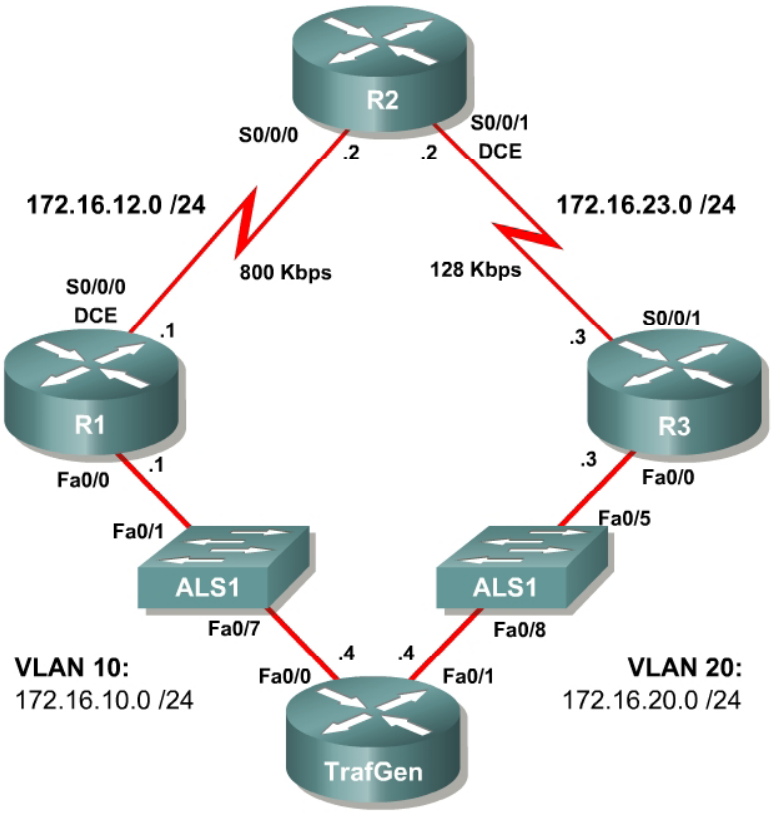
Lab 4.5 Class-based Queuing and NBAR

Learning Objectives

- Utilize NBAR for protocol detection
- Mark IP Precedence
- Allocate bandwidth using the Modular QoS Command-Line Interface
- Configure CBWFQ and LLQ queuing strategies

Topology Diagram



Scenario

In this lab, you will implement classification using Network-based Application Recognition (NBAR) and the Modular QoS CLI (MQC) to configure quality of service (QoS) on R1 and R2. You will configure both class-based marking and class-based queuing algorithms.

Preparation

This lab uses the Basic Pagent Configuration for TrafGen and the switch to generate and facilitate lab traffic in a stream from TrafGen to R1 to R2. Prior to beginning this lab, configure TrafGen (R4) and the switch according to the Basic Pagent Configuration in Lab 3.1: Preparing for QoS. You can accomplish this on R4 by loading the *basic-ios.cfg* file from flash memory into the NVRAM and reloading.

```
TrafGen# copy flash:basic-ios.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On the switch, load the *basic.cfg* file into NVRAM and reload the device.

```
Switch# copy flash:basic.cfg startup-config
Destination filename [startup-config]?
[OK]
2875 bytes copied in 1.456 secs (1975 bytes/sec)
TrafGen# reload
Proceed with reload? [confirm]
```

On TrafGen, instruct TGN to load the *basic-tgn.cfg* file and to start generating traffic.

```
TrafGen> enable
TrafGen# tgn load-config
TrafGen# tgn start
```

In addition, add the Fast Ethernet 0/5 interface on the switch to VLAN 20 since R3 will be the exit point from the network topology in this lab.

