Multicasting in IP and AppleTalk Networks

Over the past few years, the concept of end-users being able to send and receive audio and video (known collectively as *multimedia*) at the desktop has gained considerable attention and acceptance. With high-performance 486, Pentium, and PowerPC CPUs, more than 80 percent of the personal computers sold during 1995 were multimedia capable. Today, it is not uncommon for end-users to run video editing and image processing applications from the desktop.

The proliferation of more and more multimedia-enabled desktop computers has spawned a new class of multimedia applications that operate in networked environments. These network multimedia applications leverage existing network infrastructure to deliver video and audio applications to end users. Most notable are video conferencing and video server applications. With these applications, video and audio streams are transferred over the network between peers or between clients and servers.

There are three types of multimedia applications:

- *Unicast*—Unicast applications send one copy of each packet to each host that wants to receive the packet. This type of application is easy to implement, but it requires extra bandwidth because the network has to carry the same packet multiple times—even on shared links. Because unicast applications make a copy of each packet, the number of receivers is limited to the number of copies of each packet that can be made by the CPU that runs the unicast application.

- *Broadcast*—Broadcast applications send each packet to a broadcast address. This type of application is easier to implement than unicast applications, but it can have serious effects on the network. Allowing the broadcast to propagate throughout the network is a significant burden on both the network (in terms of traffic volume) and the hosts connected to the network (in terms of the CPU time that each host that does not want to receive the transmission must spend processing and discarding unwanted broadcast packets). You can configure routers to stop broadcasts at the LAN boundary (a technique that is frequently used to prevent broadcast storms), but this technique limits the receivers according to their physical location.

- *Multicast*—Multicast applications send each packet to a multicast group address. Hosts that want to receive the packets indicate that they want to be members of the multicast group. This type of application expects that networks with hosts that have joined a multicast group will receive multicast packets. Multicast applications and underlying multicast protocols control multimedia traffic and shield hosts from having to process unnecessary broadcast traffic.

This case study examines multicast protocols that have been developed for the Internet Protocol (IP) and for AppleTalk, as well as Cisco Internetwork Operating System (Cisco IOS) features that can help your network deliver video and audio smoothly.

Multicast Applications in IP Networks

Currently, support for IP multicasting comes from three protocols:

- Internet Group Management Protocol (IGMP)
- Protocol-Independent Multicast (PIM)
- Distance Vector Multicast Routing Protocol (DVMRP)

Network multimedia applications for IP use IGMP to join multicast groups. PIM and DVMRP use IGMP to determine the location of hosts that have joined a multicast group.

This section covers the following topics:

- Addressing
- Internet Group Management Protocol
- Protocol-Independent Multicast

Addressing

IP multicasting applications use Class D addresses to address packets. The high-order 4 bits of a Class D address are set to 1110, and the remaining 28 bits are set to a specific multicast group ID. Class D addresses are typically written as dotted-decimal numbers and are in the range of 224.0.0.0 through 239.255.255.255.

Some multicast group addresses are assigned as well-known addresses by the Internet Assigned Numbers Authority (IANA). These multicast group addresses are called *permanent host groups* and are similar in concept to the well-known TCP and UDP port numbers. Table 11-1 list the multicast address of three permanent host groups.

Table 11-1   Multicast Addresses for Permanent Host Groups

Permanent Host Group	Multicast Address
Network Time Protocol (NTP)	224.0.1.1
RIP-2	224.0.0.9
Silicon Graphics' Dogfight application	224.0.1.2

Internet Group Management Protocol

The Internet Group Management Protocol (IGMP) uses IP datagrams to allow IP multicast applications to join a multicast group. Membership in a multicast group is dynamic—that is, it changes over time as hosts join and leave the group.

Multicast routers that run IGMP use IGMP host-query messages to keep track of the hosts that belong to multicast groups. These messages are sent to the all-systems group address 224.0.0.1. The hosts then send IGMP report messages listing the multicast groups they would like to join.When the router receives a packet addressed to a multicast group, it forwards the packet on those interfaces that have hosts that belong to that group. If you want to prevent hosts on a particular interface from participating in a multicast group, you can configure a filter on that interface by using the **ip igmp access-group** interface configuration command.

Routers on which GMP is enabled periodically send IGMP host-query messages to refresh their knowledge of memberships present on their interfaces. If, after some number of queries, the router determines that no local hosts are members of a particular multicast group on a particular interface, the router stops forwarding packets for that group and sends a "prune" message upstream toward the source of the packet.

