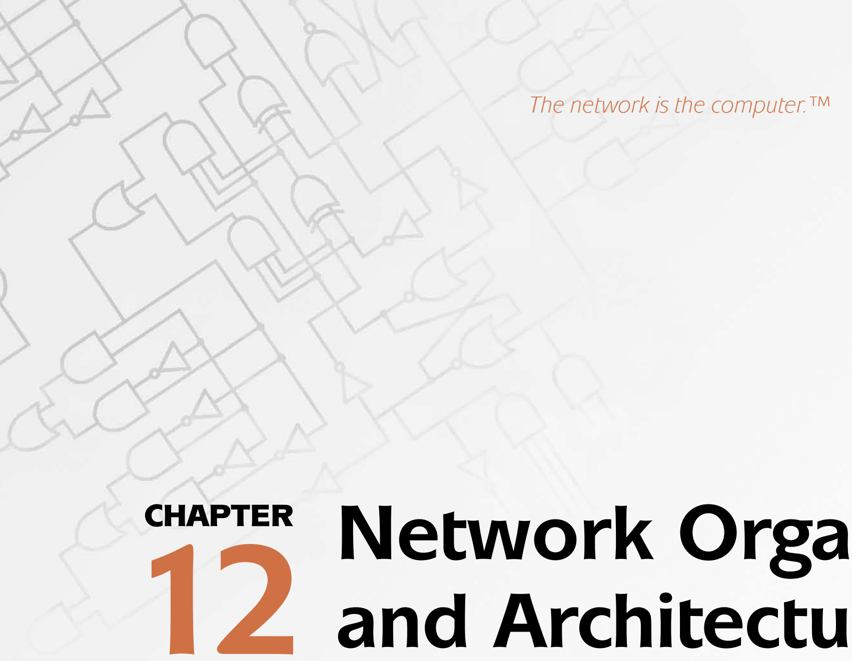


The network is the computer.™

—Sun Microsystems, Inc.

CHAPTER 12 Network Organization and Architecture

12.1 INTRODUCTION

Sun Microsystems launched a major advertising campaign in the 1980s with the catchy slogan that opens this chapter. A couple decades ago, its pitch was surely more sizzle than steak, but it was like a voice in the wilderness, heralding today's wired world with the Web at the heart of global commerce. Standalone business computers are now obsolete and irrelevant.

This chapter will introduce you to the vast and complex arena of data communications with a particular focus on the Internet. We will look at architectural models (network protocols) from a historical point of view, a theoretical point of view, and a practical point of view. Once you have an understanding of how a network operates, you will learn about many of the components that constitute network organization. Our intention is to give you a broad view of the technologies and terminology that every computer professional will encounter at some time during his or her career. To understand the computer is to also understand the network.

12.2 EARLY BUSINESS COMPUTER NETWORKS

Today's computer networks evolved along two different paths. One path was directed toward enabling fast and accurate business transactions, whereas the other was aimed at facilitating collaboration and knowledge sharing in the academic and scientific communities.

Digital networks of all varieties aspire to share computer resources in the simplest, fastest, and most cost-effective manner possible. The more costly the computer, the stronger the motivation to share it among as many users as possible. In the 1950s, when most computers cost millions of dollars, only the wealthiest companies could afford more than one system. Of course, employees in remote locations had as much need for computer resources as their central office counterparts, so some method of getting them connected had to be devised. And virtually every vendor had a different connectivity solution. The most dominant of these vendors was IBM with its **Systems Network Architecture (SNA)**. This communications architecture, with modifications, has persisted for more than three decades.

IBM's SNA is a specification for end-to-end communication between physical devices (called **physical units**, or **PUs**) over which logical sessions (known as **logical units**, or **LUs**) take place. In the original architecture, the physical components of this system consisted of terminals, printers, communications controllers, multiplexers, and front-end processors. Front-end processors sat between the host (mainframe) system and the communications lines. They managed all of the communications overhead including polling each of the communications controllers, which in turn polled each of their attached terminals. This architecture is shown in Figure 12.1.

IBM's SNA was geared toward high-speed transaction entry and customer service inquiries. Even at the modest line speed of 9,600bps (bits per second), access to data on the host was nearly instantaneous when all network components were functioning properly under normal loads. The speed of this architecture, however, came at the expense of flexibility and interoperability. The human overhead in managing and supporting these networks was enormous, and connections to other vendors' equipment and networks were often laudable feats of software and hardware engineering. Over the past 30 years, SNA has adapted to changing business needs and networking environments, but the underlying concepts are essentially what they were decades ago. In fact, this architecture was so well designed that aspects of it formed the foundation for the definitive international communications architecture, OSI, which we discuss in Section 12.4. Although SNA contributed much to the young science of data communications, the technology has just about run its course. In most installations, it has been replaced by "open" Internet protocols.

12.3 EARLY ACADEMIC AND SCIENTIFIC NETWORKS: THE ROOTS AND ARCHITECTURE OF THE INTERNET

Amid the angst of the Cold War, American scientists at far-flung research institutions toiled under government contracts, seeking to preserve the military ascendancy of the United States. At a time when the country had fallen behind in the technology race, the U.S. government created an organization called the **Advanced Research Projects Agency (ARPA)**. The sophisticated computers this organization needed to carry out its work, however, were scarce and extremely costly—even by Pentagon standards. Before long, it

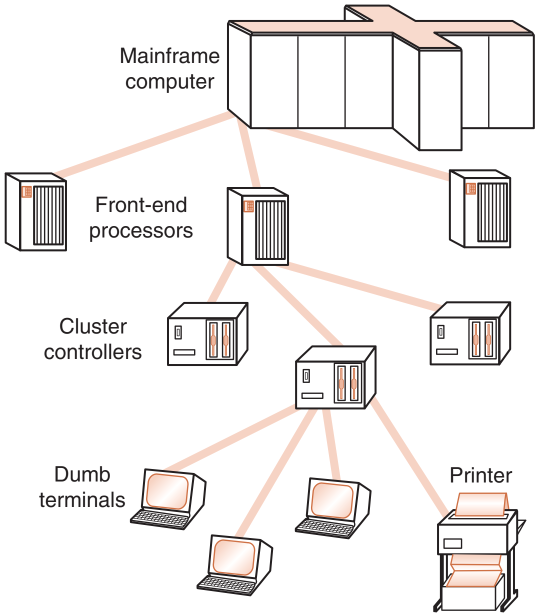


FIGURE 12.1 A Hierarchical, Polled Network

occurred to someone that by establishing communication links into the few supercomputers that were scattered all over the United States, computational resources could be shared by innumerable like-minded researchers. Moreover, this network would be designed with sufficient redundancy to provide for continuous communication, even if thermonuclear war knocked out a large number of nodes or communication lines. To this end, in December 1968, a Cambridge, Massachusetts, consulting firm called BBN (Bolt, Beranek, and Newman, now Genuity Corporation) was awarded the contract to construct such a network. In December 1969, four nodes, the University of Utah, the University of California at Los Angeles, the University of California at Santa Barbara, and the Stanford Research Institute, went online. ARPAnet gradually expanded to include more government and research institutions. When President Reagan changed the name of ARPA to the **Defense Advanced Research Projects Network (DARPA)**, ARPAnet became **DARPAnet**. Through the early 1980s, nodes were added at a rate of a little more than one per month. However, military researchers eventually abandoned DARPAnet in favor of more secure channels.

In 1985, the National Science Foundation (NSF) established its own network, **NSFnet**, to support its scientific and academic research. NSFnet and DARPAnet served a similar purpose and a similar user community, but the capabilities of NSFnet outstripped those of DARPAnet. Consequently, when the military abandoned DARPAnet, NSFnet absorbed it and became what we now know as the Internet. By the early 1990s, the NSF had outgrown NSFnet, so it began building a faster, more reliable NSFnet. Administration of the public Internet then fell to private national and regional corporations, such as Sprint, MCI, and PacBell, to name a few. These companies bought the NSFnet trunk lines, called **backbones**, and made money by selling backbone capacity to various **Internet service providers (ISPs)**.

The original DARPAnet (and now the Internet) would have survived thermonuclear war because, unlike all other networks in existence in the 1970s, it had no dedicated connections between systems. Information was instead routed along whatever pathways were available. Parts of the data stream belonging to a single dialogue could take different routes to their destinations. The key to this robustness is the idea of **datagram** message packets, which carry data in chunks instead of the streams used by the SNA model. Each datagram contains addressing information so that every datagram can be routed as a single, discrete unit.

A second revolutionary aspect of DARPAnet was that it created a uniform protocol for communications between dissimilar hosts along networks of differing speeds. Because it connected many different kinds of networks, DARPAnet was said to be an **internetwork**. As originally specified, each host computer connected to DARPAnet by way of an **Interface Message Processor (IMP)**. IMPs took care of protocol translation from the language of DARPAnet to the communications language native to the host system, so any communications protocol could be used between the IMP and the host. Today, routers (discussed in Section 12.6.7) have replaced IMPs, and the communications protocols are less heterogeneous than they were in the 1970s. However, the underlying principles have remained the same, and the generic concept of internetworking has become practically synonymous with the Internet. A modern internetwork configuration is shown in Figure 12.2. The diagram shows how four routers form the heart of the network. They connect many different types of equipment, making decisions on their own as to how datagrams should get to their destinations in the most efficient way possible.

The Internet is much more than a set of good data communication specifications. It is, perhaps, a philosophy. The foremost principle of this philosophy is the idea of a free and open world of information sharing, with the destiny of this world being shaped collaboratively by the people and ideas in it. The epitome of this openness is the manner in which Internet standards are created. Internet standards are formulated through a democratic process that takes place under the auspices of the **Internet Architecture Board (IAB)**, which itself operates under the oversight of the not-for-profit **Internet Society (ISOC)**. The **Internet Engineering Task Force (IETF)**, operating within the IAB, is a loose alliance of industry experts that develops detailed specifications for Internet protocols. The IETF publishes all proposed standards in the form of **Requests for Comment (RFCs)**, which are open to anyone's scrutiny and comment. The two most important RFCs—RFC 791 (Internet Protocol Version 4) and RFC 793 (Transmission Control Protocol)—form the foundation of today's global Internet.

The organization of all the ISOC's committees under more committees could have resulted in a tangle of bureaucracy producing inscrutable and convoluted specifications. But owing to the openness of the entire process, as well as the talents of the reviewers, RFCs are among the clearest and most readable documents in the entire body of networking literature. It is little wonder that

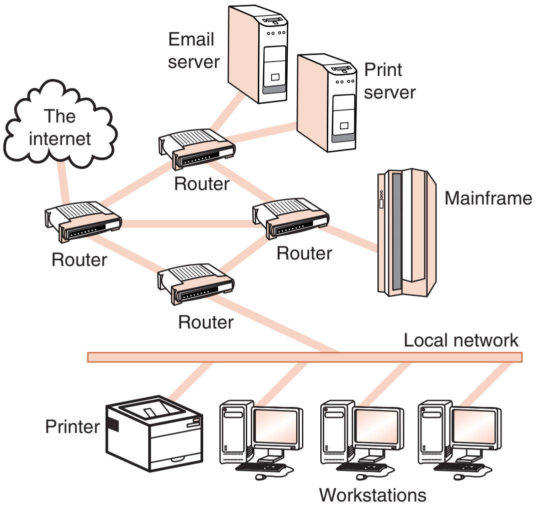


FIGURE 12.2 An Example of an Internetwork

manufacturers were so quick to adopt Internet protocols. Internet protocols are now running on all sizes of networks, both publicly and privately owned. Formerly, networking standards were handed down by a centralized committee or through an equipment vendor. One such approach resulted in the ISO/OSI protocol model, which we discuss next.

12.4 NETWORK PROTOCOLS I: ISO/OSI PROTOCOL UNIFICATION

In Chapter 13, we show how various data storage interfaces use protocol stacks. The SCSI-3 Architecture Model is one of these. In general, protocol stacks make all kinds of interfaces portable, maintainable, and easy to describe. The most important and comprehensive of these is the ISO/OSI (International Organization for Standardization/Open Systems Interconnect) protocol stack, which is the theoretical model for many storage and data communication interfaces and protocols. Although each protocol differs in implementation details, the general idea is the same: Each layer of the protocol interfaces only with layers adjacent to itself. No layer skipping is allowed. Protocol conversations take place between identical protocol layers running on two different machines. The exact manner in which this communication takes place is clearly defined in the international standards.

By the late 1970s, nearly every computer manufacturer had devised its own proprietary communication protocols. The details of these protocols were sometimes held secret by their inventors as a way to ensure a lock on the markets where their products were sold. Equipment built by Vendor A could not communicate with equipment built by Vendor B unless protocol conversion kits (black boxes) were placed between the two systems. Even then, the black boxes might not perform as expected, usually because a vendor had changed some protocol parameter after the box was built.

Two of the world's premiere standards-making bodies realized that this Tower of Babel was becoming increasingly costly, ultimately working against the advancement of information sharing. In the late 1970s and early 1980s, both the International Organization for Standardization (ISO) and the International Consultative Committee on Telephony and Telegraphy (CCITT) were independently attempting to construct an international standard telecommunications architecture. In 1984, these two entities came together to produce a unified model, now known as the **ISO Open Systems Interconnect Reference Model (ISO/OSI RM)**. (*Open systems*, in this context, means that system connectivity would not be proprietary to any single vendor.) The ISO's work is called a reference model because virtually no commercial system uses all of the features precisely as specified in the model. However, the ISO/OSI model does help us to understand how real protocols and network components fit together in the context of a standard model.

The OSI RM contains seven protocol layers, starting with physical media interconnections at Layer 1, through applications at Layer 7. We must emphasize that the OSI model defines only the functions of each of the seven layers and the interfaces between them. Implementation details are not part of the model. Many different standardization bodies, including the IEEE, the European Computer Manufacturers Association (ECMA), the International Telecommunication Union-Telecommunication Standardization Sector (ITU-T), and the ISO itself (external to the ISO model), have provided such detail. Implementations at the highest layers can be completely user defined.

12.4.1 A Parable

Before we embark on the technicalities of the OSI RM, let us offer a parable to help illustrate how layered protocols work. Suppose you have lost a bet with your sister and the price is to spend the day with your nephew, Billy. Billy is a notorious brat who throws tantrums when he doesn't get his own way. Today he has decided that he wants a roast beef sandwich from Dumpy's Deli down the street. This roast beef sandwich must be dressed with mustard and pickles. Nothing more, nothing less.

Upon entering the deli, you seat Billy and take a number from the dispenser near the counter. There is a cashier taking orders and a second person assembling orders in a food preparation area. The deli is packed with hungry workers on their lunch hours. You remark to yourself that the service seems unusually slow this day. Billy starts to announce loudly that he is hungry, while he thumps his little fists on the table.

Despite how badly you want Billy's sandwich, by waiting for your number to be called, you are obeying a protocol. You know that pushing yourself ahead of the others will get you nowhere. In fact, if you defy the protocol, you could be ejected from the deli, making matters worse.

When you finally get your turn at the counter (Billy by now is yelling quite loudly), you give the cashier your order, adding a tuna sandwich and chips for yourself. The cashier fetches your drinks and tells the food handler to prepare a tuna sandwich and a roast beef sandwich with mustard and pickles. Although the cook could hear every word of Billy's luncheon desires above the din of the crowd, she waited until the cashier told her what to prepare.

Billy, therefore, could not skip over the cashier layer of the deli protocol regardless of how loudly he yelled. Before assembling the sandwiches, the food handler had to know that the order was legitimate and that a customer was willing to pay for it. She could know this only by being told by the cashier.

Once the sandwiches have been prepared, the cook wraps them individually in deli paper, marking the paper to indicate the contents of each. The cashier fetches the sandwiches, placing them both in a brown bag, along with your chips and two cans of cola. She announces that your bill is $6.25, for which you hand her a $10.00 bill. She gives you $4.75 in change. Because she has given you the wrong change, you stand at the counter until you are given the correct change, then you proceed to your table.

Upon unwrapping the sandwich, Billy discovers that his roast beef sandwich is in fact a corned beef sandwich, triggering yet another round of whining. You have no choice but to take another number and wait in line until it is called.

Your refusal to leave the counter when given the wrong change is analogous to error-checking that takes place between the layers of a protocol. The transmission does not proceed until the receiving layer is satisfied with what it has received from the sending layer. Of course, you didn't feel comfortable unwrapping Billy's sandwich while standing at the counter (that would, after all, be icky).

The sandwiches, along with their wrapping, correspond to OSI **Protocol Data Units (PDUs)**. Once data has been encapsulated by an upper-layer protocol, a lower-layer protocol will not examine its contents. Neither you nor the cashier unwrapped the sandwiches. The cashier created yet another PDU when she placed your order in the bag. Because your order was in a bag, you could easily carry it to your table. Juggling two colas, two sandwiches, and a bag of chips through a crowded deli may indeed have had disastrous consequences.

At Dumpy's Deli, you know that if you want something to eat, you can't go directly to the cook, nor can you go to another customer, nor to a maintenance person. For lunch service, you can go only to the cashier after taking a number. The number you pull from the dispenser is analogous to an OSI **Service Access Point (SAP)**. The cashier grants you permission to place your order only when you present the ticket that proves you are the next in line.

12.4.2 The OSI Reference Model

The OSI Reference Model is shown in Figure 12.3. As you can see, the protocol consists of seven layers. Adjacent layers interface with each other through SAPs, and as the protocol data units (PDUs) pass through the stack, each protocol layer adds or removes its own header. Headers are added on the sending end, and they are removed by the receiving end. The contents of the PDUs create a conversation between peer layers on each side of the dialogue. This conversation is their protocol.