```
Switch# configure terminal
Switch(config)# interface fastethernet 0/5
Switch(config-if)# switchport access vlan 20
Switch(config-if)# switchport mode access
```

Step 1: Configure the Physical Interfaces

Configure all of the physical interfaces shown in the diagram. Set the clock rate on the serial link between R1 and R2 to 800000, the clock rate of the serial link between R2 and R3 to be 128000, and use the **no shutdown** command on all interfaces. Set the informational bandwidth parameter on the serial interfaces.

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 172.16.10.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# interface serial 0/0/0
R1(config-if)# bandwidth 800
R1(config-if)# ip address 172.16.12.1 255.255.255.0
R1(config-if)# clock rate 800000
```



```
R1(config-if)# no shutdown

R2(config)# interface serial 0/0/0
R2(config-if)# bandwidth 800
R2(config-if)# ip address 172.16.12.2 255.255.255.0
R2(config-if)# no shutdown
R2(config-if)# interface serial 0/0/1
R2(config-if)# bandwidth 128
R2(config-if)# ip address 172.16.23.2 255.255.255.0
R2(config-if)# clock rate 128000
R2(config-if)# no shutdown

R3(config)# interface fastethernet 0/0
R3(config-if)# ip address 172.16.20.3 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# interface serial 0/0/1
R3(config-if)# bandwidth 128
R3(config-if)# ip address 172.16.23.3 255.255.255.0
R3(config-if)# no shutdown
```

Issue the **show interfaces serial 0/0/0 | include Queueing** command on R1 to verify that the queuing strategy is Weighted Fair Queuing (WFQ).

```
R1# show interface serial0/0/0 | include Queueing
  Queueing strategy: weighted fair
```

If you see “fifo” as the queuing type, use the interface-level command **fair-queue** on the serial interface.

Step 2: Configure EIGRP AS 1

Configure routing between R1, R2 and R3 using Enhanced Interior Gateway Routing Protocol (EIGRP). Include the entire 172.16.0.0/16 major network in AS 1 and disable automatic summarization.

```
R1(config)# router eigrp 1
R1(config-router)# no auto-summary
R1(config-router)# network 172.16.0.0

R2(config)# router eigrp 1
R2(config-router)# no auto-summary
R2(config-router)# network 172.16.0.0

R3(config)# router eigrp 1
R3(config-router)# no auto-summary
R3(config-router)# network 172.16.0.0
```

Verify that the number of packets counted is increasing on the outbound interface of R3. Use the **show interfaces fastethernet 0/1** command**.** Issue the command twice to make sure the number of packets output has changed. If the number is not increasing, troubleshoot Layers 1, 2, and 3 connectivity and the EIGRP topology.

Step 3: Configure NBAR Protocol Discovery

NBAR is an IOS QoS feature that allows QoS decisions to be made based on individual protocols. Access control lists (ACLs) can be used to classify traffic based on headers for Layers 1 through 4 of the OSI model. NBAR, on the other hand, allows classification based on the upper layers of the OSI model—Layers 4 through 7. Since it does not rely on TCP/UDP port numbers at Layer 4, it can be used to identify traffic from applications that have dynamic port assignments. One standard feature of NBAR, known as protocol discovery, allows you to dynamically learn which application protocols are in use on your network. NBAR Protocol Discovery can also record and display the most used protocols.

For this lab, configure NBAR Protocol Discovery on the Fast Ethernet 0/0 interface on R1. The only IP traffic leaving the interface will be EIGRP Hello packets, so the majority of packets you should expect to see will be in the inbound direction. The protocols that protocol discovery shows heavy inbound traffic for are the protocols that traffic generation was configured for. To enable protocol discovery, use the interface-level command **ip nbar protocol-discovery**.

```
R1(config)# interface fastethernet0/0
R1(config-if)# ip nbar protocol-discovery
```