You can configure the router to be a member of a multicast group. This is useful for determining multicast reachability in a network. If a router is configured as a group member it can, for example, respond to an ICMP echo request packet addressed to a group for which it is a member. To configure the router as a member of a multicast group, use the **ip igmp join-group** interface configuration command.

Protocol-Independent Multicast

Protocol-Independent Multicast (PIM) is an IP multicast protocol that works with all existing unicast routing protocols. PIM has two modes that allow it to work effectively with two different types of multicast traffic distribution patterns: dense mode and sparse mode.

Dense mode PIM is designed for the following conditions:

- Senders and receivers are in close proximity to one another.

- There are few senders and many receivers.

- The volume of multicast traffic is high.

Sparse-mode PIM is designed for the following conditions:

- There are few receivers in a group.

- Senders and receivers are separated by WAN links.

Dense Mode

Dense-mode PIM uses a technique known as *reverse path forwarding*. When a router receives a packet, it sends the packet out all interfaces except the interface on which the packet was received. Reverse path forwarding allows a data stream to reach all LANs, possibly multiple times. If the router has interfaces for which no hosts are members of the multicast group for which the packet is intended or for which no downstream multicast router on that LAN has joined the group, the router sends a "prune" message up the distribution tree to inform the sender that it need not send subsequent packets for this multicast group.

Figure 11-1 shows how PIM works in dense mode.

Figure 11-1   PIM Dense-Mode Operation
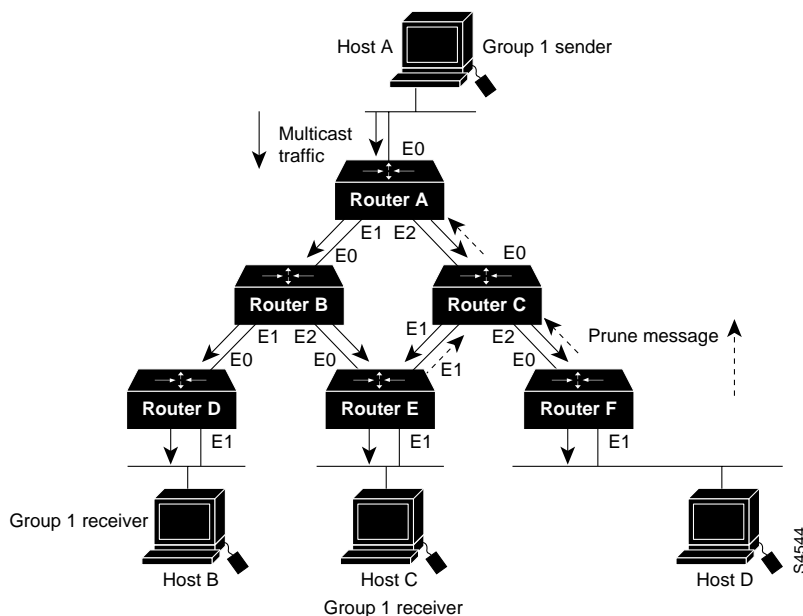
In Figure 11-1, Router A receives multicast traffic from Host A on Ethernet interface 0, duplicates each packet and sends the packets out on Ethernet interface 1 and Ethernet interface 2 to Routers B and C. Routers B and C duplicate the packets and send them out to Routers D, E, and F. Router D has a host that is a member of Group 1, so Router D does not send a prune message. Router E also has a host that is a member of Group 1, but because it receives the packets on two interfaces, Router E sends a prune message to Router C. (The decision about which router should be pruned is reached through a negotiation process conducted by Router B and Router C. If the connection between Router E and Router B had been a point-to-point link, the prune message would have been sent to Router B automatically without, thereby eliminating the need for Routers B and C to negotiate an agreement.)

Router F does not have any hosts that are members of Group 1, so it sends a prune message to Router C. Router C sends a prune message to Router A. After the prune messages are received, Router A sends multicast traffic for Group 1 to Router B only.

When you configure PIM in dense mode, you should enable IP multicast routing on every router over which multicast traffic will flow. The following commands configure dense mode PIM on Router B:

```
ip multicast-routing
interface ethernet 1
ip pim dense-mode
!
interface ethernet 2
ip pim dense-mode
```

The **ip multicast-routing** global configuration command enables IP multicast routing. You should include this command on every router that you want to participate in PIM. If some routers cannot be configured for IP multicast routing (for example, if they do not run a version of the Cisco IOS software release that supports PIM), you need to configure a tunnel so that multicast packets bypass these routers.

