CHAPTER 1

Introduction

Solutions to Review Questions and Exercises

Review Questions

1. The five components of a data communication system are the ***sender***, **receiver**, ***transmission medium***, ***message***, and ***protocol***.
2. The advantages of distributed processing are ***security***, ***access to distributed databases***, ***collaborative processing***, and ***faster problem solving***.
3. The three criteria are ***performance***, ***reliability***, and ***security***.
4. Advantages of a multipoint over a point-to-point configuration (type of connection) include ***ease of installation*** and ***low cost***.
5. Line configurations (or types of connections) are ***point-to-point*** and ***multipoint***.
6. We can divide line configuration in two broad categories:
   a. ***Point-to-point***: ***mesh***, ***star***, and ***ring***.
   b. ***Multipoint***: ***bus***
7. In ***half-duplex transmission***, only one entity can send at a time; in a ***full-duplex transmission***, both entities can send at the same time.
8. We give an advantage for each of four network topologies:
   a. ***Mesh***: secure
   b. ***Bus***: easy installation
   c. ***Star***: robust
   d. ***Ring***: easy fault isolation
9. The number of cables for each type of network is:
   a. ***Mesh***: $n\ (n - 1) / 2$
   b. ***Star***: n
   c. ***Ring***: $n - 1$
   d. ***Bus***: one backbone and n drop lines
10. The general factors are ***size***, ***distances*** (covered by the network), ***structure***, and ***ownership***.

11. An ***internet*** is an interconnection of networks. The ***Internet*** is the name of a specific worldwide network
12. A ***protocol*** defines ***what*** is communicated, in ***what way*** and ***when***. This provides accurate and timely transfer of information between different devices on a network.
13. ***Standards*** are needed to create and maintain an open and competitive market for manufacturers, to coordinate protocol rules, and thus guarantee compatibility of data communication technologies.

Exercises

14. ***Unicode*** uses **32** bits to represent a symbol or a character. We can define 2^{32} different symbols or characters.
15. With **16** bits, we can represent up to 2^{16} different colors.
16.
    a. Cable links: n $(n - 1) / 2 = (6 \times 5) / 2 =$ **15**
    b. Number of ports: $(n - 1) =$ **5** ports needed per device
17.
    a. ***Mesh topology***: If one connection fails, the other connections will still be working.
    b. **Star topology**: The other devices will still be able to send data through the hub; there will be no access to the device which has the failed connection to the hub.
    c. **Bus Topology**: All transmission stops if the failure is in the bus. If the drop-line fails, only the corresponding device cannot operate.
    d. ***Ring Topology***: The failed connection may disable the whole network unless it is a dual ring or there is a by-pass mechanism.
18. This is a ***LAN***. The Ethernet hub creates a LAN as we will see in Chapter 13.
19. Theoretically, in a ***ring topology***, unplugging one station, interrupts the ring. However, most ring networks use a mechanism that bypasses the station; the ring can continue its operation.
20. In a ***bus topology***, no station is in the path of the signal. Unplugging a station has no effect on the operation of the rest of the network.
21. See Figure 1.1
22. See Figure 1.2.
23.
    a. E-mail is not an interactive application. Even if it is delivered immediately, it may stay in the mail-box of the receiver for a while. It is not sensitive to delay.
    b. We normally do not expect a file to be copied immediately. It is not very sensitive to delay.
    c. Surfing the Internet is the an application very sensitive to delay. We except to get access to the site we are searching.
24. In this case, the communication is only between a caller and the callee. A dedicated line is established between them. The connection is ***point-to-point.***

Figure 1.1 *Solution to Exercise 21*

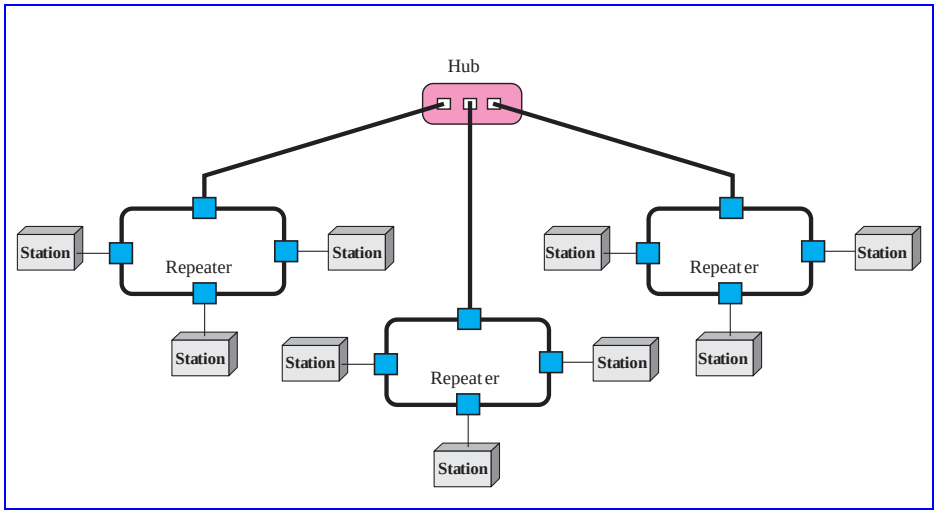
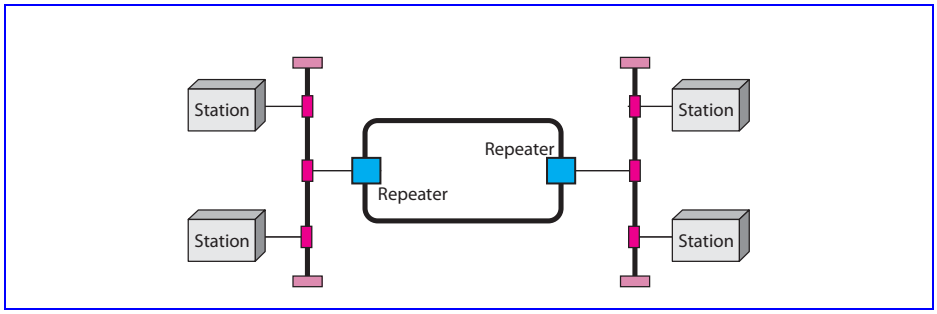


Figure 1.2 *Solution to Exercise 22*



25. The telephone network was originally designed for voice communication; the Internet was originally designed for data communication. The two networks are similar in the fact that both are made of interconnections of small networks. The telephone network, as we will see in future chapters, is mostly a circuit-switched network; the Internet is mostly a packet-switched network.

4

CHAPTER 2

Network Models

Solutions to Review Questions and Exercises

Review Questions

1. The Internet model, as discussed in this chapter, include ***physical***, ***data link***, ***network***, ***transport***, and ***application*** layers.
2. The network support layers are the ***physical***, ***data link***, and ***network*** layers.
3. The ***application*** layer supports the user.
4. The *transport layer* is responsible for ***process-to-process*** delivery of the entire message, whereas the network layer oversees ***host-to-host*** delivery of individual packets.
5. ***Peer-to-peer processes*** are processes on two or more devices communicating at a same layer
6. Each layer calls upon the ***services*** of the layer just below it using interfaces between each pair of adjacent layers.
7. ***Headers*** and ***trailers*** are control data added at the beginning and the end of each data unit at each layer of the sender and removed at the corresponding layers of the receiver. They provide source and destination addresses, synchronization points, information for error detection, etc.
8. The *physical layer* is responsible for transmitting a bit stream over a physical medium. It is concerned with
   a. ***physical characteristics of the media***
   b. ***representation of bits***
   c. ***type of encoding***
   d. ***synchronization of bits***
   e. ***transmission rate and mode***
   f. ***the way devices are connected with each other and to the links***
9. The *data link layer* is responsible for
   a. ***framing data bits***
   b. ***providing the physical addresses of the sender/receiver***
   c. ***data rate control***

d. ***detection and correction of damaged and lost frames***

10. The *network layer* is concerned with delivery of a packet across multiple networks; therefore its responsibilities include
    a. ***providing host-to-host addressing***
    b. ***routing***
11. The *transport layer* oversees the process-to-process delivery of the entire message. It is responsible for
    a. ***dividing the message into manageable segments***
    b. ***reassembling it at the destination***
    c. ***flow and error control***
12. The *physical address* is the local address of a node; it is used by the data link layer to deliver data from one node to another within the same network. The ***logical address*** defines the sender and receiver at the network layer and is used to deliver messages across multiple networks. The port address (service-point) identifies the application process on the station.
13. The *application layer services* include ***file transfer***, ***remote access***, ***shared database management***, and ***mail services***.
14. The ***application***, ***presentation***, and ***session*** layers of the OSI model are represented by the *application* layer in the Internet model. The lowest four layers of OSI correspond to the Internet model layers.

