5

Overview Of Data Communications

5.1 Introduction

The first part of the text discusses network programming and reviews Internet applications. The chapter on socket programming explains the API that operating systems provide to application software, and shows that a programmer can create applications that use the Internet without understanding the underlying mechanisms. In the remainder of the text, we will learn about the complex protocols and technologies that support communication, and see that understanding the complexity can help programmers write better code.

This part of the text explores the transmission of information across physical media, such as wires, optical fibers, and radio waves. We will see that although the details vary, basic ideas about information and communication apply to all forms of transmission. We will understand that data communications provides conceptual and analytical tools that offer a unified explanation of how communication systems operate. More important, data communications tells us what transfers are theoretically possible as well as how the reality of the physical world limits practical transmission systems.

This chapter provides an overview of data communications and explains how the conceptual pieces form a complete communication system. Successive chapters each explain one concept in detail.

5.2 The Essence Of Data Communications

What does data communications entail? As Figure 5.1 illustrates, the subject is an interesting combination of ideas and approaches from three disciplines.

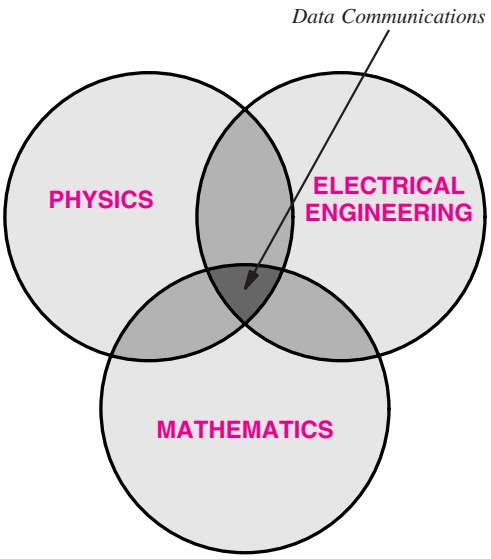


Figure 5.1 The subject of data communications lies at the intersection of Physics, Mathematics, and Electrical Engineering.

Because it involves the transmission of information over physical media, data communications touches on physics. The subject draws on ideas about electric current, light, and other forms of electro-magnetic radiation. Because information is digitized and digital data is transmitted, data communications uses mathematics and includes various forms of analysis. Finally, because the ultimate goal is to develop practical ways to design and build transmission systems, data communications focuses on developing techniques that electrical engineers can use. The point is:

> *Although it includes concepts from physics and mathematics, data communications does not merely offer abstract theories. Instead, data communications provides a foundation that is used to construct practical communication systems.*

5.3 Motivation And Scope Of The Subject

Three main ideas provide much of the motivation for data communications and help define the scope.

- The sources of information can be of arbitrary types
- Transmission uses a physical system
- Multiple sources of information can share the underlying medium

The first point is especially relevant considering the popularity of multimedia applications: information is not restricted to bits that have been stored in a computer. Instead, information can also be derived from the physical world, including audio and video. Thus, it is important to understand the possible sources and forms of information and the ways that one form can be transformed into another.

The second point suggests that we must use natural phenomena, such as electricity and electromagnetic radiation, to transmit information. Thus, it is important to understand the types of media that are available and the properties of each. Furthermore, we must understand how physical phenomena can be used to transmit information over each medium, and the relationship between data communications and the underlying transmission. Finally, we must understand the limits of physical systems, the problems that can arise during transmission, and techniques that can be used to detect or solve the problems.

The third point suggests that sharing is fundamental. Indeed, we will see that sharing plays a fundamental role in most computer networks. That is, a network usually permits multiple pairs of communicating entities to communicate over a given physical medium. Thus, it is important to understand the possible ways underlying facilities can be shared, the advantages and disadvantages of each, and the resulting modes of communication.

5.4 The Conceptual Pieces Of A Communication System

To understand data communications, imagine a working communication system that accommodates multiple sources of information, and allows each source to send to a separate destination. It may seem that communication in such a system is straightforward. Each source needs a mechanism to gather the information, prepare the information for transmission, and transmit the information across the shared physical medium. Similarly, a mechanism is needed that extracts the information for the destination and delivers the information. Figure 5.2 illustrates the simplistic view.

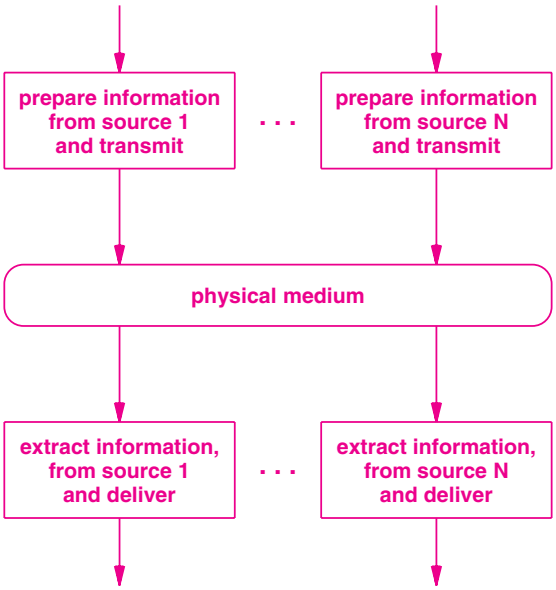


Figure 5.2 A simplistic view of data communications with a set of sources sending to a set of destinations across a shared medium.

In practice, data communications is much more complex than the simplistic diagram in Figure 5.2 suggests. Because information can arrive from many types of sources, the techniques used to handle sources vary. Before it can be sent, information must be digitized, and extra data must be added to protect against errors. If privacy is a concern, the information may need to be encrypted. To send multiple streams of information across a shared communication mechanism, the information from each source must be identified, and data from all the sources must be intermixed for transmission. Thus, a mechanism is needed to identify each source, and guarantee that the information from one source is not inadvertently confused with information from another source.

To explain the major aspects of data communications, engineers have derived a conceptual framework that shows how each subtopic fits into a communication system. The idea is that each item in the framework can be studied independently, and once all pieces have been examined, the entire subject will be understood. Figure 5.3 illustrates the framework, and shows how the conceptual aspects fit into the overall organization of a communication system.

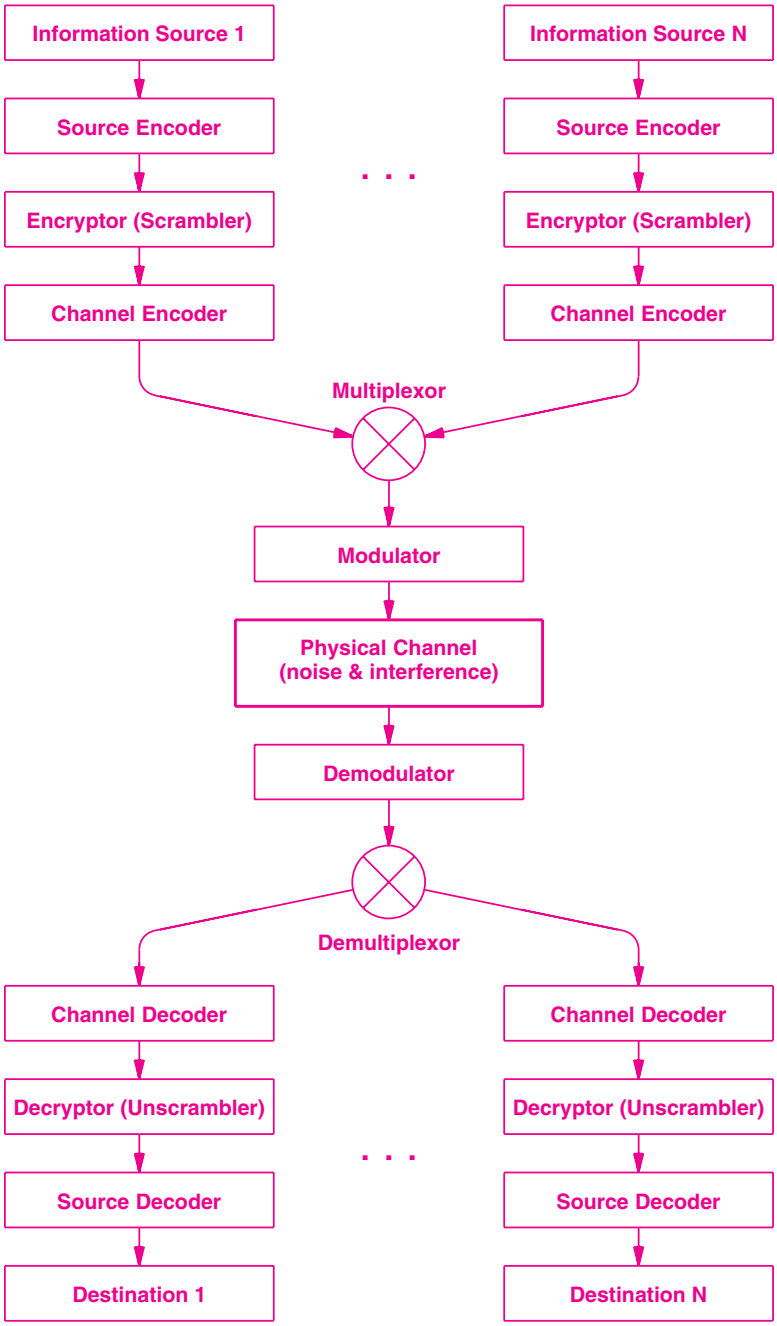


Figure 5.3 A conceptual framework for a data communications system. Multiple sources send to multiple destinations through an underlying physical channel.

5.5 The Subtopics Of Data Communications

Each of the boxes in Figure 5.3 corresponds to one subtopic of data communications. The following paragraphs explain the terminology. Successive chapters each examine one of the conceptual subtopics.

- *Information Sources.* The source of information can be either analog or digital. Important concepts include characteristics of signals, such as amplitude, frequency, and phase, and classification as either periodic or aperiodic. In addition, the subtopic focuses on the conversion between analog and digital representations of information.

- *Source Encoder and Decoder.* Once information has been digitized, digital representations can be transformed and converted. Important concepts include data compression and consequences for communications.

- *Encryptor and Decryptor.* To protect information and keep it private, the information can be encrypted (i.e., scrambled) before transmission and decrypted upon reception. Important concepts include cryptographic techniques and algorithms.