Having employed a story to explain the ideas of PDUs and SAPs, we supply another metaphor to help us further explain the OSI Reference Model. In this metaphor, suppose that you are operating your own business. This business,

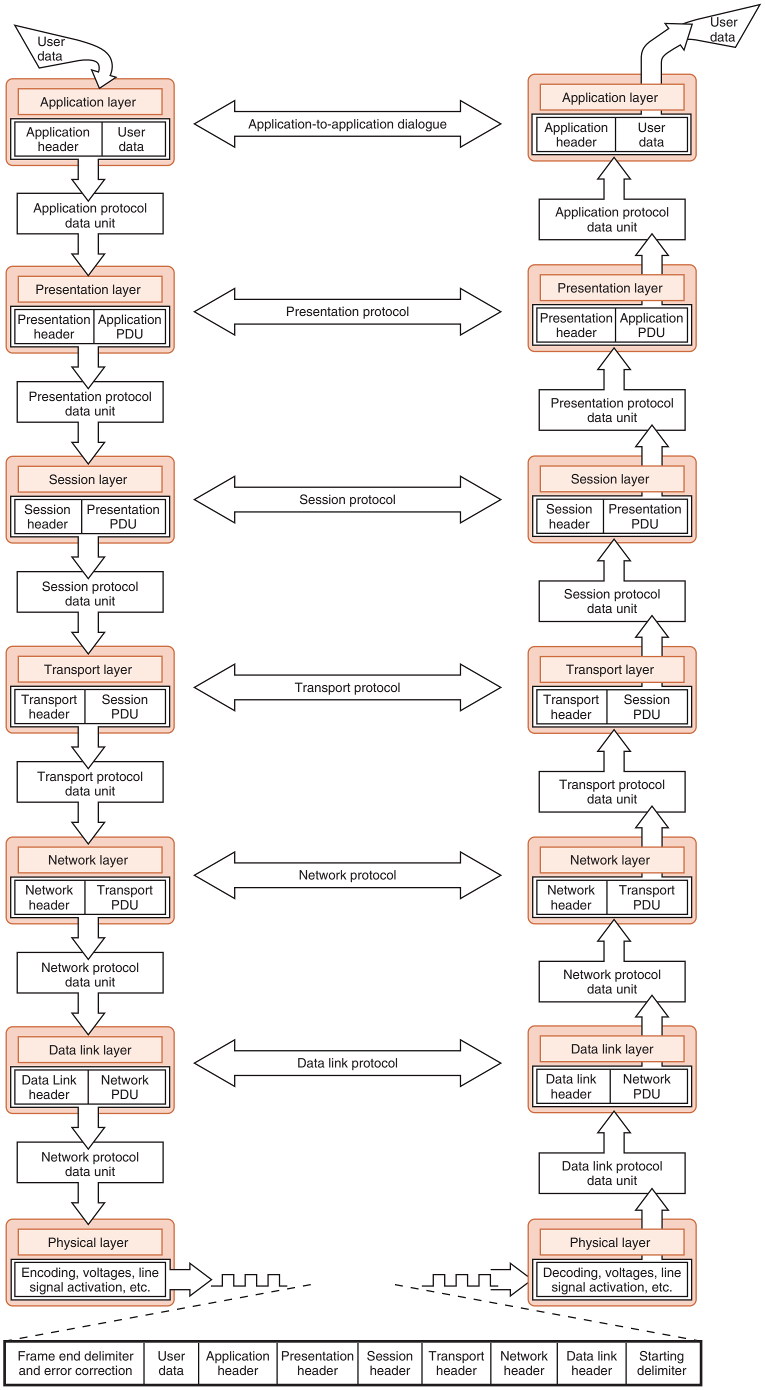


FIGURE 12.3 The OSI Reference Model—Interfaces Operate Vertically; Protocols Operate Horizontally

called Super Soups and Tasty Teas, manufactures gourmet soup and tea that is sold all over North America to people of discriminating tastes. In order to get your luscious wares where they are going, you use a private shipping company, Ginger, Lee, and Pronto, also known as GL&P. The system of manufacturing your wares and their ultimate consumption by your epicurean customers spans a series of processes analogous to those carried out by the layers of the OSI Reference Model, as we shall see in the following sections.

The OSI Physical Layer

Many different kinds of media are capable of carrying bits between a communication source (the initiator) and their destination (the responder). Neither the initiator nor the responder need have any concern as to whether their conversation takes place over copper wire, satellite links, or optical cable. The Physical layer of the OSI model assumes the job of carrying a signal from here to there. It receives a stream of bits from the Data Link layer above it, encodes those bits, and places them on the communications medium in accordance with agreed-on protocols and signaling standards.

The function of the OSI Physical layer can be compared to that of the vehicles that the GL&P shipping company uses to move products from your factory to your customer. After giving your parcel to the delivery company, you usually don't care whether the parcel is carried to its destination on a train, a truck, an airplane, or a ferryboat, as long as it arrives at its destination intact and within a reasonable amount of time. The handlers along the way have no concern as to the contents of the parcel, only as to the address on the box (sometimes even ignoring the word *fragile!*). Similar to how a freight company moves boxes, the OSI Physical layer moves transmission frames, which are sometimes called **physical Protocol Data Units**, or **physical PDUs**. Each physical PDU carries an address and has delimiter signal patterns that surround the **payload**, or contents, of the PDU.

The OSI Data Link Layer

When you send your package, the actions of placing articles in a suitable shipping container and addressing the package are comparable to the function of the OSI Data Link layer. The Data Link layer organizes message bytes into frames of suitable size for transmission along the physical medium. If you were shipping 50kg of soup and tea, and GL&P has a rule that no package can weigh more than 40kg, you would need at least two separate boxes to ship your articles. The Data Link layer does the same thing. It negotiates frame sizes and the speed at which they are sent with the Data Link layer at the other end.

The timing of frame transmission is called **flow control**. If frames are sent too fast, the receiver's buffer could overflow, causing frames to be lost. If the frames are not sent quickly enough, the receiver could time out and drop the connection. In both of these cases, the Data Link layer senses the problem when the receiver does not acknowledge the packets within a specified time interval. Lacking this acknowledgment, the sender retransmits the packet.

The OSI Network Layer

Suppose you could tell GL&P, "Send this package through Newark, New Jersey, because the terminal in New York City is always too crowded to get my package through on time." Stated another way, if you could tell the freight carrier how to route your package, you would be performing the same function as the Network layer of the OSI model. The package handlers at a Philadelphia terminal, however, would also be performing a Network layer function if they decide to route the package through Newark after they've learned of an unusual problem in New York. This kind of localized decision making is critical to the operation of every large internetwork. Because of the complexity of most networks, it is impossible for every end node computer to keep track of every possible route to every destination, so the functions of the Network layer are spread throughout the entire system.

At the originating computers, the Network layer doesn't do much except add addressing information to the PDUs from the Transport layer, and then pass them on to the Data Link layer. It does its most important and complex tasks while moving PDUs across the **intermediate nodes**—those nodes that act like freight terminals in the network. The Network layer not only establishes the route, but also ensures that the size of its PDUs is compatible with all of the equipment between the source and the destination.

The OSI Transport Layer

Let's say that the destination of your parcels, filled with canned soup and packaged teas, is a food distribution warehouse in Quebec. Upon its arrival, a shipping clerk opens the package to make sure that the goods were not damaged in transit. She opens each box, looking for dented cans and ripped tea boxes. She doesn't care whether the soup is too salty or the tea is too tart, only that the products have arrived intact and undamaged. Once the goods pass her inspection, the clerk signs a receipt that GL&P returns to you, letting you know that your products got to their destination.

Similarly, the OSI Transport layer provides quality assurance functions for the layers above it in the protocol stack. It contributes yet another level of end-to-end acknowledgment and error correction through its handshaking with the Transport layer at the other end of the connection. The Transport layer is the lowest layer of the OSI model at which there is any awareness of the network or its protocols. Once the Transport layer peels its protocol information away from the Session PDU, the Session layer can safely assume that there are no network-induced errors in the PDU.

The OSI Session Layer

The Session layer arbitrates the dialogue between two communicating nodes, opening and closing that dialogue as necessary. It controls the direction and mode, which is either **half-duplex** (in one direction at a time) or **full-duplex** (in both directions at once). If the mode is half-duplex, the Session layer determines which node has control of the line. It also supplies recovery checkpoints during file transfers. **Checkpoints** are issued each time a packet, or block of data, is acknowledged as received in good condition. If an error occurs during a large file transfer, the Session layer retransmits all data from the last checkpoint. Without checkpoint processing, the entire file would need to be retransmitted.

If the shipping clerk notices that one of your boxes was smashed in transit, she would notify you (as well as GL&P) that the goods did not arrive intact and that you must send another shipment. If the damaged parcel was a 10kg box of a 50kg shipment, you would replace only the contents of the 10kg box; the other 40kg of merchandise could be sent on its way to the consumer.

The OSI Presentation Layer

What if the consumers of your soup and tea reside in the Quebec market area and the labels on your soup cans are in English only? If you expect to sell your soup to people who speak French, you would certainly want your soup cans to have bilingual labels. If employees of the Quebec food distribution warehouse do this for you, they are doing what the Presentation layer does in the OSI model.

The Presentation layer provides high-level data interpretation services for the Application layer above it. For example, suppose one network node is an IBM zSeries Server that stores and transmits data in EBCDIC. This mainframe server needs to send some data to an ASCII-based microcomputer that has just requested it. The Presentation layers residing on the respective systems decide which of the two will perform the EBCDIC-to-ASCII translation. Either side could do it with equal effectiveness. What is important to remember is that the mainframe is sending EBCDIC to its Application layer and the Application layer on the client is receiving ASCII from the Presentation layer beneath it in the protocol stack. Presentation layer services are also called into play if we use encryption or certain types of data compression during the communication session.

The OSI Application Layer

The Application layer supplies meaningful information and services to users at one end of the communication and interfaces with system resources (programs and data files) at the other end of the communication. Application layers provide a suite of programs that can be invoked as the user sees fit. If none of the applications routinely supplied by the Application layer can do the job, we are free to write our own. With regard to communications, the only thing that these applications need to do is to send messages to the Presentation layer Service Access Points, and the lower layers take care of the hard part.

To enjoy a savory serving of Super Soups, all that the French Canadian soup connoisseur needs to do is open the can, heat, and enjoy. Because GL&P and the regional food distributor have all done their work, your soup is as tasty on the Canadian's table as it was coming out of your kitchen. (*Magnifique!*)

12.5 NETWORK PROTOCOLS II: TCP/IP NETWORK ARCHITECTURE

While the ISO and the CCITT were haggling over the finer points of the perfect protocol stack, TCP/IP was rapidly spreading across the globe. By the sheer weight of its popularity in the academic and scientific communications communities, TCP/IP quietly became the de facto global data communication standard.

Although it didn't start out that way, TCP/IP is now a lean and effective protocol stack. It has three layers that can be mapped to five of the seven layers in the OSI model. These layers are shown in Figure 12.4. Because the IP layer is loosely coupled with OSI's Data Link and Physical layers, TCP/IP can be used with any type of network, even different types of networks within a single session. The singular requirement is that all of the participating networks must be running—at minimum—Version 4 of the Internet Protocol (**IPv4**).

There are two versions of the Internet Protocol in use today: Version 4 and Version 6. IPv6 addresses many of the limitations of IPv4. Despite the many advantages of IPv6, the huge installed base of IPv4 ensures that it will be supported for many years to come. Some of the major differences between IPv4 and IPv6 are outlined in Section 12.5.5. But first, we take a detailed look at IPv4.

12.5.1 The IP Layer for Version 4

The IP layer of the TCP/IP protocol stack provides essentially the same services as the Network and Data Link layers of the OSI Reference Model: It divides TCP packets into protocol data units called datagrams, and then attaches the routing information required to get the datagrams to their destinations. The concept of the datagram was fundamental to the robustness of ARPAnet, and now the Internet. Datagrams can take any route available to them without intervention by a human network manager. Take, for example, the network shown in Figure 12.5. If intermediate node X becomes congested or fails, intermediate node Y can route datagrams through node Z until X is back up to full speed. Routers are the Internet's most critical components, and researchers are continually seeking ways to improve their effectiveness and performance. We look at routers in detail in Section 12.6.7.

The bytes that constitute any of the TCP/IP protocol data units are called **octets**. This is because at the time that the ARPAnet protocols were being designed, the word *byte* was thought to be a proprietary term for the 8-bit groups

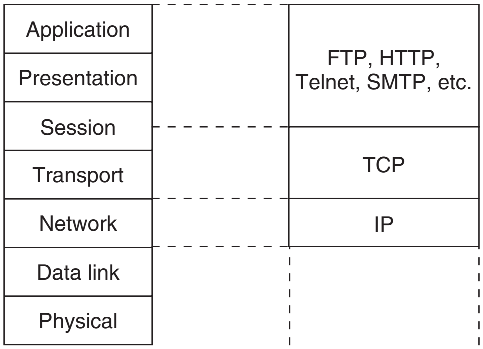


FIGURE 12.4 The TCP/IP Protocol Stack Versus the OSI Protocol Stack

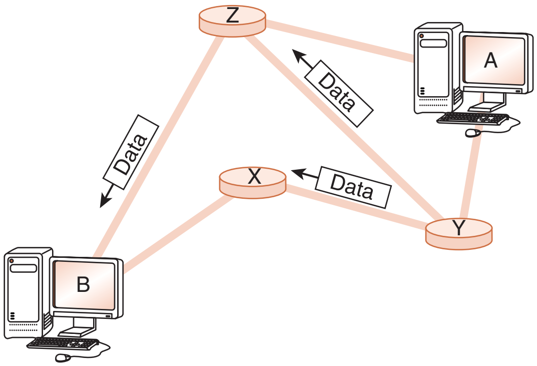


FIGURE 12.5 Datagram Routing in IP

used by IBM mainframes. Most TCP/IP literature uses the word *octet*, but we use *byte* for the sake of clarity.

12.5.2 The Trouble with IP Version 4

The number of bytes allocated for each field in the IP header reflects the technological era in which IP was designed. Back in the ARPAnet years, no one could have imagined how the network would grow, or even that there would ever be a civilian use for it.

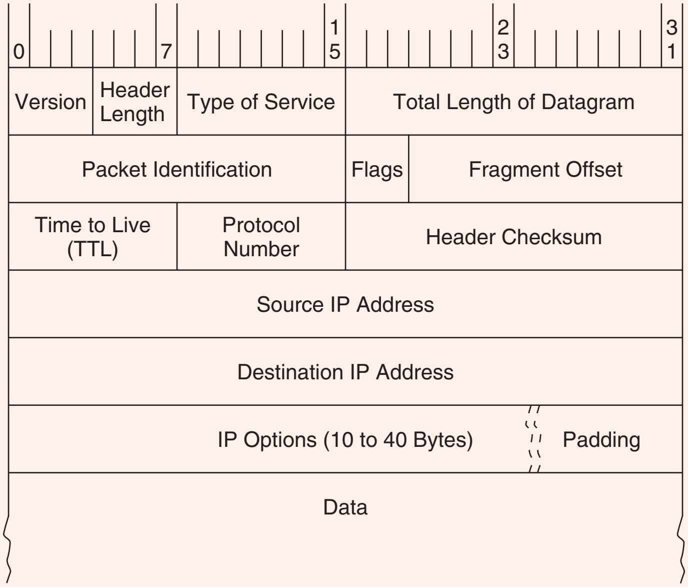
With the slowest networks of today being faster than the fastest networks of the 1960s, IP's packet length limit of 65,536 bytes has become a problem. The packets simply move too fast for certain network equipment to be sure that the packet hasn't been damaged between intermediate nodes. (At gigabit speeds, a 65,535-byte IP datagram passes over a given point in less than 1ms.)

By far the most serious problem with IPv4 headers concerns addressing. Every host and router must have an address that is unique over the entire Internet. To ensure that no Internet node duplicates the address of another Internet node, host IDs are administered by a central authority, the **Internet Corporation for Assigned Names and Numbers (ICANN)**. ICANN keeps track of groups of IP addresses, which are subsequently allocated or assigned by regional authorities. (The ICANN also coordinates the assignment of parameter values used in protocols so that everyone knows which values evoke which behaviors over the Internet.)

As you can see by looking at the IP header shown in the sidebar, there are 2^{32} or about 4.3 billion host IDs. It would be reasonable to think that there would be plenty of addresses to go around, but this is not the case. The problem lies in the fact that these addresses are not like serial numbers sequentially assigned to the next person who asks for one. It's much more complicated than that.

IP allows for three types, or **classes**, of networks, designated A, B, and C. They are distinguished from each other by the number of nodes (called **hosts**) that each can directly support. Class A networks can support the largest number of hosts; Class C, the least.

THE IP VERSION 4 DATAGRAM HEADER



Each IPv4 datagram must contain at least 40 bytes, which include a 24-byte header as shown above. The horizontal rows represent 32-bit words. Upon inspection of the figure, you can see, for example, that the Type of Service field occupies bits 8 through 15, whereas the Packet Identification field occupies bits 32 through 47 of the header. The Padding field shown as the last field of the header ensures that the data that follows the header starts on an even 32-bit boundary. The Padding always contains zeroes. The other fields in the IPv4 header are:

- Version—Specifies the IP protocol version being used. The version number tells all the hardware along the way the length of the datagram and what content to expect in its header fields. For IPv4, this field is always 0100 (because $0100_2 = 4_{10}$).
- Header Length—Gives the length of the header in 32-bit words. The size of the IP header is variable, depending on the value of the IP Options fields, but the minimum value for a correct header is 5.
- Type of Service—Controls the priority that the datagram is given by intermediate nodes. Values can range from "routine" (000) to "critical" (101). Network control datagrams are indicated with 110 and 111.

- Total Length—Gives the length of the entire IP datagram in bytes. As you can see by the layout above, 2 bytes are reserved for this purpose. Hence, the largest allowable IP datagram is $2^{16} - 1$, or 65,535.
- Packet ID—Each datagram is assigned a serial number as it is placed on the network. The combination of Host ID and Packet ID uniquely identifies each IP datagram in existence at any time in the world.
- Flags—Specifies whether the datagram may be fragmented (broken into smaller datagrams) by intermediate nodes. IP networks must be able to handle datagrams of at least 576 bytes. Most IP networks can deal with packets that are about 8KB long. With the "Don't Fragment" bit set, an 8KB datagram will not be routed over a network that says it can handle only 2KB packets, for example.
- Fragment Offset—Indicates the location of a fragment within a certain datagram. That is, it tells which part of the datagram the fragment came from.
- Time to Live (TTL)—TTL was originally intended to measure the number of seconds for which the datagram would remain valid. Should a datagram get caught in a routing loop, the TTL would (theoretically) expire before the datagram could contribute to a congestion problem. In practice, the TTL field is decremented each time it passes through an intermediate network node, so this field does not really measure the number of seconds that a packet lives, but the number of hops it is allowed before it reaches its destination.
- Protocol Number—Indicates which higher-layer protocol is sending the data that follows the header. Some of the important values for this field are:

  0 = Reserved
  1 = Internet Control Message Protocol (ICMP)
  6 = Transmission Control Protocol (TCP)
  17 = User Datagram Protocol (UDP)
  TCP is described in Section 12.5.3.