Exercises

15. The ***International Standards Organization***, or the ***International Organization of Standards***, (**ISO**) is a multinational body dedicated to worldwide agreement on international standards. An ISO standard that covers all aspects of network communications is the ***Open Systems Interconnection*** (**OSI**) model.
16.
    a. Route determination: ***network*** layer
    b. Flow control: ***data link*** and ***transport*** layers
    c. Interface to transmission media: ***physical*** layer
    d. Access for the end user: ***application*** layer
17.
    a. Reliable process-to-process delivery: **transport** layer
    b. Route selection: ***network*** layer
    c. Defining frames: ***data link*** layer
    d. Providing user services: ***application*** layer
    e. Transmission of bits across the medium: ***physical*** layer
18.
    a. Communication with user's application program: ***application*** layer
    b. Error correction and retransmission: ***data link*** and ***transport*** layers
    c. Mechanical, electrical, and functional interface: ***physical layer***

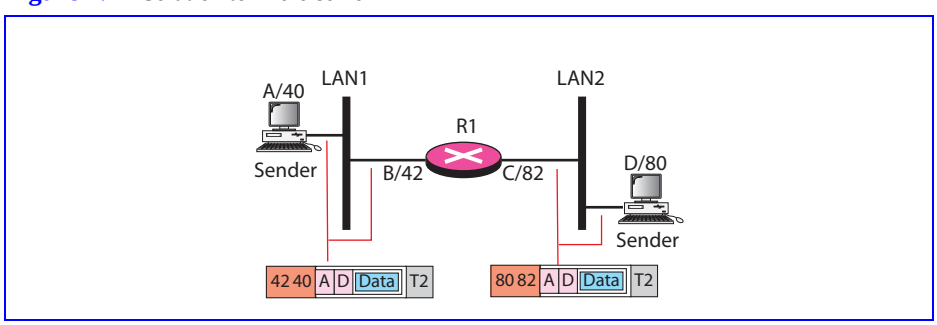
d. Responsibility for carrying frames between adjacent nodes: ***data link*** layer

19.

a. Format and code conversion services: ***presentation*** layer
b. Establishing, managing, and terminating sessions: ***session*** layer
c. Ensuring reliable transmission of data: ***data link*** and ***transport*** layers
d. Log-in and log-out procedures: ***session*** layer
e. Providing independence from different data representation: ***presentation*** layer

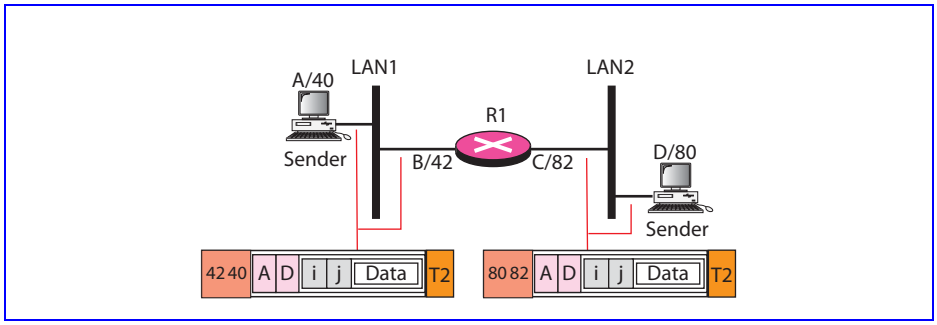
20. See Figure 2.1.

Figure 2.1 *Solution to Exercise 20*



21. See Figure 2.2.

Figure 2.2 *Solution to Exercise 21*



22. If the corrupted destination address does not match any station address in the network, the packet is lost. If the corrupted destination address matches one of the stations, the frame is delivered to the wrong station. In this case, however, the error detection mechanism, available in most data link protocols, will find the error and discard the frame. In both cases, the source will somehow be informed using one of the data link control mechanisms discussed in Chapter 11.

23. Before using the destination address in an intermediate or the destination node, the packet goes through error checking that may help the node find the corruption (with a high probability) and discard the packet. Normally the upper layer protocol will inform the source to resend the packet.

24. Most protocols issue a ***special error message*** that is sent back to the source in this case.
25. The errors ***between*** the nodes can be detected by the data link layer control, but the error ***at*** the node (between input port and output port) of the node cannot be detected by the data link layer.

CHAPTER 3

Data and Signals

Solutions to Review Questions and Exercises

Review Questions

1. ***Frequency*** and ***period*** are the inverse of each other. T = 1/ f and f = 1/T.
2. The ***amplitude*** of a signal measures the value of the signal at any point. The ***frequency*** of a signal refers to the number of periods in one second. The phase describes the position of the waveform relative to time zero.
3. Using Fourier analysis. ***Fourier series*** gives the frequency domain of a periodic signal; ***Fourier analysis*** gives the frequency domain of a nonperiodic signal.
4. Three types of transmission impairment are ***attenuation***, ***distortion***, and ***noise***.
5. ***Baseband transmission*** means sending a digital or an analog signal without modulation using a low-pass channel. ***Broadband transmission*** means modulating a digital or an analog signal using a band-pass channel.
6. A ***low-pass channel*** has a bandwidth starting from zero; a ***band-pass*** channel has a bandwidth that does not start from zero.
7. The ***Nyquist theorem*** defines the maximum bit rate of a noiseless channel.
8. The ***Shannon capacity*** determines the theoretical maximum bit rate of a noisy channel.
9. ***Optical signals*** have very high frequencies. A high frequency means a short wave length because the wave length is inversely proportional to the frequency ($\lambda = v/f$), where v is the propagation speed in the media.
10. A signal is ***periodic*** if its frequency domain plot is ***discrete***; a signal is **nonperiodic** if its frequency domain plot is ***continuous***.
11. The frequency domain of a voice signal is normally ***continuous*** because voice is a ***nonperiodic*** signal.
12. An alarm system is normally ***periodic***. Its frequency domain plot is therefore ***discrete***.
13. This is ***baseband transmission*** because no modulation is involved.
14. This is ***baseband transmission*** because no modulation is involved.
15. This is ***broadband transmission*** because it involves modulation.

Exercises

16.

a. **T = 1 / f** = 1 / (24 Hz) = 0.0417 s = 41.7×10^{-3} s = **41.7 ms**

b. **T = 1 / f** = 1 / (8 MHz) = 0.000000125 = 0.125×10^{-6} s = **0.125 μs**

c. **T = 1 / f** = 1 / (140 KHz) = 0.00000714 s = 7.14×10^{-6} s = **7.14 μs**

17.

a. f = 1 / T = 1 / (5 s) = 0.2 Hz

b. **f = 1 / T** = 1 / (12 μs) =83333 Hz = 83.333×10^{3} Hz = **83.333 KHz**

c. **f = 1 / T** = 1 / (220 ns) = 4550000 Hz = 4.55×10^{6} Hz = **4.55 MHz**

18.

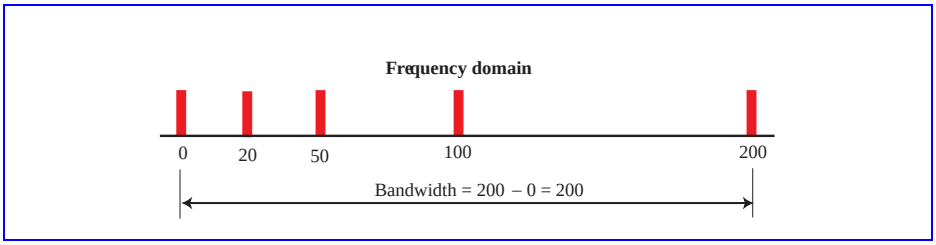
a. 90 degrees (π/2 radian)

b. 0 degrees (0 radian)

c. 90 degrees (π/2 radian)

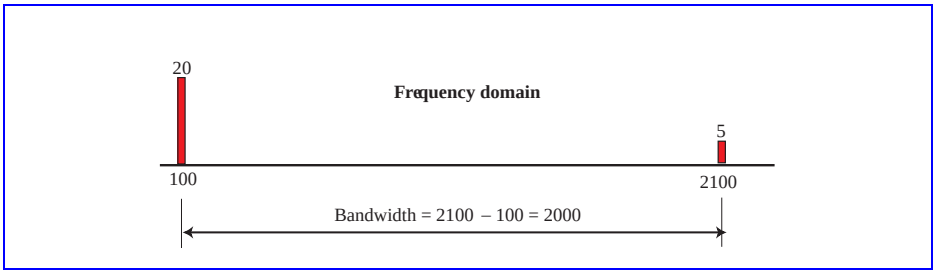
19. See Figure 3.1

Figure 3.1 *Solution to Exercise 19*



20. We know the lowest frequency, 100. We know the bandwidth is 2000. The highest frequency must be 100 + 2000 = **2100 Hz**. See Figure 3.2

Figure 3.2 *Solution to Exercise 20*



21. Each signal is a simple signal in this case. The bandwidth of a simple signal is zero. So the bandwidth of both signals are the same.

