5.8 Link Virtualization: A Network as a Link Layer

Because this chapter concerns link-layer protocols, and given that we're now nearing the chapter's end, let's reflect on how our understanding of the term *link* has evolved. We began this chapter by viewing the link as a physical wire connecting two communicating hosts, as illustrated in Figure 5.2. In studying multiple access protocols (Figure 5.9), we saw that multiple hosts could be connected by a shared wire and that the "wire" connecting the hosts could be radio spectra or other media. This led us to consider the link a bit more abstractly as a channel, rather than as a wire. In our study of Ethernet LANs (Figure 5.26) we saw that the interconnecting media could actually be a rather complex switched infrastructure. Throughout this evolution, however, the hosts themselves maintained the view that the interconnecting medium was simply a link-layer channel connecting two or more hosts. We saw, for example, that an Ethernet host can be blissfully unaware of whether it is connected to other LAN hosts by a single short LAN segment (Figure 5.9) or by a geographically dispersed switched LAN (Figure 5.26).

In Section 5.7 we saw that the PPP protocol is often used over a modem connection between two hosts. Here, the link connecting the two hosts is actually the telephone network—a logically separate, global telecommunications network with its own switches, links, and protocol stacks for data transfer and signaling. From the Internet link-layer point of view, however, the dial-up connection through the telephone network is viewed as a simple "wire." In this sense, the Internet virtualizes the telephone network, viewing the telephone network as a link-layer technology providing link-layer connectivity between two Internet hosts. You may recall from our discussion of overlay networks in Chapter 2 that an overlay network similarly views the Internet as a means for providing connectivity between overlay nodes, seeking to overlay the Internet in the same way that the Internet overlays the telephone network.

In this section, we'll consider asynchronous transfer mode (ATM) and Multiprotocol Label Switching (MPLS) networks. Unlike the circuit-switched telephone network, both ATM and MPLS are packet-switched, virtual-circuit networks in their own right. They have their own packet formats and forwarding behaviors. Thus, from a pedagogical viewpoint, a discussion of ATM and MPLS fits well into a study of either the network layer or the link layer. From an Internet viewpoint, however, we can consider ATM and MPLS, like the telephone network and switched-Ethernets, as link-layer technologies that serve to interconnect IP devices. Thus, we'll consider both MPLS and ATM in our discussion of the link layer. Frame-relay networks can also be used to interconnect IP devices, though they represent a slightly older (but still deployed) technology and will not be covered here; see the very readable book [Goralski 1999] for details. Our treatment of ATM and MPLS will be necessarily brief, as entire books could be (and have been) written on these networks. We recommend [Black 1995, Black 1997] and [Davie 2000] for details on ATM and MPLS, respectively. We'll focus here primarily on how these networks serve to interconnect IP devices, although we'll dive a bit deeper into the underlying technologies as well.

5.8.1 Asynchronous Transfer Mode (ATM) Networks

The standards for **asynchronous transfer mode (ATM)** networks were first developed in the mid-1980s, with the goal of designing a single networking technology that would transport real-time audio and video as well as text, e-mail, and image files. Two groups, the ATM Forum (now known as the MFA Forum [MFA Forum 2007]) and the International Telecommunications Union [ITU 2007], were involved in the development of ATM standards. They defined a complete end-to-end standard, ranging from the specification of the application interface to ATM down to the bit-level framing of ATM data over various fiber, copper, and radio physical layers. In practice, ATM has been used primarily within telephone and IP networks, serving, for example, as a link-layer technology to connect IP routers, as discussed above.

Principal Characteristics of ATM

As discussed in Section 4.1, ATM supports several service models, including constant bit rate service, variable bit rate service, available bit rate service, and unspecified bit rate service. ATM is a packet-switched, virtual-circuit (VC) network architecture. Recall that we've considered VCs at some length in Section 4.2.1. ATM's overall architecture is organized into three layers, as shown in Figure 5.32.

The **ATM adaptation layer (AAL)** is roughly analogous to the Internet's transport layer and is present only at the ATM devices at the edge of the ATM network. On the sending side, the AAL is passed data from a higher-level application or protocol (such as IP, if ATM is being used to connect IP devices). On the receiving side it passes data up to the higher-layer protocol or application. AALs have been defined for constant bit rate services and circuit emulation (AAL1), for variable bit-rate services such as variable bit rate video (AAL2), and for data services such as IP-datagram transport (AAL5). Among the services performed by the AAL are error detection and segmentation/reassembly. The unit of data handled by the AAL is referred to by the rather generic name of **AAL protocol data unit (PDU)**, which is roughly equivalent to a UDP or TCP segment.

