that cannot tolerate excessive or variable delay (jitter).

When a packet that matches the criteria for a low-latency queue arrives on an interface, the router immediately places it in this queue. The low-latency queue is always served first and is always given bandwidth up to the guaranteed level. Low-latency traffic can also burst past its guaranteed level when bandwidth is available. You can specify an upper limit for bursting low-latency traffic to prevent it from entirely starving out other traffic.

Packets that do not match the criteria for the low-latency queue are served by the queuing method enabled on the interface (FIFO or CBWFQ) with the remaining bandwidth.

When you configure a low-latency queue on the ProCurve Secure Router, you must configure:

- the criteria for packets placed in the queue
- the bandwidth guaranteed the queue

Determining Bandwidth for the Queue

LLQ allows you to manually determine the bandwidth a queue receives. Before you configure a low-latency queue, you should plan for every queue that will be implemented on the interface. You can then determine how much bandwidth to assign each queue. If the interface also implements CBWFQ, you should remember to take bandwidth allocated to classes into account.

This section of the guide gives you some general guidelines for determining how much bandwidth you must allocate to:

- VoIP traffic
- video streams

You should, of course, refer to the documentation for your VoIP application and follow any instructions given in that documentation first.
Determining Bandwidth for VoIP

One of the most common applications for a low-latency queue is VoIP traffic. You calculate the bandwidth necessary for VoIP traffic by:

1. calculating the bandwidth necessary for one call
2. making adjustments to this calculation according to the capabilities of your VoIP devices
3. multiplying the per-call bandwidth by the number of calls the router needs to support at once

**Calculating Per-Call Bandwidth.** VoIP standards specify the minimum bit rate necessary for acceptable voice quality. Various standards specify various rates. (See Table 8-8.)

However, this rate does not correspond exactly to the bandwidth necessary to maintain voice quality. The rate is that for voice bits, but, because frame and packet headers add overhead, a connection must provide greater bandwidth. VoIP packets are quite small; headers might add as many, or even more, bytes than the payload of actual voice data.

To calculate the per-call bandwidth for your VoIP application, you must transform its specified bit rate into a rate of packets per second. This rate depends on the size of the voice payload, which in turn depends on the codec your system’s application uses.

**Table 8-8. VoIP Standards**

<table>
<thead>
<tr>
<th>Standard</th>
<th>Bit Rate</th>
<th>Codec (Sample Time)</th>
<th>Sample Size</th>
<th>Packets Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.711</td>
<td>• 56 Kbps</td>
<td>20 ms</td>
<td>• 140 bytes</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>• 64 Kbps</td>
<td></td>
<td>• 160 bytes</td>
<td></td>
</tr>
<tr>
<td>G.722</td>
<td>• 48 Kbps</td>
<td>20 ms</td>
<td>• 120 bytes</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>• 56 Kbps</td>
<td></td>
<td>• 140 bytes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 64 Kbps</td>
<td></td>
<td>• 160 bytes</td>
<td></td>
</tr>
<tr>
<td>G.723.1</td>
<td>• 5.3 Kbps</td>
<td>30 ms</td>
<td>• 20 bytes</td>
<td>33.3</td>
</tr>
<tr>
<td></td>
<td>• 6.3 Kbps</td>
<td></td>
<td>• 24 bytes</td>
<td></td>
</tr>
</tbody>
</table>
Setting Up Quality of Service
Configuring LLQ

The codec dictates how often the voice stream is sampled, and together with the bit rate, the size of each voice sample. For example, G.722 has a 20 ms codec and 64 Kbps rate. Therefore, each sample is 160 bytes. (64,000 bits/second x .02 seconds = 1280 bits. 1280 bits/8 = 160 bytes.) This sample size is relatively large. Others, however, are very small. For example, a G.729 sample size might be only 10 bytes.

Note

The voice payload for VoIP packets is divisible by the sample size. Some VoIP applications combine several samples into a single packet to reduce overhead. For example, G.728 packets often include four samples.

Calculate the packets transmitted per second by dividing the bit rate by the number of bits (not bytes) in the total voice payload. The router must forward this many frames across the WAN connection every second to maintain an acceptable QoS.

To determine how much bandwidth the router needs to forward the required number of frames per second, first calculate the total size of a frame. Add the number of bits in the frame and packet headers to the number of bits in the voice payload. Then multiply the total size of the frames in bits by the rate of packets per second. This is the minimum bandwidth required per call over the WAN connection.

Table 8-9 shows example calculations for common sample sizes with several VoIP standards.
Table 8-9. Example Bandwidth Calculations for VoIP

<table>
<thead>
<tr>
<th>Standard</th>
<th>Packets per Second</th>
<th>Voice Payload Size</th>
<th>Total Size with MLPPP or Frame Relay header</th>
<th>Per-Call Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.711</td>
<td>50</td>
<td>140 bytes</td>
<td>187 bytes</td>
<td>74.8 Kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>160 bytes</td>
<td>207 bytes</td>
<td>82.8 Kbps</td>
</tr>
<tr>
<td>G.722</td>
<td>50</td>
<td>120 bytes</td>
<td>167 bytes</td>
<td>66.8 Kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>140 bytes</td>
<td>187 bytes</td>
<td>74.8 Kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>160 bytes</td>
<td>207 bytes</td>
<td>82.8 Kbps</td>
</tr>
<tr>
<td>G.723.1</td>
<td>33.3</td>
<td>20 bytes</td>
<td>67 bytes</td>
<td>17.8 Kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24 bytes</td>
<td>71 bytes</td>
<td>18.9 Kbps</td>
</tr>
<tr>
<td>G.728</td>
<td>100</td>
<td>20 bytes</td>
<td>67 bytes</td>
<td>53.6 Kbps</td>
</tr>
<tr>
<td>G.729</td>
<td>50</td>
<td>20 bytes</td>
<td>67 bytes</td>
<td>26.8 Kbps</td>
</tr>
</tbody>
</table>

The necessary bandwidth depends on:

- The size of the packet headers—Packet headers are 47 bytes for MLPPP and MLFR connections as well as Frame Relay connections that use fragmentation:
  - 20 for the IP header
  - 8 for the UDP header
  - 12 for the RTP header
  - 6 for the MLPPP or MLFR header
  - 1 for the MLPPP or MLFR end-of-frame flag

If you use the MLPPP long sequence number format, the MLPPP header is 8 bytes and the total size for packet headers is 49 bytes. For PPP connections, the packet headers are 45 bytes.

Real-time Transport Protocol (RTP) compression (cRTP) reduces the IP header from 40 bytes to 2 bytes (or 4 bytes with checksums), which decreases necessary bandwidth.

- The size of the voice payload—Each packet must have a set number of bytes in its header no matter how large or how small its payload. As the size of the voice payload increases, the proportion of bytes consumed in overhead decreases and so does the necessary bandwidth. Combining several samples in the same packet increases the size of the voice payload.

However, the decrease in required bandwidth comes at the price of higher latency.
Setting Up Quality of Service
Configuring LLQ

Making Adjustments. Calls typically contain bursts of noise when a person speaks and periods of silence when the person listens. Some VoIP applications use Voice Activity Detection (VAD) to suppress transmission of VoIP frames when the line is silent. If your equipment uses VAD, you can cut the per-call bandwidth requirement by as much as half.

Calculating Total Bandwidth. Your organization should study typical VoIP usage patterns. It should then generate a policy for the number of calls the router should support at once, taking into account cost and other needs for bandwidth.

Multiply the minimum number of calls the WAN must support by the per-call bandwidth.

This is the bandwidth you would specify for the VoIP low-latency queue. (See “Setting the Bandwidth Guaranteed the Queue” on page 8-42.)

Determining Bandwidth for Video Streaming

Organizations are increasingly using videoconferencing to broadcast company presentations and to allow employees at remote sites to communicate. Like VoIP, video streaming requires low jitter. Video packets are larger than VoIP packets and so require much more bandwidth and have a larger serialization delay. However, streaming video tends to be less interactive than voice, so the traffic can tolerate higher delay. Many applications hold the initial video packets and send the streaming video with a continual lag so that, overall, the video runs smoothly without jitter or delay.

Because the packets are relatively large, overhead from frame and packet headers does not greatly increase required bandwidth.

Several H.323 and Session Initiation Protocol (SIP) standards have emerged for video. The amount of bandwidth required depends on the resolution and the number of frames per second and can be very large. Even though compression greatly reduces required bandwidth, the router may need to devote a large portion of a T1 or E1 line to a video stream while it is active.

The video application should indicate minimum bit rates for different quality pictures. Determine the number of video streams the router must be able to establish at once and multiply it by the bit rate you select. The video streamer may also send the video to a Web server, which all hosts can access.
Placing Traffic in a Low-Latency Queue

The ProCurve Secure Router guarantees traffic in a low-latency queue the amount of bandwidth you specify. Traffic can burst above this bandwidth, but if the line becomes congested, the router will drop bursting packets in favor of other traffic.