- Header Checksum—This field is calculated by first calculating the one's complement sum of all 16-bit words in the header, and then taking the one's complement of this sum, with the checksum field itself originally set to all zeroes. The one's complement sum is the arithmetic sum of two of the words with the (seventeenth) carry bit added to the lowest bit position of the sum. (See Section 2.4.2.) For example, 11110011 + 10011010 = 110001101 = 10001110 using one's complement arithmetic. What this means is that if we have an IP datagram of the form shown to the right, each w_i is a 16-bit word in the IP datagram. The complete checksum would be computed over two 16-bit words at a time: $w_1 + w_2 = S_1; S_1 + w_3 = S_2; \ldots S_k + S_{k-2} = S_{k+1}$.

w_1	w_2
w_3	w_4
. . .	. . .
. . .	. . .
w_{n-1}	w_n

- Source and Destination Addresses—Tell where the datagram is going. We have much more to say about these 32-bit fields in Section 12.5.2.
- IP Options—Provides diagnostic information and routing controls. IP Options are, well, optional.

The first three bits of an IP address indicate the network class. Addresses for Class A networks always begin with 0, Class B with 10, and Class C with 110. The remaining bits in the address are devoted to the network number and the host ID within that network number, as shown in Figure 12.6.

IP addresses are 32-bit numbers expressed in dotted decimal notation, for example, 18.7.21.69 or 146.186.157.6. Each of these decimal numbers represents 8 bits of binary information and can therefore have a decimal value between 0 and 255. Note that 127.x.x.x is a Class A network that is reserved for **loopback testing**, which checks the TCP/IP protocol processes running on the host. During the loopback test, no datagrams enter the network. The 0.0.0.0 network is typically reserved for use as the default route in the network.

Allowing for the reserved networks 0 and 127, only 126 Class A networks can be defined using a 7-bit network field. Class A networks are the largest networks of all, each able to support about 16.7 million nodes. Although it is unlikely that a Class A network would need all 16 million possible addresses, the Class A addresses, 1.0.0.0 through 126.255.255.255, were long ago assigned to early Internet adopters such as MIT and the Xerox Corporation. Furthermore, all of the 16,382 Class B network IDs (128.0.0.0 to 191.255.255.255) have also been assigned. Each Class B network can contain 65,534 unique node addresses. Because very few organizations need more than 100,000 addresses, their next choice is to identify themselves as Class C network owners, giving them only 256 addresses within the Class C space of 192.0.0.0 through 233.255.255.255. This is far fewer than would meet the needs of even a moderately sized company or institution. Thus, many networks have been unable to obtain a contiguous block of IP addresses so that each node on the network can have its own address on the

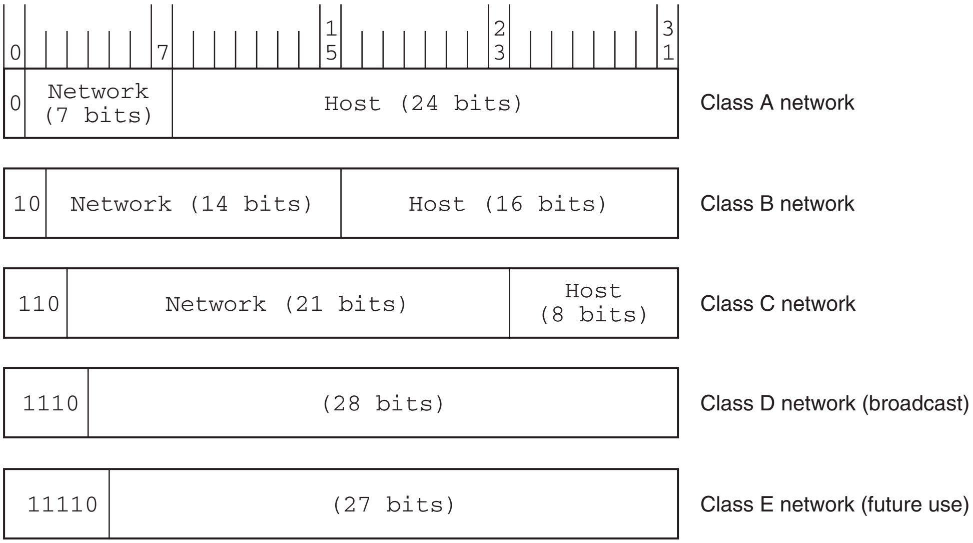


FIGURE 12.6 IP Address Classes

Internet. A number of clever workarounds have been devised to deal with this problem, but the ultimate solution lies in reworking the entire IP address structure. (Classes D and E do exist, but they aren't networks at all. Instead, they're groups of reserved addresses. The Class D addresses, 224 through 240, are used for multicasting by groups of hosts that share a common characteristic. The Class E addresses, 241 through 248, are reserved for future use.)

In addition to the eventual depletion of address space, there are other problems with IPv4. Its original designers did not anticipate the growth of the Internet and the routing problems that would result from the address class scheme. There are typically 70,000-plus routes in the routing table of an Internet backbone router. The current routing infrastructure of IPv4 needs to be modified to reduce the number of routes that routers must store. As with cache memory, larger router memories result in slower routing information retrieval. There is also a definite need for security at the IP level. A protocol called **IPSec (Internet Protocol Security)** is currently defined for the IP level. However, it is optional and hasn't been standardized or universally adopted.

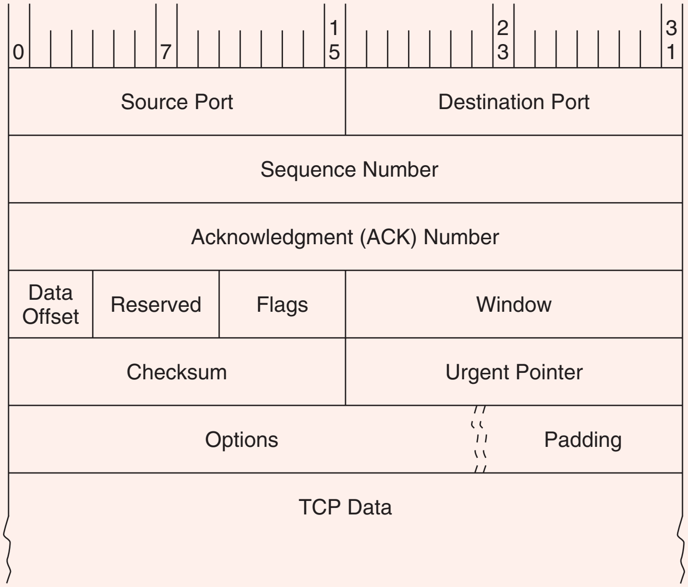
12.5.3 Transmission Control Protocol

The sole purpose of IP is to correctly route datagrams across the network. You can think of IP as a courier who delivers packages with no concern as to their contents or the order in which they are delivered. Transmission Control Protocol (TCP) is the consumer of IP services, and it does indeed care about these things and many others.

The protocol connection between two TCP processes is much more sophisticated than the one at the IP layer. Where IP simply accepts or rejects datagrams based only on header information, TCP opens a conversation, called a **connection**, with a TCP process running on a remote system. A TCP connection is very much analogous to a telephone conversation, with its own protocol "etiquette." As part of initiating this conversation, TCP also opens a service access point (SAP) in the application running above it. In TCP, this SAP is a numerical value called a **port**. The combination of the port number, the host ID, and the protocol designation becomes a **socket**, which is logically equivalent to a file name (or **handle**) to the application running above TCP. Instead of accessing data by using its disk file name, the application using TCP reads data through the socket. Port numbers 0 through 1,023 are called "well-known" port numbers because they are reserved for particular TCP applications. For example, the TCP/IP File Transfer Protocol (FTP) application uses ports 20 and 21. The Telnet terminal protocol uses port 23. Port numbers 1,024 through 65,535 are available for user-defined implementations.

TCP makes sure that the stream of data it provides to the application is complete, in its proper sequence, and with no duplicated data. TCP also compensates for irregularities in the underlying network by making sure that its **segments** (data packets with headers) aren't sent so fast that they overwhelm intermediate nodes or the receiver. A TCP segment requires at least 20 bytes for its header. The data payload is optional. A segment can be at most 65,515 bytes long, including the header, so that the entire segment fits into an IP payload. If need be, IP can fragment a TCP segment if requested to do so by an intermediate node.

THE TCP SEGMENT FORMAT



The TCP segment format is shown above. The numbers at the top of the figure are the bit positions spanned by each field. The horizontal rows represent 32-bit words. The fields are defined as follows:

- Source and Destination Ports—Specify interfaces to applications running above TCP. These applications are known to TCP by their port number.
- Sequence Number—Indicates the sequence number of the first byte of data in the payload. TCP assigns each transmitted byte a sequence number. If 100 data bytes will be sent 10 bytes at a time, the sequence number in the first segment might be 0, the second 10, the third 20, and so forth. The starting sequence number is not necessarily 0, so long as the number is unique between the sender and the receiver.
- Acknowledgment Number—Contains the next data sequence number that the receiver is expecting. TCP uses this value to determine whether any datagrams have gotten lost along the way.
- Data Offset—Contains the number of 32-bit words in the header or, equivalently, the relative location of the word where the data starts within the segment. Also known as the header length.

- Reserved—These six bits must be zero until someone comes up with a good use for them.
- Flags—Contains six bits that are used mostly for protocol management. They are set to "true" when their values are nonzero. The TCP flags and their meanings are:

  URG: Indicates that urgent data exists in this segment. The Urgent Pointer field (see below) points to the location of the first byte that follows the urgent information.

  ACK: Indicates whether the Acknowledgment Number field (see above) contains significant information.

  PSH: Tells all TCP processes involved in the connection to clear their buffers, that is, "push" the data to the receiver. This flag should also be set when urgent data exists in the payload.

  RST: Resets the connection. Usually, it forces validation of all packets received and places the receiver back into the "listen for more data" state.

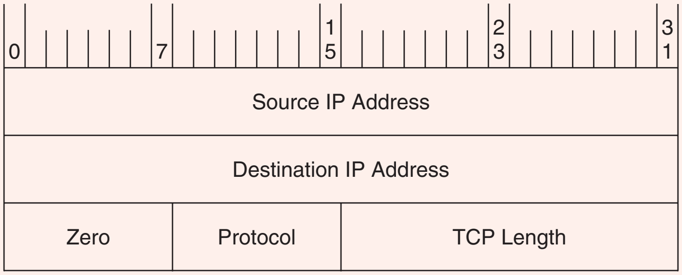
  SYN: Indicates that the purpose of the segment is to synchronize sequence numbers. If the sender transmits [SYN, SEQ# = x], it should subsequently receive [ACK, SEQ# = $x + 1$] from the receiver. At the time that two nodes establish a connection, both exchange their respective initial sequence numbers.

  FIN: This is the "finished" flag. It lets the receiver know that the sender has completed transmission, in effect starting closedown procedures for the connection.

- Window—Allows both nodes to define the size of their respective data windows by stating the number of bytes that each is willing to accept within any single segment. For example, if the sender transmits bytes numbered 0 to 1,023 and the receiver acknowledges with 1,024 in the ACK# field and a window value of 512, the sender should reply by sending data bytes 1,024 through 1,535. (This may happen when the receiver's buffer is starting to fill up so it requests that the sender slow down until the receiver catches up.) Notice that if the receiver's application is running very slowly, say it's pulling data 1 or 2 bytes at a time from its buffer, the TCP process running at the receiver should wait until the application buffer is empty enough to justify sending another segment. If the receiver sends a window size of 0, the effect is acknowledgment of all bytes up to the acknowledgment number, and to stop further data transmission until the same acknowledgment number is sent again with a nonzero window size.
- Checksum—This field contains the checksum over the fields in the TCP segment (except the data padding and the checksum itself), along with an IP pseudoheader as follows:

(continued)

THE TCP SEGMENT FORMAT (*continued*)



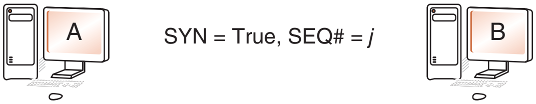
As with the IP checksum explained earlier, the TCP checksum is the 16-bit one's complement of the sum of all 16-bit words in the header and text of the TCP segment.

- Urgent Pointer—Points to the first byte that follows the urgent data. This field is meaningful only when the URG flag is set.
- Options—Concerns, among other things, negotiation of window sizes and whether selective acknowledgment (SACK) can be used. SACK permits retransmission of particular segments within a window as opposed to requiring the entire window to be retransmitted if a segment from somewhere in the middle gets lost. This concept will be clearer to you after our discussion of TCP flow control.

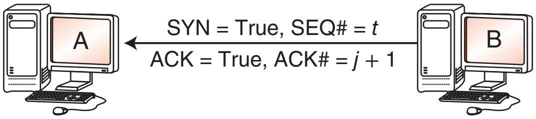
TCP provides a reliable, connection-oriented service. **Connection-oriented** means simply that the connection must be set up before the hosts can exchange any information (much like a telephone call). The reliability is provided by a sequence number assigned to each segment. Acknowledgments are used to verify that segments are received, and must be sent and received within a specific period of time. If no acknowledgment is forthcoming, the data is retransmitted. We provide a brief introduction to how this protocol works in the next section.

12.5.4 The TCP Protocol at Work

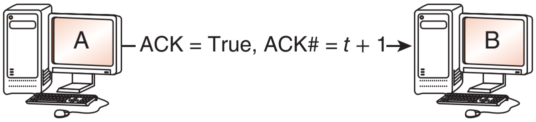
So how does all of this fit together to make a solid, sequenced, error-free connection between two (or more) TCP processes running on separate systems? Successful communication takes place in three phases: one to initiate the connection, a second to exchange the data, and a third to tear down the connection. First, the initiator, which we'll call A, transmits an "open" primitive to a TCP process running on the remote system, B. B is assumed to be listening for an "open" request. This "open" primitive has the form:



If B is ready to accept a TCP connection from the sender, it replies with:



To which A responds:



A and B have now acknowledged each other and synchronized the starting sequence numbers. A's next sequence number will be $t + 2$; B's will be $j + 2$. Protocol exchanges like these are often referred to as **three-way handshakes**. Most networking literature displays these sorts of exchanges schematically, as shown in Figure 12.7.

After the connection between A and B is established, they may proceed to negotiate the window size and set other options for their connection. The window tells the sender how much data to send between acknowledgments. For example,

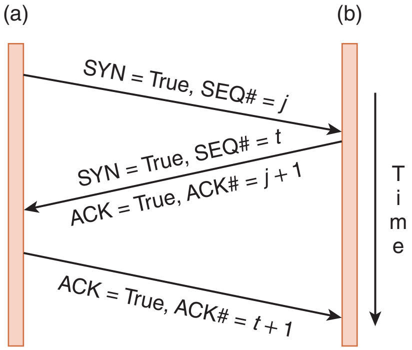


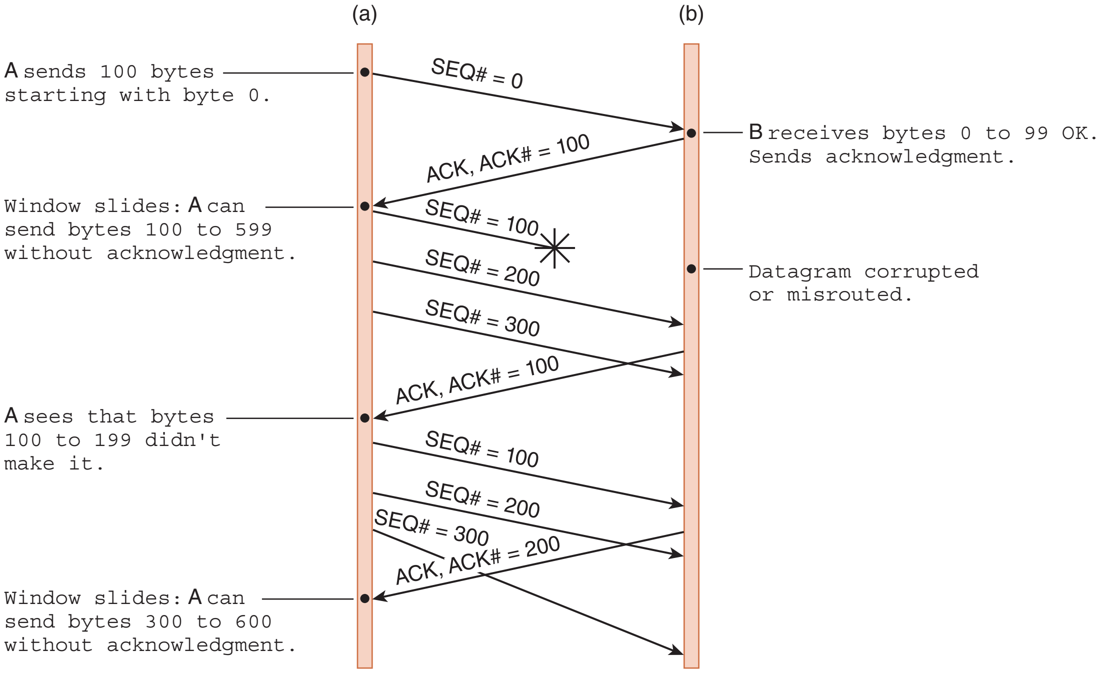
FIGURE 12.7 The TCP 3-Way Handshake

suppose A and B negotiate a window size of 500 bytes with a data payload size of 100 bytes, both agreeing not to use selective acknowledgment (discussed below). Figure 12.8 shows how TCP manages the flow of data between the two hosts. Notice what happens when a segment gets lost: The entire window is retransmitted, despite the fact that subsequent segments were delivered without error.