The AAL5 PDU is shown in Figure 5.33. The PDU's fields are relatively straightforward. The PAD ensures that the PDU is an integer multiple of 48 bytes, because the PDU will be segmented to fit into the 48-byte payloads of the underlying ATM packets (known as *ATM cells*). The length field identifies the size of the PDU payload, so that the PAD can be removed at the receiver. The CRC field provides for error detection using the same cyclic redundancy check as Ethernet. The payload field can be up to 65,535 bytes long.

Let's now drop down one layer and consider the **ATM layer**, which lies at the heart of the ATM architecture. The ATM layer defines the structure of the ATM cell

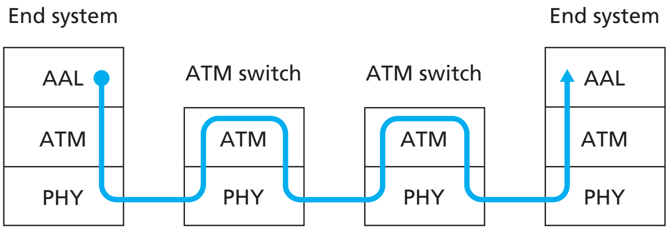
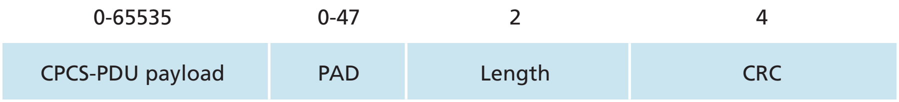


Figure 5.32 ♦ The three ATM layers. The AAL layer is only present at the edges of the ATM network.



0-65535	0-47	2	4
CPCS-PDU payload	PAD	Length	CRC

Figure 5.33 ♦ AAL5 PDU

and the meaning of the fields within the cell. The ATM cell is as important to an ATM network as the IP datagram is to an IP network. The first 5 bytes of the cell constitute the ATM header; the remaining 48 bytes constitute the ATM payload. Figure 5.34 shows the structure of the ATM cell header.

The fields in the ATM cell have the following functions:

- **Virtual-channel identifier (VCI).** Indicates the virtual channel to which the cell belongs. As with most network technologies that use virtual circuits, a cell's VCI is translated from link to link (see Section 4.2.1).
- **Payload type (PT).** Indicates the type of payload contained in the cell. There are several data payload types, several maintenance payload types, and an idle cell payload type. The PT field also includes a bit that serves to indicate the last cell in a fragmented AAL PDU.
- **Cell-loss priority (CLP) bit.** Can be set by the source to differentiate between high-priority traffic and low-priority traffic. If congestion occurs and an ATM switch must discard cells, the switch can use this bit to first discard low-priority traffic.
- **Header error control (HEC) byte.** Error-detection bits that protect the cell header.

Before a source can begin sending cells to a destination, the ATM network must first establish a **virtual channel (VC)** from source to destination. A virtual channel is nothing more than a virtual circuit, as described in Section 4.2.1. Each VC is a path consisting of a sequence of links between source and destination. A **virtual channel identifier (VCI)** is associated with each link on the VC. Whenever a VC is established or torn down, VC translation tables must be updated (see Section 4.2.1.).

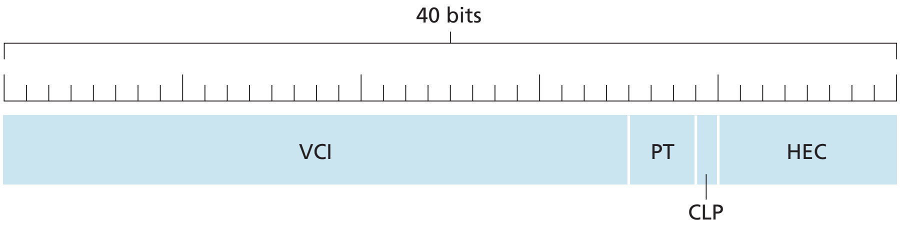


Figure 5.34 ♦ The format of the ATM cell header

If permanent VCs are used, there is no need for dynamic VC establishment and teardown. When dynamic VC establishment and teardown are called for, the Q.2931 protocol [Black 1997, ITU-T Q.2931 1994] provides signaling needed among the ATM switches and end systems.

The **ATM physical layer** is at the very bottom of the ATM protocol stack, and deals with voltages, bit timings, and framing on the physical medium. A good deal of the physical layer depends on the link's physical characteristics. There are two broad classes of physical layers: Those that have a transmission frame structure (for example, T1, T3, SONET, or SDH) and those that do not. If the physical layer has a frame structure, then it is responsible for generating and delineating frames. The use of the term *frames* here should not be confused with the link-layer (e.g., Ethernet) frames used in the earlier sections of this chapter. The transmission frame here is a physical-layer TDM-like mechanism for organizing the bits sent on a link.