The **ip pim** interface configuration command enables PIM on the specified interface, and the **dense-mode** keyword enables dense mode. When you configure PIM in dense mode, you should apply the **ip pim** command with the **dense-mode** keyword to every interface that you want to forward multicast traffic.

Note Enabling PIM automatically enables IGMP.

In dense mode, the PIM-configured interface with the highest IP address on a LAN (subnet) is responsible for sending IGMP host-query messages to all hosts on the LAN. By default, the router that is responsible for sending PIM router-query messages sends them every 30 seconds. If you want to modify this interval, use the **ip pim query-interval** interface configuration command.

By default, a PIM-configured interface forwards all multicast packets. If you want to control the forwarding of packets, use the **ip multicast-threshold ttl** interface configuration command. The **ip multicast-threshold ttl** command changes the value of time-to-live (TTL) threshold, which the router compares with the TTL field in the IP header. Only those multicast packets that have a TTL greater than the TTL threshold are forwarded. You might, for example, want to set the TTL threshold to a very high value (such as 200) to prevent multicast packets from exiting an area.
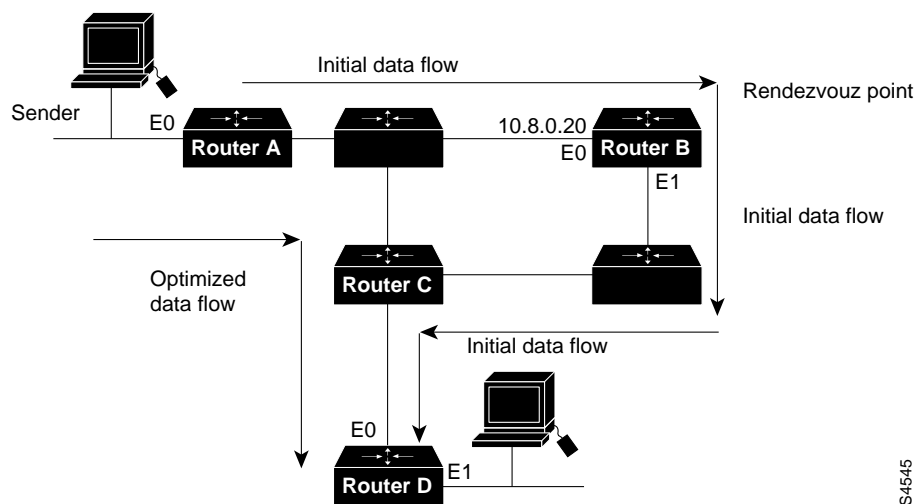
Sparse Mode

Sparse-mode PIM is designed for environments in which many multipoint data streams go to a relatively small number of the LAN segments. For this type of environment, dense mode PIM would use bandwidth inefficiently.

Sparse-mode PIM assumes that no hosts want to receive multicast traffic unless they specifically request it. In sparse-mode PIM, a router is designated as a rendezvous point. The rendezvous point collects information about multicast senders and makes that information available to potential receivers. When a sender wants to send data, it first sends the data to the rendezvous point. When a receiver wants to receive data, it registers with the rendezvous point. When the data stream begins to flow from sender to rendezvous point to receiver, the routers in the path automatically optimize the path to remove any unnecessary hops.

Figure 11-2 shows how PIM works in sparse mode.

Figure 11-2   PIM Sparse-Mode Operation

In Figure 11-2, Routers A and D are leaf routers. *Leaf routers* are routers that are directly connected either to a receiver or sender of multicast messages. The sparse-mode configuration of a leaf router designates one or more routers as rendezvous points. In this example, Router B is designated as the rendezvous point.

The leaf router that is directly connected to a sender (in this case, Router A) sends PIM register messages on behalf of the sender to the rendezvous point. The leaf router that is directly connected to a receiver (in this case, Router B) sends PIM join and prune messages to the rendezvous point to inform it about group membership.

