vation protocols (Chapter 6). Indeed, introducing new protocols into the network layer is like replacing the foundation of a house—it is difficult to do without tearing the whole house down or at least temporarily relocating the house's residents. On the other hand, the Internet has witnessed rapid deployment of new protocols at the application layer. The classic examples, of course, are the Web, instant messaging, and peer-to-peer file sharing. Other examples include audio and video streaming and chat. Introducing new application-layer protocols is like adding a new layer of paint to a house—it is relatively easy to do, and if you choose an attractive color, others in the neighborhood will copy you. In summary, in the future we can expect to see changes in the Internet's network layer, but these changes will likely occur on a time scale that is much slower than the changes that will occur at the application layer.

4.8 ♦ Multicast Routing

The transport- and network-layer protocols we have studied so far provide for the delivery of packets from a single source to a single destination. Protocols involving just one sender and one receiver are often referred to as **unicast protocols.**

A number of emerging network applications require the delivery of packets from one or more senders to a *group of receivers.* These applications include bulk data transfer (for example, the transfer of a software upgrade from the software developer to users needing the upgrade), streaming continuous media (for example, the transfer of the audio, video, and text of a live lecture to a set of distributed lecture participants), shared data applications (for example, a whiteboard or teleconferencing application that is shared among many distributed participants), data feeds (for example, stock quotes), Web cache updating, and interactive gaming (for example, distributed interactive virtual environments or multiplayer games such as Quake). For each of these applications, an extremely useful abstraction is the notion of a **multicast:** the sending of a packet from one sender to multiple receivers with a single send operation.

In this section we consider the network-layer aspects of multicast. We will see that as in the unicast case, routing algorithms again play a central role in the network layer. We will also see, however, that unlike the unicast case, Internet multicast is *not* a connectionless service—state information for a multicast connection must be established and maintained in routers that handle multicast packets sent among hosts in a so-called multicast group. This, in turn, will require a combination of signaling and routing protocols in order to set up, maintain, and tear down connection state in the routers.

4.8.1 Introduction: The Internet Multicast Abstraction and Multicast Groups

From a networking standpoint, the multicast abstraction—a single send operation that results in copies of the sent data being delivered to many receivers—can be implemented in several ways:

- *One-to-all unicast.* One possibility is for the sender to use a separate unicast transport connection to each of the receivers. An application-level data unit that is passed to the sender's transport layer is then duplicated at the sender and transmitted over each of the individual connections. This approach implements a one-sender-to-many-receivers multicast abstraction using an underlying unicast network layer [Talpade 1997], requiring no explicit multicast support from the network layer to implement the multicast abstraction. This is shown in Figure 4.47(a), with network routers shaded in grey to indicate that they are not actively involved in supporting the multicast. Here, the multicast sender uses three *separate* unicast connections to reach the three receivers.
- *Application-level multicast*. A second possibility also uses unicast transmissions but involves the receivers in the replication and forwarding of data. Rather than having the sender itself transmit a copy to each receiver, the sender transmits a copy to a smaller number of receivers, which then make copies themselves and forward these copies on to other receivers, which may then duplicate and forward copies to yet additional receivers, and so on. This requires that receivers set up and maintain an application-level distribution infrastructure [Chu 2000; Penderakis 2001], as illustrated in Figure 4.47(b). Here, a single datagram is unicast from the sender to the upper-rightmost receiver. This receiver makes two copies, unicasting one copy to the receiver on its LAN and unicasting the second copy to the receiver on the upper LAN.
- *Explicit multicast.* The third possibility is to provide explicit multicast support at the network layer. In this approach, a *single* datagram is transmitted from the sending host. This datagram (or a copy of this datagram) is then replicated at a network *router* whenever it must be forwarded on multiple outgoing links in order to reach the receivers. Figure 4.47(c) illustrates this approach, with certain routers shaded in color to indicate that they are actively involved in supporting the multicast. Here, a single datagram is transmitted by the sender. That datagram is then duplicated by the router within the network; one copy is forwarded to the uppermost receiver, and another copy is forwarded toward the rightmost receivers. At the rightmost router, the multicast datagram is broadcast over the Ethernet that connects the two receivers to the rightmost router.