After protocol discovery has been enabled for a minute or two, you can see the information it has collected by using the command **show ip nbar protocol-discovery**. This command displays statistics globally for every interface in which NBAR protocol discovery is enabled. The protocols will be ranked based on traffic usage per interface. Notice that ingress and egress traffic is separated as it is in the output of the **show interfaces** command.

```
R1# show ip nbar protocol-discovery

 FastEthernet0/0
                            Input                     Output
                            -----                     ------
   Protocol                 Packet Count              Packet Count
                            Byte Count                Byte Count
                            5min Bit Rate (bps)       5min Bit Rate (bps)
                            5min Max Bit Rate (bps)   5min Max Bit Rate (bps)
   ------------------------ ------------------------- -------------------------
   ssh                      47691                     0
                            37214753                  0
                            800000                    0
                            800000                    0
   xwindows                 46638                     0
                            36235048                  0
                            797000                    0
                            797000                    0
   pop3                     47549                     0
                            37165341                  0
                            796000                    0
                            796000                    0
   smtp                     47112                     0
                            36874672                  0
```



```
                           794000                  0
                           794000                  0
   http                    47099                   0
                           36687939                0
                           791000                  0
                           791000                  0
   ntp                     44401                   0
                           34670597                0
                           770000                  0
                           770000                  0
   ftp                     45142                   0
                           35185881                0
                           767000                  0
                           767000                  0
   telnet                  44322                   0
                           34652510                0
                           762000                  0
                           762000                  0
   eigrp                   0                       17
                           0                       1258
                           0                       0
                           0                       0
<OUTPUT OMITTED>
```

NBAR uses a preconfigured set of port numbers, which it references during protocol discovery and normal classification operation. Issue the **show ip nbar port-map** command to view the protocol-to-port mappings. This command can also come in handy if you need to find out a well-known port number for an application and do not have access to outside resources. Existing protocol mappings can be modified and custom protocols can be defined, but those NBAR features are outside of the scope of this lab.

```
R1# show ip nbar port-map
port-map bgp                    udp 179
port-map bgp                    tcp 179
port-map bittorrent             tcp 6881 6882 6883 6884 6885 6886 6887 6888
6889
port-map citrix                 udp 1604
port-map citrix                 tcp 1494
port-map cuseeme                udp 7648 7649 24032
port-map cuseeme                tcp 7648 7649
port-map dhcp                   udp 67 68
port-map directconnect          tcp 411 412 413
port-map dns                    udp 53
port-map dns                    tcp 53
port-map edonkey                tcp 4662
port-map exchange               tcp 135
port-map fasttrack              tcp 1214
port-map finger                 tcp 79
port-map ftp                    tcp 21
port-map gnutella               udp 6346 6347 6348
port-map gnutella               tcp 6346 6347 6348 6349 6355 5634
port-map gopher                 udp 70
port-map gopher                 tcp 70
port-map h323                   udp 1300 1718 1719 1720 11720
port-map h323                   tcp 1300 1718 1719 1720 11000 – 11999
<OUTPUT OMITTED>
```

According to best QoS practices, where should packets be marked?

Mark as close as possible to the source, but not so close to the edge that the marking is made on an untrusted device.

What is a trust boundary in terms of classification and marking?

A trust boundary is a delineation of where markings will be honored and where they will not.

Step 4: Classify and Mark Packets

The Modular QoS CLI (MQC) allows someone to create QoS policies on a router in a modular and easy-to-understand format. When creating QoS policies using MQC, there are normally three configuration tasks:

1. Define traffic classes and the method of classification. Classes of traffic are defined in class maps using match statements. The match criterion can be an access list, NBAR-recognized protocol, QoS marking, packet size, and so forth.

2. Create a QoS policy to provision network resources for any traffic classes created in Step 1. A QoS policy maps QoS actions, such as marking, queuing, shaping, policing, or compression, to selected classes.

3. Finally, the policy is applied to an interface directionally, in either the inbound or outbound direction.

Certain policy-map commands can only be applied in a specific direction. For instance, queuing strategies can only be applied in the outbound policies. The router sends an error message to the console if a queuing policy is applied to an interface in the inbound direction, because this is an impossible configuration option.

On R1, you will create a QoS policy to mark an IP Precedence based on the application-layer protocol of the packets. The 3-bit IP Precedence field is part of the legacy Type of Service (ToS) byte on IP packets. Internet standards later converted this byte to the differentiated services (DiffServ) byte which contained the 6-bit differentiated services code point (DSCP) field. The three bits of the IP Precedence field map to the three high-order bits of the DSCP field for backwards-compatibility. For instance, WFQ does not look at the three low-order bits in the DSCP field, but does set weights for each flow based on the three high-order bits of the ToS/DS byte that are used for the IP Precedence

You will apply this QoS policy outbound on R1's Serial 0/0/0 interface.