The QoS map entry both selects traffic for the queue and assigns the queue a bandwidth.

To configure LLQ, you must complete these steps:

1. Create a QoS map entry.
2. Match the entry to traffic that meets your criteria.
3. Set the queue’s maximum guaranteed bandwidth.
4. Apply the QoS map to a WAN interface.

You can place the following traffic in a low-latency queue:

- traffic marked with a certain ToS value
- traffic with a certain source and/or destination IP address and port
- traffic destined to a certain range of UDP protocol ports
- bridged traffic

Creating a QoS Map Entry

To create a QoS map, enter the following command from the global configuration mode context:

**Syntax:** qos map <mapname> <sequence number>

The sequence number indicates the priority for the QoS map entry. Because the ProCurve Secure Router searches entries with the lowest numbers first, the lower the map’s number, the higher its priority. For example, enter:

```
ProCurve(config)# qos map LowLatency 10
```

Selecting the Traffic to Be Placed in the Low-Latency Queue

You select the traffic that the QoS map entry will mark by entering one of the **match** commands shown in Table 8-10.
Setting Up Quality of Service
Configuring LLQ

Table 8-10. QoS Map Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Match Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToS value—IP precedence</td>
<td>match precedence &lt;0-7&gt;</td>
</tr>
<tr>
<td>ToS value—DiffServ</td>
<td>match dscp &lt;0-63&gt;</td>
</tr>
<tr>
<td>IP header—source or destination IP address and protocol port</td>
<td>match list &lt;ACL listname&gt;</td>
</tr>
<tr>
<td>destination UDP protocol port</td>
<td>match ip rtp &lt;first port number&gt;[&lt;last port number&gt;][all]</td>
</tr>
<tr>
<td>bridged traffic</td>
<td>match protocol bridge [netbeui]</td>
</tr>
</tbody>
</table>

Each QoS map entry can use only one set of criteria to match traffic. To match another group of traffic, you must configure another entry.

Placing Traffic with a Certain ToS Value in a Low-Latency Queue.

WFQ allocates more bandwidth to traffic with higher IP precedence or DiffServ values. However, the more conversation subqueues an interface accumulates, the less effect a high ToS value has. Rather than relying on packets’ ToS values to grant them the bandwidth they need, you can place packets with a certain ToS value in a low-latency queue. Use this command:

**Syntax:** match [dscp <value> | precedence <value>]

Use the `dscp` option to select DiffServ values. Valid DiffServ values are from 0 to 63; valid IP precedence values are from 0 to 7.

For example, for VoIP traffic you would enter the value set by your VoIP devices. Usually, this is the DiffServ value for Expedited Forwarding (46). Enter:

ProCurve(config-qos-map)# match dscp 46

You can only match the entry to one value. If you want to match more than one value, you must configure another entry for the QoS map.

**Note**

The ToS value for LLQ is always set by an application or device before the packet arrives on the interface. Although the router can mark traffic with ToS values, these values are used in the network to which the router forwards the packet.
Setting Up Quality of Service
Configuring LLQ

Placing Traffic Destined to a UDP Protocol Port in a Low-Latency Queue. VoIP and other real-time traffic requires special handling. Congestion affects this traffic far more negatively than it does bursty data traffic. One way of classifying VoIP traffic is noting the UDP ports on which your VoIP applications operate. You can then match a QoS map entry to these ports. You can, of course, similarly define low-latency queues for other applications.

You use the **match ip rtp** command to place RTP packets destined to a range of UDP destination ports in a low-latency queue:

**Syntax:** match ip rtp `<first port number>` `<last port number>` [all]

For example, RTP applications generally operate between ports 16,384 and 32,767. You would enter:

```
ProCurve(config-qos-map)# match ip rtp 16384 32764 all
```

**Note**
The router matches all RTP packets destined to *even* port numbers in the specified range. (Typically, servers listen for traffic on even ports.) If you want to match traffic to both even and odd ports, you must add the **all** keyword.

Placing Traffic to and/or from an IP Address in a Low-Latency Queue. You can assign packets to a low-latency queue according to the source and/or destination IP addresses in their IP headers.

You can guarantee all traffic *from* a network—for example, a subnet that transmits mission-critical data—low latency. You may also want to give packets destined to a specific address a set amount of bandwidth. For example, you can prioritize traffic to the router's Web browser interface, so that IT staff can manage the router despite network congestion.

You place such traffic in a queue by matching the QoS map entry to an ACL. The ACL actually selects the traffic. An extended ACL can define traffic according to its source and destination IP address as well as a variety of fields in the IP, TCP, or UDP header.

To place traffic with certain values in its IP header in a low-latency queue, you must:

1. Configure an ACL.
   a. Create an extended ACL.
   b. Add any necessary deny entries to the ACL.
   c. Add permit entries for the addresses to or from which you want to guarantee traffic bandwidth.
2. Match the QoS map entry to the ACL.
Configuring an ACL. Create an ACL by entering a command such as this from the global configuration mode context:

```
ProCurve(config)# ip access-list extended LowLatencyTraffic
```

ACLs exclude all traffic that you do not explicitly permit, so you may not need to enter any deny statements. However, you must explicitly deny traffic to or from a denied host within a permitted range—for example, host 99 on the permitted 192.168.3.0 /24 subnet. You must enter the deny statements first because the router processes ACL entries in order and stops processing them as soon as it finds a match.

You will often want an ACL to select an entire range of addresses or subnets. ACLs on the ProCurve Secure Router use wildcard bits (which operate on reverse logic from subnet masks) to select ranges of addresses.

You use this command to select traffic to be matched or not matched:

**Syntax:** `[permit | deny] ip [any | <source A.B.C.D> <wildcard bits> | host <source A.B.C.D>] [any | <destination A.B.C.D> <wildcard bits> | host <destination A.B.C.D>]`

Wildcard bits operate on opposite logic from subnet masks. A one means that the router ignores that bit when deciding whether a packet’s source or destination address matches the entry. For example, if you wanted to select every host in a Class B network, you would use the wildcard bits 0.0.255.255.

---

*Figure 8-6. Placing Network Traffic in a Low-Latency Queue*
Network 1 at Site A, shown in Figure 8-6, contains VoIP equipment that communicates with equipment at Network 4 at Site B. Host 26 on Network 1 is an email server; it does not send real-time data. To select the traffic to be placed in a low-latency queue, enter:

```bash
ProCurve(config)# ip access-list extended LowLatencyTraffic
ProCurve(config-ext-nacl)# deny ip host 172.16.1.26 any
ProCurve(config-ext-nacl)# permit ip 172.16.1.0 0.0.0.255 172.16.4.0 0.0.0.255
```

You can also select certain types of traffic (for example, HTTP or Telnet) by specifying a protocol such as TCP or UDP and then indicating the source or destination port after the address:

**Syntax:**

```text
[deny | permit] <protocol> [any | host <A.B.C.D> | <A.B.C.D> <wildcard bits>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>]
```

For example:

```bash
ProCurve(config-ext-nacl)# permit tcp host 172.16.1.30 eq telnet any
```

The `eq` keyword selects a single port and the `range` keyword allows you to enter a range of ports. For more information about configuring ACLs, see Chapter 5: Applying Access Control to Router Interfaces.

**Matching a QoS Map Entry to an ACL.** Move to the configuration mode context for the QoS map entry you have created. Then enter this command:

**Syntax:**

```text
match list <ACL listname>
```

For example:

```bash
ProCurve(config-qos-map)# match list LowLatencyTraffic
```

**Placing Bridged Traffic in a Low-Latency Queue.** You can configure one or more interfaces on the ProCurve Secure Router to act as a bridge. In effect, the router extends a LAN throughout two or more remote sites. Traffic between hosts at each local site can obviously travel faster than that between hosts at different sites. Local hosts are not only physically closer, but they can also take advantage of higher-speed Ethernet connections.
For Frame Relay connections, packets are queued on the Frame Relay interface. When one of the Frame Relay subinterfaces is part of a bridge group, you can place bridged traffic in a low-latency queue to speed processing and transmission. The queue also guarantees that the bridged traffic receives the portion of the bandwidth it needs. Setting the maximum bandwidth for packets in this queue also ensures that bridged traffic does not starve out non-bridged traffic.

To select bridged traffic for the queue, enter:

**Syntax:** match protocol bridge [netbeui]

NetBEUI allows hosts to communicate within the LAN. To allow only this traffic access to the low-latency queue, use the optional **netbeui** keyword.

**Setting the Bandwidth Guaranteed the Queue**

After you select which traffic the QoS map entry should place in the queue, you must select how much bandwidth it should guarantee this queue. See “Determining Bandwidth for the Queue” on page 8-32 for some general guidelines on arriving at this value.