22.

a. bit rate = 1/ (bit duration) = 1 / (0.001 s) = 1000 bps = **1 Kbps**

b. bit rate = 1/ (bit duration) = 1 / (2 ms) = **500 bps**

c. bit rate = 1/(bit duration) = 1 / (20 μs/10) = 1 / (2 μs) = **500 Kbps**

23.

a. (10 / 1000) s = **0.01 s**

b. (8 / 1000) s = 0. 008 s = **8 ms**

c. ((100,000 × 8) / 1000) s = **800 s**

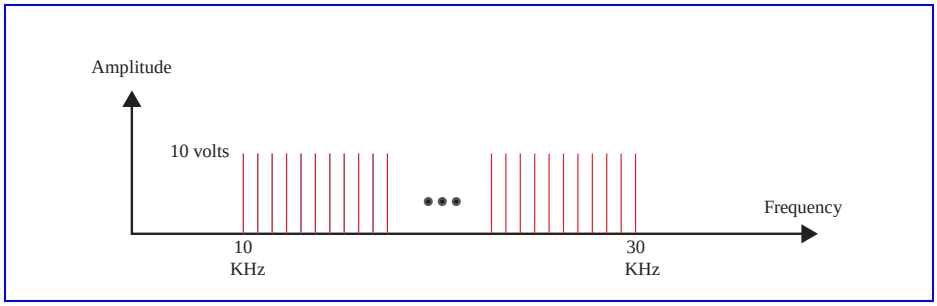
24. There are 8 bits in 16 ns. Bit rate is 8 / (16×10^{-9}) = 0.5×10^{-9} = **500 Mbps**

25. The signal makes 8 cycles in 4 ms. The frequency is 8 /(4 ms) = **2 KHz**

26. The bandwidth is 5 × 5 = **25 Hz**.

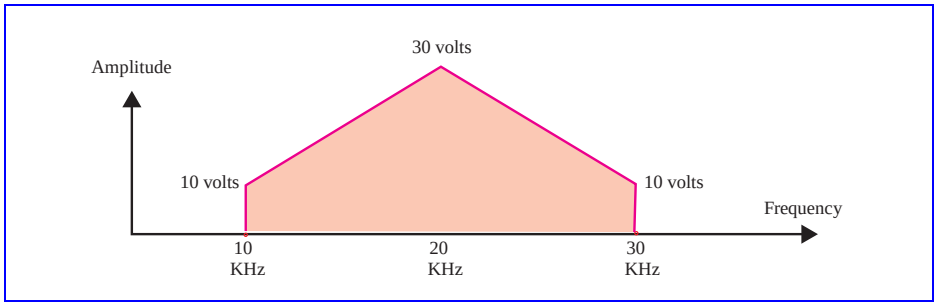
27. The signal is periodic, so the frequency domain is made of discrete frequencies. as shown in Figure 3.3.

Figure 3.3 *Solution to Exercise 27*



28. The signal is nonperiodic, so the frequency domain is made of a continuous spectrum of frequencies as shown in Figure 3.4.

Figure 3.4 *Solution to Exercise 28*



29.

Using the first harmonic, data rate = 2 × 6 MHz = **12 Mbps**
Using three harmonics, data rate = (2 × 6 MHz) /3 = **4 Mbps**
Using five harmonics, data rate = (2 × 6 MHz) /5 = **2.4 Mbps**

30. dB = 10 $\log_{10}$ (90 / 100) = **–0.46 dB**

31. –10 = 10 $\log_{10}$ (P_2 / 5) → $\log_{10}$ (P_2 / 5) = –1 → (P_2 / 5) = 10^{-1} → P_2 = **0.5 W**

32. The total gain is 3 × 4 = 12 dB. The signal is amplified by a factor $10^{1.2}$ = **15.85.**

33. 100,000 bits / 5 Kbps = **20 s**
34. 480 s × 300,000 km/s = **144,000,000 km**
35. 1 μm × 1000 = 1000 μm = **1 mm**
36. We have

$$\mathbf{4{,}000 \log_2 (1 + 1{,}000) \approx 40\ Kbps}$$

37. We have

$$\mathbf{4{,}000 \log_2 (1 + 10 / 0.005) = 43{,}866\ bps}$$

38. The file contains 2,000,000 × 8 = 16,000,000 bits. With a 56-Kbps channel, it takes 16,000,000/56,000 = **289 s**. With a 1-Mbps channel, it takes **16 s**.
39. To represent 1024 colors, we need $\log_2 1024 = 10$ (see Appendix C) bits. The total number of bits are, therefore,

$$\mathbf{1200 \times 1000 \times 10 = 12{,}000{,}000\ bits}$$

40. We have

$$\mathbf{SNR = (200\ mW) / (10 \times 2\ \times \mu W) = 10{,}000}$$

We then have

$$\mathbf{SNR_{dB} = 10 \log_{10} SNR = 40}$$

41. We have

$$\mathbf{SNR = (signal\ power)/(noise\ power).}$$

However, power is proportional to the square of voltage. This means we have

$$\mathbf{SNR = [(signal\ voltage)^2] / [(noise\ voltage)^2] =}$$
$$\mathbf{[(signal\ voltage) / (noise\ voltage)]^2 = 20^2 = 400}$$

We then have

$$\mathbf{SNR_{dB} = 10 \log_{10} SNR \approx 26.02}$$

42. We can approximately calculate the capacity as
    a. $C = B \times (SNR_{dB} /3) = 20\ KHz \times (40 /3) =$ **267 Kbps**
    b. $C = B \times (SNR_{dB} /3) = 200\ KHz \times (4 /3) =$ **267 Kbps**
    c. $C = B \times (SNR_{dB} /3) = 1\ MHz \times (20 /3) =$ **6.67 Mbps**
43.
    a. The data rate is doubled ($C_2 = 2 \times C_1$).
    b. When the SNR is doubled, the data rate increases slightly. We can say that, approximately, ($C_2 = C_1 + 1$).
44. We can use the approximate formula

$$\mathbf{C = B \times (SNR_{dB} /3)\ or\ SNR_{dB} = (3 \times C) /B}$$

We can say that the minimum

$$\mathbf{SNR_{dB} = 3 \times 100\ Kbps / 4\ KHz = 75}$$

This means that the minimum

$$\textbf{SNR} = 10^{\textbf{SNRdB}/10} = 10^{7.5} \approx \textbf{31,622,776}$$

45. We have

$$\textbf{transmission time} = \textbf{(packet length)/(bandwidth)} = \textbf{(8,000,000 bits) / (200,000 bps)} = \textbf{40 s}$$

46. We have

$$\textbf{(bit length)} = \textbf{(propagation speed)} \times \textbf{(bit duration)}$$

The bit duration is the inverse of the bandwidth.

a. Bit length = $(2 \times 10^8 \text{ m}) \times [(1 / (1 \text{ Mbps})] =$ **200 m**. This means a bit occupies 200 meters on a transmission medium.
b. Bit length = $(2 \times 10^8 \text{ m}) \times [(1 / (10 \text{ Mbps})] =$ **20 m**. This means a bit occupies 20 meters on a transmission medium.
c. Bit length = $(2 \times 10^8 \text{ m}) \times [(1 / (100 \text{ Mbps})] =$ **2 m**. This means a bit occupies 2 meters on a transmission medium.

47.

a. Number of bits = bandwidth × delay = 1 Mbps × 2 ms = **2000 bits**
b. Number of bits = bandwidth × delay = 10 Mbps × 2 ms = **20,000 bits**
c. Number of bits = bandwidth × delay = 100 Mbps × 2 ms = **200,000 bits**

48. We have

$$\textbf{Latency} = \textbf{processing time} + \textbf{queuing time} + \textbf{transmission time} + \textbf{propagation time}$$

Processing time = 10 × 1 μs = 10 μs = **0.000010 s**

Queuing time = 10 × 2 μs = 20 μs = **0.000020 s**

Transmission time = 5,000,000 / (5 Mbps) = **1 s**

Propagation time = (2000 Km) / (2×10^8) = **0.01 s**

$$\textbf{Latency} = 0.000010 + 0.000020 + 1 + 0.01 = \textbf{1.01000030 s}$$

The transmission time is dominant here because the packet size is huge.