- *Channel Encoder and Decoder.* Channel coding is used to detect and correct transmission errors. Important topics include methods to detect and limit errors, and practical techniques like parity checking, checksums, and cyclic redundancy codes that are employed in computer networks.

- *Multiplexor and Demultiplexor.* Multiplexing refers to the way information from multiple sources is combined for transmission across a shared medium. Important concepts include techniques for simultaneous sharing as well techniques that allow sources to take turns when using the medium.

- *Modulator and Demodulator.* Modulation refers to the way electromagnetic radiation is used to send information. Concepts include both analog and digital modulation schemes, and devices known as modems that perform the modulation and demodulation.

- *Physical Channel and Transmission.* The subtopic includes transmission media and transmission modes. Important concepts include bandwidth, electrical noise and interference, and channel capacity, as well as transmission modes, such as serial and parallel.

6.10 Digital Signals And Signal Levels

We said in addition to being represented by an analog signal, information can also be represented by a *digital* signal. We further defined a signal to be digital if a fixed set of valid levels has been chosen and at any time, the signal is at one of the valid levels. Some systems use voltage to represent digital values by making a positive voltage correspond to a logical one, and zero voltage correspond to a logical zero. For example, +5 volts can be used for a logical one and 0 volts for a logical zero.

If only two levels of voltage are used, each level corresponds to one data bit (0 or 1). However, some physical transmission mechanisms can support more than two signal levels. When multiple digital levels are available, each level can represent multiple bits. For example, consider a system that uses four levels of voltage: -5 volts, -2 volts, +2 volts, and +5 volts. Each level can correspond to two bits of data as Figure 6.8 illustrates.

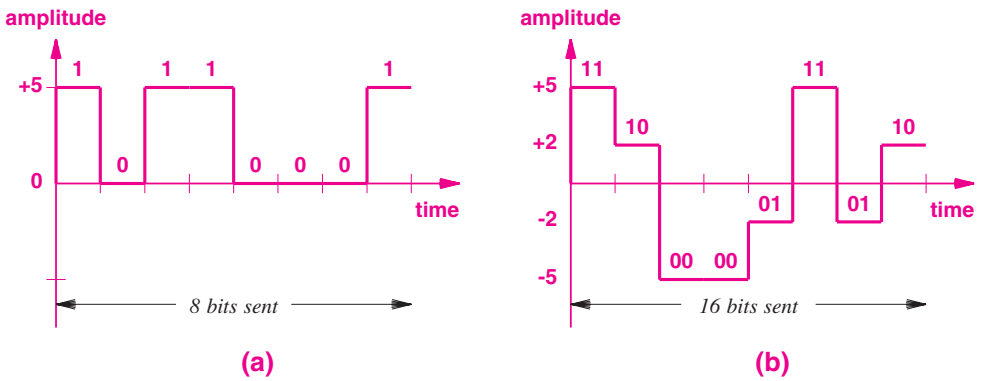


Figure 6.8 (a) A digital signal using two levels, and (b) a digital signal using four levels.

As the figure illustrates, the chief advantage of using multiple signal levels arises from the ability to represent more than one bit at a time. In Figure 6.8b, for example, -5 volts represents the two-bit sequence *00*, -2 volts represents *01*, +2 volts represents *10*, and +5 volts represents *11*. Because multiple levels of signal are used, each time slot can transfer two bits, which means that the four-level representation in Figure 6.8b sends twice as many bits per unit time as the two-level representation in Figure 6.8a.

The relationship between the number of levels required and the number of bits to be sent is straightforward. There must be a signal level for each possible combination of bits. Because 2^n combinations are possible with n bits, a communication system must use 2^n levels to represent n bits. To summarize:

A communication system that uses two signal levels can only send one bit at a given time; a system that supports 2^n signal levels can send n bits at a time.

It may seem that voltage is an arbitrary quantity, and that one could achieve arbitrary numbers of levels by dividing voltage into arbitrarily small increments. Mathematically, one could create a million levels between 0 and 1 volts merely by using 0.0000001 volts for one level, 0.0000002 for the next level, and so on. Unfortunately, practical electronic systems cannot distinguish between signals that differ by arbitrarily small amounts. Thus, practical systems are restricted to a few signal levels.

6.11 Baud And Bits Per Second

How much data can be sent in a given time? The answer depends on two aspects of the communication system. As we have seen, the rate at which data can be sent depends on the number of signal levels. A second factor is also important: the amount of time the system remains at a given level before moving to the next. For example, the diagram in Figure 6.8a shows time along the x-axis, and the time is divided into eight segments, with one bit being sent during each segment. If the communication system is modified to use half as much time for a given bit, twice as many bits will be sent in the same amount of time. The point is:

An alternative method of increasing the amount of data that can be transferred in a given time consists of decreasing the amount of time that the system leaves a signal at a given level.

As with signal levels, the hardware in a practical system places limits on how short the time can be — if the signal does not remain at a given level long enough, the receiving hardware will fail to detect it. Interestingly, the accepted measure of a communication system does not specify a length of time. Instead, engineers measure the inverse: how many times the signal can change per second, which is defined as the *baud*. For example, if a system requires the signal to remain at a given level for .001 seconds, we say that the system operates at 1000 baud.

The key idea is that both baud and the number of signal levels control the bit rate. If a system with two signal levels operates at 1000 baud, the system can transfer exactly 1000 bits per second. However, if a system that operates at 1000 baud has four signal levels, the system can transfer 2000 bits per second (because four signal levels can represent two bits). Equation 6.1 expresses the relationship between baud, signal levels, and bit rate.

$$\textit{bits per second} = \textit{baud} \times \left\lfloor \log_2(\textit{levels}) \right\rfloor \tag{6.1}$$

7.3 A Taxonomy By Forms Of Energy

Figure 7.1 illustrates how physical media can be classified according to the form of energy used to transmit data. Successive sections describe each of the media types.

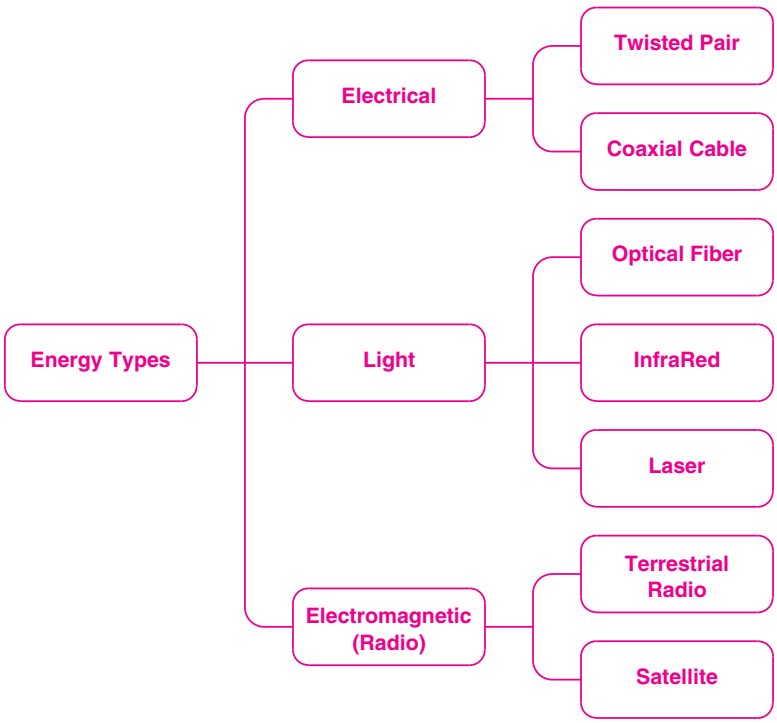


Figure 7.1 A taxonomy of media types according to the form of energy used.

Like most taxonomies, the categories are not perfect, and exceptions exist. For example, a space station in orbit around the earth might employ non-terrestrial communication that does not involve a satellite. Nevertheless, our taxonomy covers most communications.

7.4 Background Radiation And Electrical Noise

Recall from basic physics that electrical current flows along a complete circuit. Thus, all transmissions of electrical energy need two wires to form a circuit — a wire to the receiver and a wire back to the sender. The simplest form of wiring consists of a cable that contains two copper wires. Each wire is wrapped in a plastic coating, which insulates the wires electrically. The outer coating on the cable holds related wires together to make it easier for humans who connect equipment.

Computer networks use an alternative form of wiring. To understand why, one must know three facts.

1. Random electromagnetic radiation, called *noise*, permeates the environment. In fact, communication systems generate minor amounts of electrical noise as a side-effect of normal operation.
2. When it hits metal, electromagnetic radiation induces a small signal, which means that random noise can interfere with signals used for communication.
3. Because it absorbs radiation, metal acts as a *shield*. Thus, placing enough metal between a source of noise and a communication medium can prevent noise from interfering with communication.

The first two facts outline a fundamental problem inherent in communication media that use electrical or radio energy. The problem is especially severe near a source that emits random radiation. For example, florescent light bulbs and electric motors both emit radiation, especially powerful motors such as those used to operate elevators, air conditioners, and refrigerators. Surprisingly, smaller devices such as paper shredders or electric power tools can also emit enough radiation to interfere with communication. The point is:

The random electromagnetic radiation generated by devices such as electric motors can interfere with communication that uses radio transmission or electrical energy sent over wires.

8.6 An Example Block Error Code: Single Parity Checking

To understand how additional information can be used to detect errors, consider a *single parity checking (SPC) mechanism*. One form of SPC defines a block to be an 8-bit unit of data (i.e., a single *byte*). On the sending side, an encoder adds an extra bit, called a *parity bit* to each byte before transmission; a receiver removes the parity bit and uses it to check whether bits in the byte are correct.

Before parity can be used, the sender and receiver must be configured for either *even parity* or *odd parity*. When using even parity, the sender chooses a parity bit of 0 if the byte has an even number of 1 bits, and 1 if the byte has an odd number of 1 bits. The way to remember the definition is: even or odd parity specifies whether the 9 bits sent across a channel have an even or odd number of 1 bits. Figure 8.4 lists examples of data bytes and the value of the parity bit that is sent when using even or odd parity.