IP over ATM

Now let's consider how an ATM network can be used to provide connectivity between IP devices. Figure 5.35 shows an ATM backbone with four entry/exit points for Internet IP traffic. Note that each entry/exit point is a router. An ATM backbone can span an entire continent and may have tens or even hundreds of ATM switches. Most ATM backbones have a permanent VC between each pair of entry/exit points. By using permanent VCs, ATM cells are routed from entry point to exit point without having to establish and tear down VCs dynamically. Permanent VCs, however, are feasible only when the number of entry/exit points is relatively small. For n entry points, $n(n - 1)$ permanent VCs are needed to directly connect n entry/exit points.

Each router interface that connects to the ATM network will need two addresses, in much the same way that an IP host has two addresses for an Ethernet interface: an IP address and a MAC address. Similarly, an ATM interface will have an IP address and an ATM address. Consider now an IP datagram crossing the ATM network shown in Figure 5.35. In the simplest case, the ATM network appears as a single logical link—ATM interconnects these four routers just as Ethernet can be used to connect four routers. Let us refer to the router at which the datagram enters the ATM network as the "entry router" and the router at which the datagram leaves the network as the "exit router." The entry router does the following:

1. Examines the destination address of the datagram.
2. Indexes its routing table and determines the IP address of the exit router (that is, the next router in the datagram's route).
3. To get the datagram to the exit router, the entry router views ATM as just another link-layer protocol. To move the datagram to the next router, we must determine the physical address of the next-hop router. Recall from our

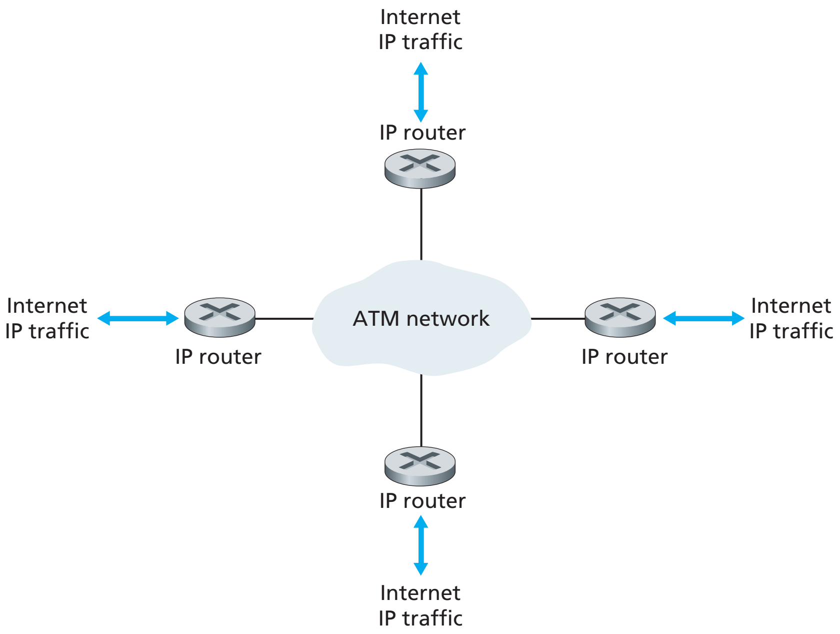


Figure 5.35 ♦ ATM network in the core of an Internet backbone

discussion in Section 5.4.2 that this is done using ARP. In the case of an ATM interface, the entry router indexes an ATM ARP table with the IP address of the exit router and determines the ATM address of the exit router. The ATMARP protocol is described in [RFC 2225].

4. IP in the entry router then passes the datagram along with the ATM address of the exit router down to the link layer (that is, ATM).

After these four steps have been completed, the job of moving the datagram to the exit router is out of the hands of IP and in the hands of ATM. ATM must now move the datagram to the ATM destination address obtained in Step 3 above. This task has two subtasks:

1. Determine the VCI for the VC that leads to the ATM destination address.
2. Segment the datagram into cells at the sending side of the VC (that is, at the entry router), and reassemble the cells into the original datagram at the receiving side of the VC (that is, at the exit router).