CHAPTER 4

Digital Transmission

Solutions to Review Questions and Exercises

Review Questions

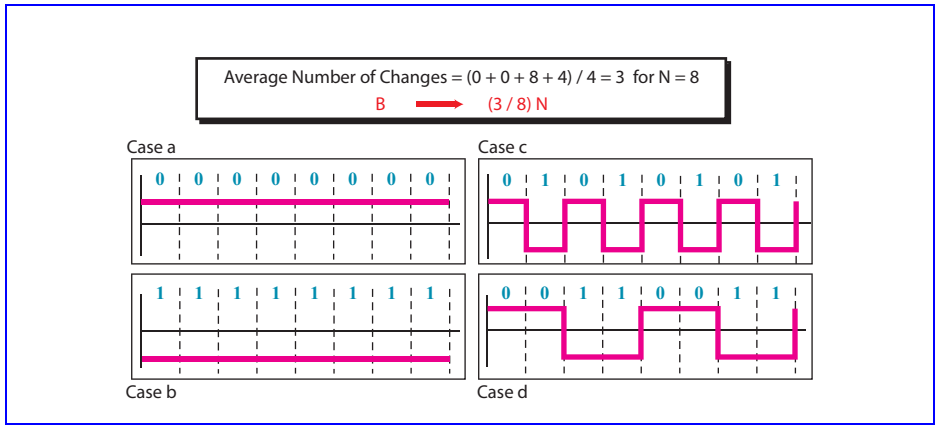
1. The three different techniques described in this chapter are ***line coding***, ***block coding***, and ***scrambling***.
2. A ***data element*** is the smallest entity that can represent a piece of information (a bit). A ***signal element*** is the shortest unit of a digital signal. Data elements are what we need to send; signal elements are what we can send. Data elements are being carried; signal elements are the carriers.
3. The ***data rate*** defines the number of data elements (bits) sent in 1s. The unit is bits per second (bps). The ***signal rate*** is the number of signal elements sent in 1s. The unit is the baud.
4. In decoding a digital signal, the incoming signal power is evaluated against the ***baseline*** (a running average of the received signal power). A long string of 0s or 1s can cause ***baseline wandering*** (a drift in the baseline) and make it difficult for the receiver to decode correctly.
5. When the voltage level in a digital signal is constant for a while, the spectrum creates very low frequencies, called ***DC components***, that present problems for a system that cannot pass low frequencies.
6. A ***self-synchronizing*** digital signal includes timing information in the data being transmitted. This can be achieved if there are transitions in the signal that alert the receiver to the beginning, middle, or end of the pulse.
7. In this chapter, we introduced ***unipolar***, ***polar***, ***bipolar***, ***multilevel***, and ***multitransition*** coding.
8. ***Block coding*** provides redundancy to ensure synchronization and to provide inherent error detecting. In general, block coding changes a block of m bits into a block of n bits, where n is larger than m.
9. ***Scrambling,*** as discussed in this chapter, is a technique that substitutes long zero-level pulses with a combination of other levels without increasing the number of bits.

10. Both ***PCM*** and ***DM*** use sampling to convert an analog signal to a digital signal. PCM finds the value of the signal amplitude for each sample; DM finds the change between two consecutive samples.
11. In ***parallel transmission*** we send data *several* bits at a time. In ***serial transmission*** we send data *one* bit at a time.
12. We mentioned ***synchronous***, ***asynchronous***, and ***isochronous***. In both synchronous and asynchronous transmissions, a bit stream is divided into independent frames. In synchronous transmission, the bytes inside each frame are synchronized; in asynchronous transmission, the bytes inside each frame are also independent. In isochronous transmission, there is no independency at all. All bits in the whole stream must be synchronized.

Exercises

13. We use the formula **s = c × N × (1/r)** for each case. We let c = 1/2.
    a. r = 1 → s = (1/2) × (1 Mbps) × 1/**1** = **500 kbaud**
    b. r = 1/2 → s = (1/2) × (1 Mbps) × 1/(**1/2**) = **1 Mbaud**
    c. r = 2 → s = (1/2) × (1 Mbps) × 1/**2** = **250 Kbaud**
    d. r = 4/3 → s = (1/2) × (1 Mbps) × 1/(**4/3**) = **375 Kbaud**
14. The number of bits is calculated as (0.2 /100) × (1 Mbps) = **2000 bits**
15. See Figure 4.1. Bandwidth is proportional to **(3/8)N** which is within the range in Table 4.1 (B = 0 to N) for the NRZ-L scheme.

Figure 4.1 *Solution to Exercise 15*



16. See Figure 4.2. Bandwidth is proportional to **(4.25/8)N** which is within the range in Table 4.1 (B = 0 to N) for the NRZ-I scheme.
17. See Figure 4.3. Bandwidth is proportional to **(12.5 / 8) N** which is within the range in Table 4.1 (B = N to B = 2N) for the Manchester scheme.
18. See Figure 4.4. B is proportional to **(12/8) N** which is within the range in Table 4.1 (B = N to 2N) for the differential Manchester scheme.