Clearly, the third approach makes more efficient use of network bandwidth in that only a *single* copy of a datagram will ever traverse a link. On the other hand, considerable network-layer support is needed to implement a multicast-aware network layer. Similarly, application-level multicast can be more efficient than one-to-all unicast but requires a significant amount of infrastructure to set up and maintain the application-level distribution architecture. In the remainder of this section, we'll focus on a multicast-aware network layer, as this approach poses a number of interesting challenges and exposes a number of the issues also faced by application-level multicast.

With multicast communication, we are immediately faced with two problems that are much more complicated than in the case of unicast—how to identify the receivers of a multicast datagram and how to address a datagram sent to these receivers.

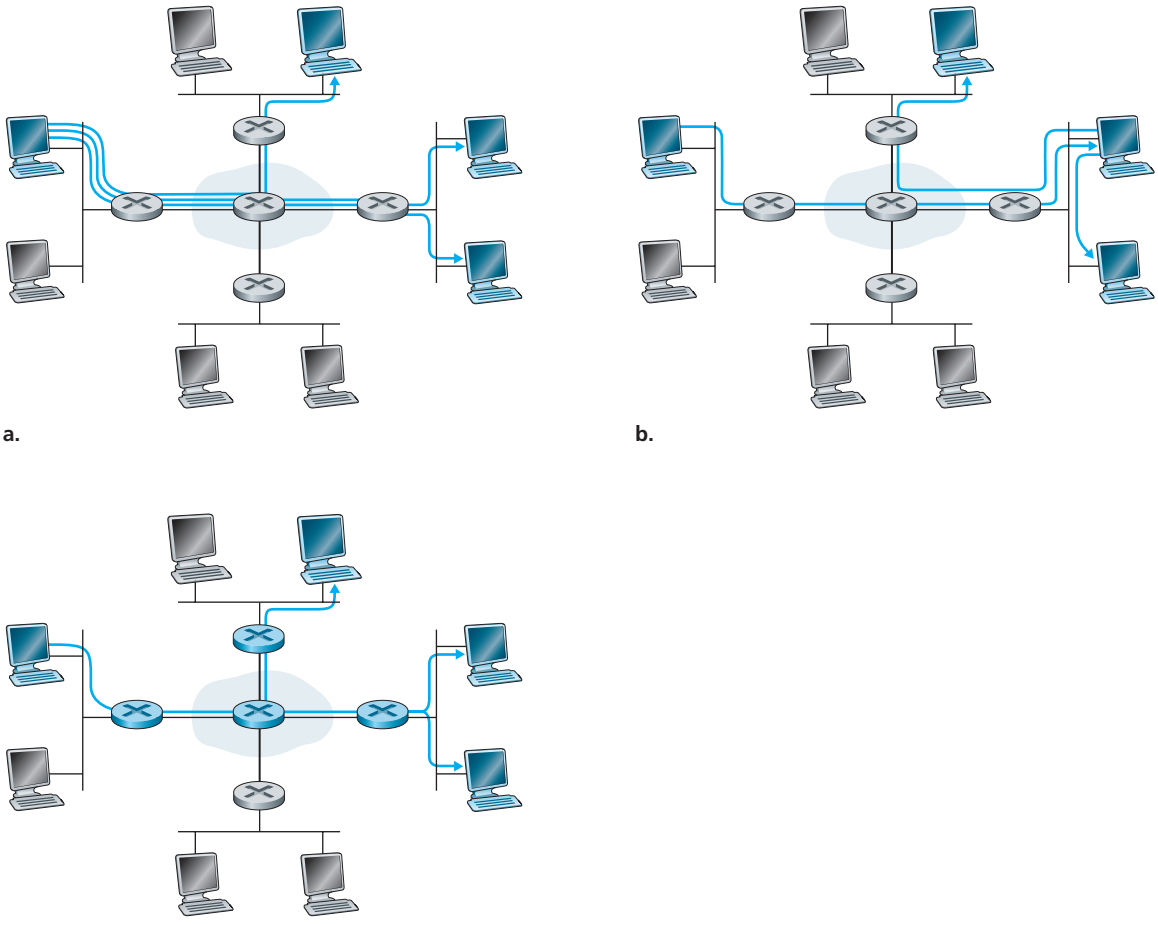

Figure 4.47 ♦ Three approaches for implementing the multicast abstraction

In the case of unicast communication, the IP address of the receiver (destination) is carried in each IP unicast datagram and identifies the single recipient. But in the case of multicast, we now have multiple receivers. Does it make sense for each multicast datagram to carry the IP addresses of all of the multiple recipients? While this approach might be workable with a small number of recipients, it would not scale well to the case of hundreds or thousands of receivers; the amount of addressing information in the datagram would swamp the amount of data actually carried in the datagram's payload field. Explicit identification of the receivers by the sender also requires that the sender know the identities and addresses of all of the receivers. We will see shortly that there are cases where this requirement might be undesirable.