Enter:

**Syntax:** priority {<bandwidth> [<burst>] | percent <value> | unlimited}

The command gives you three options for allocating bandwidth to the queue:

- specifying an absolute bandwidth in Kbps
- specifying a percentage of the total bandwidth
- allowing the queue unlimited bandwidth

The sections below describe these options in more detail. Before you allocate bandwidth, you should plan for every queue that will be implemented in this map. You can then determine how much bandwidth to assign to each queue. If the map also implements CBWFQ, remember that the bandwidth for low-latency queues is allocated first.

**Allocating Guaranteed Absolute Bandwidth to a Low-Latency Queue.** The `<bandwidth>` value in the **priority** command is the maximum bandwidth (in Kbps) **guaranteed** to the queue. The valid range for the bandwidth in Kbps is 8 to 1,000,000.

For example, enter:

ProCurve(config-qos-map)# priority 512
The traffic placed in the queue receives priority above all other traffic until it reaches the bandwidth specified in the command.

If the link is uncongested, priority traffic is allowed to burst up to the interface rate; otherwise, priority traffic above the specified bandwidth can be dropped. The router automatically calculates the burst value, which it uses to allow traffic to temporarily burst above its maximum guaranteed limit. You should almost always keep the default value. However, you can configure the burst value manually with the `priority` command's `<burst>` option (range: 3 to 1,000,000). For example, enter:

```
ProCurve(config-qos-map)# priority 512 640
```

**Allocating a Guaranteed Percentage of Bandwidth to a Low-Latency Queue.** Use the `percent <value>` option in the `priority` command to allocate bandwidth as a percentage of the total interface bandwidth. This command is especially useful for protecting bandwidth allocation in multilink applications. When one link fails in a multilink, the available bandwidth changes. Allocating bandwidth by percentages ensures that each queue continues to receive only its proper share of the new total bandwidth.

**Note**

If it is important that the high priority traffic receives a certain, set bandwidth at all times, use the `priority <bandwidth>` command to specify this bandwidth as an absolute amount.

To calculate the priority percent value, divide the minimum amount of bandwidth needed for the traffic by the line rate (or the rate-limited data rate, if this option has been configured on the interface). For example, to specify 80 Kbps of minimum bandwidth on an interface with a line rate of 512 Kbps (eight E1 or T1 channels at 64 Kbps per channel), enter:

```
ProCurve(config-qos-map)# priority percent 16
```

**Guaranteeing Unlimited Bandwidth to a Low-Latency Queue.** If you specify the `unlimited` option, priority traffic will take all of the bandwidth available on the interface.

**Note**

Do *not* assign a queue unlimited bandwidth lightly. Even if the traffic in the queue is important, it is rarely critical enough to be worth starving out all other traffic. For example, it could even starve out a Telnet or HTTP session with the router.
Marking Low Latency Packets with a ToS Value

You can use the same QoS map entry to place packets in a low-latency queue and to mark these packets with a ToS value. Simply enter a `set` command as well as a `priority` command. See “Setting the ToS Value” on page 8-50.

Assigning the QoS Map to an Interface

The QoS map does not take effect until you apply it to a logical WAN interface. Valid interfaces include:

- PPP interfaces
- HLDC interfaces
- Frame Relay interfaces
- ATM subinterfaces
- demand interfaces

QoS maps apply to packets outbound on WAN connections. You can also apply a QoS map to an Ethernet interface for which you have configured rate-limiting.

To apply the QoS map to an interface, enter:

**Syntax:** qos-policy out `<mapname>`

For example:

ProCurve(config)# interface ppp 1
ProCurve(config-ppp 1)# qos-policy out LowLatency

**Note**

When specifying the QoS map, you include only the name, not the sequence number. This allows the interface to grant QoS to many different kinds of traffic. You can only configure one QoS map for an interface, but this map can include many entries, each creating a different low-latency queue. Other entries can also perform other types of QoS, such as marking packets or placing them in a class for CBWFQ.

If you attempt to assign a second map with a different name to an interface, it will overwrite the first.
Marking Packets with a ToS value

The ProCurve Secure Router can mark the ToS field of packets it forwards with an IP precedence or DiffServ value. These ToS values grant packets different types of service according to configurations in the connecting network. You should negotiate with your service provider or other connecting network which packets will be marked with which values and what level of service these values entail.

You can also mark packets with IP precedence values for implementing WFQ on the router. (See “Configuring WFQ” on page 8-14.)

However, you cannot assign a packet a ToS value in one map entry and then match that ToS value to a low-latency queue in another entry. The router stops processing the QoS map as soon as it finds a match for the packet.

You can, however, configure a QoS map entry to both mark packets and place them in a low-latency queue or CBWFQ class. Simply enter both the set command and the priority or bandwidth command.

You can configure the router to mark:
- traffic already marked with a QoS value
- traffic with a specific source and/or destination IP address
- traffic destined to a UDP protocol port
- bridged traffic

No matter what type of traffic you want to mark, you configure the router in the same way:
1. Create a QoS map entry.
2. Select the traffic to be marked.
3. Set the QoS value.
4. Assign the QoS map to an interface.

Creating a QoS Map Entry

To create a QoS map, enter the following command from the global configuration mode context:

Syntax: qos map <mapname> <sequence number>
Setting Up Quality of Service
Marking Packets with a ToS value

For example:

ProCurve(config)# qos map PacketMarking 10

The sequence number indicates the priority for the QoS map entry. Because the ProCurve Secure Router searches entries with the lowest numbers first, the lower the map’s number, the higher its priority.

You should be careful when configuring a QoS map entry to mark packets. For example, you might configure another QoS map entry to place traffic in a low-latency queue according to its QoS value. If you want the router to include the traffic it marks in this queue, you must assign the packet-marking entry a lower sequence number than the low-latency queue entry.

Selecting the Traffic to Be Marked

You select the traffic that the QoS map entry will mark by entering one of the match commands shown in Table 8-11.

Table 8-11. QoS Map Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Match Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToS value—IP precedence</td>
<td>match precedence &lt;0-7&gt;</td>
</tr>
<tr>
<td>ToS value—DiffServ</td>
<td>match dscp &lt;0-63&gt;</td>
</tr>
<tr>
<td>IP header—source or destination IP address and protocol port</td>
<td>match list &lt;ACL listname&gt;</td>
</tr>
<tr>
<td>destination UDP protocol port</td>
<td>match ip rtp &lt;first port number&gt; [&lt;last port number&gt;] [all]</td>
</tr>
<tr>
<td>bridged traffic</td>
<td>match protocol bridge [netbeui]</td>
</tr>
</tbody>
</table>

Each QoS map entry can use only one set of criteria to match traffic. To match another group of traffic, you must configure another entry.

Marking Traffic Already Set to a ToS Value. You can change the QoS value for packets that are already marked with an IP precedence or DiffServ value. Simply match the QoS map entry that you created to packets of that value:

Syntax: match [dscp <0-63> | precedence <0-7>]
The specific type of service granted to packets with different ToS values has only been loosely standardized. Devices in your network might use different values than devices in an external network to which the router is forwarding traffic. You can match the local network values and map them to values in the external network that guarantee equivalent service.

For an example, your devices only support IP precedence. However, your ISP allows you to mark packets for Expedited Forwarding. You could match the QoS map entry to packets with an IP precedence of 5 and then set the value to DSCP 46. Enter:

ProCurve(config-qos-map)# match precedence 5

Valid DiffServ (DSCP) values are from 0 to 63; valid IP precedence values are from 0 to 7. You will learn how to set the new value in “Setting the ToS Value” on page 8-50.

You can only match the entry to one value. If you want to map more than one value to the new value, you must configure a QoS map entry for each original value. (You should assign the QoS map the same name, but a different sequence number.)

**Marking Traffic According to Fields in the IP Header.** You can mark packets according to the source and/or destination IP addresses in IP headers.

For example, you could place mission-critical traffic from a specific network into the AF31 class (DSCP 26). If your service provider has agreed to support this DiffServ, it would grant these packets relatively more bandwidth and drop them only after other packets are dropped.

You can also select traffic according to source and/or destination TCP or UDP ports. For example, if your network uses a VoIP application that does not automatically mark its signaling traffic, you could configure a QoS map entry to match and mark packets destined to the port specified by your application.

You select the traffic to be marked by matching the QoS map entry to an extended ACL. The ACL actually selects the traffic. An extended ACL can define traffic according to its source and destination IP address as well as a variety of fields in the IP, TCP, or UDP header. You also use extended ACLs to mark traffic destined to a specific TCP or UDP protocol port.
To mark traffic selected by an ACL, you must complete several steps:

1. Configure an ACL.
   a. Create an extended ACL.
   b. Add any necessary deny entries to the ACL.
   c. Add permit entries for the addresses to or from which you want to mark traffic.

2. Match the QoS map entry to the ACL.

**Configuring an ACL.** Create an extended ACL by entering this command from the global configuration mode context:

**Syntax:** `ip access-list extended <listname>`

For example:

```
ProCurve(config)# ip access-list extended PacketMarker
```

You can select a range of addresses in an ACL. You can also select certain types of traffic by specifying source or destination protocol port.