Figure 4.2 *Solution to Exercise 16*

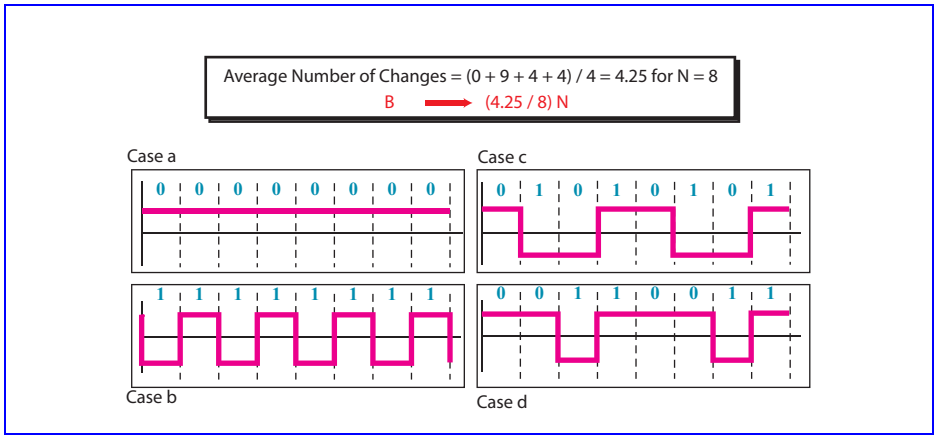


Figure 4.3 *Solution to Exercise 17*

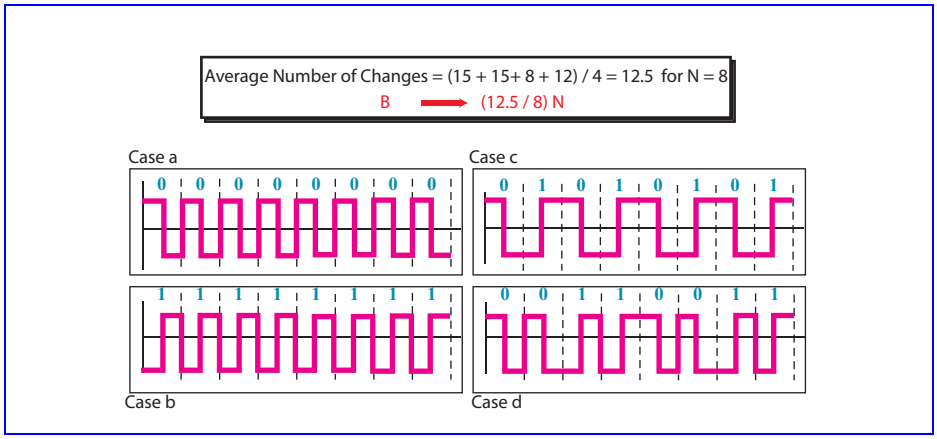
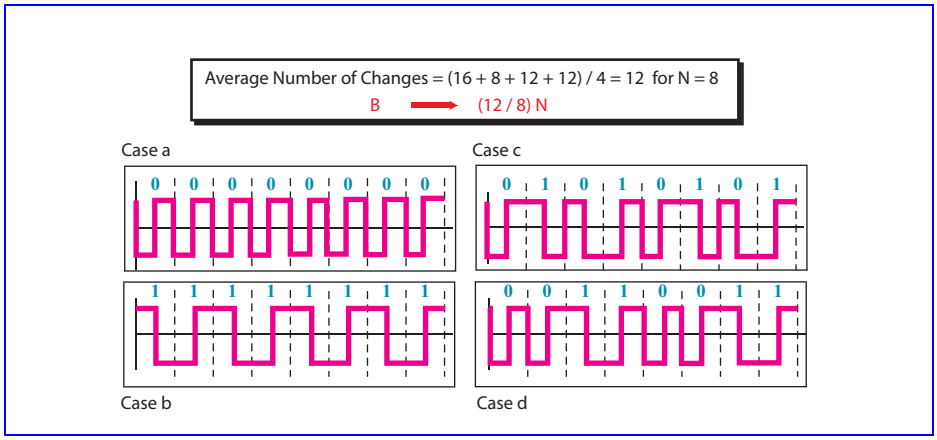
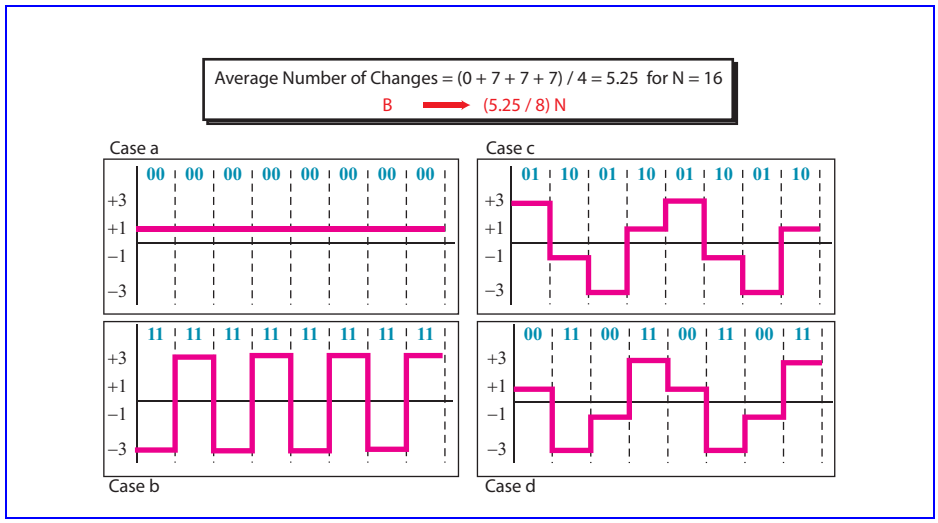


Figure 4.4 *Solution to Exercise 18*



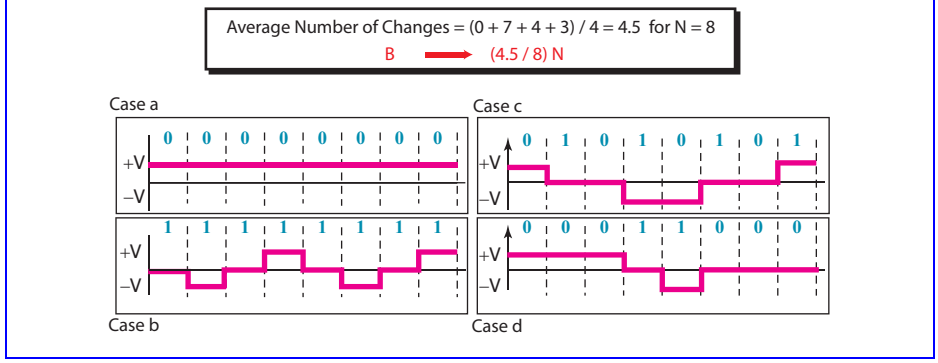
19. See Figure 4.5. B is proportional to **(5.25 / 16) N** which is inside range in Table 4.1 (B = 0 to N/2) for 2B/1Q.

Figure 4.5 *Solution to Exercise 19*



20. See Figure 4.6. B is proportional to **(5.25/8)** × **N** which is inside the range in Table 4.1 (B = 0 to N/2) for MLT-3.

Figure 4.6 *Solution to Exercise 20*



21. The data stream can be found as
    a. NRZ-I: **10011001**.
    b. Differential Manchester: **11000100**.
    c. AMI: **01110001**.
22. The data rate is 100 Kbps. For each case, we first need to calculate the value f / N. We then use Figure 4.6 in the text to find P (energy per Hz). All calculations are approximations.

a. f /N = 0/100 = 0 → **P = 1.0**

b. f /N = 50/100 = 1/2 → **P = 0.5**

c. f /N = 100/100 = 1 → **P = 0.0**

d. f /N = 150/100 = 1.5 → **P = 0.2**

23. The data rate is 100 Kbps. For each case, we first need to calculate the value f/N. We then use Figure 4.8 in the text to find P (energy per Hz). All calculations are approximations.

a. f /N = 0/100 = 0 → **P = 0.0**

b. f /N = 50/100 = 1/2 → **P = 0.3**

c. f /N = 100/100 = 1 → **P = 0.4**

d. f /N = 150/100 = 1.5 → **P = 0.0**

24.

a. The output stream is **01010 11110 11110 11110 11110 01001**.

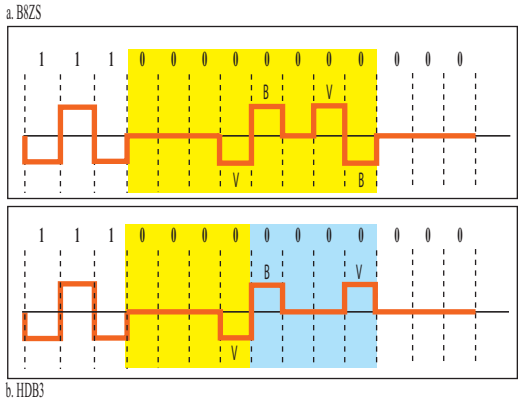
b. The maximum length of consecutive 0s in the input stream is **21**.

c. The maximum length of consecutive 0s in the output stream is **2**.

25. In 5B/6B, we have $2^5 = 32$ data sequences and $2^6 = 64$ code sequences. The number of unused code sequences is $64 - 32 = \mathbf{32}$. In 3B/4B, we have $2^3 = 8$ data sequences and $2^4 = 16$ code sequences. The number of unused code sequences is $16 - 8 = \mathbf{8}$.

26. See Figure 4.7. Since we specified that the last non-zero signal is positive, the first bit in our sequence is positive.

Figure 4.7 *Solution to Exercise 26*



27.

a. In a low-pass signal, the minimum frequency 0. Therefore, we have

$f_{max} = 0 + 200 = 200$ KHz. → $f_s = 2 \times 200{,}000 =$ **400,000 samples/s**

b. In a bandpass signal, the maximum frequency is equal to the minimum frequency plus the bandwidth. Therefore, we have

$$f_{max} = 100 + 200 = 300 \text{ KHz.} \rightarrow f_s = 2 \times 300{,}000 = \textbf{600,000 samples /s}$$

28.

a. In a lowpass signal, the minimum frequency is 0. Therefore, we can say

$$f_{max} = 0 + 200 = 200 \text{ KHz} \rightarrow f_s = 2 \times 200{,}000 = \textbf{400,000 samples/s}$$

The number of bits per sample and the bit rate are

$$n_b = \log_2 1024 = 10 \text{ bits/sample} \qquad N = 400 \text{ KHz} \times 10 = \textbf{4 Mbps}$$

b. The value of $n_b = 10$. We can easily calculate the value of SNR_{dB}

$$SNR_{dB} = 6.02 \times n_b + 1.76 = \textbf{61.96}$$

c. The value of $n_b = 10$. The minimum bandwidth can be calculated as

$$B_{PCM} = n_b \times B_{analog} = 10 \times 200 \text{ KHz} = \textbf{2 MHz}$$

29. The maximum data rate can be calculated as

$$N_{max} = 2 \times B \times n_b = 2 \times 200 \text{ KHz} \times \log_2 4 = \textbf{800 kbps}$$

30. We can first calculate the sampling rate (fs) and the number of bits per sample (nb)

$$f_{max} = 0 + 4 = 4 \text{ KHz} \rightarrow f_s = 2 \times 4 = \textbf{8000 sample/s}$$

We then calculate the number of bits per sample.

$$\rightarrow n_b = 30000 / 8000 = 3.75$$

We need to use the next integer $n_b = 4$. The value of SNR_{dB} is

$$SNR_{dB} = 6.02 \times n_b + 1.72 = \textbf{25.8}$$

31. We can calculate the data rate for each scheme:

a. NRZ → $N = 2 \times B = 2 \times 1 \text{ MHz} = \textbf{2 Mbps}$
b. Manchester → $N = 1 \times B = 1 \times 1 \text{ MHz} = \textbf{1 Mbps}$
c. MLT-3 → $N = 3 \times B = 3 \times 1 \text{ MHz} = \textbf{3 Mbps}$
d. 2B1Q → $N = 4 \times B = 4 \times 1 \text{ MHz} = \textbf{4 Mbps}$

32.

a. For synchronous transmission, we have $1000 \times 8 = \textbf{8000}$ bits.

b. For asynchronous transmission, we have $1000 \times 10 = \textbf{10000}$ bits. Note that we assume only one stop bit and one start bit. Some systems send more start bits.

c. For case a, the redundancy is 0%. For case b, we send 2000 extra for 8000 required bits. The redundancy is **25%**.