For these reasons, in the Internet architecture (and the ATM architecture as well), a multicast datagram is addressed using **address indirection.** That is, a single identifier is used for the group of receivers, and a copy of the datagram that is addressed to the group using this single identifier is delivered to all of the multicast receivers associated with that group. In the Internet, the single identifier that represents a group of receivers is a class D multicast address, as we saw earlier in Section 4.4. The group of receivers associated with a class D address is referred to as a **multicast group.** The multicast group abstraction is illustrated in Figure 4.48. Here, four hosts (shown in shaded blue) are associated with the multicast group address of 226.17.30.197 and will receive all datagrams addressed to that multicast address. The difficulty that we must still address is the fact that each host has a unique IP unicast address that is completely independent of the address of the multicast group in which it is participating.

While the multicast group abstraction is simple, it raises a host (pun intended) of questions. How does a group get started and how does it terminate? How is the group address chosen? How are new hosts added to the group (either as senders or

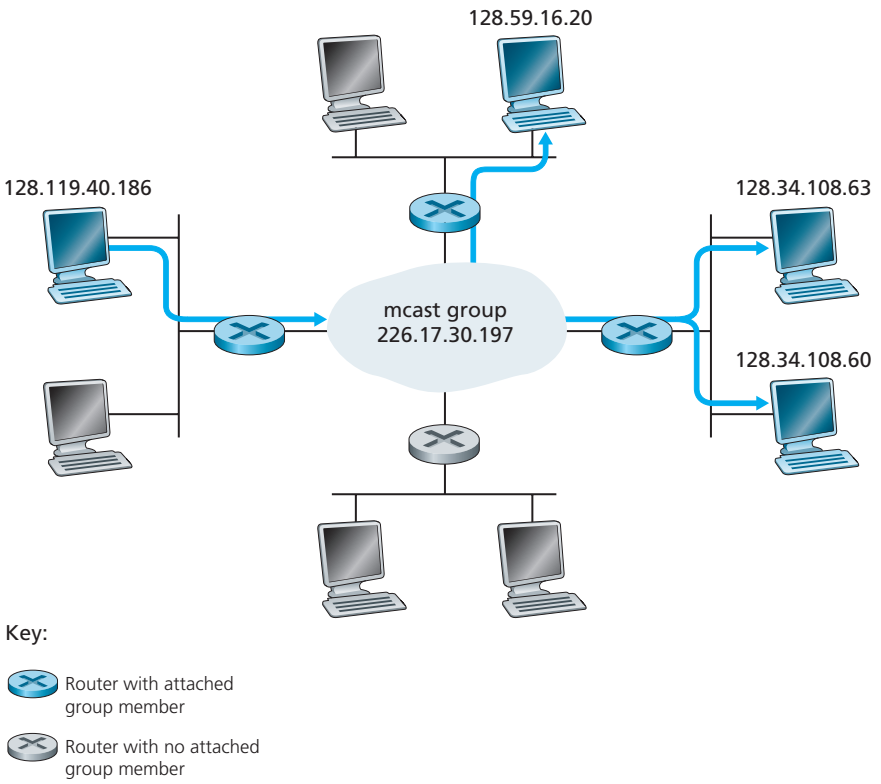

Figure 4.48 ♦ The multicast group: a datagram addressed to the group is delivered to all members of the multicast group.

receivers)? Can anyone join a group (and send to, or receive from, that group) or is group membership restricted and, if so, by whom? Do group members know the identities of the other group members as part of the network-layer protocol? How do the network routers interoperate with each other to deliver a multicast datagram to all group members? For the Internet, the answers to all of these questions involve the Internet Group Management Protocol [RFC 2236]. So, let us next consider the IGMP and then return to these broader questions.