The following commands configure Router A for sparse mode:

```
ip multicast-routing
ip pim rp-address 10.8.0.20 1
!
interface ethernet 0
ip pim sparse-mode
!
interface ethernet 1
ip pim sparse-mode
!
access-list 1 permit 224.0.1.2
```

The following commands configure Router D for sparse mode:

```
ip multicast-routing
ip pim rp-address 10.8.0.20 1
!
interface ethernet 0
ip pim sparse-mode
!
interface ethernet 1
ip pim sparse-mode
!
access-list 1 permit 224.0.1.2
```

The **ip multicast-routing** global configuration command enables IP multicast routing. When you configure PIM, IP multicast routing must be enabled on every router over which multicast traffic will flow. If some routers cannot be configured for IP multicast routing (for example, if they do not run a version of the Cisco IOS Software that supports PIM), you need to configure a tunnel so that multicast packets bypass these routers.

The **ip pim rp-address** global configuration command specifies the IP address of an interface on the router that is to be the rendezvous point and specifies that access list 1 is to be used to define the multicast groups for which the rendezvous point is to be used. The **ip pim rp-address** command must be configured on every sparse-mode router.

The **ip pim** interface configuration command enables PIM on the interface, and the **sparse-mode** keyword enables sparse mode. When you configure PIM in sparse mode, you should apply the **ip pim** command with the **sparse-mode** keyword to every interface that you want to forward multicast traffic.

The **access-list** global configuration command defines a standard IP access list that permits traffic using the multicast address 224.0.1.2 (the Silicon Graphics Dogfight application).

In sparse mode, the PIM-configured interface with the highest IP address on a LAN (subnet) is responsible for sending IGMP host-query messages to all hosts on the LAN and for sending PIM register and join messages toward the rendezvous point.

Note To configure a router as a rendezvous point, add the **ip multicast-routing** command and the **ip pim** command with the **sparse-mode** keyword to its configuration. The router recognizes its own IP address as the address of the rendezvous point and automatically assumes the functions of a rendezvous-point function.

Interoperability with Distance Vector Multicast Routing Protocol

The Distance Vector Multicast Routing Protocol (DVMRP) is another multicast protocol that has been developed for IP. DVMRP is similar to dense-mode PIM in that it uses reverse path forwarding. When a router receives a packet, it sends the packet out all interfaces except the interface that leads back to the source of the packet. If the router has interfaces for which no hosts are members of the multicast group for which the packet is intended, the router sends a "prune" message up the distribution tree to inform the sender that it need not send subsequent packets for this multicast group.

Although the Cisco IOS software does not support DVMRP, it does support interoperability with DVMRP-configured routers. PIM-configured routers dynamically discover DVMRP-configured routers on attached networks. When a DVMRP neighbor is discovered, PIM-configured routers periodically transmit DVMRP report messages advertising the unicast sources that are reachable in the PIM domain. By default, directly connected subnets and networks are advertised. The PIM-configured router forwards multicast packets that it receives from DVMRP routers into the PIM domain and, in turn, forwards multicast packets from the PIM domain to DVMRP routers.

Note When PIM-configured routers are directly connected to DVMRP routers or interoperate with DVMRP routers over a tunnel, the DVMRP routers should run mrouted Version 3.8. (The mrouted protocol is a public domain implementation of DVMRP.)

Interoperability between Directly Connected Routers

Figure 11-3 illustrates a topology in which a PIM-configured router is directly connected to a DVMRP-configured router.

Figure 11-3   PIM and DVMRP Interoperability
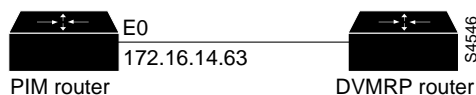
The following commands configure the PIM router for interoperability with the DVMRP router:

```
ip multicast-routing
!
interface ethernet 0
ip address 172.16.14.63 255.255.0.0
ip pim dense-mode
ip dvmrp metric 1 list 1
ip dvmrp metric 0 list 2
!
access-list 1 permit 192.168.35.0 0.0.0.255
access-list 1 permit 192.168.36.0 0.0.0.255
access-list 1 permit 192.168.37.0 0.0.0.255
access-list 1 deny 0.0.0.0 255.255.255.255
access-list 2 permit 0.0.0.0 255.255.255.255
```

The **ip dvmrp metric** interface configuration commands configure the metric that is to be associated with a set of destinations for DVMRP reports. The first **ip dvmrp metric** command causes the routes specified by access list 1 to be advertised to the DVMRP router (in this case, networks 192.168.35.0, 192.168.36.0, and 192.168.37.0). The second **ip dvmrp metric** command indicates that the routes specified by access list 2 are not to be advertised (in this case, all other routes). If you do not specify the routes that are to be advertised, only those subnets and networks that are directly connected to the PIM router will be advertised.