CHAPTER 5

Analog Transmission

Solutions to Review Questions and Exercises

Review Questions

1. Normally, ***analog transmission*** refers to the transmission of analog signals using a band-pass channel. Baseband digital or analog signals are converted to a complex analog signal with a range of frequencies suitable for the channel.
2. A ***carrier*** is a single-frequency signal that has one of its characteristics (amplitude, frequency, or phase) changed to represent the baseband signal.
3. The process of changing one of the characteristics of an analog signal based on the information in digital data is called ***digital-to-analog conversion***. It is also called modulation of a digital signal. The baseband digital signal representing the digital data modulates the carrier to create a broadband analog signal.
4.
   a. ASK changes the ***amplitude*** of the carrier.
   b. FSK changes the ***frequency*** of the carrier.
   c. PSK changes the ***phase*** of the carrier.
   d. QAM changes both the ***amplitude*** and the ***phase*** of the carrier.
5. We can say that the most susceptible technique is ***ASK*** because the amplitude is more affected by noise than the phase or frequency.
6. A ***constellation diagram*** can help us define the amplitude and phase of a signal element, particularly when we are using two carriers. The diagram is useful when we are dealing with multilevel ASK, PSK, or QAM. In a constellation diagram, a signal element type is represented as a dot. The bit or combination of bits it can carry is often written next to it.The diagram has two axes. The horizontal *X* axis is related to the in-phase carrier; the vertical *Y* axis is related to the quadrature carrier.
7. The two components of a signal are called ***I*** and ***Q***. The I component, called in-phase, is shown on the horizontal axis; the Q component, called quadrature, is shown on the vertical axis.
8. The process of changing one of the characteristics of an analog signal to represent the instantaneous amplitude of a baseband signal is called ***analog-to-analog con-***

version. It is also called the *modulation* of an analog signal; the baseband analog signal modulates the carrier to create a broadband analog signal.

9.
   a. AM changes the ***amplitude*** of the carrier
   b. FM changes the ***frequency*** of the carrier
   c. PM changes the ***phase*** of the carrier

10. We can say that the most susceptible technique is ***AM*** because the amplitude is more affected by noise than the phase or frequency.

Exercises

11. We use the formula **S = (1/r) × N**, but first we need to calculate the value of r for each case.

a. $r = \log_2 2$	$= 1$	$\rightarrow$	$S = (1/1) \times (2000 \text{ bps})$	$=$ **2000 baud**
b. $r = \log_2 2$	$= 1$	$\rightarrow$	$S = (1/1) \times (4000 \text{ bps})$	$=$ **4000 baud**
c. $r = \log_2 4$	$= 2$	$\rightarrow$	$S = (1/2) \times (6000 \text{ bps})$	$=$ **3000 baud**
d. $r = \log_2 64$	$= 6$	$\rightarrow$	$S = (1/6) \times (36{,}000 \text{ bps})$	$=$ **6000 baud**

12. We use the formula **N = r × S**, but first we need to calculate the value of r for each case.

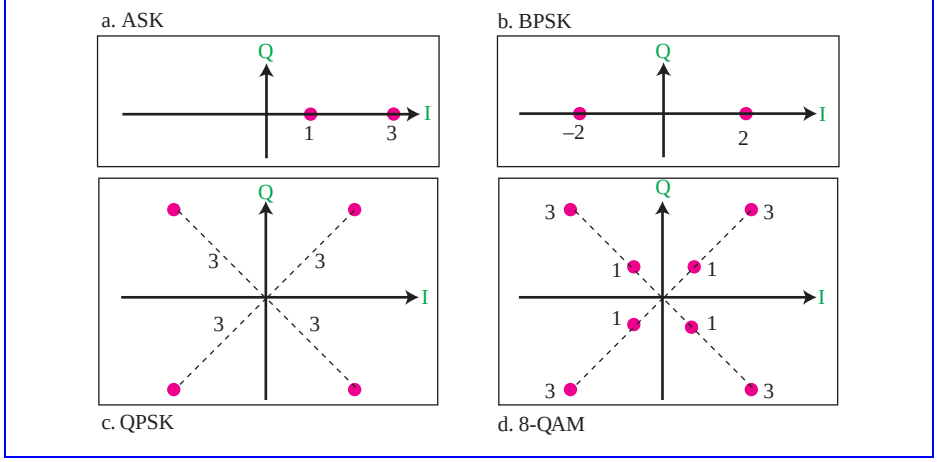
a. $r = \log_2 2$	$= 1$	$\rightarrow$	$N = (1) \times (1000 \text{ bps})$	$=$ **1000 bps**
b. $r = \log_2 2$	$= 1$	$\rightarrow$	$N = (1) \times (1000 \text{ bps})$	$=$ **1000 bps**
c. $r = \log_2 2$	$= 1$	$\rightarrow$	$N = (1) \times (1000 \text{ bps})$	$=$ **1000 bps**
d. $r = \log_2 16$	$= 4$	$\rightarrow$	$N = (4) \times (1000 \text{ bps})$	$=$ **4000 bps**

13. We use the formula **$r = \log_2 L$** to calculate the value of r for each case.

a. $\log_2 4$	$= 2$
b. $\log_2 8$	$= 3$
c. $\log_2 4$	$= 2$
d. $\log_2 128$	$= 7$

14. See Figure 5.1.
   a. We have two signal elements with peak amplitudes 1 and 3. The phase of both signal elements are the same, which we assume to be 0 degrees.
   b. We have two signal elements with the same peak amplitude of 2. However, there must be 180 degrees difference between the two phases. We assume one phase to be 0 and the other 180 degrees.
   c. We have four signal elements with the same peak amplitude of 3. However, there must be 90 degrees difference between each phase. We assume the first phase to be at 45, the second at 135, the third at 225, and the fourth at 315 degrees. Note that this is one out of many configurations. The phases can be at

Figure 5.1 *Solution to Exercise 14*

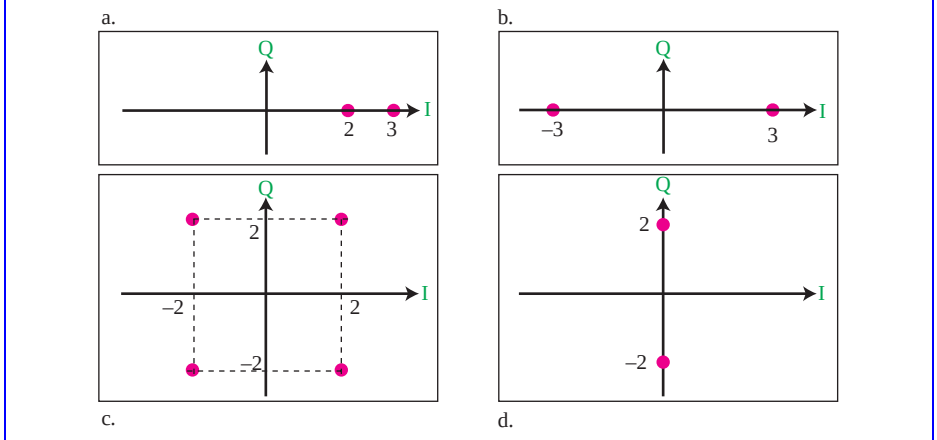


0, 90, 180, and 270. As long as the differences are 90 degrees, the solution is satisfactory.

d. We have four phases, which we select to be the same as the previous case. For each phase, however, we have two amplitudes, 1 and 3 as shown in the figure. Note that this is one out of many configurations. The phases can be at 0, 90, 180, and 270. As long as the differences are 90 degrees, the solution is satisfactory.

15. See Figure 5.2

Figure 5.2 *Solution to Exercise 15*



a. This is ASK. There are two peak amplitudes both with the same phase (0 degrees). The values of the peak amplitudes are $A_1 = 2$ (the distance between

the first dot and the origin) and A_2= 3 (the distance between the second dot and the origin).

b. This is BPSK, There is only one peak amplitude (3). The distance between each dot and the origin is 3. However, we have two phases, 0 and 180 degrees.

c. This can be either QPSK (one amplitude, four phases) or 4-QAM (one amplitude and four phases). The amplitude is the distance between a point and the origin, which is $(2^2 + 2^2)^{1/2} = 2.83$.

d. This is also BPSK. The peak amplitude is 2, but this time the phases are 90 and 270 degrees.