For example, you can mark all traffic destined to TCP port 1720 with a ToS value:

```
ProCurve(config-ext-nacl)# permit tcp any any eq 1720
```

For the complete command syntax and more information about configuring ACLs, see “Classifying Traffic According to IP Header Fields” on page 8-23 or Chapter 5: Applying Access Control to Router Interfaces.

**Matching a QoS Map Entry to an ACL.** Move to the configuration mode context for the QoS map entry you have created. Then enter this command:

**Syntax:** `match list <ACL listname>`

For example:

```
ProCurve(config-qos-map)# match list PacketMarker
```

To actually select the IP precedence value, you must enter a `set` command. See “Setting the ToS Value” on page 8-50.
Marking Traffic Destined to a UDP Protocol Port. It can be important to prioritize traffic to specific, well-known UDP ports. For example, you do not want user traffic to starve out customers accessing your business's Web server. You can mark traffic from the server destined to the HTTP port with a higher QoS value.

Also, VoIP and other real-time traffic requires special handling. Congestion affects this traffic far more negatively than it does bursty data traffic. If your VoIP applications do not themselves mark traffic with a ToS value, you should configure the router to do so. Your service provider will inform you what type of service you can request with various ToS values.

You should note the UDP ports on which your VoIP applications operate. You can then match a QoS map entry to these ports.

You use this command to mark RTP packets destined to a range of UDP destination ports:

**Syntax:** match ip rtp <first port number> <last port number> [all]

For example, many VoIP applications operate between ports 16,384 and 32,764. You would enter:

```
ProCurve(config-qos-map)# match ip rtp 16384 32764
```

**Note**

The router matches all RTP packets destined to even port numbers in the specified range. (Typically, servers listen for user traffic on even ports.) If you want to match traffic to both even and odd ports, you must add the all keyword. For example:

```
ProCurve(config-qos-map)# match ip rtp 16384 33764 all
```

**Note**

You can also configure a low-latency queue on the local router interface for VoIP traffic. This is often a better solution than relying on WFQ. WFQ only grants high priority traffic relatively more bandwidth. It works best on relatively uncongested links for which only a small percentage of the traffic has been marked with a high priority QoS value. When many conversational subqueues are running, the effect of a higher IP precedence is diluted. Voice traffic may become jittery. You should place this traffic in a low-latency queue that guarantees it a maximum absolute amount of bandwidth. For more information on configuring bandwidth for VoIP traffic, see “Determining Bandwidth for VoIP” on page 8-33.
**Marking Bridged Traffic.** You can configure one or more interfaces on a
the ProCurve Secure Router to act as a bridge. In effect, the router extends a
LAN throughout two or more remote sites. Traffic between hosts at each local
site can obviously travel faster than that between hosts at different sites. Local
hosts are not only physically closer, but they can also take advantage of higher-
speed Ethernet connections.

To mark bridged traffic, you must first match the QoS map entry to it. Enter:

```
ProCurve(config-qos-map)# match protocol bridge
```

To mark only NetBEUI traffic, which manages communications between hosts
in a LAN, enter:

```
ProCurve(config-qos-map)# match protocol bridge netbeui
```

**Setting the ToS Value**

After you have selected the traffic to be marked by the QoS entry, you enter a
set command to specify the value with which it will be marked.

The router can mark packets with an IP precedence or with a DiffServ value.

For example, you may want to match packets that have an IP precedence value
of 5 and change the IP precedence value to 4. Or you may want to assign VoIP
traffic (for example, traffic destined to UDP ports 16,384 to 20,000) an IP
precedence value of 5.

To mark packets matched to the QoS map entry with an IP precedence value,
enter:

**Syntax:** set precedence <0-7>

For example, set packets to critical priority:

```
ProCurve(config-qos-map)# set precedence 5
```

Although the ProCurve Secure Router itself does not support DiffServ except
as mapped to IP precedence, devices in the forwarding network may do so. In
this case, you should have the router mark packets that match the criteria you
have selected with the appropriate value for a PHB:

**Syntax:** set dscp <0-63>

For example, mark packets for Expedited Forwarding:

```
ProCurve(config-qos-map)# set dscp 46
```
Assigning the QoS Map to an Interface

The QoS map does not take effect until you apply it to a logical interface. Valid interfaces include:

- PPP interfaces
- HDLC interfaces
- Frame Relay interfaces
- ATM subinterfaces
- demand interfaces
- Ethernet interfaces for which you have set a rate limit

To apply the QoS map to an interface, move to the interface configuration mode and enter:

**Syntax:** qos-policy out <mapname>

For example:

ProCurve(config)# interface frame-relay 1
ProCurve(config-fr 1)# qos-policy out PacketMarking

**Note**

When specifying the QoS map, you include only the name, not the sequence number. This allows the interface to grant QoS to many different kinds of traffic. You can only configure one QoS map for an interface, but this map can include many entries, each marking different packets. You can also mark packets in one QoS entry and establish a low-latency queue or a class for CBWFQ in an entry in the same map.

If you attempt to assign a second map with a different name to an interface, it will overwrite the first.

Example Packet Marking Configuration

Your LAN connects to an external network through the Internet. The local network sometimes uses the IP precedence value 5. However, your service provider charges a great deal for this type of service. You decide to lower the IP precedence of all packets with precedence 5 to 3. Also, your network uses SIP VoIP devices, and your devices do not mark their signaling traffic for VoIP. You must assign this traffic DiffServ PHB AF31.
You would complete the following configurations:

1. Create a QoS map entry for lowering the precedence for traffic with IP precedence 5:
   
   ProCurve(config)# qos map InternetConnection 10
   ProCurve(config-qos-map)# match precedence 5
   ProCurve(config-qos-map)# set precedence 3

2. Configure an ACL to select SIP signaling traffic, which travels to TCP and UDP port 5060:
   
   ProCurve(config)# ip access-list extended VoiceSignaling
   ProCurve(config-ext-nacl)# permit tcp any any eq 5060
   ProCurve(config-ext-nacl)# permit udp any any eq 5060

3. Configure another entry in the QoS map to mark traffic in the VoiceSignaling ACL for the AF31 PHB:
   
   ProCurve(config)# qos map InternetConnection 11
   ProCurve(config-qos-map)# match list VoiceSignaling
   ProCurve(config-qos-map)# set dscp 26

4. Assign the QoS map to the PPP interface:
   
   ProCurve(config)# int ppp 1
   ProCurve(config-ppp 1)# qos-policy out InternetConnection

   If so desired, you could first configure more entries for the map to place other traffic in low-latency queues or CBWFQ classes.
Configuring Rate Limiting for Frame Relay

Overview
Rate limiting helps to maintain QoS on a Frame Relay connection and to minimize the number of packets dropped during congestion.

Rate Limiting
The permanent virtual circuits (PVCs) established on a Frame Relay connection must share the bandwidth available to the carrier line. (For information on how to increase this bandwidth by aggregating several carrier lines, see Chapter 2: Increasing Bandwidth.) You can configure rate limiting to preserve QoS on each PVC.

Rate limiting defines the maximum amount of bandwidth a PVC is allowed to consume. You can set different rate limits for when the line is congested and when it is free.

FRF.12
When running voice or other real-time, delay-sensitive applications over a Frame Relay interface, you may want to fragment Frame Relay frames. Fragmenting frames reduces latency and improves performance. You should use fragmentation for lower-speed connections, particularly those with speeds of 512 Kbps or less.

VoIP traffic is sensitive to delay. The quality of a voice transmission is seriously decreased if its delay is more than 150 ms. Transmission delays accumulate at several points: they include, for example, the time it takes the router to process and encapsulate a frame and to transmit the frame across the WAN connection.
Serialization delay is the time it takes the router to transmit data out an interface. The larger the frame, the longer it takes to serialize it. VoIP frames are relatively small and have a short serialization delay, which maintains the quality of the voice stream at the other end of the link. However, large data frames can clog the interface, delaying VoIP frames as well. Frame Relay fragmentation reduces the times it takes to serialize large data frames by breaking large data frames into smaller frames. VoIP frames can then be interleaved among the fragments. (See Figure 8-7.)

Configuring Rate Limiting

By default, Frame Relay interfaces always forward packets at their transmission rate. However, because Frame Relay networks operate over shared lines, the network may sometimes be congested and unable to forward all the traffic the router sends it. Rate limiting prevents the router from forwarding packets only to have those packets dropped.

Rate limiting can also help maintain QoS between two peers that have a different amount of bandwidth. For example, the local router may have an MLFR connection over 2 E1 lines, providing 4 Mbps, while the router at the remote site only has 2 Mbps available for the connection. It is pointless for the local router to send 4 Mbps of data only to have the remote router drop the packets it cannot receive.
You shape Frame Relay traffic by setting the committed burst value \((B_c)\) and the excessive burst value \((B_e)\). These values determine how much bandwidth the Frame Relay subinterface can use when the line is and is not congested.