Interoperability over a Tunnel

DVMRP tunnels are used when one or more routers on a path do not support multicast routing. The router then sends and receives multicast packets over the tunnel. This allows a PIM domain to connect to a DVMRP router.

When a PIM-configured router interoperates with DVMRP over a tunnel, it advertises source routes in DVMRP report messages. In addition, the router caches any DVMRP report messages that it receives. The router uses the cached report messages as part of its reverse path forwarding calculation. This allows the router to forward multicast packets that it receives over the tunnel.

Figure 11-4 illustrates interoperability with DVMRP over a tunnel interface.

Figure 11-4   PIM and DVRMP Interoperability over a Tunnel
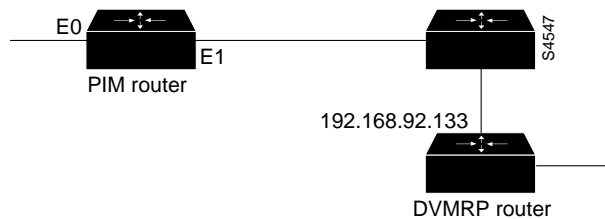
The following commands configure the PIM router:

```
ip multicast-routing
!
interface tunnel 0
ip address 192.168.47.1 255.255.255.0
ip pim dense-mode
tunnel source ethernet 1
tunnel destination 192.168.92.133
tunnel mode dvmrp
!
interface ethernet 1
ip address 192.168.23.23 255.255.255.0 secondary
ip address 192.168.243.2 255.255.255.0
ip pim dense-mode
ip dvmrp accept-filter 1
!
access-list 1 permit 192.168.48.0 0.0.0.255
access-list 1 permit 192.168.49.0 0.0.0.255
access-list 1 permit 192.168.50.0 0.0.0.255
access-list 1 deny 0.0.0.0 255.255.255.255
```

The **interface tunnel** global configuration command creates a tunnel (that is, a virtual interface).

The **tunnel source** interface configuration command specifies the interface that participates in the tunnel.

The **tunnel destination** interface configuration command specifies the IP address of the mrouted multicast router at the other end of the tunnel.

The **tunnel mode** interface configuration command use the **dvmrp** keyword to configure the tunnel as a DVMRP tunnel.

The **ip address** interface configuration command assigns an address to the tunnel to enable the sending of IP packets over the tunnel and to cause the router to perform DVMRP summarization. Alternatively, the **ip unnumbered** interface configuration command can be used. Either method allows IP multicast packets to flow over the tunnel. If the tunnel has a different network number than the subnet, subnets will not be advertised over the tunnel. In this case, only the network number is advertised over the tunnel.

By specifying the **dense-mode** keyword, the **ip pim** interface configuration command configures dense-mode PIM on the interface.

The **ip dvmrp accept-filter** interface configuration command configures an acceptance filter for incoming DVMRP reports. Routes that match the specified access list (in this case, access list 1) are stored in the DVMRP routing table (in this case, 192.168.48.0, 192.168.49.0, and 192.168.50.0). If a Cisco router is a neighbor to router running mrouted Version 3.6, the Cisco router can be configured to advertise network 0.0.0.0 to the DVMRP neighbor by using the **ip dvmrp default-information** command and specifying the **originate** keyword.

AppleTalk Multicasting

For AppleTalk, the Simple Multicast Routing Protocol (SMRP) supports the routing of multicast packets to multicast groups, with packet replication occurring only on those interfaces that have hosts that belong to the multicast group.

Network multimedia applications, such as QuickTime Conferencing (QTC), allow two or more hosts to participate in a QuickTime Conferencing session. End-users join the multicast group for the multicast transmissions they want to receive. SMRP conserves bandwidth by routing AppleTalk packets to all members of a multipoint group without producing duplicate packets on a particular network segment.

Figure 11-5 shows how SMRP works in an AppleTalk network.