To summarize:

> *Single parity checking (SPC) is a basic form of channel coding in which a sender adds an extra bit to each byte to make an even (or odd) number of 1 bits and a receiver verifies that the incoming data has the correct number of 1 bits.*

Original Data	Even Parity	Odd Parity
0 0 0 0 0 0 0 0	0	1
0 1 0 1 1 0 1 1	1	0
0 1 0 1 0 1 0 1	0	1
1 1 1 1 1 1 1 1	0	1
1 0 0 0 0 0 0 0	1	0
0 1 0 0 1 0 0 1	1	0

Figure 8.4 Data bytes and the corresponding value of a single parity bit when using even parity or odd parity.

Single parity checking is a weak form of channel coding that can detect errors, but cannot correct them. Furthermore, parity mechanisms can only handle errors where an odd number of bits are changed. If one of the nine bits (including the parity bit) is changed during transmission, the receiver will declare that the incoming byte is invalid.

13.8 LAN Topologies

Because many LAN technologies have been invented, it is important to know how specific technologies are similar and how they differ. To help understand similarities, each network is classified into a category according to its *topology* or general shape. This section describes four basic topologies that are used to construct LANs; a later chapter discusses specific technologies. Figure 13.7 illustrates the topologies.

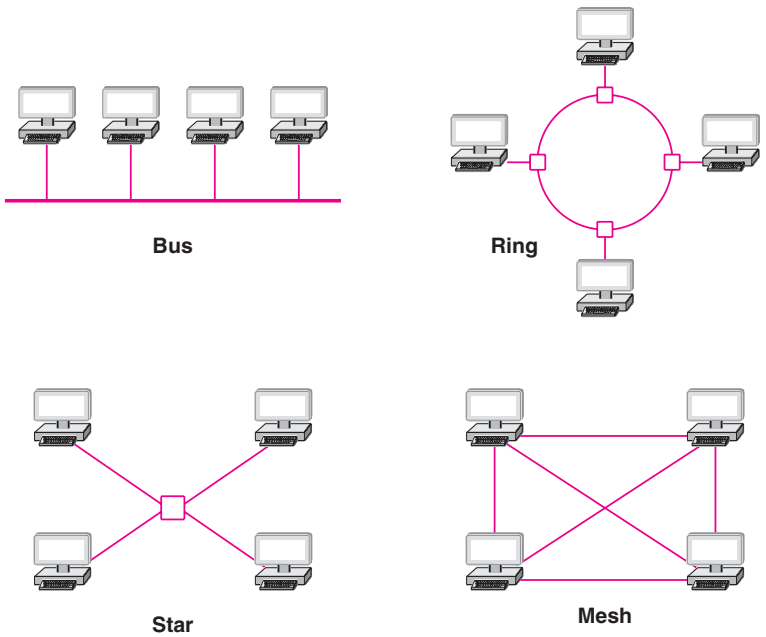


Figure 13.7 Four network topologies used with LANs.

13.8.1 Bus Topology

A network that uses a *bus topology* usually consists of a single cable to which computers attach†. Any computer attached to a bus can send a signal down the cable, and all computers receive the signal. Because all computers attach directly to the cable, any computer can send data to any other computer. Of course, the computers attached to a bus network must coordinate to ensure that only one computer sends a signal at any time.

13.8.2 Ring Topology

A network that uses a *ring topology* arranges for computers to be connected in a closed loop — a cable connects the first computer to a second computer, another cable connects the second computer to a third, and so on, until a cable connects the final computer back to the first. Some technologies that use a ring topology require a computer to connect to a small device that forms the ring. The advantage of using a separate device lies in the ability of the ring to continue operation even if some of the computers are disconnected. The name *ring* arises because one can imagine the computers and the cables connecting them arranged in a circle as Figure 13.7 illustrates. In practice, the cables in a ring network do not form a circle. Instead, they run along hallways or rise vertically from one floor of a building to another.

13.8.3 Mesh Topology

A network that uses a *mesh topology* provides a direct connection between each pair of computers. The chief disadvantage of a mesh arises from the cost: a mesh network connecting n computers requires:

$$\textit{connections in a mesh network} = \frac{n!}{(n-2)!\,2!} = \frac{n^2 - n}{2} \tag{13.1}$$

The important point is that the number of connections needed for a mesh network grows faster than the number of computers. Because connections are expensive, few LANs employ a mesh topology.

13.8.4 Star Topology

A network uses a *star topology* when all computers attach to a central point. Because a star-shaped network resembles the spokes of a wheel, the center of a star network is often called a *hub*. A typical hub consists of an electronic device that accepts data from a sending computer and delivers it to the appropriate destination.

In practice, star networks seldom have a symmetric shape in which the hub is located an equal distance from all computers. Instead, a hub often resides in a location

†In practice, the ends of a bus network must be terminated to prevent electrical signals from reflecting back along the bus.

separate from the computers attached to it. For example, computers can reside in individual offices, while the hub resides in a location accessible to an organization's networking staff.

13.8.5 The Reason For Multiple Topologies

Each topology has advantages and disadvantages. A ring topology makes it easy for computers to coordinate access and to detect whether the network is operating correctly. However, an entire ring network is disabled if one of the cables is cut. A star topology helps protect the network from damage to a single cable because each cable connects only one machine. A bus requires fewer wires than a star, but has the same disadvantage as a ring: a network is disabled if someone accidentally cuts the main cable. Later chapters that describe specific network technologies provide additional details about differences. For now, it is sufficient to understand:

Networks are classified into broad categories according to their general shape. Although a mesh topology is possible, the primary topologies used with LANs are star, ring, and bus; each has advantages and disadvantages.

13.9 Packet Identification, Demultiplexing, MAC Addresses

In addition to standards that specify the details of various LAN technologies, IEEE has created a standard for *addressing*. To understand addressing, consider packets traversing a shared medium as Figure 13.2 illustrates†. In the simplest case, each packet that travels across the shared medium is intended for a specific recipient, and only the intended recipient should process the packet. In packet switching systems, demultiplexing uses an identifier known as an *address*. Each computer is assigned a unique address, and each packet contains the address of the intended recipient.

In the IEEE addressing scheme, each address consists of 48 bits. IEEE uses the term *Media Access Control address* (*MAC address*). Because 48-bit addresses originated with Ethernet technology, networking professionals often use the term *Ethernet address*. To guarantee that each address is unique, IEEE allocates an address for each piece of network interface hardware. Thus, if a consumer purchases a *Network Interface Card* (*NIC*) for their PC, the NIC contains a unique IEEE address assigned when the device was manufactured.

Rather than assign individual addresses, IEEE assigns a block of addresses to each equipment vendor, and allows the vendor to assign a unique value to each device they manufacture. Thus, a 48-bit address is divided into a 3-byte *Organizationally Unique ID* (*OUI*) that identifies the equipment vendor and a 3-byte block that identifies a particular *Network Interface Controller* (*NIC*). Figure 13.8 illustrates the division.

†Figure 13.2 can be found on page 223.

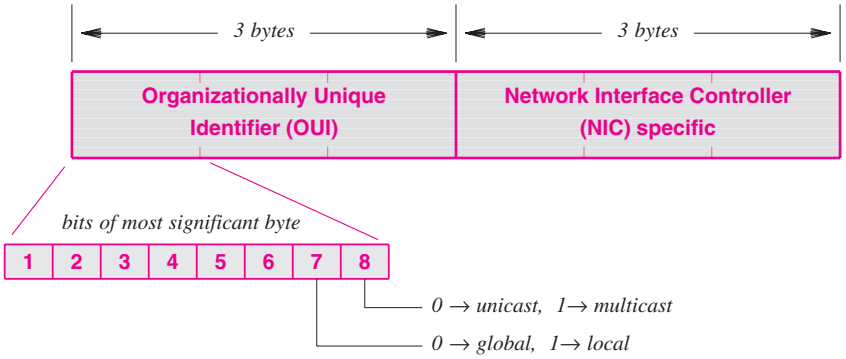


Figure 13.8 The division of a 48-bit IEEE MAC address.

Interestingly, the two low-order bits of the most significant byte of the OUI are assigned a special meaning as the figure indicates. The least significant bit of the most significant byte is a *multicast* bit that specifies whether the address is *unicast* (0) or *multicast* (1), and the next bit specifies whether the OUI is globally unique (0) or locally assigned (1). The next section explains multicast. Globally unique addresses are assigned by the IEEE; locally assigned addresses are available for experimental work or for organizations that desire to create their own address space.

13.12 Frames And Framing

Chapter 9 introduces the concept of framing in the context of synchronous communication systems as a mechanism that allows a receiver to know where a message begins and ends. In a more general sense, we use the term *framing* to refer to the structure added to a sequence of bits or bytes that allows a sender and receiver to agree on the exact format of the message. In a packet-switched network, each *frame* corresponds to a packet. A frame consists of two conceptual parts:

- Header that contains metadata, such as an address
- Payload that contains the data being sent

A frame *header* contains information used to process the frame. In particular, a header usually contains an address that specifies the intended recipient. The *payload* area contains the message being sent, and is usually much larger than the frame header. In most network technologies, the message is *opaque* in the sense that the network only examines the frame header. Thus, the payload can contain an arbitrary sequence of bytes that are only meaningful to the sender and receiver.

A frame is usually arranged so the header is transmitted before the payload, which allows a receiver to begin processing the frame as the bits arrive. Some technologies delineate each frame by sending a short prelude before the frame and a short postlude after the frame. Figure 13.10 illustrates the concept.

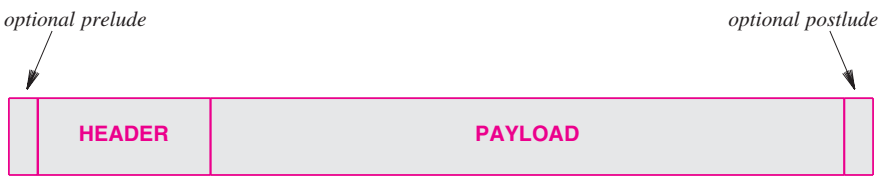


Figure 13.10 Typical structure of a frame in a packet-switched network.

To understand how framing works, consider an example using bytes. That is, suppose a data communication mechanism can transfer an arbitrary 8-bit byte from a sender to a receiver, and imagine that the mechanism is used to send packets. Assume that a packet

header consists of 6 bytes and the payload consists of an arbitrary number of bytes. We will use a single byte to mark the start of a frame, and a single byte to mark the end of a frame. In the ASCII character set, the *Start Of Header (SOH)* character marks the beginning of a frame, and the *End Of Transmission* (*EOT*) character marks the end of a frame. Figure 13.11 illustrates the format.