The total burst values for all PVCs on an interface should be less than the interface's access rate to save bandwidth for overhead bits the router does not count when calculating transmission rates. If you set the burst rates too high, the connection will become oversubscribed and queues will build. The access rate is the physical rate limit; manually limiting traffic to this rate is unnecessary and defeats the purpose of traffic shaping.

**Setting the Committed Burst Rate**

Your service level agreement (SLA) with your Frame Relay provider specifies the bandwidth that the provider guarantees each PVC, or the PVC's committed information rate (CIR). The CIR is calculated from the \(B_c\), which is the maximum number of bits that the service provider guarantees to forward during a certain interval of time \((T)\). The CIR is \(B_c/T\).

You should set a \(B_c\) for each Frame Relay subinterface to ensure that the PVC does not exceed its CIR. Some service providers penalize you for consistently transmitting more than the agreed-upon amount of traffic.

You configure the value in bits. Because the Secure Router OS, like most service providers, always considers the time interval to be one second, the \(B_c\) is effectively the CIR. To set the \(B_c\), enter the following command from a Frame Relay subinterface configuration mode context:

**Syntax:** `frame-relay bc <bits>`

You can set a \(B_c\) between 0 and 4,294,967,294 bps. For example, if your SLA guarantees a CIR of 1 Mbps, enter:

```
ProCurve(config-fr 1.101)# frame-relay bc 1000000
```

**Setting the Excessive Burst Rate**

The \(B_e\) sets the maximum number of bits that the router can transmit during \(T\). The \(B_e\) determines the rate at which the router can burst data above the committed rate when the Frame Relay network is not congested.

On the ProCurve Secure Router, the \(B_e\) is the number of bits *beyond* the \(B_c\) that the router can transmit, so the upper limit for bandwidth available on the interface is the sum of the \(B_c\) and \(B_e\).
Your SLA probably includes terms for bursting traffic past the CIR. Some providers allow subscribers to burst any amount of traffic. You could set the \( B_e \) so that, with the \( B_c \), it equals the physical access rate. You should be aware, however, that packets transmitted beyond the CIR may be dropped.

Frame Relay switches mark each packet burst past a PVC’s CIR with a Discard Eligible (DE) bit. For example, if a PVC’s \( B_c \) is 1.0 Mb, its \( B_e \) is 1.5 Mb, and it is transmitting traffic at full capacity, then the last 500 Kb of packets will be marked DE. Frame Relay switches drop DE packets first if the network becomes congested.

Some service providers do not allow you to burst traffic or only allow you to burst a limited amount of traffic. A PVC’s Excessive Information Rate (EIR) limits its maximum transmission rate when the network is not congested. If you have agreed upon an EIR, your service provider may penalize you for exceeding it.

You should set the \( B_e \) so that the PVC does not burst too much traffic. Enter the following command from the Frame Relay subinterface configuration mode context:

**Syntax:** frame-relay be <bits>

For example, your provider allows you to burst 500 Kbps past your CIR, so you enter:

```
ProCurve(config-fr 1.101)# frame-relay be 500000
```

You can set a \( B_e \) between 0 and 4,294,967,294 bps.

The sum of the committed and excessive burst values is the upper limit for bandwidth available on the interface, which should always exceed 8000 bps. You can dedicate up to 75 percent of the available bandwidth to low-latency queues and CBWFQ classes.

**Configuring Frame Relay Fragmentation**

You must first configure rate limiting as described above to structure how the router bursts traffic. You can then enable Frame Relay fragmentation and set the size for fragments.

You set the FRF.12 fragment size for individual Frame Relay subinterfaces. Move to the configuration mode context for the subinterface and enter:

**Syntax:** frame-relay fragment <threshold>
Setting Up Quality of Service
Configuring Rate Limiting for Frame Relay

For example:

ProCurve(config-fr 1.101)# frame-relay fragment 100

The threshold is the fragment size in bytes. Valid fragment sizes are between 64 and 1600 bytes. Because fragmentation is not implemented by default, there is no default fragment size.

Data frames can be fragmented and reconstructed without harming their integrity. They are not delay sensitive, and it does not matter if some of the data arrives slightly after the rest. VoIP frames, on the other hand, cannot be fragmented. Frames must arrive in order and in a continuous stream. With FRF.12 fragmentation, the router only fragments frames that exceed the threshold size. You should take care to set this threshold greater than the size of VoIP frames so that the router does not fragment them.

Almost all devices now use a maximum transmission unit (MTU) size of 1500 bytes. However, if any device to which the router will be forwarding traffic has a smaller MTU, the fragment threshold must be set at or below this MTU.

Example Frame Relay QoS Configuration

Your organization has a PVC carried on two E1 lines that together provide 4.096 Mbps to a remote site. This PVC carries both data and voice.

The PVC’s CIR is 1.5 Mbps. Your Frame Relay provider allows you to burst an unlimited amount traffic. However, the E1 lines carry several PVCs. This PVC connects to a site that only has a 2.048 Mbps Frame Relay connection. You do not want the router to transmit packets only to have them dropped on the other end, so you decide to set the maximum transmission rate to 2.048 Mbps.

You would shape traffic on the Frame Relay PVC as follows:

ProCurve(config)# interface frame-relay 1.1
ProCurve(config-fr 1.1)# frame-relay bc 1500000
ProCurve(config-fr 1.1)# frame-relay be 548000
ProCurve(config-fr 1.1)# frame-relay fragment 210
Configuring QoS for Ethernet

The ProCurve Secure Router allows you to apply rate limiting and QoS to Ethernet interfaces. These QoS mechanisms affect traffic passed through the router to the LAN.

Overview

You can configure these QoS mechanisms on Ethernet interfaces:

- rate limiting
- outbound QoS policies

You must enable rate limiting in order to apply a QoS map to the Ethernet interface.

Rate Limiting

Rate limiting helps maintain QoS when traffic transits across connections of varying bandwidth. For example, an Ethernet interface may pass traffic at 100 Mbps to a device that can forward it at only 10 Mbps. The second device will be forced to queue and finally drop many of these packets.

Rate limiting traffic passing through the Ethernet interfaces minimizes the number of packets the router ultimately drops.

Configuring Rate Limiting on an Ethernet Interface

Move to the Ethernet interface configuration mode context. Set the maximum bandwidth that the Ethernet interface will transmit with this command:

Syntax: traffic-shape rate <bps>

You can set the bandwidth between 1000 and 100,000,000 bps (100 Mbps).
Configuring QoS Policies on an Ethernet Interface

When you apply a QoS map to the interface, it can only draw on 75 percent of the maximum bandwidth set with the `traffic-shape rate` command.

For example, Network 1, which is shown in Figure 8-8, uses VoIP and maintains a Web server. The LAN uses 10-Mbps Ethernet connections. You could limit the Ethernet interface on the ProCurve Secure Router to 8 Mbps. You could then configure a QoS map to implement several queues on the Ethernet interface according to your organization's policies. You could configure a QoS map to grant VoIP, streaming video, and other real-time traffic 2 Mbps in a low-latency queue. You could then allocate, using CBWFQ, 25 percent of the remaining bandwidth to traffic destined to the Web server (IP address 192.168.1.20) and 25 percent to traffic from a remote site (network 192.168.4.0/24).

Because you are using the `remaining percent` keyword, the bandwidth allocated to the classes is calculated from the bandwidth that remains after the Secure Router OS subtracts the 2 Mbps guaranteed to low-latency queues from the 8-Mbps rate-limited bandwidth. Therefore, each class would receive 1.5 Mbps (25 percent of 6 Mbps).

Figure 8-8. Configuring Ethernet QoS
You would enter these commands to configure the QoS policy:

ProCurve(config)# ip access-list extended WebTraffic
ProCurve(config-ext-nacl)# permit tcp any host 192.168.1.20 eq www
ProCurve(config-ext-nacl)# exit
ProCurve(config)# ip access-list extended RemoteTraffic
ProCurve(config-ext-nacl)# permit ip 192.168.4.0 0.0.0.255 any
ProCurve(config)# qos map Outside 10
ProCurve(config-qos-map)# match ip rtp 16384 32764 all
ProCurve(config-qos-map)# priority 2000
ProCurve(config-qos-map)# qos map Outside 20
ProCurve(config-qos-map)# match list WebTraffic
ProCurve(config-qos-map)# bandwidth remaining percent 25
ProCurve(config-qos-map)# qos map Outside 30
ProCurve(config-qos-map)# match list RemoteTraffic
ProCurve(config-qos-map)# bandwidth remaining percent 25
ProCurve(config-qos-map)# interface eth 0/1
ProCurve(config-eth 0/1)# traffic-shape rate 8000000
ProCurve(config-eth 0/1)# qos-policy out Outside
Example: Configuring QoS for VoIP

You should now be able to configure QoS for specific applications. You will be guided through the process of configuring VoIP for the Frame Relay network shown in Figure 8-9.