Figure 11-5   SMRP in an AppleTalk Network
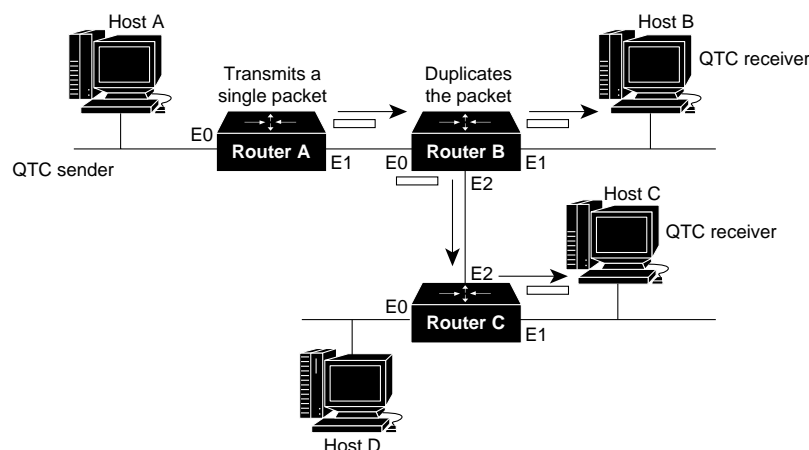
Router A receives a multicast packet from Host A and sends it to Router B. Two interfaces on Router B have hosts that have registered to receive this multicast transmission, so Router B duplicates the packet and sends one packet out on Ethernet interface 1 and the other packet out on Ethernet interface 2. Only one interface on Router C has hosts that have registered to receive this multicast transmission, so Router C sends the packet out on Ethernet interface 1.

The following commands configure SMRP on Router A:

```
smrp routing
!
interface ethernet 0
smrp protocol appletalk
!
interface ethernet 1
smrp protocol appletalk
```

The following commands configure SMRP on Router B:

```
smrp routing
!
interface ethernet 0
smrp protocol appletalk
!
interface ethernet 1
smrp protocol appletalk
```

```
interface ethernet 2
smrp protocol appletalk
```

The following commands configure SMRP on Router C:

```
smrp routing
!
interface ethernet 0
smrp protocol appletalk
!
interface ethernet 1
smrp protocol appletalk
!
interface ethernet 2
smrp protocol appletalk
```

The **smrp routing** global configuration command enables SMRP routing.

The **smrp protocol** interface configuration command enables SMRP on the interface, and the **appletalk** keyword specifies AppleTalk as the OSI Layer 3 protocol for SMRP.

Multicasting over WAN Connections

For the most part, users cannot detect the irregular arrival of data packets, but they can easily detect the irregular arrival of multimedia data, especially when that data includes an audio portion. Irregularly delivered video data is characterized by visible jitter and audible distortion. Smoothing jitter and distortion is especially desirable when multimedia data shares a low-bandwidth link with data traffic, as shown in Figure 11-6.

Figure 11-6   Multicast over WAN Connections
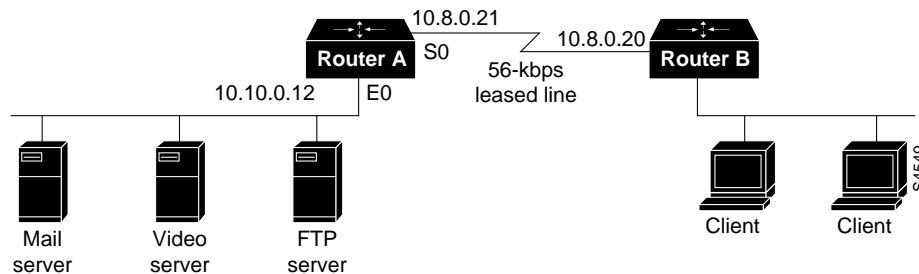
The Cisco IOS software provides three queuing algorithms that you can use to ensure that multicast traffic arrives at its destination without jitter and distortion: weighted fair queuing, priority queuing, and custom queuing. The queuing algorithm that is best for any particular network depends on the traffic flow characteristics of that network. You might want to try all three algorithms to determine the algorithm that provides the smoothest delivery for a particular network connection.

Weighted Fair Queuing

Weighted fair queuing (introduced in Cisco IOS Software Release 11.0) is enabled by default for all interfaces whose bandwidth is less than or equal to 2048 megabits per second (Mbps) and that do not use Link Access Procedure, Balanced (LAPB), X.25, PPP, or Synchronous Data Link Control (SDLC) encapsulations. (Weighted fair queuing cannot be enabled on interfaces that use these encapsulations.) Weighted fair queuing is a traffic priority management algorithm that identifies

conversations (traffic streams) and breaks them up to ensure that capacity is shared fairly. The algorithm examines fields in the packet header to identify unique conversations. For example, for AppleTalk, the algorithm uses the source network, node, and socket number; the destination network, node, and socket number; and the type. For IP, the algorithm uses the protocol, source and destination IP address; source and destination port number; and the TOS (type of service) field.