4.8.2 IGMP

The **Internet Group Management Protocol, IGMP** version 2 [RFC 2236], operates between a host and its directly attached router (informally, think of the directly attached router as the "first-hop" router that a host would see on a path to any other host outside its own local network, or the "last-hop" router on any path to that host), as shown in Figure 4.49. Figure 4.49 shows three first-hop multicast routers, each connected to its attached hosts via one outgoing local interface. This local interface is attached to a LAN in this example, and while each LAN has multiple attached hosts, at most a few of these hosts will typically belong to a given multicast group at any given time.

IGMP provides the means for a host to inform its attached router that an application running on the host wants to join a specific multicast group. Given that the scope of IGMP interaction is limited to a host and its attached router, another protocol is clearly required to coordinate the multicast routers (including the attached routers) throughout the Internet, so that multicast datagrams are routed to their final destinations. This latter functionality is accomplished by the **network-layer multicast routing algorithms,** such as PIM, DVMRP, and MOSPF, which we will examine in

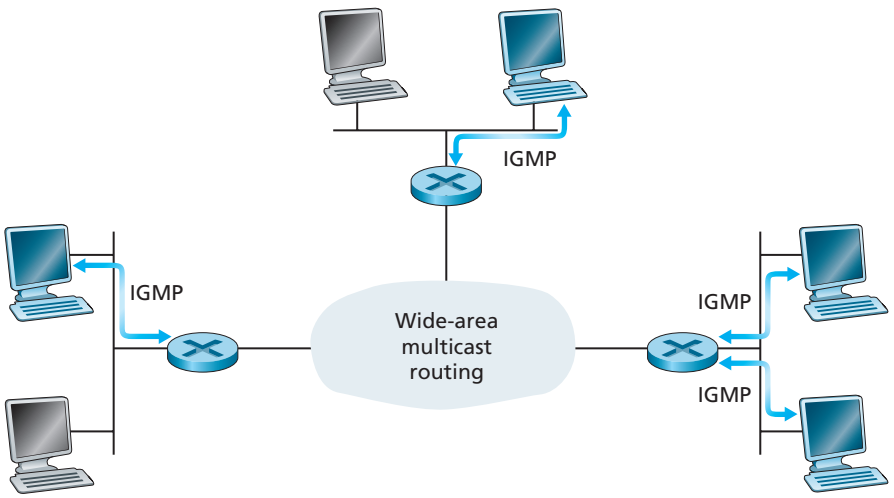

Figure 4.49 ♦ The two components of network-layer multicast: IGMP and multicast routing protocols

Sections 4.8.3 and 4.8.4. Network-layer multicast in the Internet thus consists of two complementary components: IGMP and multicast routing protocols.

Although IGMP is referred to as a "group membership protocol," the term is a bit misleading since IGMP operates locally, between a host and an attached router. Despite its name, IGMP is *not* a protocol that operates among all the hosts that have joined a multicast group, hosts that may be spread around the world. Indeed, there is *no* network-layer multicast group membership protocol that operates among all the Internet hosts in a group. There is no network-layer protocol, for example, that allows a host to determine the identities of all of the other hosts, network-wide, that have joined the multicast group. (See the homework problems for a further exploration of the consequences of this design choice.)

IGMP version 2 [RFC 2236] has only three message types, as shown in Table 4.4. A general membership_query message is sent by a router to all hosts on an attached interface (for example, to all hosts on a local area network) to determine the set of all multicast groups that have been joined by the hosts on that interface. A router can also determine whether a specific multicast group has been joined by hosts on an attached interface using a specific membership_query. The specific query includes the multicast address of the group being queried in the multicast group address field of the IGMP membership_query message, as shown in Figure 4.51.