Figure 13.11 An example frame format that uses SOH and EOT characters to delineate a frame.

The example format appears to have unnecessary overhead. To understand why, consider what happens when a sender transmits two frames with no delay between them. At the end of the first frame, the sender transmits EOT, and then with no delay, the sender transmits SOH to start the second frame. In such circumstances, only one character is needed to separate two blocks of data — a framing scheme that delimits both the beginning and end of each frame appears to send an extra, unnecessary character between frames.

The advantage of sending a character at the end of a frame becomes clear when one considers that packet transmission is asynchronous and that errors can occur. For asynchronous communication, using an EOT to mark the end of a frame allows a receiver to process the frame without waiting for the start of a successive frame. In the case of an error, using SOH and EOT to bracket the frame helps with recovery and synchronization — if a sender crashes during transmission of a frame, a receiver will be able to determine that a partial frame arrived.

13.13 Byte And Bit Stuffing

In the ASCII character set, SOH has hexadecimal value 201 and EOT has the hexadecimal value 204. The question arises: what happens if the payload of a frame includes one or more bytes with value 201 or 204? The answer lies in a technique known as *byte stuffing* that allows transmission of arbitrary data without confusion.

In general, to distinguish between data and control information, such as frame delimiters, a sender changes the data to replace each control byte with a sequence and the receiver replaces the sequence with the original value. As a result, a frame can transfer arbitrary data and the underlying system never confuses data with control information. The technique is known as *byte stuffing*; the terms *data stuffing* and *character stuffing* are sometimes used. A related technique used with systems that transfer a bit stream is known as *bit stuffing*.

As an example of byte stuffing, consider a frame as illustrated in Figure 13.11. Because SOH and EOT are used to delimit the frame, those two bytes must not appear in the payload. Byte stuffing solves the problem by reserving a third character to mark occurrences of reserved characters in the data. For example, suppose the ASCII character ESC (hexadecimal value 1B) has been selected as the third character. When any of the three special characters occur in the data, the sender replaces the character with a two-character sequence. Figure 13.12 lists one possible mapping.

Byte In Payload	Sequence Sent
SOH	ESC A
EOT	ESC B
ESC	ESC C

Figure 13.12 An example of byte stuffing that maps each special character into a 2-character sequence.

As the figure specifies, the sender replaces each occurrence of SOH by the two characters ESC and A, each occurrence of EOT by the characters ESC and B, and each occurrence of ESC by the two characters ESC and C. A receiver reverses the mapping by looking for ESC followed by one of A, B, or C and replacing the 2-character combination with the appropriate single character. Figure 13.13 shows an example payload and the same payload after byte stuffing has occurred. Note that once byte stuffing has been performed, neither SOH nor EOT appears anywhere in the payload.

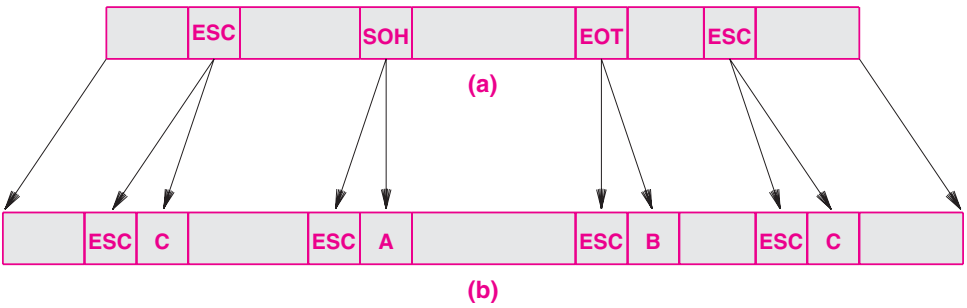


Figure 13.13 Illustration of (a) original data, and (b) a version after byte-stuffing has been performed.

14.5.3 Token Passing

Token passing has been used in several LAN technologies, and is most often associated with ring topologies†. To understand token passing, imagine a set of computers connected in a ring, and imagine that at any instant, exactly one of the computers has received a special control message called a *token*. To control access, each computer follows Algorithm 14.3

Algorithm 14.3

```
Purpose:
    Control transmission of packets through token passing
Method:
    Each computer on the network repeats {
        Wait for the token to arrive;
        Transmit a packet if one is waiting to be sent;
        Send the token to the next station;
    }
```

Algorithm 14.3 Controlled access through token passing.

In a token passing system, when no station has any packets to send, the token circulates among all stations continuously. For a ring topology, the order of circulation is defined by the ring. That is, if a ring is arranged to send messages in a clockwise fashion, the *next station* mentioned in the algorithm refers to the next physical station in a clockwise order. When token passing is applied to other topologies (e.g., a bus), each station is assigned a position in a logical sequence, and the token is passed according to the assigned sequence.

†Although older LANs used token passing ring technology, popularity has decreased, and few token passing networks remain.

14.6.2 CSMA/CD

In 1973, researchers at Xerox PARC created an extremely successful network technology that used a random access protocol. In 1978, a standard (informally called the *DIX standard*) was created by Digital Equipment Corporation, Intel, and Xerox. Known as *Ethernet*, the original Ethernet technology consisted of a single long cable to which computers attach†. The cable served as a shared medium — instead of broadcasting radio frequency transmissions through the atmosphere, Ethernet transmitted signals down a cable. Furthermore, instead of using two frequencies and a central transmitter, Ethernet allows all communication to proceed across the shared cable. Despite their differences, Ethernet and ALOHAnet had to solve the same basic problem: if two stations attempt to transmit at the same time, the signals interfere and a collision occurs.

Ethernet offered three innovations in the way collisions are handled:

- Carrier sense
- Collision detection
- Binary exponential backoff

Carrier Sense. Instead of allowing a station to transmit whenever a packet becomes ready, Ethernet requires each station to monitor the cable to detect whether another transmission is already in progress. The mechanism, which is known as *carrier*

sense, prevents the most obvious collision problems, and substantially improves network utilization.

Collision Detection. Although carrier sense is used, a collision can occur if two stations wait for a transmission to stop, find the cable idle, and both start transmitting. A small part of the problem is that even at the speed of light, some time is required for a signal to travel down the cable. Thus, a station at one end of the cable cannot know instantly when a station at the other end begins to transmit.

To handle collisions, each station monitors the cable during transmission. If the signal on the cable differs from the signal that the station is sending, it means that a collision has occurred. The technique is known as *collision detection*. When a collision is detected, the sending station aborts transmission.

Many details complicate Ethernet transmission. For example, following a collision, transmission does not abort until enough bits have been sent to guarantee that the collided signals reach all stations. Furthermore, following a transmission, stations must wait for an *interpacket gap* (9.6 μsec for a 10 Mbps Ethernet) to insure that all stations sense an idle network and have a chance to transmit. Such details illustrate how carefully the technology was designed.

Binary Exponential Backoff. Ethernet does more than merely detect collisions — it also recovers from them. After a collision occurs, a computer must wait for the cable to become idle again before transmitting a frame. As with ALOHAnet, randomization is used to avoid having multiple stations transmit simultaneously as soon as the cable is idle. That is, the standard specifies a maximum delay, d, and requires each station to choose a random delay less than d after a collision occurs. In most cases, when two stations each choose a random value, the station that chooses the smallest delay will proceed to send a packet and the network will return to normal operation.

In the case where two or more computers happen to choose nearly the same amount of delay, they will both begin to transmit at nearly the same time, producing a second collision. To avoid a sequence of collisions, Ethernet requires each computer to double the range from which a delay is chosen after each collision. A computer chooses a random delay from 0 to d after one collision, a random delay between 0 and $2d$ after a second collision, between 0 and $4d$ after a third, and so on. After a few collisions, the range from which a random value is chosen becomes large. Thus, some computer will choose a random delay shorter than the others, and will transmit without a collision.

Doubling the range of the random delay after each collision is known as *binary exponential backoff*. In essence, exponential backoff means that an Ethernet can recover quickly after a collision because each computer agrees to wait longer times between attempts when the cable becomes busy. Even in the unlikely event that two or more computers choose delays that are approximately equal, exponential backoff guarantees that contention for the cable will be reduced after a few collisions.

The combination of techniques described above is known by the name *Carrier Sense Multi-Access with Collision Detection*. (*CSMA/CD*). Algorithm 14.4 summarizes CSMA/CD.

Algorithm 14.4

Purpose:

Use CSMA/CD to send a packet

Method:

```
Wait for a packet to be ready;
Wait for the medium to be idle (carrier sense);
Delay for the interpacket gap;
Set variable x to the standard backoff range, d ;
Attempt to transmit the packet (collision detection);
While (a collision occurred during previous transmission) {
    Choose q to be a random delay between 0 and x ;
    Delay for q microseconds;
    Double x in case needed for the next round;
    Attempt to retransmit the packet (collision detection);
}
```

Algorithm 14.4 Packet transmission using CSMA/CD.

14.6.3 CSMA/CA

Although it works well on a cable, CSMA/CD does not work as well in wireless LANs because a transmitter used in a wireless LAN has a limited range, δ. That is, a receiver that is more than δ away from the transmitter will not receive a signal, and will not be able to detect a carrier. To see why limits cause problems for CSMA/CD, consider three computers with wireless LAN hardware positioned as Figure 14.6 illustrates.

Figure 14.6 Three computers with wireless LAN hardware at maximal distance.

In the figure, computer 1 can communicate with computer 2, but cannot receive the signal from computer 3. Thus, if computer 3 is transmitting a packet to computer 2, computer 1's carrier sense mechanism will not detect the transmission. Similarly, if computers 1 and 3 simultaneously transmit, only computer 2 will detect a collision. The problem is sometimes called the *hidden station problem* because some stations are not visible to others.

To ensure that all stations share the transmission media correctly, wireless LANs use a modified access protocol known as *Carrier Sense Multiple Access With Collision Avoidance* (*CSMA/CA*). Instead of depending on all other computers to receive all transmissions, the CSMA/CA used with wireless LANs triggers a brief transmission from the intended receiver before transmitting a packet. The idea is that if both the sender and receiver transmit a message, all computers within range of either will know a packet transmission is beginning. Figure 14.7 illustrates the sequence.