Begin by implementing the first task: classification. Create traffic classes using NBAR for protocol recognition.

Class-maps are defined with the global configuration command **class-map** [*match-type*] *name*. The optional *match-type* argument can be set to either **match-any** or the default, **match-all**. This argument defines whether all of the successive match statements must be met in order for traffic to be classified into this class, or if only one is necessary.

Once in the class-map configuration mode, matching criteria can be defined with the **match** *criteria* command. To view all the possibilities of what can be matched on, use the **?** command. Choose to use NBAR for classification using the **match protocol** *name* command.

Create three traffic classes:

Critical: EIGRP or Network Time Protocol (NTP) traffic. These protocols are used for network control.

Interactive: Telnet, SSH, and XWindows traffic. These protocols are used for remote administration.

Web: HTTP, POP3, and SMTP traffic. These protocols are used for web and email access.

When creating these traffic classes, should you use the **match-any** or the **match-all** keyword?

You should employ the **match-any** keyword so that more than one protocol can be selected for each traffic class.

The classes created must match with the match-any mode so that any of the protocols listed can be matched. Obviously, it would be impossible for a packet to be two protocols at once.

```
R1(config)# class-map match-any critical
R1(config-cmap)# match ?
  access-group         Access group
  any                  Any packets
  class-map            Class map
  cos                  IEEE 802.1Q/ISL class of service/user priority values
  destination-address  Destination address
  discard-class        Discard behavior identifier
  dscp                 Match DSCP in IP(v4) and IPv6 packets
  flow                 Flow based QoS parameters
  fr-de                Match on Frame-relay DE bit
  fr-dlci              Match on fr-dlci
  input-interface      Select an input interface to match
  ip                   IP specific values
  mpls                 Multi Protocol Label Switching specific values
```



```
  not                  Negate this match result
  packet               Layer 3 Packet length
  precedence           Match Precedence in IP(v4) and IPv6 packets
  protocol             Protocol
  qos-group            Qos-group
  source-address       Source address
  vlan                 VLANs to match
R1(config-cmap)# match protocol eigrp
R1(config-cmap)# match protocol ntp
R1(config-cmap)# class-map match-any interactive
R1(config-cmap)# match protocol telnet
R1(config-cmap)# match protocol ssh
R1(config-cmap)# match protocol xwindows
R1(config-cmap)# class-map match-any web
R1(config-cmap)# match protocol http
R1(config-cmap)# match protocol pop3
R1(config-cmap)# match protocol smtp
```

You can verify created class-maps with the command **show class-map**.

```
R1# show class-map
 Class Map match-any critical (id 1)
   Match protocol eigrp
   Match protocol ntp

 Class Map match-any class-default (id 0)
   Match any

 Class Map match-any interactive (id 2)
   Match protocol telnet
   Match protocol ssh
   Match protocol xwindows

 Class Map match-any web (id 3)
   Match protocol http
   Match protocol pop3
   Match protocol smtp
```

The next task will be to define the QoS policy in a policy map. Create a policy map in global configuration mode using the **policy-map** *name* command. Segment the policy map by traffic class by issuing the **class** *name* command. The names of the classes will be the same as the class maps you created above. Additionally, there is the built-in class “class-default,” which matches any traffic not included in any other class.

```
R1(config)# policy-map markingpolicy
```

At the class configuration prompt, you can use various commands that will affect traffic of that class (use **?** to see what is available). To modify packets, use the command **set** *property value*. Create a new policy named “markingpolicy” and set the IP Precedence for matched packets as follows:

Critical: Set the IP Precedence to Network Control, represented by the value 7.

Interactive: Set the IP Precedence to Critical, represented by the value 5.

Web: Set the IP Precedence to Flash, represented by the value 3.