The first subtask is straightforward. The interface at the sending side maintains a table that maps ATM addresses to VCIs. Because we're assuming that the VCs are permanent, this table is static and up-to-date. (If the VCs were not permanent, then the ATM Q.2931 signaling protocol would be needed to establish and tear down the VCs dynamically.) The second task merits more careful consideration. One approach is to use IP fragmentation, as discussed in Section 4.4. With IP fragmentation, the sending router would first break the original datagram into fragments, with each fragment being no more than 48 bytes, so that the fragment could fit into the payload of the ATM cell. But this fragmentation approach has a big problem—each IP fragment typically has 20 bytes of header, so that an ATM cell carrying a fragment would have 25 bytes of "overhead" and only 28 bytes of useful information. ATM thus uses AAL5 to provide more efficient segmentation/reassembly of a datagram.

The ATM network then moves each cell across the network to the ATM destination address. At each ATM switch between the ATM source and the ATM destination, the ATM cell is processed by the ATM physical and ATM layers, but not by the AAL layer. At each switch the VCI is typically translated (see Section 4.2.1) and the HEC is recalculated. When the cells arrive at the ATM destination address, they are directed to an AAL buffer that has been allocated to the particular VC. The AAL5 PDU is then reconstructed and the IP datagram is extracted and passed up the protocol stack to the IP layer.

5.8.2 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) evolved from a number of industry efforts in the mid-to-late 1990s to improve the forwarding speed of IP routers by adopting a key concept from the world of virtual-circuit networks: a fixed-length label. The goal was not to abandon the destination-based IP datagram-forwarding infrastructure for one based on fixed-length labels and virtual circuits, but to augment it by selectively labeling datagrams and allowing routers to forward datagrams based on fixed-length labels (rather than destination IP addresses) when possible. Importantly, these techniques work hand-in-hand with IP, using IP addressing and routing. The IETF unified these efforts in the MPLS protocol [RFC 3031, RFC 3032], effectively blending VC techniques into a routed datagram network.

Let's begin our study of MPLS by considering the format of a link-layer frame that is handled by an MPLS-capable router. Figure 5.36 shows that a link-layer frame transmitted on a PPP link or LAN (such as Ethernet) has a small MPLS header added between the layer-2 (i.e., PPP or Ethernet) header and layer-3 (i.e., IP) header. RFC 3032 defines the format of the MPLS header for such links; headers are defined for ATM and frame-relayed networks as well in other RFCs. Among the fields in the MPLS header are the label (which serves the role of the virtual-circuit identifier that we encountered back in Section 4.2.1), 3 bits reserved for experimental use, a single S bit, which is used to indicate the end of a series of "stacked" MPLS headers (an advanced topic that we'll not cover here), and a time-to-live field.

It's immediately evident from Figure 5.36 that an MPLS-enhanced frame can only be sent between routers that are both MPLS capable (since a non-MPLS-capable

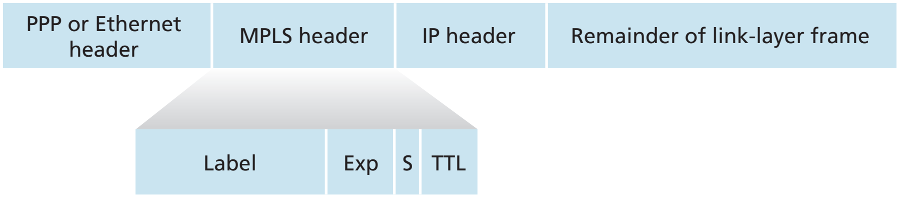
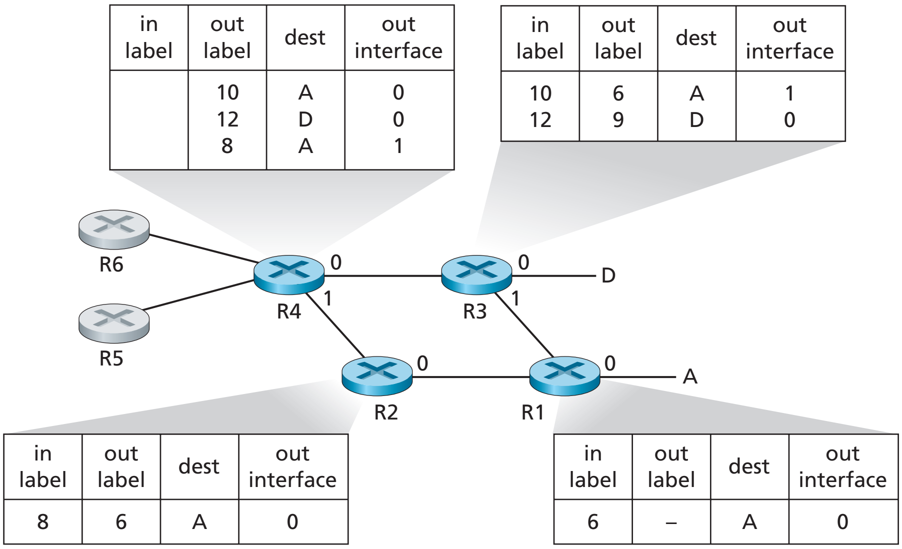