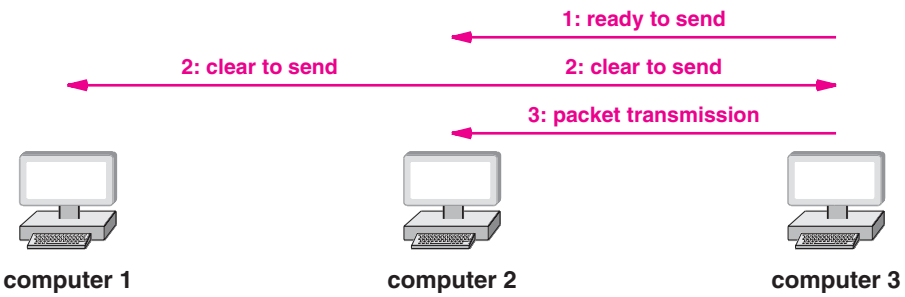


Figure 14.7 A sequence of messages sent when computer 3 transmits a packet to computer 2.

In the figure, computer 3 sends a short message to announce that it is ready to transmit a packet to computer 2, and computer 2 responds by sending a short message announcing that it is ready to receive the packet. All computers in range of computer 3 receive the initial announcement, and all computers in the range of computer 2 receive the response. As a result, even though it cannot receive the signal or sense a carrier, computer 1 knows that a packet transmission is taking place.

Collisions of control messages can occur when using CSMA/CA, but they can be handled easily. In the figure, for example, if computers 1 and 3 each attempt to transmit a packet to computer 2 at exactly the same time, their control messages will collide. Computer 2 will detect the collision, and will not reply. When a collision occurs, the sending stations apply random backoff before resending the control messages. Because control messages are much shorter than a packet, the probability of a second collision is low. Eventually, one of the two control messages arrives intact, and computer 2 transmits a response.

We can summarize:

Because computers on a wireless LAN can span distances greater than a signal can propagate, wireless LANs use CSMA/CA in which the sending and receiving computers each send a control message before packet transmission occurs.

17

LAN Extensions: Fiber Modems, Repeaters, Bridges, and Switches

17.1 Introduction

Previous chapters describe LAN topologies and wiring schemes. A typical LAN is designed to span a few hundred meters, which means LAN technology works well within a single building or a small campus.

This chapter discusses two important concepts: mechanisms that can extend a LAN across a longer distance and LAN switching. The chapter introduces repeaters, bridges, and the spanning tree algorithm used to prevent forwarding loops.

17.2 Distance Limitation And LAN Design

Distance limitation is a fundamental part of LAN designs. When designing a network technology, engineers choose a combination of capacity, maximum delay, and distance that can be achieved at a given cost. One limitation on distance arises because hardware is engineered to emit a fixed amount of energy — if wiring is extended beyond the design limits, stations will not receive a sufficiently strong signal, and errors will occur. The point is:

A maximum length specification is a fundamental part of LAN technology; LAN hardware will not work correctly over wires that exceed the bound.

17.3 Fiber Modem Extensions

Engineers have developed a variety of ways to extend LAN connectivity. As a general rule, extension mechanisms do not increase the strength of signals, nor do they merely extend cables. Instead, most extension mechanisms use standard interface hardware, and insert additional hardware components that can relay signals across longer distances.

The simplest LAN extension mechanism consists of an optical fiber and a pair of *fiber modems*, used to connect a computer to a remote Ethernet. Figure 17.1 illustrates the interconnection.

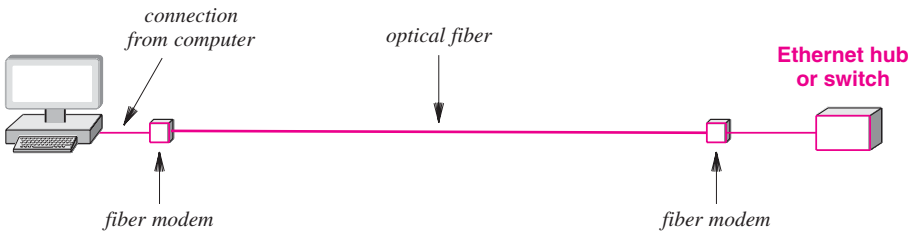


Figure 17.1 Illustration of fiber modems used to provide a connection between a computer and a remote Ethernet.

Each of the fiber modems contains hardware to perform two chores: accept packets over the Ethernet interface and send them over the optical fiber, and accept packets that arrive over the optical fiber and send them over the Ethernet interface†. If the modems offer a LAN interface on each end, standard interfaces can be used on the computer and the LAN device.

To summarize:

A pair of fiber modems and optical fibers can be used to provide a connection between a computer and a remote LAN such as an Ethernet.

†In practice, implementations use a pair of fibers to allow simultaneous transmission in both directions.

17.4 Repeaters

A *repeater* is an analog device used to propagate LAN signals over long distances. A repeater does not understand packets or signal coding. Instead, it merely amplifies the signal received, and transmits the amplified version as output.

Repeaters were used extensively with the original Ethernet, and have been used with other LAN technologies. Recently, repeaters have been introduced with infrared receivers to permit a receiver to be located at a longer distance from a computer. For example, consider a situation in which the infrared receiver for a cable television controller must be in a different room than the controller. A repeater can extend the connection, as Figure 17.2 illustrates.

Figure 17.2 Illustration of an infrared sensor extended with a repeater.

To summarize:

> *A repeater* is an analog hardware device used to extend a LAN. The repeater amplifies and sends all incoming signals to the other side.

17.5 Bridges And Bridging

A *bridge* is a mechanism that connects two LAN segments (e.g., two hubs) and transfers packets between them. The bridge listens in *promiscuous mode* on each segment (i.e., receives all packets sent on the segment). When it receives an intact frame from one segment, the bridge forwards a copy of the frame to the other segment. Thus, two LAN segments connected by a bridge appear to behave like a single LAN — a computer connected to either segment can send a frame to any computer on the two segments. Furthermore, a broadcast frame is delivered to all computers on the two segments. Thus, computers do not know whether they are connected to a single LAN segment or a bridged LAN.

Originally, bridges were sold as independent hardware devices that each had two network connections. Currently, bridge technology is incorporated into other devices, such as a cable modem. Figure 17.3 illustrates the conceptual architecture.

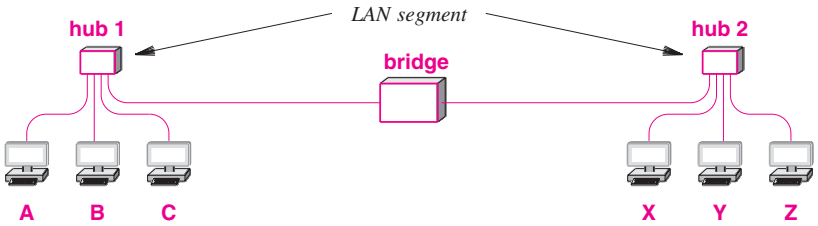


Figure 17.3 Illustration of six computers connected to a pair of bridged LAN segments.

To summarize:

A bridge is a mechanism used to connect two LAN segments and forward frames from one segment to another; computers cannot tell whether they are on a single segment or a bridged LAN.

18.3 Traditional WAN Architecture

Traditional WAN technologies were developed before Local Area Networks emerged, before personal computers were available, and before the Internet had been created†. Thus, traditional WAN architectures were designed to connect a set of sites, where each site had a few, large computers.

Without LAN technologies available, WAN designers chose to create a special-purpose hardware device that could be placed at each site. Known as a *packet switch*, the device provides local connections for computers at the site as well as connections for data circuits that lead to other sites.

Conceptually, a packet switch consists of a small computer system with a processor, memory, and I/O devices used to send and receive packets. Early packet switches were constructed from conventional computers; the packet switches used in the highest-speed WANs require special-purpose hardware. Figure 18.1 illustrates the internal architecture.

†Early WANs were called *long-haul networks*.

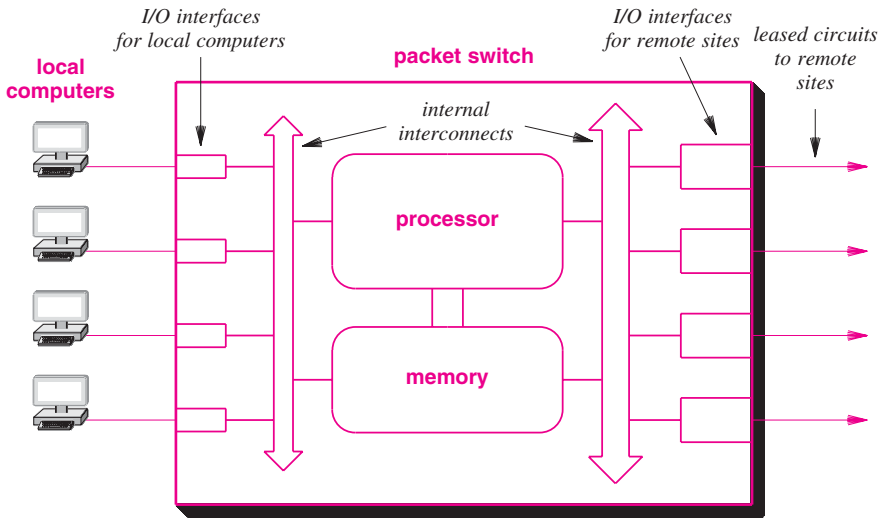


Figure 18.1 Illustration of traditional packet switch architecture.

As the figure shows, a packet switch contains two types of I/O devices. The first, which operates at high-speed, is used to connect the switch to a digital circuit that leads to another packet switch. The second type of I/O device, which operates at a lower speed, is used to connect the switch to an individual computer.

Since the advent of LAN technology, most WANs separate a packet switch into two parts: a Layer 2 switch that connects local computers and a router that connects to other sites. Part 4 of the text discusses Internet routers in detail, and explains how the concepts covered here apply to the Internet; for now, it is sufficient to understand that communication with local computers can be separated from transmission across a WAN. Figure 18.2 illustrates the separation.

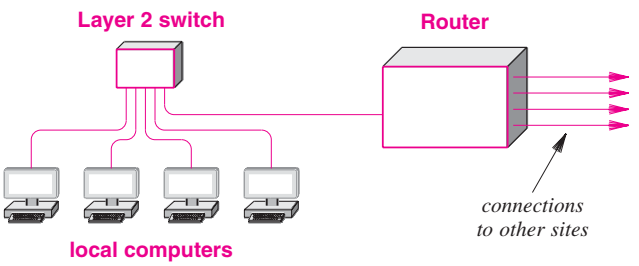


Figure 18.2 Illustration of a modern WAN site with local communication handled by a separate LAN.