If an acknowledgment gets lost, however, a subsequent acknowledgment can prevent retransmission of the one that got lost, as shown in Figure 12.9. Of course, the acknowledgment must be sent in time to prevent a "timeout" retransmission.

Using acknowledgment numbers, the receiver can also ask the sender to slow down or halt transmission. It is necessary to do so when the receiver's buffer gets too full. Figure 12.10a illustrates how this is done. Figure 12.10b shows how B keeps the connection alive while it cannot receive any more data.

Upon completion of the data exchange, one or both of the TCP processes gracefully terminates the connection. One side of the connection, say A, may indicate to the other side, B, that it is finished by sending a segment with its FIN flag set to true. This effectively closes down the connection from A to B. B, however, could continue its side of the conversation until it no longer has data to send. Once B is finished, it also transmits a segment with the FIN flag set. If A acknowledges B's FIN, the connection is terminated on both ends. If B receives no acknowledgment for the duration of its timeout interval, it automatically terminates the connection.



A sends 100 bytes starting with byte 0.

B receives bytes 0 to 99 OK. Sends acknowledgment.

Window slides: A can send bytes 100 to 599 without acknowledgment.

Datagram corrupted or misrouted.

A sees that bytes 100 to 199 didn't make it.

Window slides: A can send bytes 300 to 600 without acknowledgment.

FIGURE 12.8 TCP Data Transfer with a Lost Segment

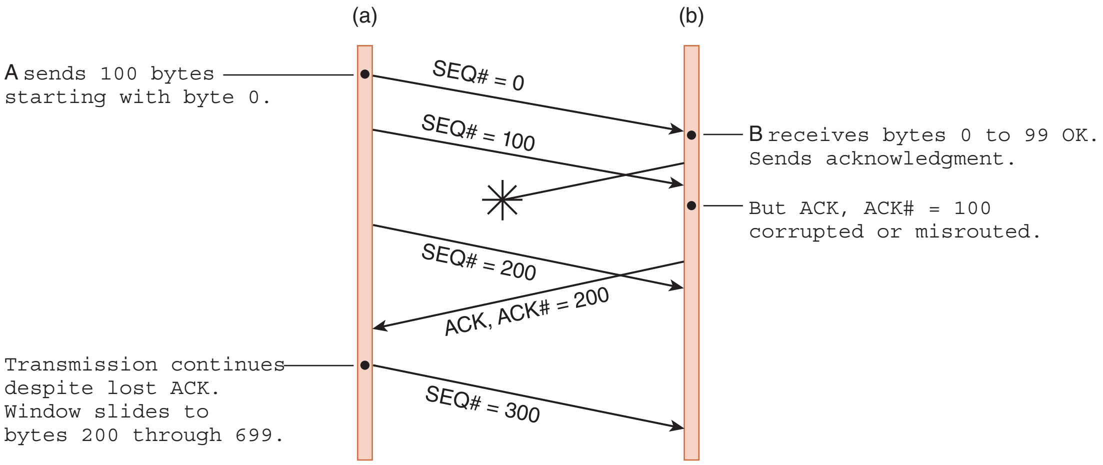

(a)
(b)
A sends 100 bytes starting with byte 0.
SEQ# = 0
B receives bytes 0 to 99 OK. Sends acknowledgment.
SEQ# = 100
But ACK, ACK# = 100 corrupted or misrouted.
SEQ# = 200
ACK, ACK# = 200
Transmission continues despite lost ACK. Window slides to bytes 200 through 699.
SEQ# = 300

FIGURE 12.9 An Acknowledgment Gets Lost

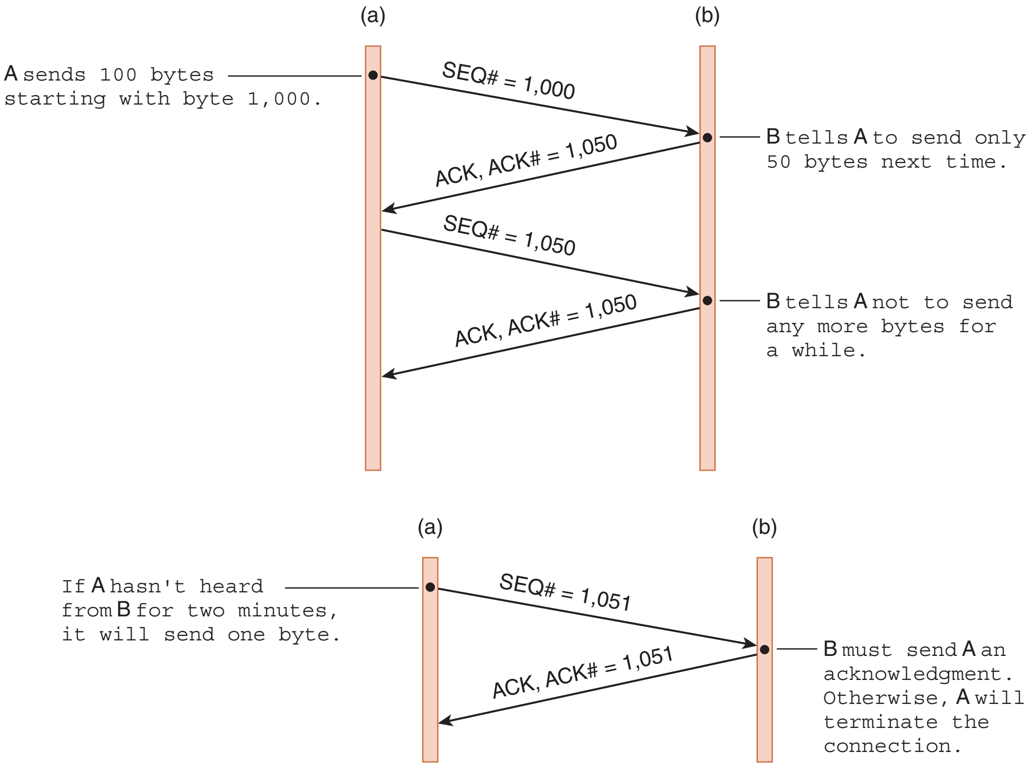

(a)
(b)
A sends 100 bytes starting with byte 1,000.
SEQ# = 1,000
B tells A to send only 50 bytes next time.
ACK, ACK# = 1,050
SEQ# = 1,050
B tells A not to send any more bytes for a while.
ACK, ACK# = 1,050
(a)
(b)
If A hasn't heard from B for two minutes, it will send one byte.
SEQ# = 1,051
B must send A an acknowledgment. Otherwise, A will terminate the connection.
ACK, ACK# = 1,051

FIGURE 12.10 TCP Flow Control

(a) B Tells A to Slow Down

(b) B Keeps the Connection Alive while Unable to Receive More Data

As opposed to having hard-and-fast rules, TCP allows the sender and receiver to negotiate a timeout period. The timeout should be set to a greater value if the connection is slower than when it is faster. The sender and receiver can also agree to use selective acknowledgment. When **selective acknowledgment (SACK)** is enabled, the receiver must acknowledge each datagram. In other words, no sliding window is used. SACK can save some bandwidth when an error occurs, because only the segment that has not been acknowledged (instead of the entire window) will be retransmitted. But if the exchange is error-free, bandwidth is wasted by sending acknowledgment segments. For this reason, SACK is chosen only when there is little TCP buffer space on the receiver. The larger the receiver's buffer, the more "wiggle room" it has for receiving segments out of sequence. TCP does whatever it can to provide the applications running above it with an error-free, sequenced stream of data.

12.5.5 IP Version 6

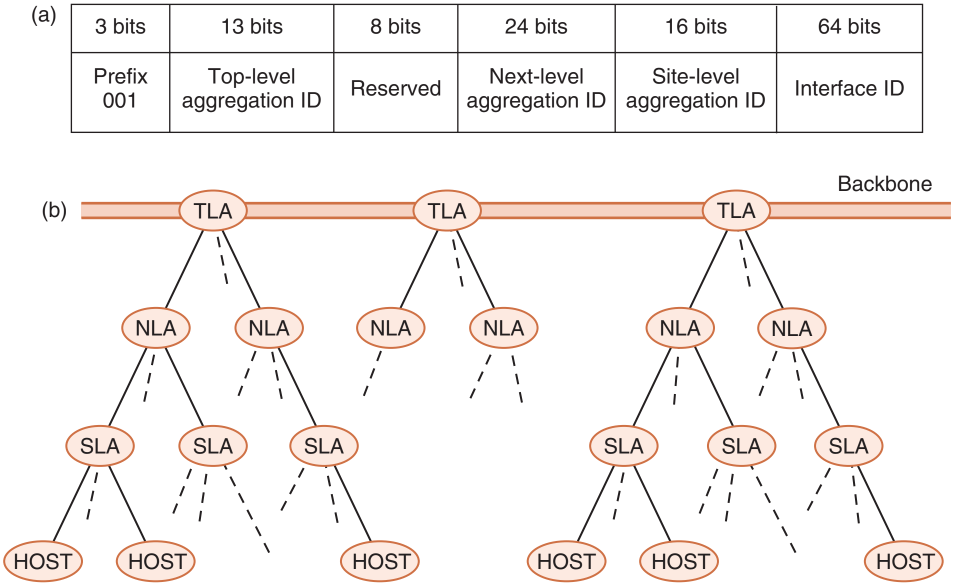
By 1994, it appeared that IP's Class B address problem was a crisis in the making, having the potential to bring the explosive growth of the Internet to an abrupt halt. Spurred by this sense of approaching doom, the IETF began concerted work on a successor to IPv4, now called IPv6. IETF participants released a number of experimental protocols that, over time, became known as IPv5. The corrected and enhanced versions of these protocols became known as IPv6. Experts predict that IPv6 won't be widely implemented until late in the first decade of the twenty-first century. (Every day more Internet applications are being modified to work with IPv6.) In fact, some opponents argue that IPv6 will "never" be completely deployed because so much costly hardware will need to be replaced and because workarounds have been found for the most vexing problems inherent in IPv4. But, contrary to what its detractors would have you believe, IPv6 is much more than a patch for the Class B address shortage problem. It fixes many things that most people don't realize are broken, as we will explain.

The IETF's primary motivation in designing a successor to IPv4 was, of course, to extend IP's address space beyond its current 32-bit limit to 128 bits for both the source and destination host addresses. This is an incredibly large address space, giving 2^{128} possible host addresses. In concrete terms, if each of these addresses were assigned to a network card weighing 28 grams (1 oz), 2^{128} network cards would have a mass 1.61 *quadrillion* times that of the entire Earth! So it would seem that the supply of IPv6 addresses is inexhaustible.

The downside of having such a large address space is that address management becomes critical. If addresses are assigned haphazardly with no organization in mind, effective packet routing would become impossible. Every router on the Internet would eventually require the storage and speed of a supercomputer to deal with the ensuing routing table explosion. To head off this problem, the IETF came up with a hierarchical address organization that it calls the **Aggregatable Global Unicast Address Format** shown in Figure 12.11a. The first 3 bits of the IPv6 address constitute a flag indicating that the address is a Global Unicast

Address. The next 13 bits form the **Top-Level Aggregation Identifier (TLA ID)**, which is followed by 8 reserved bits that allow either the TLA ID or the 24-bit **Next-Level Aggregation Identifier (NLA ID)** to expand, if needed. A TLA entity may be a country or perhaps a major global telecommunications carrier. An NLA entity could be a large corporation, a government, an academic institution, an ISP, or a small telecommunications carrier. The 16 bits following the NLA ID are the **Site-Level Aggregation Identifier (SLA ID)**. NLA entities can use this field to create their own hierarchy, allowing each NLA entity to have 65,536 subnetworks, each of which can have 2^{64} hosts. This hierarchy is shown graphically in Figure 12.11b.

At first glance, the notion of making allowances for 2^{64} hosts on each subnet seems as wasteful of address space as the IPv4 network class system. However, such a large field is necessary to support **stateless address autoconfiguration**, a new feature in IPv6. In stateless address autoconfiguration, a host uses the 48-bit address burned into its network interface card (its MAC address, explained in Section 12.6.2), along with the network address information that it retrieves from a nearby router to form its entire IP address. If no problems occur during this process, each host on the network configures its own address information with no intervention by the network administrator. This feature will be a blessing to network administrators if an entity changes its ISP or telecommunications carrier. Network administrators will have to change only the IP addresses of their routers.

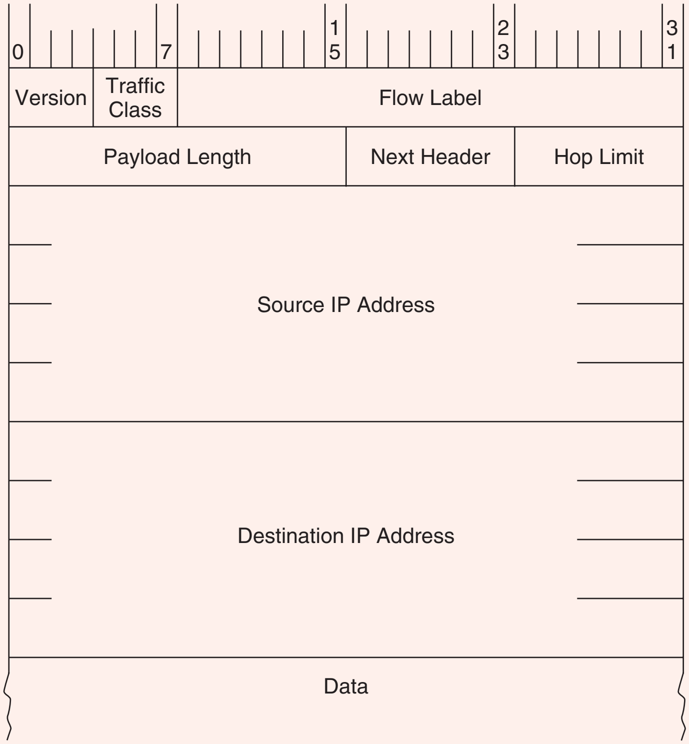


3 bits	13 bits	8 bits	24 bits	16 bits	64 bits
Prefix 001	Top-level aggregation ID	Reserved	Next-level aggregation ID	Site-level aggregation ID	Interface ID

FIGURE 12.11 (a) Aggregatable Global Unicast Address Format
(b) Aggregatable Global Unicast Hierarchy

THE IP VERSION 6 HEADER

The obvious problem with IPv4, of course, is its 32-bit address fields. IPv6 corrects this shortcoming by expanding the address fields to 128 bits. In order to keep the IPv6 header as small as possible (which speeds routing), many of the rarely used IPv4 header fields are not included in the main header of IPv6. If these fields are needed, a Next Header pointer has been provided. With the Next Header field, IPv6 could conceivably support a large number of header fields. Thus, future enhancements to IP would be much less disruptive than the switch from Version 4 to Version 6. The IPv6 header fields are explained on the next page.



Stateless address autoconfiguration will automatically update the TLA or SLA fields in every node on the network.

The written syntax of IPv6 addresses also differs from that of IPv4 addresses. Recall that IPv4 addresses are expressed using a dotted decimal notion, as in 146.186.157.6. IPv6 addresses are instead given in hexadecimal, separated by colons, as follows:

- Version—Always 0110.
- Traffic Class—IPv6 will eventually be able to tell the difference between real-time transmissions (e.g., voice and video) and less time-sensitive data transport traffic. This field will be used to distinguish between these two traffic types.
- Flow Label—This is another field for which specifications are still in progress. A “flow” is a conversation, either broadcast to all nodes or initiated between two particular nodes. The Flow Label field identifies a particular flow stream, and intermediate routers will route the packets in a manner consistent with the code in the flow field.
- Payload Length—Indicates the length of the payload in bytes, which includes the size of additional headers.
- Next Header—Indicates the type of header, if any, that follows the main header. If an IPv6 protocol exchange requires more protocol information than can be carried in a single header, the Next Header field provides for an extension header. These extension headers are placed in the payload of the segment. If there is no IP extension header, then this field will contain the value for “TCP,” meaning that the first header data in the payload belongs to TCP, not IP. In general, only the destination node will examine the contents of the extension headers. Intermediate nodes pass them on as if they were common payload data.
- Hop Limit—With 16 bits, this field is much larger than in Version 4, allowing 256 hops. As in Version 4, this field is decremented by each intermediate router. If it ever becomes zero, the packet is discarded and the sender is notified through an ICMP (for IPv6) message.
- Source and Destination Addresses—Much larger, but with the same meaning as in Version 4. See text for a discussion of the format for this address.

```
30FA:505A:B210:224C:1114:0327:0904:0225
```

making it much easier to recognize the binary equivalent of an IP address.

IPv6 addresses can be abbreviated, omitting zeros where possible. If a 16-bit group is 0000, it can be written as 0, or omitted altogether. If more than two consecutive colons result from this omission, they can be reduced to two colons (provided there is only one group of more than two consecutive colons). For example, the IPv6 address:

```
30FA:0000:0000:0000:0010:0002:0300
```

can be written

```
30FA:0:0:0:10:2:300
```

or even

```
30FA::10:2:300
```

However, an address such as 30FA::24D6::12CB is invalid.

The IETF is also proposing two other routing improvements: implementation of **multicasting** (where one message is placed on the network and read by multiple nodes) and **anycasting** (where any one of a logical group of nodes can be the recipient of a message, but no particular receiver is specified by the packet). This feature, along with stateless address autoconfiguration, facilitates support for mobile devices, an increasingly important sector of Internet users, particularly in countries where most telecommunications take place over wireless networks.

As previously mentioned, security is another major area in which IPv6 differs from IPv4. All of IPv4's security features (IPSec) are "optional," meaning that no one is forced to implement any type of security, and most installations don't. In IPv6, IPSec is mandatory. Among the security improvements in IPv6 is a mechanism that prevents **address spoofing**, where a host can engage in communications with another host using a falsified IP address. (IP spoofing is often used to subvert filtering routers and firewalls that are intended to keep outsiders from accessing private intranets, among other things.) IPSec also supports encryption and other measures that make it more difficult for miscreants to sniff out unauthorized information.