Hosts respond to a membership_query message with an IGMP membership_report message, as illustrated in Figure 4.50. Membership_report messages can also be generated by a host when an application first joins a multicast group without waiting for a membership_query message from the router. Membership_report messages are received by the router, as well as all hosts on the attached interface (for example, in the case of a LAN). Each membership_report contains the multicast address of a single group that the responding host has joined. Note that an attached router doesn't really care *which* hosts have joined a given multicast group or even *how many* hosts on the same LAN have joined the same group. (In either case, the router's work is the same—it must run a multicast routing protocol together with other routers to ensure that it receives the multicast datagrams for the appropriate multicast groups.) Since a router really only cares about whether one or more of its

IGMP Message Types	Sent by	Purpose
Membership query: general	router	Query multicast groups joined by attached hosts
Membership query: specific	router	Query if specific multicast group joined by attached hosts
Membership report	host	Report host wants to join or is joined to given multicast group
Leave group	host	Report leaving given multicast group

Table 4.4. ♦ IGMP v2 Message Types

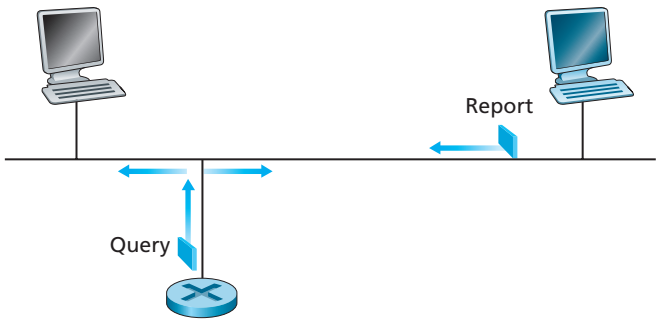

Figure 4.50 ♦ IGMP member query and membership report

attached hosts belong to a given multicast group, it would ideally like to hear from only one of the attached hosts that belongs to each group (why waste the effort to receive identical responses from multiple hosts?). IGMP thus provides an explicit mechanism aimed at decreasing the number of membership_report messages generated when multiple attached hosts belong to the same multicast group.

Specifically, each membership_query message sent by a router also includes a "maximum response time" field, as shown in Figure 4.51. After receiving a membership_query message and before sending a membership_report message for a given multicast group, a host waits a random amount of time between zero and the maximum response time value. If the host observes a membership_report message from some *other* attached host for that given multicast group, it *suppresses* (discards) its own pending membership_report message, since the host now knows that the attached router already knows that one or more hosts are joined to that multicast group. This form of **feedback suppression** is thus a performance optimization—it avoids the transmission of unnecessary membership_report messages. Similar feedback suppression mechanisms have been used in a number of Internet protocols, including reliable multicast transport protocols [Floyd 1997].

The final type of IGMP message is the leave_group message. Interestingly, this message is optional! But if it is optional, how does a router detect that there are no longer any hosts on an attached interface that are joined to a given multicast group? The answer to this question lies in the use of the IGMP membership_query message. The

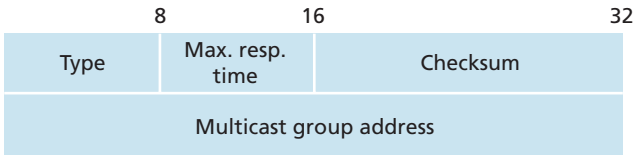

Figure 4.51 ♦ IGMP message format

router infers that no hosts are joined to a given multicast group when no host responds to a membership_query message with the given group address. This is an example of what is sometimes called **soft state** in an Internet protocol. In a soft-state protocol, the state (in this case of IGMP, the fact that there are hosts joined to a given multicast group) is removed via a timeout event (in this case, via a periodic membership_query message from the router) if it is not explicitly refreshed (in this case, by a membership_report message from an attached host). It has been argued that soft-state protocols result in simpler control than hard-state protocols, which not only require state to be explicitly added and removed, but also require mechanisms to recover from the situation where the entity responsible for removing state has terminated prematurely or failed [Sharma 1997]. An excellent discussion of soft state can be found in [Raman 1999].

The IGMP message format is summarized in Figure 4.51. Like ICMP, IGMP messages are carried (encapsulated) within an IP datagram, with an IP protocol number of 2.