18.4 Forming A WAN

Conceptually, a WAN can be formed by interconnecting a set of sites. The exact details of the interconnections depend on the data rate needed, the distance spanned, and the delay that can be tolerated. Many WANs use leased data circuits as described in Chapter 12 (e.g., a T3 circuit or an OC-12 circuit). However, other forms are also available, such as microwave and satellite channels. In addition to choosing the technology for a connection, a designer must choose a topology. For a given set of sites, many topologies are possible. For example, Figure 18.3 illustrates a possible way to interconnect four traditional packet switches and eight computers.

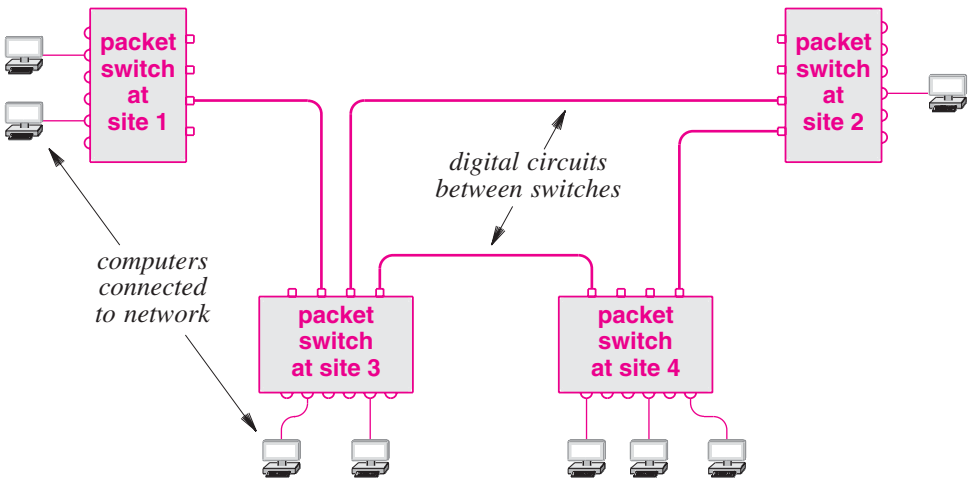


Figure 18.3 An example WAN formed by interconnecting packet switches.

As the figure shows, a WAN does not need to be symmetric — the interconnections among packet switches and the capacity of each connection can be chosen to accommodate the expected traffic and provide redundancy in case of failure. In the figure, the packet switch at site *1* has only one connection to the rest of the network, while the packet switches at other sites have at least two external connections. The point is:

> *A traditional WAN is formed by interconnecting packet switches; a packet switch at each site connects to computers. The topology and capacity of connections are chosen to accommodate expected traffic and need for redundancy.*

18.5 Store And Forward Paradigm

The goal of a WAN is to allow as many computers as possible to send packets simultaneously. The fundamental paradigm used to achieve simultaneous transmission is known as *store and forward*. To perform store and forward processing, a packet switch *buffers* packets in memory. The *store* operation occurs when a packet arrives: I/O hardware inside the packet switch places a copy of the packet in memory. The *forward* operation occurs once a packet has arrived and is waiting in memory. The processor examines the packet, determines its destination, and sends the packet over the I/O interface that leads to the destination.

A system that uses the store and forward paradigm can keep each data link busy, and thus, increase overall performance. More important, if multiple packets are sent to the same output device, the packet switch can accept and hold packets in memory until the device is ready. For example, consider packet transmission on the network in Figure 18.3. Suppose the two computers at site *1* each generate a packet destined for a computer at site *3* at approximately the same time. The two computers can send their packet to the packet switch simultaneously. As each packet arrives, I/O hardware on the packet switch places the packet in memory and informs the packet switch processor. The processor examines each packet's destination, and determines that the packets should be sent to site *3*. If the output interface that leads to site *3* is idle when a packet arrives, transmission starts immediately. If the output device is busy, the processor places the outgoing packet in a queue associated with the device. As soon as it finishes sending a packet, the device extracts and sends the next packet in the queue.

The concept can be summarized:

> *Wide area packet switching systems use the store-and-forward technique in which packets arriving at a packet switch are placed in a queue until the packet switch can forward them on toward their destination. The technique allows a packet switch to buffer a short burst of packets that arrive simultaneously.*

18.6 Addressing In A WAN

From the view of an attached computer, a traditional WAN network operates similar to a LAN. Each WAN technology defines the exact frame format a computer uses when sending and receiving data. Furthermore, each computer connected to a WAN is assigned an address. When sending a frame to another computer, the sender must supply the destination address.

Although details vary, WANs addresses follow a key concept that is used in the Internet: *hierarchical addressing*. Conceptually, hierarchical addressing divides each address into two parts:

(site, computer at the site)

In practice, instead of a identifying a site, each packet switch is assigned a unique number, which means that the first part of an address identifies a packet switch and the second part identifies a specific computer. For example, Figure 18.4 shows two-part hierarchical addresses assigned to computers connected to a pair of packet switches.

Figure 18.4 Example of an address hierarchy where each address identifies a packet switch and a computer attached to the switch.

The figure shows each address as a pair of decimal integers. For example, a computer connected to port *6* on packet switch *2* is assigned address *[2,6]*. In practice, an address is represented as a single binary value, with some bits of the binary value used to represent a packet switch and others used to identify a computer. In Part 4 of the text, we will see that the Internet uses the same scheme: each Internet address consists of a binary number where a prefix of the bits identify a specific network in the Internet and the remainder of the bits identify a computer attached to the network.

18.7 Next-Hop Forwarding

The importance of hierarchical addressing becomes clear when one considers packet processing. When a packet arrives, a packet switch must choose an outgoing path over which to forward the packet. If a packet is destined for a local computer, the switch sends the packet directly to the computer. Otherwise, the packet must be forwarded over one of the connections that leads to another switch. To make the choice, a packet switch examines the destination address in the packet, and extracts the packet switch number. If the number in the destination address is identical to the packet switch's own ID, the packet is intended for a computer on the local packet switch. Otherwise, the packet is intended for a computer on another packet switch. Algorithm 18.1 explains the computation.

The important idea is that a packet switch does not need to keep complete information about how to reach all possible computers, nor does a switch need to compute the entire route a packet will follow through the network. Instead, a switch bases forwarding on packet switch IDs, which means that a switch only needs to know which outgoing link to use to reach a given switch.

Algorithm 18.1

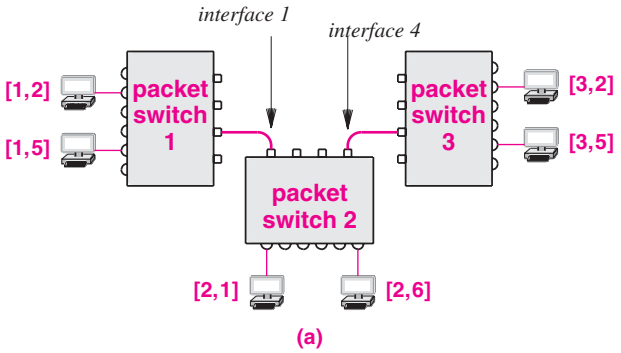
```
Given:
    A packet that has arrived at packet switch Q
Perform:
    The next-hop forwarding step
Method:
    Extract the destination address from the packet;
    Divide the address into a packet switch number, P, and a
        computer identification, C;
    if ( P == Q ) {         /* the destination is local */
        Forward the packet to local computer C;
    } else {
        Select a link that leads to another packet switch, and for-
ward
            the packet over the link;
    }
```

Algorithm 18.1 The two steps a packet switch uses to forward a packet when using next-hop forwarding.

We say that a switch only needs to compute the *next hop* for a packet. The process is called *next-hop forwarding*, and is analogous to the way airlines list flights. Suppose an airline passenger traveling from San Francisco to Miami finds that the only available itinerary involves three flights: the first from San Francisco to Dallas, the second from Dallas to Atlanta, and the third from Atlanta to Miami. Although the ultimate destination (Miami) remains the same throughout the trip, the next hop changes at each airport. When the passenger leaves San Francisco, the next hop is Dallas. When the passenger is in Dallas, the next hop is Atlanta, and when the passenger is in Atlanta, the next hop is Miami.

To make the computation efficient, packet switches use table lookup. That is, each packet switch contains a *forwarding table*† that lists all possible packet switches and gives a next hop for each. Figure 18.5 illustrates next-hop forwarding with a trivial example.

†Although purists insist on the name *forwarding table*, such tables were originally called *routing tables*, and the terminology is still widely used in the networking industry.



to reach	send to
switch 1	interface 1
switch 2	local delivery
switch 3	interface 4

(b)

Figure 18.5 (a) A network of three packet switches, and (b) the next-hop forwarding table for switch *2*.

To use a forwarding table, a switch extracts a packet's destination address, and uses the packet switch portion of the address as an index in the forwarding table. For example, consider the table in Figure 18.5b. If a packet is destined for *[3,5]*, the switch extracts *3*, consults the table, and forwards the packet to interface *4*, which leads to switch *3*.

Using only one part of a two-part hierarchical address to forward a packet has two practical consequences. First, the computation time required to forward a packet is reduced because the forwarding table can be organized as an array that uses indexing instead of searching. Second, the forwarding table contains one entry per packet switch instead of one entry per destination computer. The reduction in table size can be substantial, especially for a large WAN that has many computers attached to each packet switch.

In essence, a two-part hierarchical addressing scheme allows packet switches to use only the first part of the destination address until the packet reaches the final switch (i.e., the switch to which the destination computer is attached). Once the packet reaches the final switch, the switch uses the second part of the address to choose a specific computer, as Algorithm 18.1 describes.

To summarize:

Only the first part of a destination address is used when forwarding a packet across a WAN. Once the packet reaches the switch to which the destination computer attaches, the second part of the address is used to forward the packet to the correct local computer.

18.8 Source Independence

Note that next-hop forwarding does not depend on the packet's original source or on the path the packet has taken before it arrives at a particular packet switch. Instead, the next hop to which a packet is sent depends only on the packet's destination. The concept, which is known as *source independence*, is a fundamental idea in networking, and will be implicit in our discussions throughout the chapter and in later chapters that describe Internet forwarding.