Perhaps the best feature of IPv6 is that it provides a transition plan that allows networks to gradually move to the new format. Support for IPv4 is built into IPv6. Devices that use both protocols are called **dual stack** devices, because they support protocol stacks for both IPv4 and IPv6. Most routers on the market today are dual stack devices, with the expectation that IPv6 will become a reality in the not-too-distant future.

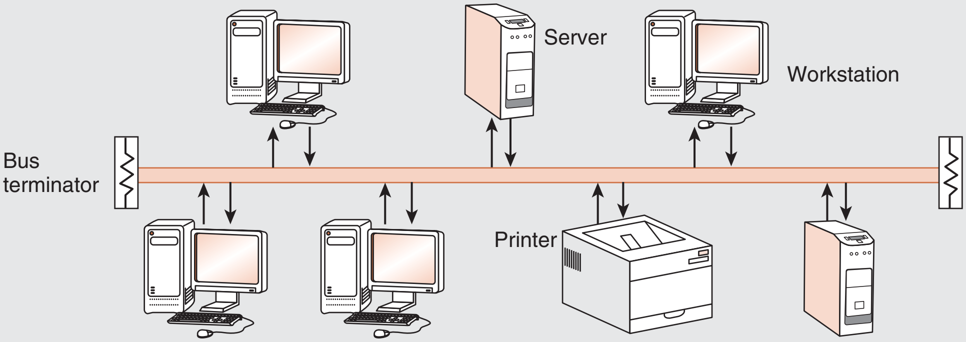
The benefits of IPv6 over IPv4 are clear: a greater address space, better and built-in quality of service, and better and more efficient routing. It is not a question of *if* but of *when* we will move to IPv6. The transition will be driven by the business need for IPv6 and development of the necessary applications. Although hardware replacement cost is a significant barrier, technician training and replacement of minor IP devices (such as network fax machines and printers) will contribute to the overall cost of conversion. With the advent of IP-ready automobiles, as well as many other Internet devices, IPv4 no longer meets the needs of many current applications.

Ethernet Then and Now

Ethernet is today's dominant architecture for **local area networks (LANs)**. Ethernet devices can be found on desktops, in data centers, and in living rooms. Its ubiquity was never envisioned by its inventors, Robert Metcalf and David Boggs of the Xerox Corporation, when they developed Ethernet in the early 1970s. In the early 1980s, DEC and Intel were helping to refine Ethernet, and they brought many new and innovative products to market. By the time the IEEE began its Project 802, Ethernet was a well-established networking architecture. As such, its operational theory became part of the IEEE 802.3 standard, but Ethernet is not entirely congruent with the standard, because Ethernet defines Logical Link functionality where IEEE 802.3 does not. (IEEE 802.2 is the Data Link specification.) There are also some slight differences between the Ethernet Protocol Data Unit and the IEEE 802.3 PDU. Because of its wide deployment, our discussion will center on Ethernet.

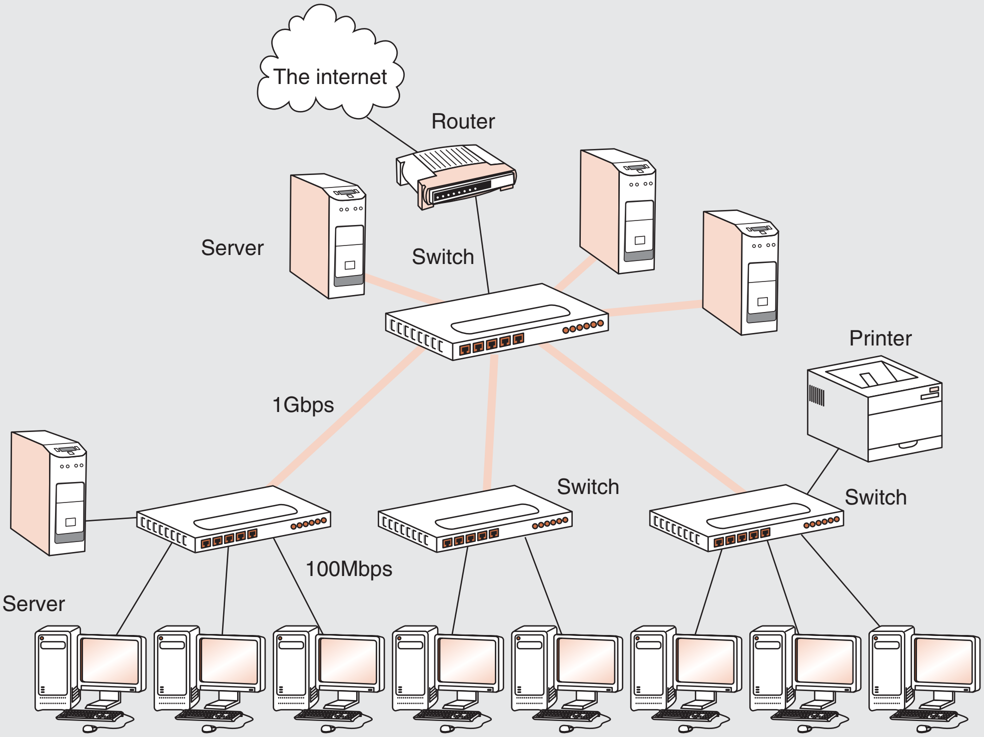
Ethernet uses a **Carrier Sense Multiple Access / Collision Detection (CSMA/CD)** medium access control. Before placing data on the network, the interface circuit checks to see that the carrier (i.e., the network) is alive; then it listens to see if any other stations are using the line. If the line is silent (quiesced), the interface places its transmission frame on the line. If another station happens to do this at the same instant, the frames will collide. This collision should be detected by both transmitting stations.

Traditional Ethernet networks run at 10Mbps, and they use a bus topology as shown in the figure below. All stations connect directly to the bus (Multiple Access); thus all nodes receive every frame. As each LAN station receives a frame, it checks the (MAC) address information in the frame header. If a station sees its own address, it strips off the framing bits and then passes the PDU to its Logical Link layer.



A Traditional Ethernet Bus Architecture

In order to detect collisions, an Ethernet interface card must listen to the line while it is transmitting. If it sees a voltage higher than allowed, that means that another node has begun transmitting at nearly the same time and the frames have collided somewhere along the bus. Once the collision is discovered, the NIC broadcasts a 32-bit jamming signal to notify all stations that a collision has occurred. The stations that were involved in the collision will both cease transmitting and wait a (pseudo) random amount of time before trying once more to transmit. This approach works well when the PDU is long enough to span the distance between the sending node and the destination. Because of this, the maximum speed on a traditional shared-bus CSMA/CD Ethernet system is about 100Mbps. At faster speeds, the traditional Ethernet PDU is too short. (The bit cells are narrower at faster bit rates.) Supporting speeds much beyond that calls for adjustments to Ethernet's architecture. Gigabit speeds require changing the topology of the network from a shared-bus to a switched star-shaped network topology, as shown below. Switches in a star topology funnel one signal at a time from the user nodes onto the backbone of the network. The end nodes connect to the network at 100Mbps, and upstream switches interconnect at 1Gbps or 10Gbps. Medium access management is taken care of at the switch level, so collision detection at the end nodes may be turned off. Switches can be connected to other network components such as hubs and repeaters to further extend the network and provide downward compatibility to slower segments. Gigabit Ethernet competes directly with more radical and costly network solutions such as Fibre Channel. We discuss and Fibre Channel in Chapter 13.



A Gigabit Ethernet LAN

12.6 NETWORK ORGANIZATION

Computer networks are often classified according to their geographic service areas. The smallest networks are **local area networks (LANs)**. Although they can encompass thousands of nodes, LANs typically are used in a single building, or a group of buildings that are near each other. When a LAN covers more than one building, it is sometimes called a **campus network**. Usually, the region (the property) covered by a LAN is under the same ownership (or control) as the LAN itself. **Metropolitan area networks (MANs)** are networks that cover a city and its environs. They often span areas that are not under the ownership of the people who also own the network. **Wide area networks (WANs)** can cover multiple cities or span the entire world.

At one time, the protocols employed by LANs, MANs, and WANs differed vastly from one another. MANs and WANs were usually designed for high-speed throughput because they served as backbone systems for multiple slower LANs, or they offered access to large host computers in data centers far away from end users. As network technologies have evolved, however, these networks are now distinguished from each other not so much by their speed or by their protocols, but by their ownership. One person's campus LAN might be another person's MAN. In fact, as LANs are becoming faster and more easily integrated with WAN technology, it is conceivable that eventually the concept of a MAN may disappear entirely.

This section discusses the physical network components common to LANs, MANs, and WANs. We start at the lowest level of network organization, the physical medium level, Layer 1.

12.6.1 Physical Transmission Media

Virtually any medium with the ability to carry a signal can support data communication. There are two general types of communications media: **Guided transmission media** and **unguided transmission media**. Unguided media broadcast data over the airwaves using infrared, microwave, satellite, or broadcast radio carrier signals. Guided media are physical connectors such as copper wire or fiber-optic cable that directly connect to each network node.

The physical and electrical properties of guided media determine their ability to accurately convey signals of given frequencies over various distances. In Chapter 7, we mentioned that signals attenuate (get weaker) over long distances. The longer the distance and the higher the signal frequency, the greater the attenuation. Attenuation in copper wire results from the interactions of several electrical phenomena. Chief among these are the internal resistance of copper conductors and the electrical interference (inductance and capacitance) that occurs when signal-carrying wires are in close proximity to each other. External electrical fields such as those surrounding fluorescent lights and electric motors can also attenuate—or even garble—signals as they are transmitted over copper wire. Collectively, the electrical phenomena that work against the accurate transmission of signals are called **noise**. Signal and noise strengths are both measured in decibels (dB). Cables are rated according to how well they convey signals at different frequencies in the presence of noise. The resulting quantity is the **signal-to-noise** rating for the communications channel, and it is also measured in decibels:

$$\text{Signal-to-Noise Ration (dB)} = 10 \log_{10} \frac{\text{Signal dB}}{\text{Noise dB}}$$

The **bandwidth** of a medium is technically the range of frequencies that it can carry, measured in hertz. The wider the medium's bandwidth, the more information it can carry. In digital communications, bandwidth is the general term for the information-carrying capacity of a medium, measured in bits per second (bps). Another important measure is **bit error rate (BER)**, which is the ratio of the number of bits received in error to the total number of bits received. If signal frequencies exceed the signal-carrying capacity of the line, the BER may become so extreme that the attached devices will spend more of their time doing error recovery than in doing useful work.

Coaxial Cable

Coaxial cable was once the medium of choice for data communications. It can carry signals up to trillions of cycles per second with low attenuation. Today, it is used mostly for broadcast and closed circuit television applications. Coaxial cable also carries signals for residential Internet services that piggyback on cable television lines.

The heart of a coaxial cable is a thick (12- to 16-gauge) inner conductor surrounded by an insulating layer called a **dielectric**. The dielectric is surrounded by a foil shield to protect it from transient electromagnetic fields. The foil shield is itself wrapped in a steel or copper braid to provide an electrical ground for the cable. The entire cable is then encased in a durable plastic coating (see Figure 12.12).

The coaxial cable employed by cable television services is called **broadband cable** because it has a capacity of at least 2Mbit/sec. Broadband communication provides multiple channels of data, using a form of multiplexing. Computer networks now infrequently use **narrowband cable**, which is optimized for a typical bandwidth of 64kbit/sec, consisting of a single channel.

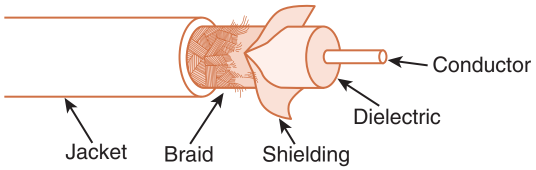


FIGURE 12.12 The Parts of a Coaxial Cable

Twisted Pair

The easiest way to connect two computers is simply to run a pair of copper wires between them. One of the wires is used for sending data, the other for receiving. Of course, the farther apart the two systems are, the stronger the signal has to be to prevent them from attenuating into oblivion over long distances. The distance between the two systems also affects the speed at which data can be transferred. The farther apart they are, the slower the line speed must be to avoid excessive errors. Using thicker conductors (smaller wire gauge numbers) can reduce attenuation. Of course, thick wire is more costly than thin wire.

In addition to attenuation, cable makers are also challenged by an electrical phenomenon known as **inductance**. When two wires lie perfectly flat and adjacent to each other, strong high-frequency signals in the wires create magnetic (inductive) fields around the copper conductors, which interfere with the signals in both lines.

The easiest way to reduce the electrical inductance between conductors is to twist them together. To a point, the more twists that are introduced in a pair of wires per linear foot, the less attenuation is caused by the wires interfering with each other. Twisted wire is more costly to manufacture than untwisted wire because more wire is consumed per linear foot and the twisting must be carefully controlled. **Twisted pair** cabling, with two twisted wire pairs, is used in most LAN installations today (see Figure 12.13). It comes in two varieties: shielded and unshielded. Unshielded twisted pair is the most popular.

Shielded twisted pair cable is suitable for environments having a great deal of electrical interference. Today's business environments are teeming with sources of electromagnetic radiation that can interfere with network signals. These sources can be as seemingly benign as fluorescent lights or as obviously hostile as large, humming power transformers. Any device that produces a magnetic field has the potential for scrambling network communication links. Interference can limit the speed of a network because higher signal frequencies are more sensitive to any kind of signal distortion. As a safeguard against environmental interference (called **electromagnetic interference [EMI]** or **radio-frequency interference [RFI]**), shielded twisted pair wire can be installed to help maintain the integrity of network communications in hostile environments.

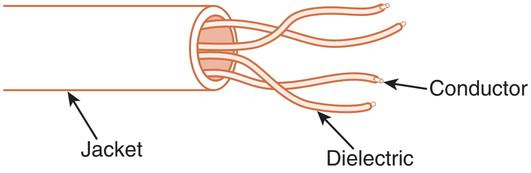


FIGURE 12.13 Twisted Pair Cable

Experts disagree as to whether this shielding is worth the higher material and installation costs. They point out that if the shielding is not properly grounded, it can actually cause more problems than it solves. Specifically, it can act as an antenna that actually attracts radio signals to the conductors!

Whether shielded or unshielded, network conductors must have signal-carrying capacity appropriate to the network technology being used. The Electronic Industries Alliance (EIA), along with the Telecommunications Industry Association (TIA), established a rating system for network cabling in 1991. The latest revision of this rating system is EIA/TIA-568B. The EIA/TIA **category** ratings specify the maximum frequency that the cable can support without excessive attenuation. The ISO rating system, which is not used as often as the EIA/TIA category system, refers to these wire grades as **classes**. These ratings are shown in Table 12.1. Most LANs installed today are equipped with Category 5 or better cabling. Many installations are abandoning copper entirely and installing fiber-optic cable instead (see next Section).

Note that the signal-carrying capacity of the cable grades shown in Table 12.1 is given in terms of megahertz. This is not the same as megabits. As we saw in Chapter 2, Section 2.A, the number of bits carried at any given frequency is a function of the encoding method used in the network. Networks running below 100Mbps could easily afford to use Manchester coding, which requires two signal transitions for every bit transmitted. Networks running at 100Mbps and above use different encoding schemes, one of the most popular being the 4B/5B, 4 bits in 5 baud using NRZI signaling, as shown in Figure 12.14.

Baud is the unit of measure for the number of signal transitions supported by a transmission medium or transmission method over a medium. For networks other than the voice telephone network, the line speed is rated in hertz, but hertz and baud are equivalent with regard to digital signals. As you can see in Figure 12.14, if a network uses 4B/5B encoding, a signal-carrying capacity of 125MHz is required for the line to have a bit rate of 100Mbps.

Fiber-Optic Cable

Optical fiber network media can carry signals faster and farther than either twisted pair or coaxial cable. Fiber-optic cable is theoretically able to support frequencies in the terahertz range, but transmission speeds are more commonly in the range of about 2GHz, carried over runs of 10 to 100km (without

EIA/TIA	ISO	Maximum Frequency
Category 1		Voice and "low-speed" data (4–9.6kHz)
Category 2	Class A	1Mbps or less
Category 3	Class B	10MHz
Category 4	Class C	20MHz
Category 5	Class D	100MHz
Category 6	Class E	250MHz
Category 7	Class F	600MHz

TABLE 12.1 EIA/TIA-568B and ISO Cable Specifications

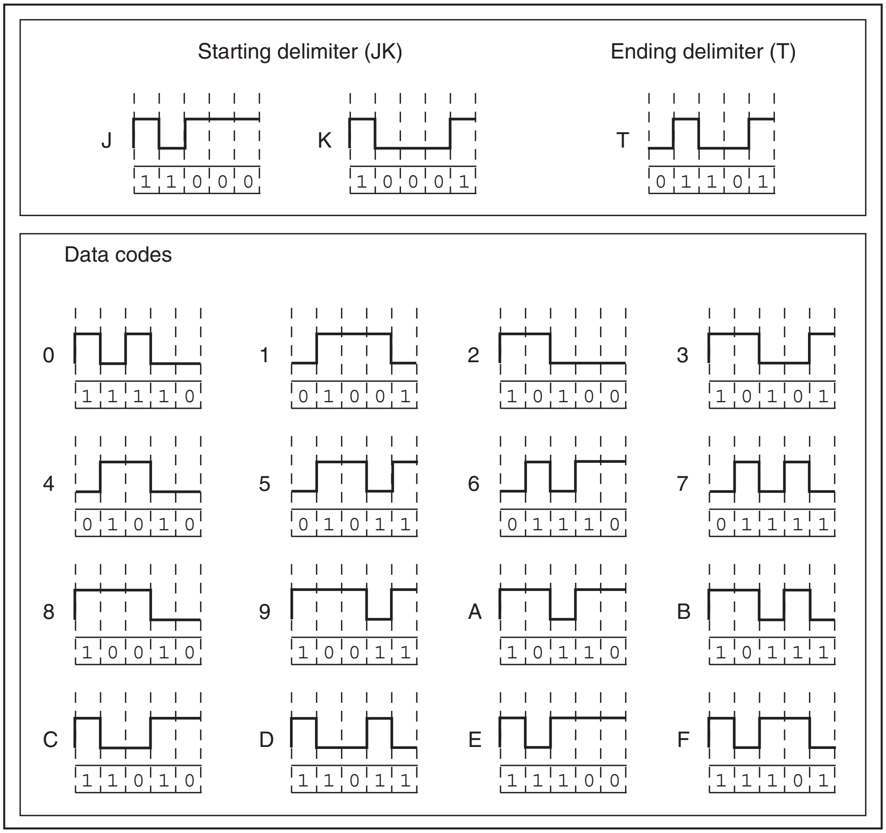


FIGURE 12.14 4B/5B Encoding

repeaters). Optical cable consists of bundles of thin (1.5 to 125μm) glass or plastic strands surrounded by a protective plastic sheath. Although the underlying physics is quite different, you can think of a fiber-optic strand as a conductor of light just as copper is a conductor of electricity. The cable is a type of "light guide" that routes the light from one end of the cable to the other. At the sending end, a light emitting diode or laser diode emits pulses of light that travel through the glass strand, much as water goes through a pipe. On the receiving end, photodetectors convert the light pulses into electrical signals for processing by electronic devices.