All other traffic: Set the IP Precedence of all other traffic to Routine, represented by the value 0. This value is the default value for IP Precedence.

There are different names for each value (these can be found out with the **?** command, and this is shown in the following output for reference).

```
R1(config-pmap)# class critical
R1(config-pmap-c)# set precedence ?
  <0-7>           Precedence value
  cos             Set packet precedence from L2 COS
  critical        Set packets with critical precedence (5)
  flash           Set packets with flash precedence (3)
  flash-override  Set packets with flash override precedence (4)
  immediate       Set packets with immediate precedence (2)
  internet        Set packets with internetwork control precedence (6)
  network         Set packets with network control precedence (7)
  priority        Set packets with priority precedence (1)
  qos-group       Set packet precedence from QoS Group.
  routine         Set packets with routine precedence (0)

R1(config-pmap-c)# set precedence 7
R1(config-pmap-c)# class interactive
R1(config-pmap-c)# set precedence 5
R1(config-pmap-c)# class web
R1(config-pmap-c)# set precedence 3
R1(config-pmap-c)# class class-default
R1(config-pmap-c)# set precedence 0
```

Verify the policy map configuration using the **show policy-map** command.

```
R1# show policy-map
  Policy Map markingpolicy
    Class critical
      set precedence 7
    Class interactive
      set precedence 5
    Class web
      set precedence 3
    Class class-default
      set precedence 1
```

Finally, apply the configuration outbound towards R2 with the interface-level command **service-policy** *direction name*.

```
R1(config)# interface serial 0/0/0
R1(config-if)# service-policy output markingpolicy
```

Once a policy map is applied to an interface, you can use an extended form of the **show policy-map** command by issuing the **show policy-map interface** *interface-name* command. This will give you detailed information and statistics on policy maps applied to an interface.

```
R1# show policy-map interface serial0/0/0
 Serial0/0/0

  Service-policy output: markingpolicy
```



```
Class-map: critical (match-any)
  13822 packets, 10617832 bytes
  5 minute offered rate 264000 bps, drop rate 0 bps
  Match: protocol eigrp
    5 packets, 320 bytes
    5 minute rate 0 bps
  Match: protocol ntp
    13817 packets, 10617512 bytes
    5 minute rate 264000 bps
  QoS Set
    precedence 7
      Packets marked 13822

Class-map: interactive (match-any)
  44974 packets, 34630670 bytes
  5 minute offered rate 830000 bps, drop rate 0 bps
  Match: protocol telnet
    15300 packets, 11765411 bytes
    5 minute rate 289000 bps
  Match: protocol ssh
    14451 packets, 11209788 bytes
    5 minute rate 270000 bps
  Match: protocol xwindows
    15223 packets, 11655471 bytes
    5 minute rate 282000 bps
  QoS Set
    precedence 5
      Packets marked 44984

Class-map: web (match-any)
  44600 packets, 34404320 bytes
  5 minute offered rate 857000 bps, drop rate 0 bps
  Match: protocol http
    13688 packets, 10530109 bytes
    5 minute rate 269000 bps
  Match: protocol pop3
    14513 packets, 11240708 bytes
    5 minute rate 290000 bps
  Match: protocol smtp
    16399 packets, 12633503 bytes
    5 minute rate 312000 bps
  QoS Set
    precedence 3
      Packets marked 44620

Class-map: class-default (match-any)
  13745 packets, 10547088 bytes
  5 minute offered rate 261000 bps, drop rate 0 bps
  Match: any
  QoS Set
    precedence 0
      Packets marked 13743
```

If a BGP packet with an IP precedence marking of 3 enters the Fast Ethernet 0/0 interface on R1 and is destined for R2, into which traffic class will the packet be classified?

Into the class-default class.

What IP precedence will the packet be assigned at the egress port?

The BGP packet will be assigned IP Precedence 0 for Routine Traffic.