Source independence allows the forwarding mechanism in a computer network to be compact and efficient. Because all packets follow the same path, only one table is required. Because forwarding does not use source information, only the destination address needs to be extracted from a packet. Furthermore, a single mechanism handles forwarding uniformly — packets that originate on directly connected computers and packets that arrive from other packet switches use the same mechanism.

18.9 Dynamic Routing Updates In A WAN

For a WAN to operate correctly, each switch must have a forwarding table, and must forward packets. Furthermore, values in the forwarding table must guarantee the following:

- Universal communication. The forwarding table in each switch must contain a valid next-hop route for each possible destination address.
- Optimal routes. In a switch, the next-hop value in the forwarding table for a given destination must point to the shortest path to the destination.

Network failures further complicate forwarding. For example, if two paths exist to a given destination and one of the paths becomes unavailable because hardware fails (e.g., a circuit is disconnected), forwarding should be changed to avoid the unavailable path. Thus, a manager cannot merely configure a forwarding table to contain static values that do not change. Instead, software running on the packet switches continually tests for failures, and reconfigures the forwarding tables automatically. We use the term *routing software* to describe software that automatically reconfigures forwarding tables.

The easiest way to think about route computation in a WAN is to think of a graph that models the network, and imagine software using the graph to compute the shortest path to all possible destinations. Each *node* in the graph corresponds to a packet switch in the network (individual computers are not part of the graph). If the network contains a direct connection between a pair of packet switches, the graph contains an *edge* or *link* between the corresponding nodes†. For example, Figure 18.6 shows an example WAN and the corresponding graph.

†Because the relationship between graph theory and computer networking is strong, one often hears a packet switch called a *network node* and a data circuit between two sites called a *link*.

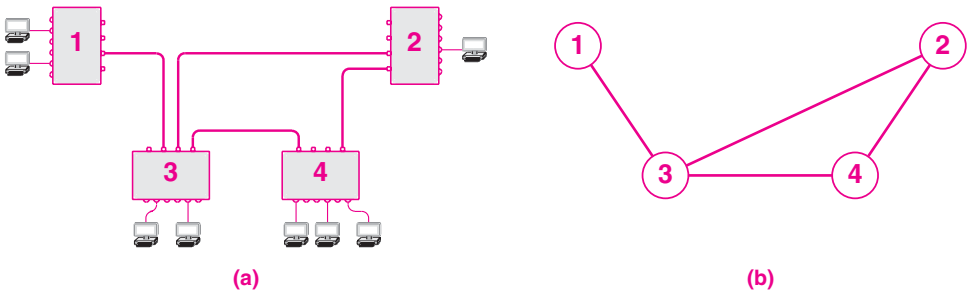


Figure 18.6 Illustration of a WAN and the corresponding graph.

As the figure shows, nodes in the graph are given a label that is the same as the number assigned to the corresponding packet switch. A graph representation is especially useful in computing next-hop forwarding because graph theory has been studied and efficient algorithms have been developed. Furthermore, a graph abstracts away details, allowing routing software to deal with the essence of the problem.

When it computes next-hop forwarding for a graph, a routing algorithm must identify a link. Our examples will used the notation (k, j) to denote a link from node *k* to node *j*. Thus, when a routing algorithm runs on the graph in Figure 18.6b, the algorithm produces output as shown in Figure 18.7.

to reach	next hop
1	–
2	(1,3)
3	(1,3)
4	(1,3)

node 1

to reach	next hop
1	(2,3)
2	–
3	(2,3)
4	(2,4)

node 2

to reach	next hop
1	(3,1)
2	(3,2)
3	–
4	(3,4)

node 3

to reach	next hop
1	(4,3)
2	(4,2)
3	(4,3)
4	–

node 4

Figure 18.7 A forwarding table for each node in the graph of Figure 18.6b.

18.10 Default Routes

The forwarding table for node *1* in Figure 18.7 raises an important point: a forwarding table may contain many entries that point to the same next hop. An examination of the WAN in Figure 18.6a reveals why all remote entries contain the same next hop: the packet switch has only one connection to the network. Therefore, all outgoing

traffic must be sent across the same connection. Consequently, except for the entry that corresponds to the node itself, all entries in node *1*'s forwarding table have a next hop that points to the link from node *1* to node *3*.

In our trivial example, the list of duplicate entries in the forwarding table is short. However a large WAN may contains hundreds of duplicate entries. Most WAN systems include a mechanism that can be used to eliminate the common case of duplicate entries. Called a *default route*, the mechanism allows a single entry in a forwarding table to replace a long list of entries that have the same next-hop value. Only one default entry is allowed in a forwarding table, and the entry has lower priority than other entries. If the forwarding mechanism does not find an explicit entry for a given destination, it uses the default. Figure 18.8 shows the forwarding tables from Figure 18.7 revised to use a default route.

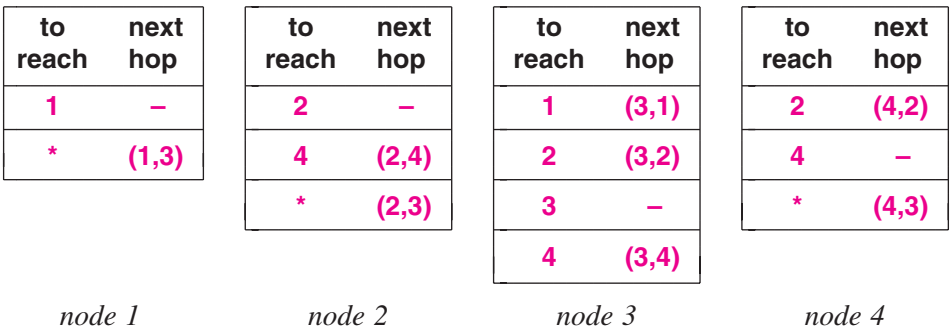


Figure 18.8 The forwarding tables from Figure 18.7 with default routes denoted by an asterisk.

Default routing is optional — a default entry is present only if more than one destination has the same next-hop value. For example, the forwarding table for node *3* does not contain a default route because each entry has a unique next hop. However, the forwarding table for node *1* benefits from a default route because all remote destinations have the same next hop.

18.11 Forwarding Table Computation

How is a forwarding table constructed? There are two basic approaches.

- *Static routing*. A program computes and installs routes when a packet switch boots; the routes do not change.
- *Dynamic routing*. A program builds an initial forwarding table when a packet switch boots; the program then alters the table as conditions in the network change.

20

Internetworking: Concepts, Architecture, and Protocols

20.1 Introduction

Previous chapters describe basic networking, including the hardware components used in LAN and WAN networks as well as general concepts such as addressing and routing. This chapter begins an examination of another fundamental idea in computer communication — an internetworking technology that can be used to connect multiple physical networks into a large, uniform communication system. The chapter discusses the motivation for internetworking, introduces the hardware components used, describes the architecture in which the components are connected, and discusses the significance of the concept. The remaining chapters in this section expand the internetworking concept, and provide additional details about the technology. They examine individual protocols, and explain how each uses techniques from earlier chapters to achieve reliable, error-free communication.

20.2 The Motivation For Internetworking

Each network technology is designed to fit a specific set of constraints. For example, LAN technologies are designed to provide high-speed communication across short distances, while WAN technologies are designed to provide communication across large areas. Consequently,

No single networking technology is best for all needs.

A large organization with diverse networking requirements needs multiple physical networks. More important, if the organization chooses the type of network that is best for each task, the organization will have several types of networks. For example, a LAN technology like Ethernet might be the best solution for connecting computers at a given site, but a leased data circuit might be used to interconnect a site in one city with a site in another.

20.3 The Concept Of Universal Service

The chief problem with multiple networks should be obvious: a computer attached to a given network can only communicate with other computers attached to the same network. The problem became evident in the 1970s as large organizations began to acquire multiple networks. Each network in the organization formed an island. In many early installations, each computer attached to a single network, and employees had to choose a computer appropriate for each task. That is, an employee was given access to multiple screens and keyboards, and the employee was forced to move from one computer to another to send a message across the appropriate network.

Users are neither satisfied nor productive when they must use a separate computer for each network. Consequently, most modern computer communication systems allow communication between any two computers analogous to the way a telephone system provides communication between any two telephones. Known as *universal service*, the concept is a fundamental part of networking. With universal service, a user on any computer in any organization can send messages or data to any other user. Furthermore, a user does not need to change computer systems when changing tasks — all information is available to all computers. As a result, users are more productive. To summarize:

A communication system that supplies universal service allows arbitrary pairs of computers to communicate.

20.4 Universal Service In A Heterogeneous World

Does universal service mean that everyone needs to adopt a single network technology, or is it possible to have universal service across multiple networks that use multiple technologies? Incompatibilities make it impossible to form a large network merely by interconnecting the wires among networks. Furthermore, extension techniques such as bridging cannot be used with heterogeneous network technologies because each technology uses its own packet format and addressing scheme. Thus, a frame created for one network technology cannot be transmitted on a network that uses a different technology. The point can be summarized:

Although universal service is highly desirable, incompatibilities among network hardware, frames, and addresses prevent a bridged network from including arbitrary technologies.

20.5 Internetworking

Despite the incompatibilities among network technologies, researchers have devised a scheme that provides universal service among heterogeneous networks. Called *internetworking*, the scheme uses both hardware and software. Additional hardware systems are used to interconnect a set of physical networks. Software on the attached computers then provides universal service. The resulting system of connected physical networks is known as an *internetwork* or *internet*.

Internetworking is quite general. In particular, an internet is not restricted in size — internets exist that contain a few networks and the global Internet contains tens of thousands of networks. Similarly, the number of computers attached to each network in an internet can vary — some networks have no computers attached, while others have hundreds.

20.6 Physical Network Connection With Routers

The basic hardware component used to connect heterogeneous networks is a *router*. Physically, a router is an independent hardware system dedicated to the task of interconnecting networks. Like a bridge, a router contains a processor and memory as well as a separate I/O interface for each network to which it connects. The network treats a connection to a router the same as a connection to any other computer. Figure 20.1 illustrates that the physical connection of networks with a router is straightforward.

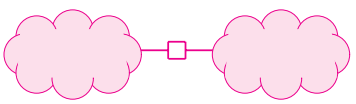


Figure 20.1 Two physical networks connected by a router, which has a separate interface for each network connection. Computers can attach to each network.