Optical fiber supports three different transmission modes depending on the type of fiber used. The types of fiber are shown in Figure 12.15. The narrowest fiber, **single-mode** fiber, conveys light at only one wavelength, typically 850, 1,300, or 1,500nm. It allows the fastest data rates over the longest distances.

Multimode fiber can carry several different light wavelengths simultaneously through a larger fiber core. In multimode fiber, the laser light waves bounce off the sides of the fiber core, causing greater attenuation than single-mode fiber.

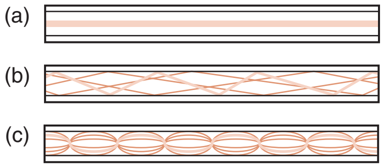


FIGURE 12.15 Optical Fiber
(a) Single Mode
(b) Multimode
(c) Graded Index

Not only do the light waves scatter, but they also collide with one another to some degree, causing further attenuation.

Multimode graded index fiber also supports multiple wavelengths concurrently, but it does so in a more controlled manner than regular multimode fiber. Multimode graded index fiber consists of concentric layers of plastic or glass, each with refractive properties that are optimized for carrying specific light wavelengths. Like regular multimode fiber, light travels in waves through multimode graded index optical fiber. But unlike multimode fiber, the waves are confined to the area of the optical fiber that is suitable to propagating its particular wavelength. Thus, the different wavelengths concurrently transmitted through the fiber do not interfere with each other.

The fiber-optic medium offers many advantages over copper, the most obvious being its enormous signal-carrying capacity. It is also immune to EMI and RFI, making it ideal for deployment in industrial facilities. Fiber-optic cable is small and lightweight, one fiber being capable of replacing hundreds of pairs of copper wires.

But optical cable is fragile and costly to purchase and install. Because of this, fiber is most often used as network **backbone cable**, which bears the traffic of hundreds or thousands of users. Backbone cable is like an interstate highway. Access to it is limited to specific entrance and exit points, but a large volume of traffic is carried at high speed. For a vehicle to get to its final destination, it has to exit the highway and perhaps drive through a residential street. The network equivalent of a residential street most often takes the form of twisted pair copper wire. This "residential street" copper wire is sometimes called **horizontal cable**, to differentiate it from backbone (**vertical**) cable. Undoubtedly, "fiber to the desktop" will eventually become a reality as costs decrease. At the same time, demand is steadily increasing for the integration of data, voice, and video over the same cable. With the deployment of these new technologies, network media probably will be stretched to their limits before the next generation of high-speed cabling is introduced.

Unguided Media—Wireless Data Communications

Bit patterns can be conveyed over any medium capable of supporting a signal. Accordingly, the transmission hardware and methods employed in wireless data communications vary widely. We can't even begin to cover them all here. We will, however, say a few words about the wireless data communications standards that most of us encounter in our daily activities: cellular wireless, Bluetooth, and the 802.11x family of standards.

As their name implies, cellular wireless networks transmit data over the cellular telephone network. Like the early residential telephone network, the cellular system was not intended to be a data network. Consequently, the so-called first- and second-generation cellular data transmission networks have limited features and transmission rates (usually between 0.3 and 1Mbps). There is indeed as much to be gained by establishing broadband cellular service as there was in establishing broadband residential service over guided media (cable and telephone). Fast cellular data communications provides added convenience for people who are already accustomed to fast networking at home and at the office. It also has the potential to reach out to millions of new customers who have been underserved by landlines. With this in mind, the ITU has defined a third-generation wireless communication structure, commonly called **3G**. The features of 3G include data rates up to 2.048Mbps, support for a wide array of equipment, and the integration of low-Earth-orbiting (LEO) satellites into a unified system. Although access speeds may vary, it is possible that with 3G, the World Wide Web will finally be open to the entire world.

Bluetooth, otherwise known as IEEE 802.15.1-2002, was conceived of by Ericsson (Telefonaktiebolaget LM Ericsson) in 1994. Bluetooth is the namesake of a tenth-century king of Denmark and Norway who is famous for ending the hostilities that had been raging among a number of Danish, Norwegian, and Swedish tribes. The analogous aim of Bluetooth is to bring together differing technologies for interconnecting computers and other equipment over very short distances, officially known as **personal area networks (PANs)**, or **piconets**. The first Bluetooth specification was released in 1999 by a consortium consisting of Ericsson, IBM, Intel, Nokia, and Toshiba.

A Bluetooth network consists of a master device and up to seven slave devices, to which the master communicates in round-robin fashion. Data transmission of 720Kbps occurs at very low power (no more than 100mw) over an unregulated frequency of 2.45GHz. Bluetooth is very popular for connecting portable computing devices (such as tablet PCs, PDAs, and cell phones) to a variety of peripheral devices without the need for cables or cable sockets that take up precious space in portable devices.

Wireless local area networks (**WLANs**) are much slower than hardwired LANs, but they provide many other benefits that have contributed to their proliferation. The most important of these is the fact that a network can be set up just about anywhere, and it can be reconfigured with unparalleled ease. The IEEE 802.11 family of WLAN standards has grown steadily since the first standard was published in 1997. We have provided a brief synopsis of the various components of the standard in Table 12.2.

A typical WLAN consists of one or more interconnected (hardwired) **wireless access points (WAPs)** that broadcast data to nodes on the network. The node maintains a connection through its assigned frequency for the duration of its session, or until the connection is broken. The range of the WAPs

IEEE Specification	Description
802.11 - 2007	The basic wireless standard includes amendments a – j and the maintenance revision, 802.11.RevMA.
802.11k - 2007	Radio resource measurements to allow remote management of services such as roaming.
802.11n - 2009	Improvements for throughput, up to 600Mbps over the 2.4 and 5GHz bands with a range of around 250ft indoors.
802.11p - 2010	Extensions specifically for automobiles and other vehicles in the 5.9GHz spectrum.
802.11r - 2008	Fast roaming: Allows reliable handoffs between base stations to accommodate wireless devices in motion (e.g., in cars).
802.11s - 2010	Wireless mesh networking
802.11u - 2010	Interworking with non-802 networks, such as 3G cellular.
802.11v - 2010	Wireless network management.
802.11.w - 2009	Wireless LAN management frame protection.
802.11.y - 2008	MAC layer enhancements for the 3,650MHz band for wireless LANs.

TABLE 12.2 IEEE 802.11 Wireless Network Standards

is limited by ambient electromagnetic interference and obstructions such as walls and furniture. In general, the faster a wireless network's data transmission speed, the more susceptible it is to obstructions, and the shorter its range of transmission. Nodes farthest from the WAP will have the lowest throughput, because the transmission speed is stepped down to accommodate the greater distance.

Security is a continual problem in wireless networking. Certain measures can be taken to make unauthorized access difficult, such as the use of the 128-bit encryption mode of **wired equivalent privacy (WEP)**. Security experts caution, however, that it is impossible to block a determined and sophisticated hacker. If security is a concern, any form of wireless networking should be deployed with extreme care. Even piconets can open the door for malicious access to the entire enterprise network.

12.6.2 Interface Cards

Transmission media are connected to clients, hosts, and other network devices through network interfaces. Because these interfaces are often implemented on removable circuit boards, they are commonly called **network interface cards**, or simply **NICs**. (Please don't say "NIC card"!) A NIC usually embodies the lowest three layers of the OSI protocol stack. It forms the bridge between the physical components of the network and your system. NICs attach directly to a system's main bus or dedicated I/O bus. They convert the parallel data passed on the system bus to the serial signals broadcast on a communications medium. NICs change the encoding of the data from binary to the Manchester or 4B/5B of the network (and vice versa). NICs also provide physical connections and negotiate permission to place signals on the network medium.

Every network card has a unique physical address burned into its circuits. This is called a **Medium Access Control (MAC)** address, and it is 6 bytes long. The first 3 bytes are the manufacturer's identification number, which is designated by the IEEE. The last 3 bytes are a unique identifier assigned to the NIC by the manufacturer. No two cards anywhere in the world should ever have the same MAC address. Network protocol layers map this physical MAC address to at least one logical address. The logical address is the name or address by which the node is known to other nodes on the network. It is possible for one computer (logical address) to have two or more NICs, but each NIC will have a distinct MAC address.

12.6.3 Repeaters

A small office LAN installation will have many NICs within a few feet of each other. In an office complex, however, NICs may be separated by hundreds of feet of cable. The longer the cable, the greater the signal attenuation. The effects of attenuation can be mitigated either by reducing transmission speed (usually an unacceptable option) or by adding repeaters to the network. **Repeaters** counteract attenuation by amplifying signals as they are passed through the physical cabling of the network. The number of repeaters required for any network depends on the distance over which the signal is transmitted, the medium used, and the signaling speed of the line. For example, high-frequency copper wire needs more repeaters per kilometer than optical cable operating at a comparable frequency.

Repeaters are part of the network medium. In theory, they are dumb devices functioning entirely without human intervention. As such, they would contain no network-addressable components. However, some repeaters now offer higher-level services to assist with network management and troubleshooting. Figure 12.16 is a representation of how a repeater regenerates an attenuated digital signal.

12.6.4 Hubs

Repeaters are Physical layer devices having one input and one output port. **Hubs** are also Physical layer devices, but they can have many ports for input and

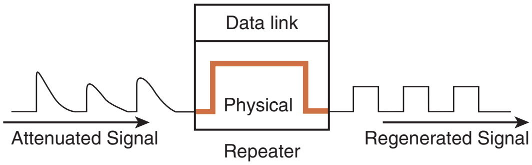


FIGURE 12.16 The Function of a Repeater in the OSI Reference Model

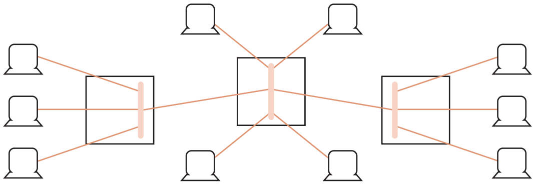


FIGURE 12.17 A Network Connected with Hubs

output. They receive incoming packets from one or more locations and broadcast the packets to one or more devices on the network. Hubs allow computers to be joined to form **network segments**. The simplest hubs are nothing more than repeaters that connect various branches of a network. Physical network branches stuck together by hubs do not partition the network in any way; they are strictly Layer 1 devices and are not aware of a packet's source or its destination. Every station on the network continues to compete for bandwidth with every other station on the network, regardless of the presence or absence of intervening hubs. Because hubs are Layer 1 devices, the physical medium must be the same on all ports of the hub. You can think of simple hubs as being nothing more than repeaters that provide multiple station access to the physical network. Figure 12.17 shows a network equipped with three hubs.

As hub architectures have evolved, many now have the ability to connect dissimilar physical media. Although such media interconnection is a Layer 2 function, manufacturers continue to call these devices "hubs." **Switching hubs** and **intelligent hubs** are still further removed from the notion of a hub being a "Layer 1 device." These sophisticated components not only connect dissimilar media, but also perform rudimentary routing and protocol conversion, which are all Layer 3 functions.

12.6.5 Switches

A switch is a Layer 2 device that creates a point-to-point connection between one of its input ports and one of its output ports. Although hubs and switches perform the same function, they differ in how they handle the data internally. Hubs broadcast the packets to all computers on the network and handle only one packet at a time. Switches, on the other hand, can handle multiple communications between the computers attached to them. If there were only two computers on the network, a hub and a switch would behave in exactly the same way. If more than two computers were trying to communicate on a network, a switch gives better performance because the full bandwidth of the network is available at both sides of the switch. Therefore, switches are preferred to hubs in most network installations. In Chapter 9, we introduced switches that connect processors to memories or processors to processors. Those switches are the same kind of switches we discuss here. Switches contain some number of buffered input ports, an equal number of output ports, a **switching fabric** (a combination of the switching units, the integrated circuits that they contain, and the programming that allows switching paths to be controlled), and digital hardware that interprets address information encoded on network frames as they arrive in the input buffers.

As with most of the network components we have been discussing, switches have been improved by adding addressability and management features. Most switches today can report on the amount and type of traffic they are handling and can even filter out certain network packets based on user-supplied parameters. Because all switching functions are carried out in hardware, switches are the preferred devices for interconnecting high-performance network components.

12.6.6 Bridges and Gateways

The purpose of both bridges and gateways is to provide a link between two dissimilar network segments. Both can support different media (and network speeds), and they are both "store and forward" devices, holding an entire frame before sending it on. But that's where their similarities end.

Bridges join two similar types of networks so they look like one network. With bridges, all computers on the network belong to the same **subnet** (the network consisting of all devices whose IP addresses have the same prefix). Bridges are relatively simple devices with functionality primarily at Layer 2. This means that they know nothing about protocols, but simply forward data depending on the destination address. Bridges can connect different media having different media access control protocols, but the protocol from the MAC layer through all higher layers in the OSI stack must be identical in both segments. This relationship is shown in Figure 12.18.

Each node connected to any particular bridge must have a unique address. (The MAC address is most often used.) The network administrator must program simple bridges with the addresses and segment numbers of each valid node on the network. The only data that is allowed to cross the bridge is data that is being sent to a valid address on the other side of the bridge. For large networks that change frequently (most networks), this continual reprogramming is tedious, time-consuming, and error prone. **Transparent bridges** were invented to alleviate this problem. They are sophisticated devices that have the ability to learn the address of every device on each segment. Transparent bridges can also supply management information such as throughput reports. Such functionality implies that a bridge is not entirely a Layer 2 device. However, bridges still require

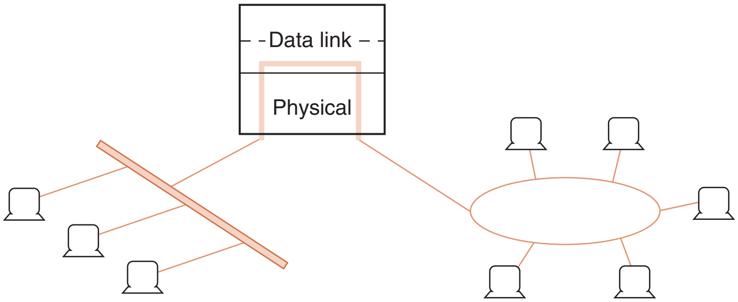


FIGURE 12.18 A Bridge Connecting Two Networks

identical Network layer protocols and identical interfaces to those protocols on both interconnected segments.

Figure 12.18 shows two different kinds of local area networks connected to each other through a bridge. This is typically how bridges are used. If, however, users on these LANs needed to connect to a system that uses a radically different protocol, for example, a public switched telephone network or a host computer that uses a nonstandard proprietary protocol, then a **gateway** is required. A gateway is a point of entrance to another network. Gateways are full-featured computers that supply communications services spanning all seven OSI layers. Gateway system software converts protocols and character codes, and can provide encryption and decryption services. Because they do so much work in their software, gateways cannot provide the throughput of hardware-based bridges, but they make up for it by providing enormously more functionality. Gateways are often connected directly to switches and routers.

12.6.7 Routers and Routing

After gateways, routers are the next most complicated components in a network. They are, in fact, small special-purpose computers. A **router** is a device (or a piece of software) connected to at least two networks that determines the destination to which a packet should be forwarded. Routers are normally located at gateways. Operating correctly, routers make the network fast and responsive. Operating incorrectly, one faulty router can bring down the whole system. In this section, we reveal the inner workings of routers and discuss the thorny problems that routers are called upon to solve.

Despite their complexity, routers are usually referred to as Layer 3 devices, because most of their work is done at the Network layer of the OSI Reference Model. However, most routers also provide some network monitoring, management, and troubleshooting services. Because routers are by definition Layer 3 devices, they can bridge different network media types (fiber to copper, for example) and connect different network protocols running at Layer 3 and below. Because of their abilities, routers are sometimes referred to as “intermediate systems” or “gateways” in Internet standards literature. (When the first Internet standards were written, the word *router* hadn’t yet been coined.)

Routers are designed specifically to connect two networks together, typically a LAN to a WAN. They are complex devices because not only do they contain buffers and switching logic, but they also have enough memory and processing power to calculate the best way to send a packet to its destination. A conceptual model of the internals of a router is shown in Figure 12.19.

In large networks, routers find an approximate solution to a problem that is fundamentally NP complete. An NP-complete problem is one in which its optimal solution is theoretically impossible within a time period that is short enough for that solution to be useful.