Figure 8-9. Example QoS Configuration

This organization uses general switched telephone network (GSTN) telephones that follow the G.711 standard. VoIP calls must be carried from the headquarters to each remote site. The organization anticipates that at the busiest time of day the network should support up to 12 calls. You would configure a low-latency queue to guarantee this bandwidth.

To deliver QoS for relatively small, time-sensitive VoIP frames, you must:
- consider the special needs of your VoIP application
- define VoIP traffic
- determine the amount of bandwidth necessary for VoIP traffic
- mark signaling traffic for special treatment
- configure rate limiting and fragmentation if you are using Frame Relay
Enabling Application-Level Gateways for Applications with Special Needs

G.711 is an H.323 application, which handles VoIP traffic. The application may cause the VoIP traffic to behave in a different manner than data traffic. For example, it sends VoIP traffic on one port and receives it on another port. If you have enabled the Secure Router OS firewall, you must also enable the H.323 application-level gateway (ALG) so that the firewall will automatically permit return traffic. The ALG will also prevent the firewall from discarding VoIP traffic for exhibiting unusual behavior.

The H.323 ALG is disabled by default. Enter:

ProCurve(config)# ip firewall alg h323

Notes

If your VoIP application uses SIP, then you should determine to which port or ports the application sends traffic. The default port enabled for SIP on the ProCurve Secure Router is UDP port 5060. If your application uses a different port than you should open it with this command:

Syntax: ip firewall alg sip udp <port number>

If your RPT application uses unexpected ports, then the firewall may drop return traffic. You should enable firewall traversal to allow all packets that are part of an RTP session to pass through the firewall:

ProCurve(config)# ip rtp firewall-traversal

See Chapter 4: ProCurve Secure Router OS Firewall—Protecting the Internal, Trusted Network for more information on ALGs and firewall traversal.

Enabling SIP Services

The VoIP application in this example uses H.323. However, if users in your network use SIP applications, then your router may need to act as a SIP proxy and registrar server.

SIP is a peer-to-peer protocol that initiates, regulates, and terminates sessions that carry VoIP, instant messages, video streams, or other RTP packets between two users. SIP was designed to provide VoIP calls with many of the features associated with traditional telephone calls, such as dialing a number or receiving a busy signal.
Although SIP can theoretically operate directly between two end users, in practice, SIP proxy and registrar servers are usually necessary. For example, a user’s SIP device needs certain information in order to invite a second user to open a call, or session. This information includes:

- the user’s registration information, which associates the SIP user with its current location
- a route to the user’s current location

It is not feasible that each SIP device store all of this information for every user that it may need to reach.

Instead, devices send their registration information to registrar servers, which map users to their current locations. When one user calls another, the SIP device sends the invitation to its proxy servers. Proxy servers look up registrations and forward the invitations to the correct end user. Proxy servers can also forward other SIP messages used to maintain and terminate a session. Because the same server often provides both proxy and registrar services, the two can often be considered in conjunction with each other.

The ProCurve Secure Router supports both proxy and registrar services.

Follow these steps to configure SIP services on your ProCurve Secure Router:

1. Enable SIP from the global configuration mode context:
   
   **Syntax:** ip sip

2. Enable SIP proxy:
   
   **Syntax:** ip sip proxy

3. You should also configure your router to act as a registrar:
   
   a. From the global configuration mode context, enable the SIP registrar:
      
      **Syntax:** ip sip registrar
   
   b. Enable the proxy server to save the registration information that it receives from users to a local location database:
      
      **Syntax:** ip sip database local
Setting Up Quality of Service
Example: Configuring QoS for VoIP

c. You can configure various settings for the router’s registrar functions, including user authentication, expire times, and the registrar’s realm. Use these commands:

   **Syntax:** ip sip registrar [authentication | default-expires <1-2592000> | max-expires <1-2592000> | min-expires <0-3600> | realm <name>]

   Users must periodically refresh their registration. The **min-expires** keyword determines how often the router will allow a user to update its information. The **max-expires** keyword sets the upper limit before the router times out registrations for which it has not received an update.

4. You can also add static entries to the router’s location database. Use this global configuration mode command to associate a username with an IP address:

   **Syntax:** ip sip location <username> <A.B.C.D>

Defining VoIP Traffic

You define the traffic by configuring a QoS map entry and matching it to a set of traffic.

First, create the QoS map entry:

   ProCurve(config)# qos map VoiceMap 10

Next, configure the criteria for traffic to be placed in the low-latency queue.

   One of the most important steps in configuring a low-latency queue is determining the best way of defining its traffic. You want all VoIP traffic, but no other traffic, to match the criteria. Selecting traffic according to source address may not be a good solution: IP addresses are usually too dynamic and will not properly select all or only VoIP frames.

   One of the best solutions is to have your VoIP application mark VoIP frames with the DSCP 46 for Expedited Forwarding. (The closer to the edge a packet is marked, the better ToS packet marking functions.) You would then match packets with a DSCP 46 to the QoS map entry:

   ProCurve(config-qos-map)# match dscp 46
If the VoIP application cannot implement DiffServ or IP precedence, you can match packets according to their UTP RTP port destination. The documentation for your VoIP application should indicate this port. However, it can sometimes be difficult to determine the ports used by an application because they can vary widely. You may need to select a broad range of ports. For example:

```
ProCurve(config-qos-map)# match ip rtp 16384 32764 all
```

Remember to use the `all` option to match odd ports as well as even.

---

**Note**

You should select one method only for defining VoIP traffic. You can only configure one `match` command for each QoS map entry.

---

**Determining the Required Bandwidth**

Next, set the bandwidth for the low-latency queue. You perform the calculations described in “Determining Bandwidth for the Queue” on page 8-32.

The per-call bandwidth for G.711 traffic over an MLFR connection is 82.8 Kbps. However, in this example, your VoIP application supports VAD, so you reduce this value to 41.4 Kbps. You want the connection to support at least 20 calls at once, which means VoIP traffic must receive a guaranteed 828 Kbps of bandwidth.

The Frame Relay connection uses two T1 carrier lines, and each T1 carrier line provides 1.5 Mbps. Even if one line fails, the router should be able to provide the necessary bandwidth.

Enter:

```
ProCurve(config-qos-map)# priority 828
```

VoIP traffic will always receive at least 828 Kbps. Beyond this level, VoIP traffic can be dropped. When the lines are busy, VoIP traffic might try to burst far past the level you have decided is adequate. Setting the maximum limit of the queue to 828 Kbps ensures that VoIP traffic receives the bandwidth it needs, but does not starve out all other traffic.

When the connection is not congested, VoIP traffic can burst up to the 3.0 Mbps provided by the two T1 carrier lines.

If your VoIP devices automatically mark signaling packets for special treatment, you can now apply the QoS map to the WAN interface.
In this example, you would move to the Frame Relay interface and enter:

`ProCurve(config-fr 1)# qos-policy out VoiceMap`

**Marking Signaling Traffic for Special Treatment**

H.323 specifies that peers exchange signaling information to establish and maintain the call. H.323 sends this information over TCP port 1720 and UDP ports 1718 and 1719. Your VoIP devices may already mark signaling packets for assured forwarding. If they do not, however, you must do so.

You should place signaling traffic in at least the AF31 class (DSCP 26). Packets in this class are given medium-high priority and are the packets in the AF3 class dropped last during congestion.

You should configure a QoS map entry to mark signaling packets with the DiffServ value 26.

First, create an extended ACL to match packets sent to the TCP and UDP ports specified by your VoIP standard. For the application in this example, you would enter:

`ProCurve(config)# ip access-list extended VoiceSignaling`
`ProCurve(config-ext-nacl)# permit tcp any any eq 1720`
`ProCurve(config-ext-nacl)# permit udp any any range 1718 1719`

Use Table 8-12 to find the TCP and UDP ports used by your application.

**Table 8-12. Destination Ports for Voice Signaling**

<table>
<thead>
<tr>
<th>VoIP Standard</th>
<th>TCP Port</th>
<th>UDP Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.323</td>
<td>1720</td>
<td>1718</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1719</td>
</tr>
<tr>
<td>SIP</td>
<td>5060</td>
<td>5060</td>
</tr>
<tr>
<td>Multimedia Gateway Control Protocol</td>
<td>—</td>
<td>2427</td>
</tr>
<tr>
<td>(MGCP)</td>
<td></td>
<td>2727</td>
</tr>
<tr>
<td>Cisco Skinny Client Control Protocol</td>
<td>2000</td>
<td>—</td>
</tr>
</tbody>
</table>

Then configure the QoS map entry. It should have the same name as, but a different number than, the low-latency queue. For example:

`ProCurve(config-ext-nacl)# qos map VoiceMap 21`
Match the map to the ACL and set the DiffServ value:

ProCurve(config-qos-map)# match list VoiceSignaling
ProCurve(config-qos-map)# set dscp 26

Finally, apply the entire QoS map to the Frame Relay interface:

ProCurve(config)# interface frame-relay 1
ProCurve(config-fr 1)# qos-policy out VoiceMap

Configuring Frame Relay Rate Limiting

In addition to guaranteeing a set amount of bandwidth for VoIP, you should also shape all Frame Relay traffic to ensure that you do not exceed your service agreement. (You do not need to configure rate limiting if your VoIP traffic crosses over a PPP connection.)