Step 5: Shape Traffic and Queue with CBWFQ and LLQ

One of the QoS actions that can be performed in a policy map is shaping. Shaping limits traffic for a traffic class to a specific rate and buffers excess traffic. Policing, a related concept drops the excess traffic. Thus, the purpose of shaping is to buffer traffic so that more traffic is sent than if you policed at the same rate because not only will the traffic conforming to the policy be sent, but also buffered excess traffic when permitted.

Policing and shaping can each be configured within a policy map as a QoS action for a specific traffic class, or you can nest policy maps to create an aggregate shaper or policer. Multiple QoS actions can be taken on a specific class of traffic so you could use shaping in conjunction with marking or compression, or various other actions. Keep this in mind for the remaining labs

The first task in creating the QoS policy is to enumerate classes. This time, use uncreative names such as “prec7” and “prec5” for packets with IP Precedences 7 and 5, respectively. Create classes like this for IP Precedences 0, 3, 5, and 7—the in Module 4.

In this circumstance, however, you will view the class-based shapers in conjunction with low-latency queuing (LLQ). There are two class-based queuing tools, class-based weighted fair queuing (CBWFQ) and low-latency queuing (LLQ). CBWFQ is similar to custom queuing (CQ) in that it provisions an average amount or percent of bandwidth to a traffic class. However, the classification mechanism in class-based tools is much more powerful because it can also use NBAR to discover application protocols and even application protocol parameters, such as the URL in an HTTP request. LLQ is a simple improvement on CBWFQ, adding the ability to designate some classes as priority traffic and ensure that they are sent before others.

On R2, create a policy map to be applied on its Serial 0/0/1 interface. This policy map will be used to shape traffic based on markings by R1.possibilities for marking from the last step. To match on IP Precedence in a class definition, use the **match precedence** *precedence* command, where the *precedence* argument is the value or representative name. You must reclassify and mark EIGRP packets because each of the EIGRP packets is link-local traffic and the EIGRP packets which you marked on ingress at R1 were not sent to R2. The new packets for the link between R1 and R2 must now be classified by an access list or NBAR. However, any NTP packets traversing the link will already be marked with IP precedence 7. You should to treat EIGRP and NTP packets in the same traffic class for consistency.

Would you use the **match-all** or **match-any** keyword when creating the “prec7” class map? Explain.

You would use the **match-any** keyword so that you can match EIGRP traffic based on NBAR and NTP traffic based on IP Precedence 7.

Create the class map as follows.

```
R2(config)# class-map prec0
R2(config-cmap)# match precedence 0
R2(config-cmap)# class-map prec3
R2(config-cmap)# match precedence 3
R2(config-cmap)# class-map prec5
R2(config-cmap)# match precedence 5
R2(config-cmap)# class-map match-any prec7
R2(config-cmap)# match precedence 7
R2(config-cmap)# match protocol eigrp
```

Next, create the QoS policy to shape and queue the traffic. The syntax for entering the policy map and per-class configuration will be the same as above. However, rather than changing packet properties, we will set up low-latency queuing (LLQ) for the interface. LLQ is a variant of class-based weighted fair queuing (CBWFQ). Configuring CBWFQ involves assigning each traffic class dedicated bandwidth, either through exact bandwidth amounts or relative percentage amounts. LLQ is the configured the same way, except that one or more traffic classes are designated as priority traffic and assigned to an expedite queue. All traffic that enters the expedite queue up to the bandwidth limit will be sent as soon as possible, preempting traffic from non-priority classes.

While you configure either CBWFQ or LLQ, you can allocate a certain bandwidth for a traffic class, using the **bandwidth** *rate* command, where *rate* is a bandwidth amount in kilobits per second. Alternatively, use the **bandwidth percentage** *percent* command to allocate a percentage of bandwidth, where 100 percent of the bandwidth is set by the informational bandwidth parameter that you configured in Step 1.

For LLQ solely, issue the **priority** *rate* command or the **priority percentage** *percent* command in policy map configuration mode. These commands have the same arguments, which have the same effect as the **bandwidth** commands, except that they designate that queue as the priority queue.