16. The number of points define the number of levels, L. The number of bits per baud is the value of r. Therefore, we use the formula **$r = \log_2 L$** for each case.

a. $\log_2 2$ = 1
b. $\log_2 4$ = 2
c. $\log_2 16$ = 4
d. $\log_2 1024$ = 10

17. We use the formula **$B = (1 + d) \times (1/r) \times N$**, but first we need to calculate the value of r for each case.

a. r = 1 → B= (1 + 1) × (1/1) × (4000 bps) = **8000 Hz**
b. r = 1 → B = (1 + 1) × (1/1) × (4000 bps) + 4 KHz = **8000 Hz**
c. r = 2 → B = (1 + 1) × (1/2) × (4000 bps) = **2000 Hz**
d. r = 4 → B = (1 + 1) × (1/4) × (4000 bps) = **1000 Hz**

18. We use the formula **$N = [1/(1 + d)] \times r \times B$**, but first we need to calculate the value of r for each case.

a. r = $\log_2 2$ = 1 → N= [1/(1 + 0)] × 1 × (4 KHz) = **4 kbps**
b. r = $\log_2 4$=2 → N = [1/(1 + 0)] × 2 × (4 KHz) = **8 kbps**
c. r = $\log_2 16$= 4 → N = [1/(1 + 0)] × 4 × (4 KHz) = **16 kbps**
d. r = $\log_2 64$= 6 → N = [1/(1 + 0)] × 6 × (4 KHz) = **24 kbps**

19.

First, we calculate the bandwidth for each channel = (1 MHz) / 10 = 100 KHz. We then find the value of r for each channel:

$$B = (1 + d) \times (1/r) \times (N) \rightarrow r = N / B \rightarrow r = (1 \text{ Mbps}/100 \text{ KHz}) = 10$$

We can then calculate the number of levels: $L = 2^r = 2^{10}$ = **1024.** This means that that we need a **1024-QAM** technique to achieve this data rate.

20. We can use the formula: $N = [1/(1 + d)] \times r \times B = 1 \times 6 \times 6$ MHz = 36 Mbps

21.

a. $B_{AM} = 2 \times B = 2 \times 5$ = **10 KHz**

b. $B_{FM} = 2 \times (1 + \beta) \times B = 2 \times (1 + 5) \times 5$ **= 60 KHz**
c. $B_{PM} = 2 \times (1 + \beta) \times B = 2 \times (1 + 1) \times 5$ = **20 KHz**

22. We calculate the number of channels, not the number of coexisting stations.

a. n = (1700 - 530) KHz / 10 KHz = **117**
b. n = (108 - 88) MHz / 200 KHz = **100**

CHAPTER 6

Bandwidth Utilization:

Solutions to Review Questions and Exercises

Review Questions

1. ***Multiplexing*** is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.
2. We discussed ***frequency-division multiplexing*** (**FDM**), ***wave-division multiplexing*** (**WDM**), and ***time-division multiplexing*** (**TDM**).
3. In ***multiplexing***, the word ***link*** refers to the physical path. The word ***channel*** refers to the portion of a link that carries a transmission between a given pair of lines. One link can have many (n) channels.
4. ***FDM*** and ***WDM*** are used to combine ***analog signals***; the bandwidth is shared. ***TDM*** is used to combine ***digital signals***; the time is shared.
5. To maximize the efficiency of their infrastructure, telephone companies have traditionally multiplexed analog signals from lower-bandwidth lines onto higher-bandwidth lines. The ***analog hierarchy*** uses voice channels (4 KHz), ***groups*** (48 KHz), ***supergroups*** (240 KHz), ***master groups*** (2.4 MHz), and ***jumbo groups*** (15.12 MHz).
6. To maximize the efficiency of their infrastructure, telephone companies have traditionally multiplexed digital signals from lower data rate lines onto higher data rate lines. The ***digital hierarchy*** uses ***DS-0*** (64 Kbps), ***DS-1*** (1.544 Mbps), ***DS-2*** (6.312 Mbps), ***DS-3*** (44.376 Mbps), and ***DS-4*** (274.176 Mbps).
7. ***WDM*** is common for multiplexing ***optical signals*** because it allows the multiplexing of signals with a very high frequency.
8. In ***multilevel TDM***, some lower-rate lines are combined to make a new line with the same data rate as the other lines. ***Multiple slot TDM***, on the other hand, uses multiple slots for higher data rate lines to make them compatible with the lower data rate line. ***Pulse stuffing TDM*** is used when the data rates of some lines are not an integral multiple of other lines.
9. In ***synchronous TDM***, each input has a reserved slot in the output frame. This can be inefficient if some input lines have no data to send. In ***statistical TDM***, slots are

dynamically allocated to improve bandwidth efficiency. Only when an input line has a slot's worth of data to send is it given a slot in the output frame.

10. In ***spread spectrum***, we spread the bandwidth of a signal into a larger bandwidth. Spread spectrum techniques add redundancy; they spread the original spectrum needed for each station. The expanded bandwidth allows the source to wrap its message in a protective envelope for a more secure transmission. We discussed ***frequency hopping spread spectrum*** (***FHSS***) and direct sequence spread spectrum (***DSSS***).
11. The ***frequency hopping spread spectrum*** (***FHSS***) technique uses *M* different carrier frequencies that are modulated by the source signal. At one moment, the signal modulates one carrier frequency; at the next moment, the signal modulates another carrier frequency.
12. The ***direct sequence spread spectrum*** (***DSSS***) technique expands the bandwidth of the original signal. It replaces each data bit with *n* bits using a spreading code.

Exercises

13. To multiplex 10 voice channels, we need nine guard bands. The required bandwidth is then B = (4 KHz) × 10 + (500 Hz) × 9 = **44.5 KHz**
14. The bandwidth allocated to each voice channel is 20 KHz / 100 = 200 Hz. As we saw in the previous chapters, each digitized voice channel has a data rate of 64 Kbps (8000 sample × 8 bit/sample). This means that our modulation technique uses 64,000/200 = 320 bits/Hz.
15.
    a. Group level: overhead = 48 KHz – (12 × 4 KHz) = **0 Hz**.
    b. Supergroup level: overhead = 240 KHz – (5 × 48 KHz) **= 0 Hz**.
    c. Master group: overhead = 2520 KHz – (10 × 240 KHz) = **120 KHz**.
    d. Jumbo Group: overhead = 16.984 MHz – (6 × 2.52 MHz) = **1.864 MHz**.
16.
    a. Each output frame carries 1 bit from each source plus one extra bit for synchronization. Frame size = 20 × 1 + 1 = **21 bits.**
    b. Each frame carries 1 bit from each source. Frame rate = **100,000 frames/s.**
    c. Frame duration = 1 /(frame rate) = 1 /100,000 = **10 μs**.
    d. Data rate = (100,000 frames/s) × (21 bits/frame) = **2.1 Mbps**
    e. In each frame 20 bits out of 21 are useful. Efficiency = 20/21= **95%**
17.
    a. Each output frame carries 2 bits from each source plus one extra bit for synchronization. Frame size = 20 × 2 + 1 = **41 bits.**
    b. Each frame carries 2 bit from each source. Frame rate = 100,000/2 **= 50,000 frames/s.**
    c. Frame duration = 1 /(frame rate) = 1 /50,000 = **20 μs**.
    d. Data rate = (50,000 frames/s) × (41 bits/frame) = **2.05 Mbps.** The output data rate here is slightly less than the one in Exercise 16.

e. In each frame 40 bits out of 41 are useful. Efficiency = 40/41= **97.5%**. Efficiency is better than the one in Exercise 16.

18.

a. Frame size = 6 × (8 + 4) = **72 bits**.

b. We can assume that we have only 6 input lines. Each frame needs to carry one character from each of these lines. This means that the frame rate is **500 frames/s**.

c. Frame duration = 1 /(frame rate) = 1 /500 = **2 ms**.

d. Data rate = (500 frames/s) × (72 bits/frame) = **36 kbps**.