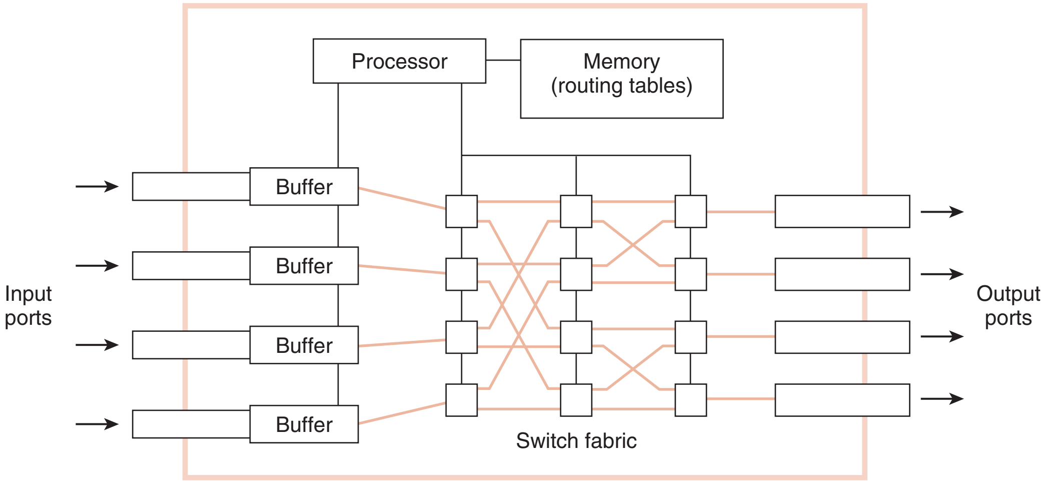


FIGURE 12.19 Anatomy of a Router

Consider the network shown in Figure 12.20. You may recognize this figure as a complete graph (K_5). There are $n(n - 1) / 2$ edges in a complete graph containing n nodes. In our illustration, we have 5 nodes and 10 edges. The edges represent routes—or **hops**—between each of the nodes.

If Node 1 (Router 1) needs to send a packet to Node 2, it has the following choices of routes:

one route of one hop:
1 → 2

three routes of two hops:
1 → 3 → 2 1 → 4 → 2 1 → 5 → 2

six routes of three hops:
1 → 3 → 4 → 2 1 → 3 → 5 → 2 1 → 5 → 4 → 2
1 → 4 → 3 → 2 1 → 5 → 3 → 2 1 → 4 → 5 → 2

six routes of four hops:
1 → 3 → 4 → 5 → 2 1 → 4 → 3 → 5 → 2 1 → 5 → 4 → 3 → 2
1 → 3 → 5 → 4 → 2 1 → 4 → 5 → 3 → 2 1 → 5 → 3 → 4 → 2

When Node 1 and Node 2 are not directly connected, the traffic between them must pass through at least one intermediate node. Considering all options, the number of possible routes is on the algorithmic order of $N!$. This problem is further complicated when costs or weights are applied to the routing paths. Worse yet, the weights can change depending on traffic flow. For example, if the connection between Nodes 1 and 2 were a tariffed **high-latency** (slow) line, we

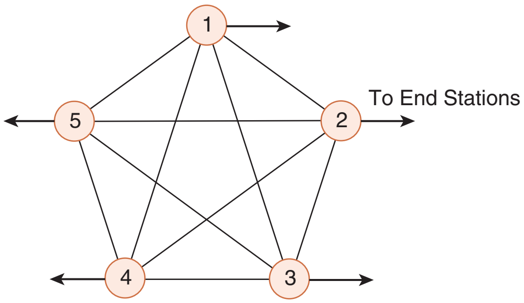


FIGURE 12.20 A Fully Connected Network

might be a whole lot better off using the 1 → 4 → 5 → 3 → 2 route. Clearly, in a real-world network, with hundreds of routers, the problem becomes enormous. If each router had to come up with the perfect outbound route for each incoming packet by considering all the possibilities, the packets would never get where they were going quickly enough to make anyone happy.

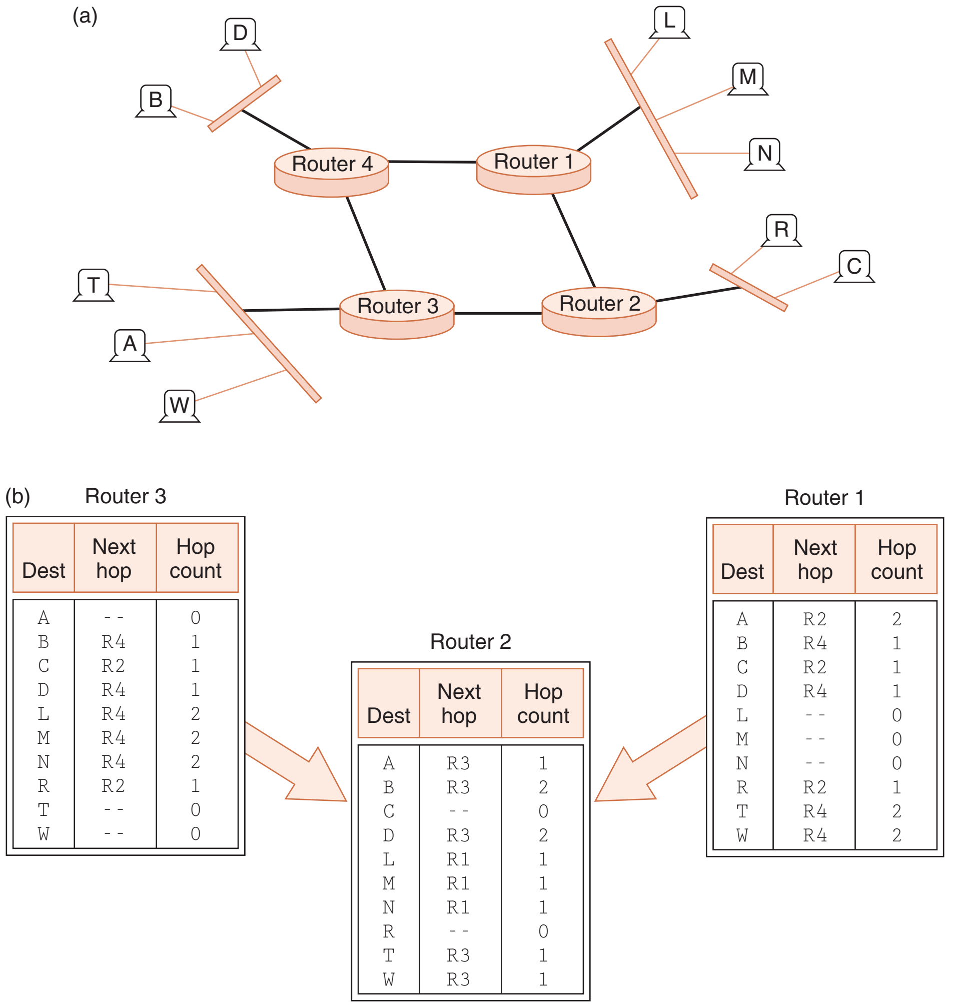
Of course, in a very stable network with only a few nodes, it is possible to program each router so that it always uses the same optimal route. This is called **static routing**, and it is feasible in networks where a large number of users in one location use a centralized host, or gateway, in another location. In the short term, this is an effective way to interconnect systems, but if a problem arises in one of the interconnecting links or routers, users are disconnected from the host. A human being must quickly respond to restore service. Static routing just isn't a reasonable option for networks that change frequently. This is to say that static routing isn't an option for most networks. Conversely, static networks are predictable, because the path (and thus the number of hops) a packet will take is always known and can be controlled. Static routing is also very stable, and it creates no routing protocol exchange traffic.

Dynamic routers automatically set up routes and respond to the changes in the network. These routers can also select an optimal route as well as a backup route should something happen to the route of choice. They do not change routing instructions, but instead allow for dynamic altering of routing tables.

Dynamic routers automatically explore their networks through information exchanges with other routers on the network. The information packets exchanged by the routers reveal their addresses and costs of getting from one point to another. Using this information, each router assembles a table of values in memory. This routing table is, in truth, a reachability list for every node on the network, plus some default values. Typically, each destination node is listed along with the neighboring, or **next-hop**, router to which it is connected.

When creating their tables, dynamic routers consider one of two metrics. They can use either the distance to travel between two nodes or the condition of the network in terms of measured latency. The algorithms using the first metric are **distance vector routing** algorithms. **Link state routing** algorithms use the second metric.

Distance vector routing is derived from a pair of similar algorithms invented in 1957 and 1962 known, respectively, as the **Bellman-Ford** and **Ford-Fulkerson** algorithms. The *distance* in distance vector routing is usually a measure of the number of nodes (hops) through which a packet must pass before reaching its destination, but any metric can be used. For example, suppose we have the network shown in Figure 12.21a. There are 4 routers and 10 nodes connected as indicated. If node B wants to send a packet to node L, there are two choices: One is B → Router 4 → Router 1 → L, with one hop between Router 4 and Router 1. The other routing choice has three hops between the routers: B → Router 4 → Router 3 → Router 2 → Router 1 → L. With distance vector routing, the objective is to always use the shortest route, so our B → Router 4 → Router 1 → L route is the obvious choice.



Router 3

Dest	Next hop	Hop count
A	--	0
B	R4	1
C	R2	1
D	R4	1
L	R4	2
M	R4	2
N	R4	2
R	R2	1
T	--	0
W	--	0

Router 2

Dest	Next hop	Hop count
A	R3	1
B	R3	2
C	--	0
D	R3	2
L	R1	1
M	R1	1
N	R1	1
R	--	0
T	R3	1
W	R3	1

Router 1

Dest	Next hop	Hop count
A	R2	2
B	R4	1
C	R2	1
D	R4	1
L	--	0
M	--	0
N	--	0
R	R2	1
T	R4	2
W	R4	2

FIGURE 12.21 (a) An Example of a Network with 4 Routers and 10 Nodes
(b) Routing Tables from Router 1 and Router 3 Are Used for Building the Routing Table for Router 2

In distance vector routing, every router needs to know the identities of each node connected to each router as well as the hop counts between them. To do this efficiently, routers exchange node and hop count information with their adjacent neighbors. For example, using the network shown in Figure 12.21a, Router 1 and Router 3 would have routing tables as shown in Figure 12.21b. These routing tables are then sent to Router 2. As shown in the figure, Router 2 selects the shortest path to any of the nodes considering all of the routes that are reported in the routing tables. The final routing table contains the addresses of nodes directly connected to Router 2 along with a list of destination nodes that are reachable through other routers and a hop count to those nodes. Notice that the hop counts in the final table for Router 2 are increased by 1 to account for the one hop between Router 2 and Router 1, and between Router 2 and Router 3. A real routing table would also contain a default router address that would be used for nodes that are not directly connected to the network, such as stations on a remote LAN or Internet destinations, for example.

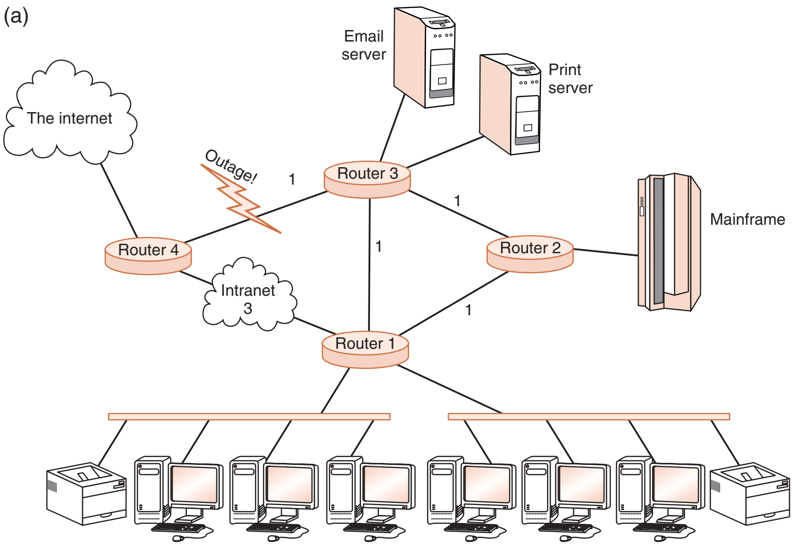
Distance vector routing is easy to implement, but it does have a few problems. For one thing, it can take a long time for the routing tables to stabilize (or **converge**) in a large network. Additionally, a considerable amount of traffic is placed on the network as the routing tables are updated. And third, obsolete routes can persist in the routing tables, causing misrouted or lost packets. This last problem is called the **count-to-infinity** problem.

You can understand the count-to-infinity problem by studying the network shown in Figure 12.22a. Notice that there are redundant paths through the network. Note also that the path through the intranet requires 3 hops. For example, if Router 3 goes offline, clients can still get to the mainframe and the Internet, but they won't be able to print anything until Router 3 is again operational.

The paths from all of the routers to the Internet are shown in Figure 12.22b. We call the time that this snapshot was taken $t = 0$. As you can see, Router 1 and Router 2 use Router 3 to get to the Internet. Sometime between $t = 0$ and $t = 1$, the link between Router 3 and Router 4 goes down (say someone unplugs the cable that connects these routers). At $t = 1$, Router 3 discovers this break, but has just received the routing table update from its neighbors, both of which **advertise** themselves as being able to get to the Internet in two hops. Router 3 then assumes that it can get to the Internet using one of these two routers and updates its table accordingly. It picks Router 1 as its next hop to the Internet. (Router 3 is one hop from Router 1, so the total hops from Router 3 to the Internet is 1 + 2 = 3.) Router 3 then sends its routing table to Router 1 and Router 2 at $t = 2$. At $t = 3$, Router 1 and Router 2 receive Router 3's updated hop count for getting to the Internet, so they add 1 to Router 3's value (because they know that Router 3 is one hop away) and subsequently broadcast their tables. This cycle continues until all of the routers end up with a hop count of infinity, meaning that the registers that hold the hop count eventually overflow, crashing the whole network.

Two methods are commonly used to prevent this situation. One is to use a small value for infinity, facilitating early problem detection (before a register overflows), and the other is to somehow prevent short cycles like the one that happened in our example.

(a)



(b)

From router	Next hop	Hop count
R1	R3	2
R2	R3	2
R3	R4	1
R4	--	0

Time *t* = 0

Routes to the Internet as seen by four routers.

(c)

From router	Next hop	Hop count
R1	R3	2
R2	R3	2
R3	Unreachable	
R4	--	0

Time *t* = 1

Router 3 (R3) detects broken link.

(d)

From router	Next hop	Hop count
R1	R3	2
R2	R3	2
R3	R1	3
R4	--	0

Time *t* = 2

R3 discovers route through R1 then advertises its route of 3 hops to get to the Internet.

(e)

From router	Next hop	Hop count
R1	R3	4
R2	R3	4
R3	R1	3
R4	--	0

Time *t* = 3

The other routers, knowing that they need R3 to get to the Internet, update their tables to reflect the new distance to the Internet as reported by R3.

(f)

From router	Next hop	Hop count
R1	R3	5
R2	R3	5
R3	R1	4
R4	--	0

Time *t* = 4

R3 now sees that the distance between R1 and the Internet is 5, so it adds 1 to R1's hop count to update its own route to the Internet.

(g)

From router	Next hop	Hop count
R1	R3	6
R2	R3	6
R3	R1	5
R4	--	0

Time *t* = 5

R1 and R2, upon receiving R3's route advertisement, update their tables with the updated hop count from R3 to the Internet. This process continues until the counters overflow.

FIGURE 12.22 (a) An Example of a Network with Redundant Paths
(b) The Routing that Router 1, Router 2, and Router 3 would use to get to the Internet
(c–g) Routing Table Updates for the Paths to the Internet

Sophisticated routers use a method called **split horizon** routing to keep short cycles out of the network. The idea is simple: No router will use a route given by its neighbor that includes itself in the route. (Similarly, the router could go ahead and use the self-referential route, but set the path value to infinity. This is called **split horizon with poison reverse**. The route is "poisoned" because it is marked as unreachable.) The routing table exchange for our example path to the Internet using split horizon routing would converge as shown in Figure 12.23. Of course, we still have the problem of larger cycles occurring. Say Router 1 points to Router 2, which points to Router 3, which points to Router 1. To some extent, this problem can be remedied if the routers exchange their tables only when a link needs to be updated. (These are called **triggered updates**.) Updates done in this manner cause fewer cycles in the routing graph and also reduce traffic on the network.

In large internetworks, hop counts can be a misleading metric, particularly when the network includes a variety of equipment and line speeds. For example, suppose a packet has two ways of getting somewhere. One path traverses six routers on a 100Mbps LAN, and the other traverses two routers on a 64Kbps leased line. Although the 100Mbps LAN could provide more than ten times the throughput, the hop count metric would force traffic onto the slower leased line. If instead of counting hops we measure the actual line latency, we could prevent such anomalies. This is the idea behind **link state routing**.

As with distance vector routing, link state routing is a self-managing system. Each router discovers the speed of the lines between itself and its neighboring routers by periodically sending out *Hello* packets. At the instant it releases the packet, the router starts a timer. Each router that subsequently receives the packet immediately dispatches a reply. Once the initiator gets a reply, it stops its timer and divides the result by 2, giving the one-way time

From router	Next hop	Hop count
R1	R3	2
R2	R3	2
R3	Unreachable	
R4	--	0

Time $t = 1$

R3 detects failed link on its path to the Internet.

From router	Next hop	Hop count
R1	R4	3
R2	R1	4
R3		∞
R4	--	0

Time $t = 2$

Because all other routes to the Internet include R3, it advertises its hop count to the Internet as infinity. Upon seeing this, the other routers find a shorter route.

From router	Next hop	Hop count
R1	R4	3
R2	R1	4
R3	R1	4
R4	--	0

Time $t = 3$

R3 now sees that its shortest route to the Internet is through R1 and updates its table accordingly.

FIGURE 12.23 Split Horizon with Poison Reverse Routing

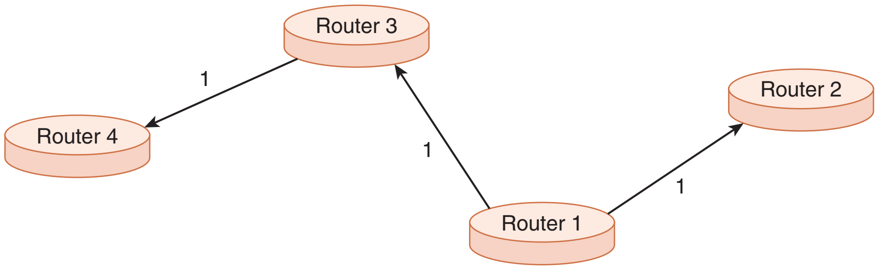


FIGURE 12.24 How Router 1 Sees the Network in Figure 12.22a Using Link State Routing and Dijkstra's Algorithm

estimate for the link to the router that replied to the packet. Once all the replies are received, the router assembles the timings into a table of link state values. This table is then broadcast to all other routers, except its adjacent neighbors. Nonadjacent routers then use this information to update all routes that include the sending router. Eventually, all routers within the routing domain end up with identical routing tables. Simply stated, after convergence takes place, a single snapshot of the network exists in the tables of each router. The routers then use this image to calculate the optimal path to every destination in its routing table.