When setting the committed and excessive burst rate, refer to your SLA with your Frame Relay provider. The CIR is the bandwidth the service provider guarantees will be available to the connection at any given moment. The EIR is the rate at which traffic can burst when the network is uncongested. In this example, the CIR for the connection to each remote site is 1.5 Mbps. The provider allows you to burst any amount of traffic, so you could set the excessive burst rate to the interface’s access rate. However, because two PVCs will be drawing on the same bandwidth, you might want to limit the excessive burst rate.

First, configure the Bc to match the CIR:

ProCurve(config-fr 1.102)# frame-relay bc 1500000

You can also limit the rate at which the PVC can burst past the committed rate. For example, you can force the PVC to use no more than two-thirds of the connection’s bandwidth. Set the Be to 500 Kbps so that the total available bandwidth will be 2.0 Mbps (determined by summing the Bc and the Be).

ProCurve(config-fr 1.102)# frame-relay be 500000
Configuring Frame Relay Fragmentation

It does not matter how much bandwidth you guarantee a queue if other frames clog up the interface when it is their turn to be transmitted. You should enable the interface to fragment large data frames to reduce serialization delay. Take care to set the size at which the router starts fragmenting frames above the size of VoIP frames, which, including the MLFR and IP headers, is 207 bytes. Set the fragmentation threshold:

```
ProCurve(config-fr 1.102)# frame-relay fragment 210
```

Monitoring QoS

You should periodically monitor your network to ensure that traffic is actually receiving the QoS specified by your organization. Problems with QoS may include:

- an excessive number of dropped packets
- dropped telephone calls or breaks in the voice stream
- jittery video streams

If the router is dropping an excessive number of packets, you may need to:

- configure rate limiting on the Ethernet interfaces or on Frame Relay subinterfaces
- raise the threshold for WFQ conversational subqueues

To solve problems with low-quality VoIP and video streams, you should reexamine the amount of bandwidth allocated to this traffic in low-latency queues. You should also confirm that the QoS map entries for such queues are actually selecting the correct traffic.

If the WAN simply demands more bandwidth from connections than the router can provide, no number of QoS configurations will eliminate all problems. You may need to reconsider your network and your organization's priorities and place limits on certain kinds of traffic from certain subnets. Your organization may also decide to renegotiate the WAN connection's CIR and EIR with a Frame Relay provider or to increase the connection's bandwidth. (See Chapter 2: Increasing Bandwidth.)
Setting Up Quality of Service
Monitoring QoS

Viewing QoS Maps

When monitoring QoS on the router, you should first eliminate problems arising from misconfigurations that result in the QoS policy not being applied to the traffic at all. The following are possible scenarios:

- Criteria was misconfigured—Examples include a miskeyed IP precedence value or misconfigured wildcard bits in an ACL.
- Priority settings were miskeyed.
- An entry order results in packets being selected by the wrong QoS map entry—Remember that the ProCurve Secure Router processes QoS map entries starting with the lowest sequence number.
- QoS map entries were misnamed—You apply the QoS map as a whole to the interface. If you have inadvertently given entries different names, the interface will not implement all the classes or low-latency queues that you have configured for it.
- A QoS map was not applied to the WAN interface.

Use the commands shown in Table 8-13 to view your configurations for QoS maps. You enter `show` commands from the enable mode context. You can also enter the commands from any configuration mode context by prefacing them with `do`. For example:

```
ProCurve(config)# do show qos map
```

Table 8-13. QoS Map show Commands

<table>
<thead>
<tr>
<th>View</th>
<th>Command Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>all QoS maps:</td>
<td><code>show qos map</code></td>
</tr>
<tr>
<td>• entries:</td>
<td></td>
</tr>
<tr>
<td>- criteria for matching traffic</td>
<td><code>show qos map &lt;mapname&gt;</code></td>
</tr>
<tr>
<td>- bandwidth or priority bandwidth (if set)</td>
<td><code>show qos map &lt;mapname&gt; &lt;sequence number&gt;</code></td>
</tr>
<tr>
<td>- ToS value with which packets will be marked (if set)</td>
<td></td>
</tr>
<tr>
<td>- number of packets the router has matched to the entry</td>
<td></td>
</tr>
<tr>
<td>• interfaces using the map and the map’s status (active or inactive)</td>
<td><code>show running-config qos-map</code> [verbose]</td>
</tr>
<tr>
<td>a specific QoS map</td>
<td></td>
</tr>
<tr>
<td>a specific QoS map entry</td>
<td></td>
</tr>
<tr>
<td>the QoS map applied to an interface</td>
<td></td>
</tr>
<tr>
<td>portion of the running-config dealing with QoS maps</td>
<td></td>
</tr>
</tbody>
</table>
Setting Up Quality of Service

Monitoring QoS

You can modify a QoS map entry by entering its configuration mode context and reentering commands.

You can delete a QoS map entry by entering:

**Syntax:** no qos map <mapname> [sequence number]

For example:

ProCurve(config)# no qos map VoiceMap 20

You can then reconfigure the map entry.

You can also delete the entire map by entering the mapname without the sequence number. For example:

ProCurve(config)# no qos map VoiceMap

If you determine that the QoS maps are properly configured, but the QoS is still unacceptable, you may need to redesign your QoS policies.

Managing Queues

You can monitor queues on logical interfaces with the following enable mode command:

**Syntax:** show queue [atm <subinterface number> | ethernet <slot>/<port> | frame-relay <interface number> | ppp <interface number>]

Enter `show queue` to see subqueues on all router interfaces. Enter the interface ID to see information about one interface only.

The screen displays statistics for interface queues, including:

- the current number of packets
- the highest number of packets
- the threshold level
- the number of packets dropped
- available bandwidth

If you notice that a great number of packets have been dropped, you may want to raise the threshold level. On Frame Relay interfaces, you should also configure rate limiting. You could consider renegotiating your CIR as well. In addition, you could configure rate limiting on the Ethernet interface, which
controls the amount of traffic passed to the lower-speed WAN interfaces. Rate limiting Ethernet traffic prevents the router from receiving and processing a great number of packets that it will only have to drop.

The `show queue` command also displays the number of currently active conversations on an interface as well as the highest number of conversations ever active at once. Remember that the effect of WFQ can be diluted by a large number of queues. If you find that an interface is constantly using an excessive number of subqueues, you should consider configuring LLQ for real-time traffic (if you have not already done so).

If the interface is only implementing WFQ, you should also consider configuring CBWFQ: scan through the queues, look for common criteria, and group several queues into a class.

### Troubleshooting Common Configuration Problems

Common problems you may encounter while configuring QoS policies on your router include:

- the map becoming inactive
- an Ethernet interface refusing to take a QoS map

#### A Map Becoming Inactive

The Secure Router OS will only activate a map if the interface has sufficient bandwidth to grant every queue established in the map the guaranteed rate. By default, the bandwidth available for special queues on an interface is 75 percent of the interface’s speed.

If you use the `percent` or `remaining percent` keywords to set the bandwidth, the Secure Router OS will not allow you to exceed the limit. However, if you use absolute bandwidths, the Secure Router OS will not recognize the problem until you assign the QoS map to the logical interface. The Secure Router OS will deactivate a map that guarantees more bandwidth than an interface has available.

When you use the `percent` keywords to configure CBWFQ in conjunction with low-latency queues, you may also inadvertently exceed an interface’s available bandwidth. For example, a Frame Relay interface is bound to a single E1 line. You allocate 60 percent of the interface’s bandwidth (1.23 Mbps) to several CBWFQ classes. You also guarantee a low-latency queue 600 Kbps, bringing the total up to 1.83 Mbps and exceeding the 1.536 Mbps limit. When you assign the map to the Frame Relay interface, the router will force the map to become inactive.
Using the **percent remaining** keywords helps to avoid this problem. The Secure Router OS allocates bandwidth from only that which remains after low-latency queues have been served. However, you can still make errors, so plan carefully before configuring the map. (See “Allocating Bandwidth to a Class” on page 8-26.)

A QoS map will also become inactive on an interface when the interface has not been bound to a physical carrier line. The Secure Router OS will report that the map is inactive because the interface has insufficient bandwidth.

**An Ethernet Interface Refusing to Take a QoS-Policy**

You must configure rate limiting in order to implement QoS on an Ethernet interface. Use the **traffic-shape rate** command as described in “Configuring Rate Limiting on an Ethernet Interface” on page 8-58.