Create a policy named “llqpolicy” on R2. The policy should allocate 10 percent of traffic to the “prec7” traffic class, 15 percent to the “prec5” traffic class, 30 percent to the “prec3” traffic class, and 20 percent to the “prec0” traffic class. Expedite traffic that falls into the “prec7” traffic class. Also, select weighted fair-queuing as the queuing method in the default traffic class with the **fair-queue** command.


```
R2(config)# policy-map llqpolicy
R2(config-pmap)# class prec7
R2(config-pmap-c)# priority percent 10
R2(config-pmap-c)# class prec5
R2(config-pmap-c)# bandwidth percent 15
R2(config-pmap-c)# class prec3
R2(config-pmap-c)# bandwidth percent 30
R2(config-pmap-c)# class prec0
R2(config-pmap-c)# bandwidth percent 20
R2(config-pmap-c)# class class-default
R2(config-pmap-c)# fair-queue
```

Verify your QoS policy configuration using the **show policy-map** command. Notice that the priority queue is a variant on the regular queues.

```
R2# show policy-map
  Policy Map llqpolicy
    Class prec7
      Strict Priority
      Bandwidth 10 (%)
    Class prec5
      Bandwidth 15 (%) Max Threshold 64 (packets)
    Class prec3
      Bandwidth 30 (%) Max Threshold 64 (packets)
    Class prec0
      Bandwidth 20 (%) Max Threshold 64 (packets)
    Class class-default
      Flow based Fair Queueing
      Bandwidth 0 (kbps) Max Threshold 64 (packets)
```

What traffic types would usually belong in a priority queue in a production environment?

Routing protocol traffic belongs in a priority queue so that adjacencies do not get lost. Any delay-sensitive traffic, such as Voice over IP (VoIP) or interactive video traffic, also belongs in a priority queue.

Use the same **service-policy** command from earlier to apply this policy map to the Serial 0/0/1 interface on R2 in an outbound direction.

```
R2(config)# interface serial 0/0/1
R2(config-if)# service-policy output llqpolicy
```

Verify using the interface-specific version of **show policy-map**.

```
R2# show policy-map interface serial0/0/1
 Serial0/0/1

  Service-policy output: llqpolicy

    Class-map: prec7 (match-any)
      3995 packets, 3387767 bytes
      5 minute offered rate 81000 bps, drop rate 80000 bps
      Match:  precedence 7
        3941 packets, 3384319 bytes
```



```
    5 minute rate 81000 bps
  Match: protocol eigrp
    54 packets, 3448 bytes
    5 minute rate 0 bps
  Queueing
    Strict Priority
    Output Queue: Conversation 40
    Bandwidth 10 (%)
    Bandwidth 12 (kbps) Burst 300 (Bytes)
    (pkts matched/bytes matched) 3947/3384695
    (total drops/bytes drops) 3524/3314514

Class-map: prec5 (match-all)
  8378 packets, 7165609 bytes
  5 minute offered rate 165000 bps, drop rate 146000 bps
  Match:  precedence 5
  Queueing
    Output Queue: Conversation 41
    Bandwidth 15 (%)
    Bandwidth 19 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 8378/7165609
    (depth/total drops/no-buffer drops) 64/7459/0

Class-map: prec3 (match-all)
  10295 packets, 8813462 bytes
  5 minute offered rate 197000 bps, drop rate 163000 bps
  Match:  precedence 3
  Queueing
    Output Queue: Conversation 42
    Bandwidth 30 (%)
    Bandwidth 38 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 10293/8810571
    (depth/total drops/no-buffer drops) 64/8500/0

Class-map: prec0 (match-all)
  3239 packets, 2830395 bytes
  5 minute offered rate 73000 bps, drop rate 52000 bps
  Match:  precedence 0
  Queueing
    Output Queue: Conversation 43
    Bandwidth 20 (%)
    Bandwidth 25 (kbps)Max Threshold 64 (packets)
    (pkts matched/bytes matched) 3239/2830395
    (depth/total drops/no-buffer drops) 60/1988/0

Class-map: class-default (match-any)
  26 packets, 1524 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any
  Queueing
    Flow Based Fair Queueing
    Maximum Number of Hashed Queues 32
    (total queued/total drops/no-buffer drops) 0/0/0
```