In calculating optimal routes, each router is programmed to think of itself as the root node of a tree with every destination being an internal leaf node of the tree. Using this conceptualization, the router computes an optimal path to each destination using Dijkstra's algorithm.[1] Once found, the router stores only the next hop along the path. It doesn't store the entire path. The next (downstream) router should also have computed the same optimal path—or a better one by the time the packet gets there—so it would use the next link in the optimal path that was computed by its upstream predecessor. After Router 1 in Figure 12.22a has applied Dijkstra's algorithm, it sees the network as shown in Figure 12.24.

Clearly, routers can retain only a finite amount of information. Once a network gets to a size where performance starts to degrade (usually this happens for reasons other than routing table saturation), the network must be split into subnetworks, or **segments**. In very large networks, hierarchical topologies that involve a combination of switching and routing technologies are employed to help keep the system manageable. The best network designers know when each technology is called for in the system design. The ultimate aim is to maximize throughput while keeping the network manageable and robust.

[1] For an explanation of Dijkstra's algorithm, see Appendix A.

What Is a Firewall?

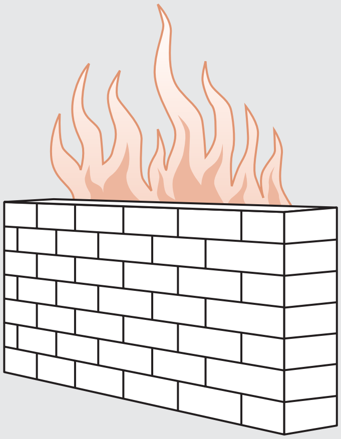
Virtually everyone in government, industry, and academia uses the Internet during the course of daily business. Yet the Internet invites everyone on board—even those persons who would plunder or destroy a company's computing resources. So how do you keep a network open enough for people to do their jobs but sufficiently secure to protect the assets of the business? The preferred solution to this problem is to place a firewall between the internal network and the Internet.

Firewalls get their name by drawing an analogy to the high brick walls that are sometimes placed between adjacent buildings. If a fire erupts in one of the buildings, the adjacent structure has some protection from becoming involved in the blaze. So it is with a network firewall: The internal users are partitioned from external users who may do harm to the internal network structure.

Firewalls come in many varieties. Two types that are most popular are router-based firewalls and host-based—or **proxy server**—firewalls. Both types are programmed with a rule base known as a **policy**. The firewall policy defines which network addresses can have access to which services. A good example of a policy involves file transfers. A firewall could be programmed to allow internal users (on the protected side of the network) to download files from the Internet. Users outside the protected network would be prohibited from downloading files from the internal network. The assumption is that data on the inside of the network may contain sensitive, private information. Any firewall can also be programmed with a list of forbidden addresses. (This is sometimes called a **blacklist**.) Blacklisted addresses often include the websites of groups disseminating objectionable material.

Both types of firewalls also distinguish between inbound and outbound traffic. This prevents the address spoofing that attempts to fool a firewall into thinking that a user is inside the network, when in fact the user is outside the network. If the firewall were fooled by a spoofed address, an external user would have free run of the internal network.

Both router-based firewalls and proxy servers have the ability to encrypt network traffic. **Encryption** is the process of scrambling a message using an algorithm and a key value so that the only device that can read the message is the device having the corresponding key. Key values are changed periodically, usually daily. This process happens automatically when firewalls are programmed with key exchange routines. Routers tend to use simpler encryption algorithms, usually based on simple bit shifts and logical ANDs using the message and the key value. (One such algorithm is



the U.S. federal **Data Encryption Standard [DES]**. For more security, the message is sometimes encrypted three times. This is called **Triple-DES**.)

As you might expect, proxy servers are slower and more prone to failure than router-based firewalls, but they also have many more features than router-based firewalls. First among these is their ability to act as an agent for users on the internal network (hence the name *proxy* server). These systems are usually equipped with two network cards, that is, they are **dual homed**. One network card connects to the internal network, and the other connects to the outside network. With this configuration, the server can completely mask the characteristics of the internal network from anyone on the outside. All that the external users can see is the address of the network interface that is connected to the outside.

Server-based firewalls can also maintain extensive network logs. Through these logs, security administrators can detect most invasion attempts by external evildoers. In some cases, logs can provide information regarding the source of a penetration attempt.

CHAPTER SUMMARY

This chapter has presented an overview of the network components and protocols that are used in building data communications systems. Each network component—each network process—carries out a task at some level within a layered protocol stack. Network engineers use layers of the OSI Reference Model to describe the roles and responsibilities of all network components. When a computer is engaged in communications with another computer, each layer of the protocol stack it is running converses with a corresponding layer running on the remote system. Protocol layers interface with their adjacent layers using service access points.

Most Internet applications rely on TCP/IP, which is by far the most widely deployed data communications protocol. Although often referred to as TCP/IP, this combination is actually two protocols. TCP provides a means for setting up a reliable communications stream on top of the unreliable IP. Version 4 of its IP component is constrained by its 32-bit address fields. Version 6 of IP will solve this problem because its address fields are 128 bits wide. With these larger address fields, routing could be a formidable task. With this in mind, the IETF has devised a hierarchical address scheme, the Aggregatable Global Unicast Address Format, which makes routing of packets both easier and faster.

We have described a number of components common to most data communications networks. The most important of these components are the physical media and the routers. Physical media must be chosen with consideration to the anticipated load and the distance to be covered. Physical media can be extended with repeaters when necessary. Routers are complex devices that monitor the state of the network. Their programming allows them to select nearly optimal paths for network traffic.

As the Internet continues its exponential growth as a vehicle for commerce, routing problems will grow proportionately. The solution to these problems may ultimately reside in rethinking the architecture and some of the assumptions that form the foundation of the Internet as we know it today.

FURTHER READING

There is no shortage of literature on the topic of computer networking. The challenge is in finding *good* networking material these days. Among the best data communications books available are those written by Tanenbaum (2010), Stallings (2013), and Kurose and Ross (2012). Following the OSI protocol stack in its organization, Tanenbaum's work is an easy-to-read introduction to most of the important concepts of data communications and networks. Kurose and Ross discuss most of the topics presented in this chapter with good detail and at a level that is accessible to most interested readers. The book by Stallings covers most of the same material as Tanenbaum's book, but with much more rigor and detail. Sherman (1990) also provides a well-written (but aging) introduction to data communications. The historical perspective that Sherman furnishes is most enjoyable.

The definitive source for information concerning Internet standards (requests for comment, or RFCs) is the Internet Engineering Task Force website at www.ietf.org. The RFCs relevant to material presented in this chapter are:

- RFC 791 "Internet Protocol Version 4 (IPv4)"
- RFC 793 "Transmission Control Protocol (TCP)"
- RFC 1180 "A TCP/IP Tutorial"
- RFC 1887 "An Architecture for IPv6 Unicast Address Allocation"
- RFC 2460 "Internet Protocol, Version 6 (IPv6) Specification"
- RFC 2026 "The Internet Standards Process"
- RFC 1925 "The Fundamental Truths of Networking"

IBM's TCP/IP tutorial Redbook by Rodriguez, Getrell, Karas, and Peschke (2001) is one of the most inexpensive and readable resources outside of the IETF. Unlike the IETF site, it also discusses ways in which a particular vendor's products implement TCP/IP (with no hype). Minoli and Schmidt (1999) discuss the Internet infrastructure, with a particular focus on quality-of-service issues.

Clark (1997) gives us a detailed and comprehensive account of telephone communications (centering in the UK). It relates important aspects of public telephone networks, including their ability to carry data traffic. Burd's (1997) ISDN and de Prycker's (1996) ATM books are both definitive accounts of their subjects.

The IBM (1995) Redbook on ATM, though less rigorous than de Prycker, provides excellent, objective detail concerning ATM's salient features.

For more information relevant to the Internet backbone router instability problem, see the papers by Labovitz, Malan, and Jahanian (1998, 1999). The University of Michigan maintains a website devoted to Internet performance issues. It can be found at www.merit.edu/ipma/.

The only way to keep abreast of the latest data networking technologies is to *constantly* read professional and trade periodicals. The most *avant-garde* information can be found in publications by the ACM and IEEE. Outstanding among these are *IEEE/ACM Transactions on Networking* and *IEEE Network*. Trade journals are another source of good information, particularly for understanding how various vendors are implementing the latest in networking technology. Two such magazines published by CMP are *Network Computing* (www.networkcomputing.com) and *Network Magazine* (www.networkmagazine.com). *Network World* is a weekly magazine published by CW Communications that not only provides an excellent print version, but its related website, www.nwfusion.com, teems with information and resources.

Many equipment vendors are gracious enough to post excellent, low-hype tutorial information on their websites. These sites include those by IBM, Cisco Systems, and Corning Glass. Certainly you will discover other great commercial sites as you explore specific technologies related to the topics presented in this chapter. It seems that one can never learn enough when it comes to data communications (no matter how hard one tries!).

REFERENCES

Burd, N. *The ISDN Subscriber Loop.* London: Chapman & Hall, 1997.

Clark, M. P. *Networks and Telecommunications: Design and Operation,* 2nd ed. Chichester, England: John Wiley & Sons, 1997.

Kurose, J. F., & Ross, K. W. *Computer Networking: A Top-Down Approach Featuring the Internet.* Boston, MA: Addison Wesley Longman, 2001.

Labovitz, C., Malan, G. R., & Jahanian, F. "Internet Routing Instability." *IEEE/ACM Transactions on Networking 6*:5, October 1998, pp. 515–528.

Labovitz, C., Malan, G. R., & Jahanian, F. "Origins of Internet Routing Instability." *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE 1,* 1999, pp. 218–226.

Liotta, A., "The Cognitive NET is Coming," *IEEE Spectrum 50*:8, August 2013, pp. 26–31.

Liu, W., Matthews, C., Parziale L., et al. *TCP/IP Tutorial and Technical Overview*, 8th ed. Armonk, NY: IBM Corporation, 2006.

Minoli, D., & Schmidt, A. *Internet Architectures.* New York: John Wiley & Sons, 1999.

Sherman, K. *Data Communications: A User's Guide*, 3rd ed. Englewood Cliffs, NJ: Prentice Hall, 1990.

Stallings, W. *Data and Computer Communications*, 10th ed. Upper Saddle River, NJ: Prentice Hall, 2013.

Tanenbaum, A. S. *Computer Networks*, 5th ed. Upper Saddle River, NJ: Prentice Hall, 2010.

REVIEW OF ESSENTIAL TERMS AND CONCEPTS

1. How is the organization of a polled network different from that of an internetwork?
2. What protocol device was the key to the robustness of DARPAnet?
3. Who establishes standards for the Internet?
4. What is the formal name given to Internet standards?
5. Which layer of the ISO/OSI Reference Model takes care of negotiating frame size and transmission speed?
6. If a communications session were to employ encryption or compression, which layer of the ISO/OSI Reference Model would perform this service?
7. According to the IPv4 format described in Section 12.5.1, what bit positions does the IP Protocol Number occupy? What is the purpose of this field?
8. Why have certain types of IP addresses become scarce?
9. Explain the general purpose of the TCP protocol.
10. How does IPv6 improve upon IPv4?
11. What is the difference between guided and unguided data transmission media? List some examples of each.
12. What determines the quality of a transmission medium? What metric is used?
13. What are the principal causes of attenuation? What can help reduce it?
14. What is the difference between the baud rate and the bit rate of a line?
15. What are the three types of fiber-optic cable? Which of these can transmit signals the fastest?
16. Where does one find a MAC address? How many bytes are in a MAC address?
17. Briefly describe how repeaters, hubs, switches, and routers differ from one another.
18. What is the difference between a bridge and a gateway? Which one is faster and why?
19. When is it not a very good idea to use static routing?
20. Give two important ways in which link state routing differs from distance vector routing.
21. What are the three main problems that arise from distance vector routing?
22. In what ways does a firewall provide security?
23. What are SCADA systems?
24. In what ways is the Internet threatened?

EXERCISES

1. In what way is the traffic of an early business computer network different from that of an early scientific-academic network? Is there such a distinction between these two types of systems today?
2. Why is the ISO/OSI protocol stack called a reference model? Do you think this will always be the case?

3. How is a Network layer protocol different from a Transport layer protocol?

4. Internet protocol standards are devised through the efforts of thousands of people all over the world—regardless of their having any particular background in data communications. On the other hand, proprietary protocols are created by a much smaller group of people, all of whom are directly or indirectly working for the same employer.

a) What advantages and disadvantages do you think are offered by each approach? Which would produce a better product? Which would produce a product more quickly?

b) Why do you think that the IETF approach has achieved ascendancy over the proprietary approach?

◆**5.** In our description of the Window field in the TCP header, we said:

Notice that if the receiver's application is running very slowly, say it's pulling data 1 or 2 bytes at a time from its buffer, the TCP process running at the receiver should wait until the application buffer is empty enough to justify sending another segment.

What is the "justification" for sending another segment?

6. The OSI protocol stack includes Session and Presentation layers in addition to its Application layer. TCP/IP applications, such as Telnet and FTP, have no such separate layers defined. Do you think that such a separation should be made? Give some advantages and disadvantages of incorporating the OSI approach into TCP/IP.

7. Why is the length of a TCP segment limited to 65,515 bytes? (Hint: Look at the definition of the Data Offset field of the TCP segment format.)

8. Why does the IETF use the word *octet* instead of *byte*? Do you think this practice should continue?

◆**9.** Into which class of networks do the following IP addresses fall?

◆**a)** 180.265.14.3

◆**b)** 218.193.149.222

◆**c)** 92.146.292.7

10. Into which class of networks do the following IP addresses fall?

a) 223.52.176.62

b) 127.255.255.2

c) 191.57.229.163

◆**11.** A station running TCP/IP needs to transfer a file to a host. The file contains 1,024 bytes. How many bytes, including all of the TCP/IP overhead, would be sent, assuming a payload size of 128 bytes and that both systems are running IPv4? (Also assume that the three-way handshake and window size negotiation have been completed and that no errors occur during transmission.)

◆**a)** What is the protocol overhead (stated as a percentage)?

◆**b)** Perform the same calculation, this time assuming that both clients are using IPv6.

12. A station running TCP/IP needs to transfer a file to a host. The file contains 2,048 bytes. How many bytes, including all of the TCP/IP overhead, would be sent, assuming a payload size of 512 bytes and that both systems are running IPv4? (Also assume that the three-way handshake and window size negotiation have been completed and that no errors occur during transmission.)

a) What is the protocol overhead (stated as a percentage)?

b) Perform the same calculation, this time assuming that both clients are using IPv6.

◆**13.** Two stations running TCP/IP are engaged in transferring a file. This file is 100KB long, the payload size is 100 bytes, and the negotiated window size is 300 bytes. The sender receives an ACK 1,500 from the receiver.

◆**a)** Which bytes will be sent next?

◆**b)** What is the last byte number that can be sent without an ACK being sent by the receiver?

14. Two stations running TCP/IP are engaged in transferring a file. This file is 10KB long, the payload size is 100 bytes, and the negotiated window size is 2,000 bytes. The sender receives an ACK 900 from the receiver.

a) Which bytes will be sent next?

b) What is the last byte number that can be sent without an ACK being sent by the receiver?

15. What problems would present themselves if TCP did not allow senders and receivers to negotiate a timeout window?

16. IP is a connectionless protocol, whereas TCP is connection-oriented. How can these two protocols coexist in the same protocol stack?

17. Section 12.6.1 states that when using 4B/5B encoding, a signal-carrying capacity of 125MHz is required for a transmission medium to have a bit rate of 100Mbps.

a) What signal-carrying capacity would be required if Manchester coding were used instead?

b) What signal-carrying capacity would be required if modified frequency modulation (MFM) coding were used, assuming that the occurrence of a 0 and the occurance of a 1 are equally likely events?

(Manchester and MFM coding are explained in Chapter 2, Section 2.A.)

18. **a)** The signal power for a particular class of network wiring is 8,733.26dB, and the noise rating at that particular signal strength at 100MHz is 41.8dB. Find the signal-to-noise ratio for this conductor.

b) Suppose the noise rating for the network wiring in part a is 9.5dB and the noise rating is 36.9dB when a 200MHz signal is transmitted. What is the signal strength?

◆**19.** ◆**a)** The signal power for a particular class of network wiring is 2,898dB, and the noise rating at that particular signal strength at 100MHz is 40dB. Find the signal-to-noise ratio for this conductor.

◆**b)** Suppose the noise rating for the network wiring in part a is 0.32dB and the noise rating is 35dB when a 200MHz signal is transmitted. What is the signal strength?

20. How big is a physical PDU? The answer to this question determines the number of simultaneous transmissions for many network architectures.

If a signal propagates through copper wire at a rate of 2×10^8m/s, then on a carrier running at 10Mbps the length of each bit pulse is given by:

$$\frac{\text{Speed of propagation}}{\text{Speed of bus}} = \frac{2 \times 10^8 \text{ m/s}}{10 \times 10^6 \text{ b/s}} = 20\text{m/bit}$$

If a data frame is 512 bits long, then the entire frame occupies:

$$(\text{Length of one bit}) \times (\text{Frame size}) = 20 \times 512 = 10{,}240 \text{ meters.}$$

a) How big is a 1,024-bit packet if the network runs at 100Mbps?

b) How big is it if the network speed is increased to 155Mbps?

c) At 100Mbps, how much time elapses as one of these frames passes a particular point in the network?

21. It looks like the 4B/5B bit cells in Figure 12.14 are fairly small. How long, in reality, is such a bit cell on a 125MHz line? (Use the constants and formulas from the previous question.)

22. With reference to Figure 12.21, suppose Router 4 derives its routing table from the routing tables of Router 1 and Router 3. Complete the routing table for Router 4 using the same format as the routing table of the other three routers.