---

**Quick Start**

This section provides the commands you must enter to quickly configure:

- **WFQ**
- **CBWFQ**
- **LLQ**
- Frame Relay fragmentation
- QoS on Ethernet interfaces

Only a minimal explanation is provided. If you need additional information about any of these options, check “Contents” on page 8-1 to locate the section that contains the explanation you need.

**Configuring WFQ**

1. WFQ is enabled by default on all interfaces with an E1 data rate or less. To re-enable WFQ on a connection on which it has been disabled, move to the logical interface configuration mode context for the connection:

   **Syntax:** `interface <interface ID>`

   For example:

   `ProCurve(config)# int ppp 1`
2. Enable WFQ and set the threshold level for how many packets each subqueue can hold (between 16 and 512):

   ProCurve(config-ppp 1)# fair-queue <packet threshold>

Configuring CBWFQ

1. If you plan to define classes according to the traffic’s source and destination IP address, you must create an extended ACL to select the network or networks that belong to a class. You should configure one ACL for each class:

   ProCurve(config)# ip access-list extended <listname>

2. Add statements to define networks as part of the class. Use wildcard bits to indicate the range of addresses. Enter first the source address and then the destination address. If you want to place certain types of traffic in the class, you can optionally specify a protocol such as TCP or UDP and indicate the source and destination port, or both.

   Syntax: [deny | permit] <protocol> [any | host <A.B.C.D> | <A.B.C.D> <wildcard bits>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>] [any | host <A.B.C.D> | <A.B.C.D> <wildcard bits>] [any | eq <port> | gt <port> | lt <port> | range <first port> <last port> | neq <port> | host <port>]

   For example:

   ProCurve(config-ext-nacl)# permit ip 192.168.8.0 0.0.0.255 any
   You could also permit HTTP traffic:

   ProCurve(config-ext-nacl)# permit tcp any any eq www

3. Determine how you will allocate the bandwidth for each class. You can assign classes an absolute amount of bandwidth, a percentage of bandwidth, or a percentage of the bandwidth remaining on an interface after low-latency queues have been served. All classes defined on an interface must use the same method.

4. Create a QoS map entry to define each class:

   Syntax: qos map <mapname> <map sequence number>

   For example:

   ProCurve(config)# qos map Class 10

   You can configure up to four classes for each interface and up to 16 classes total on the router.
5. Match the entry to the criterion for the class with one of the commands shown in Table 8-14.

For example:

```
ProCurve(config-qos-map)# match list Network1
```

### Table 8-14. QoS Map Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Match Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToS value—IP precedence</td>
<td>match precedence (&lt;0-7&gt;)</td>
</tr>
<tr>
<td>ToS value—DiffServ</td>
<td>match dscp (&lt;0-63&gt;)</td>
</tr>
<tr>
<td>IP header—source or destination</td>
<td>match list (&lt;ACL listname&gt;)</td>
</tr>
<tr>
<td>IP address and protocol port</td>
<td></td>
</tr>
<tr>
<td>destination UDP real-time protocol (RTP) port</td>
<td>match ip rtp (&lt;first port number&gt;) ([&lt;last port number&gt;]) [all]</td>
</tr>
<tr>
<td>bridged traffic</td>
<td>match protocol bridge [netbeui]</td>
</tr>
</tbody>
</table>

6. Set the bandwidth for the class as an absolute value, a percentage, or a percentage of bandwidth remaining after low latency queues have been served:

**Syntax:**

```
bandwidth [<Kbps> | percent <percentage> | remaining percent <percentage>]
```

For example:

```
ProCurve(config-qos-map)# bandwidth remaining percent 15
```

Percentages must be whole values. You can reserve up to 75 percent of the interface's total or rate-limited bandwidth for CBWFQ classes.

The range for absolute bandwidth is 8 to 200,000 Kbps. However, you are still limited to 75 percent of the interface's physical access rate or rate-limited rate.

7. Configure another class by repeating steps 3 through 5. If you want to establish low-latency queues as well, configure them as described on page 8-75. Assign each QoS map entry the same name but a different number.

For example:

```
ProCurve(config)# qos map Class 20
```

**Note**

QoS map names are case-sensitive.
8. Assign the QoS map to the logical interface for the WAN connection on which you want to enable CBWFQ. For example:

   ProCurve(config)# interface ppp 1
   ProCurve(config-ppp 1)# qos-policy out Class

Configuring a Low-Latency Queue

1. Create a QoS map entry to define the queue. For example:

   ProCurve(config)# qos map LowLatency 10

   **Syntax:** qos map <mapname> <map sequence number>

2. Match the entry to the criteria for the queue. Criteria include:
   - ToS value
   - IP header fields
   - UDP destination port
   - bridged traffic

   From the QoS map configuration mode context, enter this command:

   **Syntax:** match [dscp <0-63> | precedence <0-7> | list <ACL listname> | ip rtp <first port number> {<last port number>} {all} | protocol bridge {netbeui}]

   For example, you can select VoIP traffic by selecting the ToS value the VoIP application assigns it. Or you can select traffic destined to UDP ports associated with certain application:

   ProCurve(config-qos-map)# match ip rtp 16000 20000

3. Set the maximum bandwidth guaranteed to the queue:

   **Syntax:** priority [<Kbps> | percent <value> | unlimited] [burst <value>]

   Leave the burst value at the default. You can set the bandwidth to between 8 and 100,000 Kbps. However, you cannot allocate more than 75 percent of the interface’s bandwidth to queues. For example, enter:

   ProCurve(config-qos-map)# priority 500

   You can alternatively specify the guaranteed bandwidth needed for the low-latency traffic as a percentage of the total interface bandwidth. For example, enter:

   ProCurve(config-qos-map)# priority percent 16

   You should be wary of assigning a queue unlimited priority. Typically, even the most important traffic should not starve out all other traffic.
4. You can also mark traffic placed in a low-latency queue with a ToS value. Use a `set` command from the QoS map entry for the queue. (See step 3 in “Marking Packets” on page 8-76.)

5. If so desired, configure another queue. You can also configure classes for CBWFQ to be used with traffic that does not meet the criteria for low-latency queues. The map entries for the low-latency queues and the CBWFQ classes should use the same map name, but different map numbers.

6. Assign the QoS map to the logical interface for the WAN connection to support the low-latency queues. For example:

   ```
   ProCurve(config)# interface frame-relay 1
   ProCurve(config-fr 1)# qos-policy out <mapname>
   ```

**Marking Packets**

1. Create a QoS map entry to mark packets transmitted out an interface. If you are implementing other QoS mechanisms on the interface, you should give this entry the same map name as the entries for those mechanisms:

   **Syntax:** `qos map <mapname> <map sequence number>`

   For example:

   ```
   ProCurve(config)# qos map LowLatency 15
   ```

2. Match the entry to the criteria you have determined for traffic to be marked. Criteria include:

   - ToS value
   - IP header fields
   - UDP destination port
   - bridged traffic

   The command syntax for the QoS map configuration mode `match` command is:

   **Syntax:** `match [dscp <0-63> | precedence <0-7> | list <ACL listname> | ip rtp <first port number> {<last port number>} {all} | protocol bridge {netbeui}]`

   For example, you can mark bridged traffic with a ToS value and request better service for it from your service provider:

   ```
   ProCurve(config-qos-map)# match protocol bridge
   ```
3. Set the ToS value:

   **Syntax:** set [dscp 0-63 | precedence 0-7]

   For example:
   ProCurve(config-qos-map)# set dscp 34

4. If so desired, configure another entry to mark other packets.

5. Assign the QoS map to the logical interface that transmits the packets:
   ProCurve(config)# interface ppp 1
   ProCurve(config-ppp 1)# qos-policy out <mapname>

**Configuring Frame Relay Fragmentation**

1. Enable rate limiting on a PVC:
   a. Move to the Frame Relay subinterface for the PVC:
      ProCurve(config)# interface frame-relay <subinterface number>
      Set the committed burst rate:
      **Syntax:** frame-relay bc <bps>
      For example:
      ProCurve(config-fr 1.101)# frame-relay bc 700000
   b. Set the excessive burst rate:
      ProCurve(config-fr 1.101)# frame-relay be 400000
      **Syntax:** frame-relay be <bps>

2. Enable Frame Relay fragmentation for the PVC and set the fragment size:
   **Syntax:** frame-relay fragment <bytes>
   The threshold for fragmenting frames can be between 64 and 1600 bytes.
   For example, enter:
   ProCurve(config-fr 1.101)# frame-relay fragment 210
Configuring QoS on an Ethernet Interface

1. Move to the Ethernet interface configuration mode context and enable rate limiting:
   
   **Syntax:** traffic-shape rate \(<bps>\)

2. If you want the Ethernet interface to implement CBWFQ or low-latency queues, configure the QoS map as described above.

3. Apply the QoS map to the Ethernet interface. From the Ethernet interface configuration mode context, enter:

   **Syntax:** qos-policy out \(<mapname>\)