Challenge: Verifying IP Precedence

The topic of IP accounting is outside the scope of this curriculum. However, it is a useful tool for the verification of a marking policy. Issue the **ip accounting precedence** *direction* command in interface configuration mode to enable IP accounting on an interface. Apply this command on R3 for the Serial 0/0/1

interface that shows incoming markings from R2. View the accounting records for IP precedence by issuing the **show interfaces precedence** command.

```
R3(config)# interface serial0/0/1
R3(config-if)# ip accounting precedence input

R3# show interface precedence
Serial0/0/1
  Input
    Precedence 0:  10 packets, 5121 bytes
    Precedence 1:  230 packets, 85385 bytes
    Precedence 3:  193 packets, 127000 bytes
    Precedence 5:  88 packets, 62727 bytes
    Precedence 6:  5 packets, 320 bytes
    Precedence 7:  148 packets, 16984 bytes
```

Can you think of another simple way to count packets with each IP Precedence marking? You do not need to actually implement it. HINT: Think access lists.

You can create an extended access list with a permit statement for each IP Precedence, and then use **show access-lists** to look at the counters for each line. This is shown in the following output.

```
R3(config)# access-list 100 permit ip any any precedence 0
R3(config)# access-list 100 permit ip any any precedence 1
R3(config)# access-list 100 permit ip any any precedence 2
R3(config)# access-list 100 permit ip any any precedence 3
R3(config)# access-list 100 permit ip any any precedence 4
R3(config)# access-list 100 permit ip any any precedence 5
R3(config)# access-list 100 permit ip any any precedence 6
R3(config)# access-list 100 permit ip any any precedence 7
R3(config)# interface serial 0/0/1
R3(config-if)# ip access-group 100 in

R3# show access-list
Extended IP access list 100
    10 permit ip any any precedence routine
    20 permit ip any any precedence priority (88 matches)
    30 permit ip any any precedence immediate
    40 permit ip any any precedence flash (99 matches)
    50 permit ip any any precedence flash-override
    60 permit ip any any precedence critical (48 matches)
    70 permit ip any any precedence internet (9 matches)
    80 permit ip any any precedence network (74 matches)
```

Final Configurations

```
R1# show run
hostname R1
!
class-map match-any critical
 match protocol eigrp
 match protocol ntp
class-map match-any interactive
 match protocol telnet
 match protocol ssh
```



```
 match protocol xwindows
class-map match-any web
 match protocol http
 match protocol pop3
 match protocol smtp
!
policy-map markingpolicy
 class critical
  set precedence 7
 class interactive
  set precedence 5
 class web
  set precedence 3
 class class-default
  set precedence 0
!
interface FastEthernet0/0
 ip address 172.16.10.1 255.255.255.0
 ip nbar protocol-discovery
 no shutdown
!
interface Serial0/0/0
 ip address 172.16.12.1 255.255.255.0
 clock rate 800000
 service-policy output markingpolicy
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end

R2# show run
hostname R2
!
class-map match-all prec5
 match  precedence 5
class-map match-any prec7
 match  precedence 7
 match protocol eigrp
class-map match-all prec0
 match  precedence 0
class-map match-all prec3
 match  precedence 3
!
policy-map llqpolicy
 class prec7
  priority percent 10
 class prec5
  bandwidth percent 15
 class prec3
  bandwidth percent 30
 class prec0
  bandwidth percent 20
 class class-default
  fair-queue
!
interface Serial0/0/0
 ip address 172.16.12.2 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 bandwidth 128
```



```
 ip address 172.16.23.2 255.255.255.0
 clock rate 128000
 service-policy output llqpolicy
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
!
end

R3# show run
hostname R3
!
interface FastEthernet0/1
 ip address 172.16.20.3 255.255.255.0
 no shutdown
!
interface Serial0/0/1
 ip address 172.16.23.3 255.255.255.0
 no shutdown
!
router eigrp 1
 network 172.16.0.0
 no auto-summary
end
```