The figure uses a cloud to depict each network because router connections are not restricted to a particular network technology. A router can connect two LANs, a LAN and a WAN, or two WANs. Furthermore, when a router connects two networks in the same general category, the networks do not need to use the same technology. For example, a router can connect an Ethernet to a Wi-Fi network. Thus, each cloud represents an arbitrary network technology.

To summarize:

An Internet router is a special-purpose hardware system dedicated to the task of interconnecting networks. A router can interconnect networks that use different technologies, including different media, physical addressing schemes, or frame formats.

20.7 Internet Architecture

Routers make it possible for organizations to choose network technologies appropriate for each need and to use routers to connect all networks into an internet. For example, Figure 20.2 illustrates how three routers can be used to connect four arbitrary physical networks into an internet.

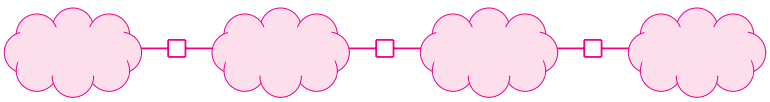


Figure 20.2 An internet formed by using three routers to interconnect four physical networks.

Although the figure shows each router with exactly two connections, commercial routers can connect more than two networks. Thus, a single router could connect all four networks in the example. Despite the feasibility, an organization seldom uses a single router to connect all of its networks. There are two reasons:

- Because the router must forward each packet, the processor in a given router is insufficient to handle the traffic passing among an arbitrary number of networks.
- Redundancy improves internet reliability. To avoid a single point of failure, protocol software continuously monitors internet connections and instructs routers to send traffic along alternative paths when a network or router fails.

Thus, when planning an internet, an organization must choose a design that meets the organization's need for reliability, capacity, and cost. In particular, the exact details of internet topology often depend on the bandwidth of the physical networks, the expected traffic, the organization's reliability requirements, and the cost and performance of available router hardware. To summarize:

An internet consists of a set of networks interconnected by routers. The internet scheme allows each organization to choose the number and type of networks, the number of routers to use to interconnect them, and the exact interconnection topology.

20.8 Achieving Universal Service

The goal of internetworking is universal service across heterogeneous networks. To provide universal service among all computers on an internet, routers must agree to forward information from a source on one network to a specified destination on another. The task is complex because frame formats and addressing schemes used by the underlying networks can differ. As a result, protocol software is needed on computers and routers to make universal service possible.

Later chapters describe Internet† protocol software in detail. They show how Internet protocols overcome differences in frame formats and physical addresses to make communication possible among networks that use different technologies. Before considering how Internet protocols work, it is important to understand the effect that an internet system presents to attached computers.

20.9 A Virtual Network

In general, Internet software provides the appearance of a single, seamless communication system to which many computers attach. The system offers universal service: each computer is assigned an address, and any computer can send a packet to any other computer. Furthermore, Internet protocol software hides the details of physical network connections, physical addresses, and routing information — neither users nor application programs are aware of the underlying physical networks or the routers that connect them.

We say that an internet is a *virtual network* system because the communication system is an abstraction. That is, although a combination of hardware and software provides the illusion of a uniform network system, no such network exists. Figure 20.3 illustrates the virtual network concept as well as a corresponding physical structure.

†Recall that when written with an uppercase *I*, the term *Internet* refers to the global Internet and the associated protocols.

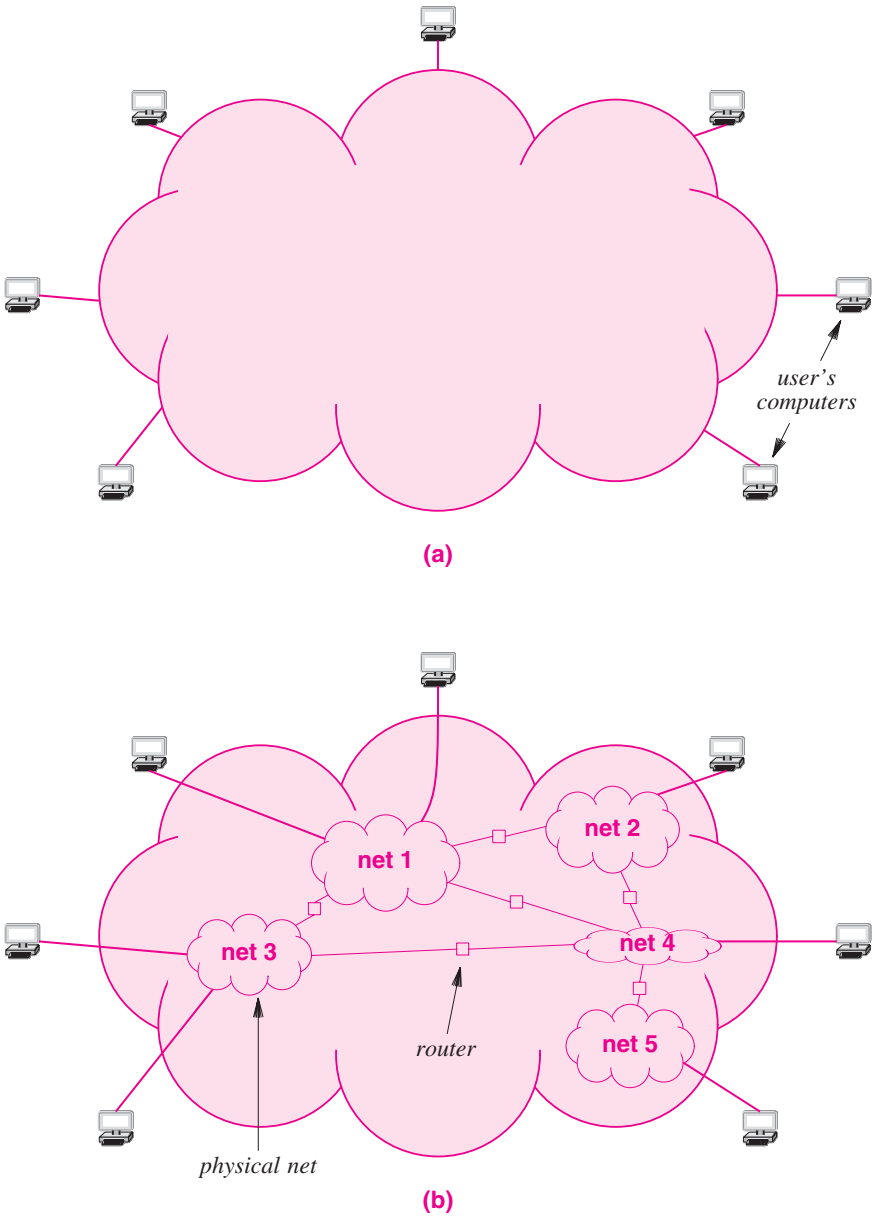


Figure 20.3 The Internet concept. (a) The illusion of a single network provided to users and applications, and (b) the underlying physical structure with routers interconnecting networks.

20.10 Protocols For Internetworking

Although several protocols have been proposed for use with internets, one suite stands out as the most widely used. The suite is formally known as the *TCP/IP Internet Protocols*; most networking professionals simply refer to the suite as *TCP/IP*†.

TCP/IP was developed at the same time as the global Internet. In fact, the same researchers who proposed TCP/IP also proposed the Internet architecture described above. Work on TCP/IP began in the 1970s, approximately the same time that Local Area Networks were being developed, and continued until the early 1990s when the Internet became commercial.

20.11 Review Of TCP/IP Layering

Recall from Chapter 1 that the Internet protocols use a five-layer reference model as Figure 20.4 illustrates.

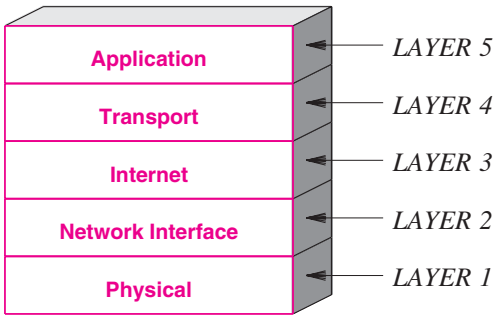


Figure 20.4 The five layers of the TCP/IP reference model.

We have already explored three of the layers. Chapters in part 1 of the text consider applications; chapters in parts 2 and 3 of the text discuss layer 1 and layer 2 protocols. Chapters in this part of the text consider the two remaining layers in detail:

Layer 3: Internet

Layer *3* (IP) specifies the format of packets sent across the Internet as well as the mechanisms used to forward packets from a computer through one or more routers to a final destination.

†TCP and IP are acronyms for two of the most important protocols in the suite; the name is pronounced by spelling out T-C-P-I-P.

Layer 4: Transport

Layer *4* (TCP) specifies the messages and procedures that are used to insure reliable transfer.

To summarize:

> *Internet protocols are organized into five conceptual layers, with IP at layer 3 and TCP at layer 4.*

20.12 Host Computers, Routers, And Protocol Layers

We use the term *host computer* to refer to a computer that connects to the Internet and runs applications. A host can be as small as a cell phone or as large as a mainframe. Furthermore, a host's CPU can be slow or fast, the memory can be large or small, and the network to which a host connects can operate at high or low speed. TCP/IP protocols make it possible for any pair of hosts to communicate, despite hardware differences.

Both hosts and routers need TCP/IP protocol software. However, routers do not use protocols from all layers. In particular, a router does not need layer *5* protocols for applications like file transfer because routers do not run conventional applications†. The next chapters discuss TCP/IP protocol software in more detail, and show how Internet layering works.

20.13 Summary

Logically, the Internet appears to be a single, seamless communication system. An arbitrary pair of computers connected to the Internet can communicate as if they were attached to a single network. That is, a computer can send a packet to any other computer that is attached to the Internet. Physically, the Internet is a collection of networks interconnected by devices called *routers*. Each router is a special-purpose device that connects to two or more networks and is dedicated to transferring Internet packets among the networks.

Computers that attach to the Internet are called *hosts*. A host may be a large computer (e.g., a supercomputer) or a small computer (e.g., a cell phone). Each host attaches to one of the physical networks in the Internet.

The illusion of a single communication system is provided by Internet protocol software. Each host or router in the Internet must run the software, which hides the details of the underlying physical connections and takes care of sending each packet to its destination.

†Some routers do run special applications that permit a manager to control the router remotely.