Having examined the protocol for joining and leaving multicast groups, we are now in a better position to reflect on the current Internet multicast service model, which is based on the work of Steve Deering [RFC 1112; Deering 1990]. In this multicast service model, any host can join a multicast group at the network layer. A host simply issues a membership_report IGMP message to its attached router. That router, working in concert with other Internet routers, will soon begin delivering multicast datagrams to the host. Joining a multicast group is thus receiver-driven. A sender need not be concerned with explicitly adding receivers to the multicast group, but neither can it control who joins the group and therefore who receives datagrams sent to that group. Similarly, there is no control over who sends to the multicast group. Datagrams sent by different hosts can be arbitrarily interleaved at the various receivers (with different interleavings possible at different receivers). A malicious sender can inject datagrams into the multicast group datagram flow. Even with benign senders, since there is no network-layer coordination of the use of multicast addresses, it is possible that two different multicast groups will choose to use the same multicast address. From a multicast application viewpoint, this will result in interleaved extraneous multicast traffic.

These problems may seem to be insurmountable drawbacks for developing multicast applications. All is not lost, however. Although the network layer does not provide for filtering, ordering, or privacy of multicast datagrams, these mechanisms can all be implemented at the application layer. There is also ongoing work aimed at adding some of this functionality into the network layer [Cain 1999]. In many ways, the current Internet multicast service model reflects the same philosophy as the Internet unicast service model—an extremely simple network layer with additional functionality being provided in the upper-layer protocols in the hosts at the edges of the network. This philosophy has been unquestionably successful for the unicast case; whether the minimalist network-layer philosophy will be equally successful for the multicast service model still remains an open question. An alternate multicast service model is presented in [Holbrook 1999]. An interesting discussion of the current Internet multicast service model and deployment issues is [Diot 2000].

4.8.3 Multicast Routing: The General Case

In the preceding section we have seen how the IGMP protocol operates at the edge of the network between a router and its attached hosts, allowing a router to determine what multicast group traffic it needs to receive for forwarding to its attached hosts. We can now focus our attention on just the multicast routers: how should they route packets amongst themselves in order to ensure that each router receives the multicast group traffic that it needs?

Figure 4.52 illustrates the setting for the **multicast routing problem.** Let us consider a single multicast group and assume that any router that has an attached host that has joined this group may either send or receive traffic addressed to this group. In Figure 4.52, hosts joined to the multicast group are shaded in color; their immediately attached router is also shaded in color. As shown in Figure 4.52, among the population of multicast routers, only a subset of these routers (those with attached hosts that are joined to the multicast group) actually need to receive the multicast traffic. In Figure 4.52, only routers *A, B, E,* and F need to receive the multicast traffic. Since none of the hosts attached to router D are joined to the multicast group and since router C has no attached hosts, neither C nor D needs to receive the multicast group traffic.

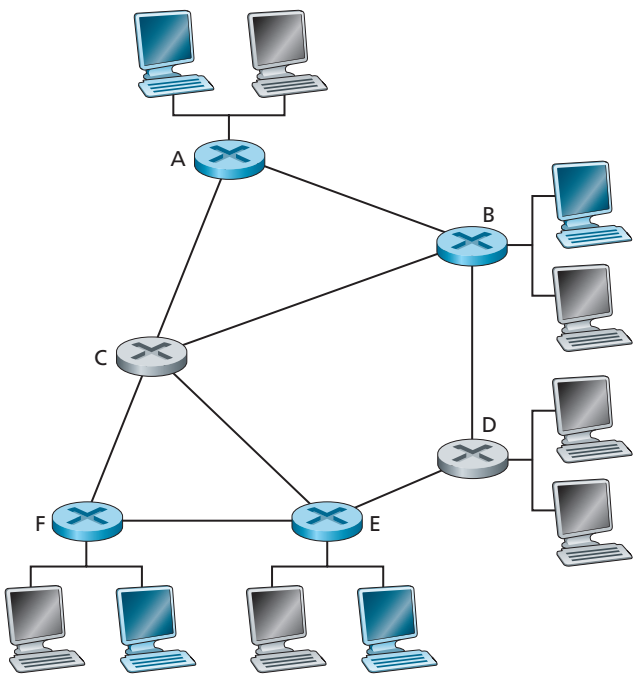

Figure 4.52 ♦ Multicast hosts, their attached routers, and other routers