The weighted fair queuing algorithm sorts conversations into two categories: those that have high bandwidth requirements with respect to the capacity of the interface (such as FTP traffic) and those that have low bandwidth requirements (such as interactive sessions). For streams that have low-bandwidth requirements, the algorithm provides access with little or no queuing, and it shares the remaining bandwidth among other conversations. In effect, weighted fair queuing gives low-bandwidth traffic priority over high-bandwidth traffic, and high-bandwidth traffic shares the transmission service proportionally.

When weighted fair queuing is enabled on an interface, new messages for high-bandwidth conversations are discarded when the congestive-messages threshold is reached (the default congestive-messages threshold is 64 messages). To change the congestive-messages threshold, enter the following command, where **number** is a value between 1 and 512:

```
fair-queue number
```

Priority Queuing

Priority queuing allows you to establish queuing priorities based on protocol type. When you enable priority queuing on an interface, weighted fair queuing is disabled for that interface automatically.

The following commands configure priority queuing to ensure a certain quality of service level for Intel ProShare video conferencing on Router A in Figure 11-6:

```
interface serial 0
ip address 10.8.0.21 255.0.0.0
priority-group 1
!
access-list 101 permit ip any any
!
priority-list 1 protocol IP high UDP 5715
priority-list 1 protocol IP medium TCP 25
priority-list 1 protocol IP normal TCP 20
```

The **priority-group** interface configuration command assigns priority list 1 to serial interface 0.

The **priority-list protocol** global configuration commands establish a priority list that is associated with priority group 1. The priority list gives high priority to UDP packets destined for port number 5715 (the port number used by Intel ProShare), medium priority to TCP packets destined for port number 25 (SMTP mail), and normal priority to TCP packets destined for port number 20 (FTP data).

Custom Queuing

Another way to assure the timely delivery of multicast packets is to use custom queuing. With custom queuing, you can define up to 16 queues, assigning normal data to queues 1 through 15 and assigning system messages, such as keepalive messages, to queue 16. The router services each queue sequentially, transmitting a configurable percentage of traffic on each queue before transmitting packets from the next queue.

Custom queuing guarantees that mission-critical data is always assigned a certain percentage of the bandwidth, and it also assures predictable throughput for other traffic. For that reason, custom queuing is recommended for networks that need to provide a guaranteed level of service for all traffic.

When you enable custom queuing on an interface, weighted fair queuing is disabled for that interface automatically.

The following commands configure custom queuing for Router A in Figure 11-6:

```
interface serial 0
ip address 10.8.0.21 255.0.0.0
custom-queue-list 1
!
access-list 101 permit ip any any
!
queue-list 1 queue 1 byte-count 57900
queue-list 1 queue 2 byte-count 19300
queue-list 1 queue 3 byte-count 19300
!
queue-list 1 protocol IP 1 UDP 5715
queue-list 1 protocol IP 2 TCP 20
queue-list 1 protocol IP 3 TCP 25
```

The **custom-queue-list** interface configuration command assigns custom queue list 1 to serial interface 0.

The **queue-list queue byte-count** global configuration commands specify the size in bytes for three custom queues (in this case, 57,900, 19,300, and 19,300). Together, these **queue-list queue byte-count** commands have the effect of assigning 60 percent of the interface's bandwidth to packets in queue 1, 20 percent of the interface's bandwidth to queue 2, and 20 percent of the interface's bandwidth to queue 3.

The first **queue-list protocol** global configuration command assigns UDP packets destined for port 5715 to queue 1. The second **queue-list protocol** command assigns TCP packets destined for port 20 (SMTP mail) to queue 2, and the third **queue-list protocol** command assigns TCP packets destined for port 25 (FTP data) to queue 3.

Summary

The growing popularity of network multimedia applications, such as video conferencing, is driving the development of protocols that channel the flow of multicast packets to the networks and hosts that want to receive them. As multicasting protocols are deployed, unicast and broadcast applications will be upgraded to take advantage of multicast support, and new multicast applications will be developed.