Figure 5.36 ♦ MPLS header: Located between link- and network-layer headers

router would be quite confused when it found an MPLS header where it had expected to find the IP header!). An MPLS-capable router is often referred to as a **label-switched router**, since it forwards an MPLS frame by looking up the MPLS label in its forwarding table and then immediately passing the datagram to the appropriate output interface. Thus, the MPLS-capable router need *not* extract the destination IP address and perform a lookup of the longest prefix match in the forwarding table. But how does a router know if its neighbor is indeed MPLS capable, and how does a router know what label to associate with the given IP destination? To answer these questions, we'll need to take a look at the interaction among a group of MPLS-capable routers.

In the example in Figure 5.37, routers R1 through R4 are MPLS capable. R5 and R6 are standard IP routers. R1 has advertised to R2 and R3 that it (R1) can route to destination A, and that a received frame with MPLS label 6 will be forwarded to destination A. Router R3 has advertised to router R4 that it can route to destinations A and D, and that incoming frames with MPLS labels 10 and 12, respectively, will be switched toward those destinations. Router R2 has also advertised to router R4 that it (R2) can reach destination A, and that a received frame with MPLS label 8 will be switched toward A. Note that router R4 is now in the interesting position of having *two* MPLS paths to reach A: via interface 0 with outbound MPLS label 10, and via interface 1 with an MPLS label of 8. The broad picture painted in Figure 5.37 is that IP devices R5, R6, A, and D are connected together via an MPLS infrastructure (MPLS-capable routers R1, R2, R3, and R4) in much the same way that a switched LAN or an ATM network can connect together IP devices. And like a switched LAN or ATM network, the MPLS-capable routers R1 through R4 do so *without ever touching the IP header of a packet*.

In our discussion above, we've not specified the specific protocol used to distribute labels among the MPLS-capable routers, as the details of this signaling are well beyond the scope of this book. We note, however, that the IETF working group on MPLS has specified in [RFC 3468] that an extension of the RSVP protocol (which we'll study in Chapter 7), known as RSVP-TE [RFC 3209], will be the focus of its efforts for MPLS signaling. Thus, the interested reader is encouraged to consult RFC 3209.



in label	out label	dest	out interface
	10	A	0
	12	D	0
	8	A	1

in label	out label	dest	out interface
10	6	A	1
12	9	D	0

in label	out label	dest	out interface
8	6	A	0

in label	out label	dest	out interface
6	–	A	0

Figure 5.37 ♦ MPLS-enhanced forwarding

Thus far, the emphasis of our discussion of MPLS has been on the fact that MPLS performs switching based on labels, without needing to consider the IP address of a packet. The true advantages of MPLS and the reason for current interest in MPLS, however, lie not in the potential increases in switching speeds, but rather in the new traffic management capabilities that MPLS enables. As noted above, R4 has *two* MPLS paths to A. If forwarding were performed up at the IP layer on the basis of IP address, the IP routing protocols we studied in Chapter 4 would specify only a single, least-cost path to A. Thus, MPLS provides the ability to forward packets along routes that would not be possible using standard IP routing protocols. This is one simple form of **traffic engineering** using MPLS [RFC 3346; RFC 3272; RFC 2702; Xiao 2000], in which a network operator can override normal IP routing and force some of the traffic headed toward a given destination along one path, and other traffic destined toward the same destination along another path (whether for policy, performance, or some other reason).

It is also possible to use MPLS for many other purposes as well. It can be used to perform fast restoration of MPLS forwarding paths, e.g., to reroute traffic over a precomputed failover path in response to link failure [Kar 2000; Huang 2002; RFC 3469]. MPLS can also be used to implement the differentiated service framework (“diff-serv”) that we will study in Chapter 7. Finally, we note that MPLS can, and has, been used to implement so-called **virtual private networks** (VPNs). In implementing a VPN for a customer, an ISP uses its MPLS-enabled network to connect together the customer's various networks. MPLS can be used to isolate both the resources and addressing used by the customer's VPN from that of other users crossing the ISP's network; see [DeClercq 2002] for details.

Our discussion of MPLS has been necessarily brief, and we encourage you to consult the references we've mentioned. We note that with so many possible uses for MPLS, it appears that it is rapidly becoming the Swiss Army knife of Internet traffic engineering!