19. We combine six 200-kbps sources into three 400-kbps. Now we have seven 400-kbps channel.

a. Each output frame carries 1 bit from each of the seven 400-kbps line. Frame size = 7 × 1 = **7 bits.**

b. Each frame carries 1 bit from each 400-kbps source. Frame rate = **400,000 frames/s.**

c. Frame duration = 1 /(frame rate) = 1 /400,000 = **2.5 μs**.

d. Output data rate = (400,000 frames/s) × (7 bits/frame) = **2.8 Mbps.** We can also calculate the output data rate as the sum of input data rate because there is no synchronizing bits. Output data rate = 6 × 200 + 4 × 400 = **2.8 Mbps**.

20.

a. The frame carries 4 bits from each of the first two sources and 3 bits from each of the second two sources. Frame size = 4 × 2 + 3 × 2 = **14 bits.**

b. Each frame carries 4 bit from each 200-kbps source or 3 bits from each 150 kbps. Frame rate = 200,000 / 4 = 150,000 /3 = **50,000 frames/s.**

c. Frame duration = 1 /(frame rate) = 1 /50,000 = **20 μs**.

d. Output data rate = (50,000 frames/s) × (14 bits/frame) = **700 kbps.** We can also calculate the output data rate as the sum of input data rates because there are no synchronization bits. Output data rate = 2 × 200 + 2 × 150 = 700 kbps.

21. We need to add extra bits to the second source to make both rates = 190 kbps. Now we have two sources, each of 190 Kbps.

a. The frame carries 1 bit from each source. Frame size = 1 + 1 = **2 bits.**

b. Each frame carries 1 bit from each 190-kbps source. Frame rate = **190,000 frames/s.**

c. Frame duration = 1 /(frame rate) = 1 /190,000 = **5.3 μs**.

d. Output data rate = (190,000 frames/s) × (2 bits/frame) = **380 kbps.** Here the output bit rate is greater than the sum of the input rates (370 kbps) because of extra bits added to the second source.

22.

a. T-1 line sends 8000 frames/s. Frame duration = 1/8000 = **125 μs**.

b. Each frame carries one extra bit. Overhead = 8000 × 1 = **8 kbps**

23. See Figure 6.1.

24. See Figure 6.2.

Figure 6.1 *Solution to Exercise 23*

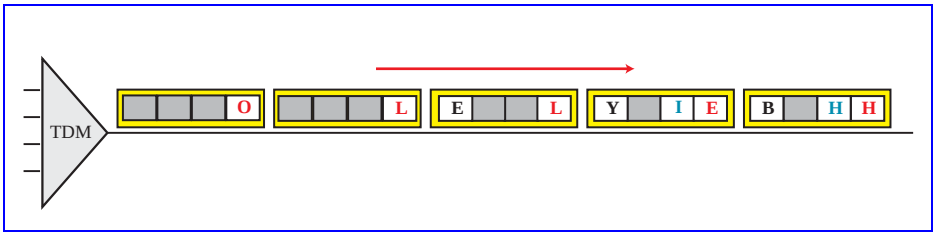
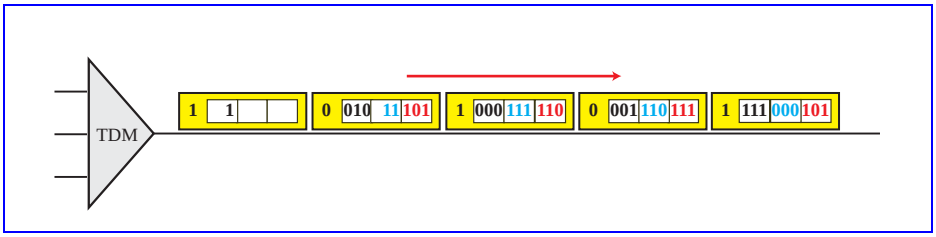
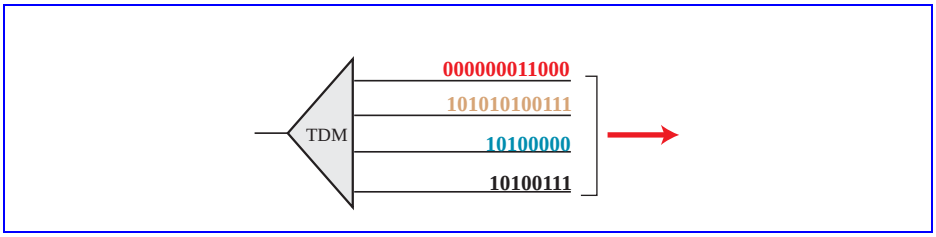


Figure 6.2 *Solution to Exercise 24*



25. See Figure 6.3.

Figure 6.3 *Solution to Exercise 25*



26.

a. DS-1 overhead = 1.544 Mbps – (24 × 64 kbps) = **8 kbps**.

b. DS-2 overhead = 6.312 Mbps – (4 × 1.544 Mbps) = **136 kbps**.

c. DS-3 overhead = 44.376 Mbps – (7 × 6.312 Mbps) = **192 kbps**.

d. DS-4 overhead = 274.176 Mbps – (6 × 44.376 Mbps) = **7.92 Mbps**.

27. The number of hops = 100 KHz/4 KHz = 25. So we need $\log_2 25 = 4.64 \approx$ **5 bits**

28.

a. 2^4 = **16 hops**

b. (64 bits/s) / 4 bits = **16 cycles**

29. Random numbers are 11, 13, 10, 6, 12, 3, 8, 9 as calculated below:

$N_1 = 11$

$N_2 = (5 + 7 \times \mathbf{11}) \bmod 17 - 1 = \mathbf{13}$

$N_3 = (5 + 7 \times \mathbf{13}) \bmod 17 - 1 = \mathbf{10}$

$N_4 = (5 + 7 \times \mathbf{10}) \bmod 17 - 1 = \mathbf{6}$

$$N_5 = (5 + 7 \times \mathbf{6}) \bmod 17 - 1 = \mathbf{12}$$
$$N_6 = (5 + 7 \times \mathbf{12}) \bmod 17 - 1 = \mathbf{3}$$
$$N_7 = (5 + 7 \times \mathbf{3}) \bmod 17 - 1 = \mathbf{8}$$
$$N_8 = (5 + 7 \times \mathbf{8}) \bmod 17 - 1 = \mathbf{9}$$

30. The Barker chip is 11 bits, which means that it increases the bit rate 11 times. A voice channel of 64 kbps needs 11 × 64 kbps = 704 kbps. This means that the bandpass channel can carry (10 Mbps) / (704 kbps) or approximately **14 channels**.

CHAPTER 7

Transmission Media

Solutions to Review Questions and Exercises

Review Questions

1. The ***transmission media*** is located ***beneath the physical layer*** and controlled by the physical layer.
2. The two major categories are ***guided*** and ***unguided*** media.
3. ***Guided media*** have physical boundaries, while ***unguided media*** are unbounded.
4. The three major categories of guided media are ***twisted-pair***, ***coaxial***, and ***fiber-optic*** cables.
5. ***Twisting*** ensures that both wires are equally, but ***inversely***, affected by external influences such as noise.
6. ***Refraction*** and ***reflection*** are two phenomena that occur when a beam of light travels into a less dense medium. When the angle of incidence is less than the critical angle, ***refraction*** occurs. The beam crosses the interface into the less dense medium. When the angle of incidence is greater than the critical angle, ***reflection*** occurs. The beam changes direction at the interface and goes back into the more dense medium.
7. The ***inner core*** of an optical fiber is surrounded by ***cladding***. The core is denser than the cladding, so a light beam traveling through the core is reflected at the boundary between the core and the cladding if the incident angle is more than the critical angle.
8. We can mention three advantages of optical fiber cable over twisted-pair and coaxial cables: ***noise resistance***, ***less signal attenuation***, and ***higher bandwidth.***
9. In ***sky propagation*** radio waves radiate upward into the ionosphere and are then reflected back to earth. In ***line-of-sight propagation*** signals are transmitted in a straight line from antenna to antenna.
10. ***Omnidirectional*** waves are propagated in all directions; ***unidirectional*** waves are propagated in one direction.

Exercises

11. See Table 7.1 (the values are approximate).

Table 7.1 *Solution to Exercise 11*

Distance	*dB at 1 KHz*	*dB at 10 KHz*	*dB at 100 KHz*
1 Km	**–3**	**–5**	**–7**
10 Km	**–30**	**–50**	**–70**
15 Km	**–45**	**–75**	**–105**
20 Km	**–60**	**–100**	**–140**

12. As the Table 7.1 shows, for a specific maximum value of attenuation, the highest frequency decreases with distance. If we consider the bandwidth to start from zero, we can say that the bandwidth decreases with distance. For example, if we can tolerate a maximum attenuation of 50 dB (loss), then we can give the following listing of distance versus bandwidth.

Distance	**Bandwidth**
1 Km	100 KHz
10 Km	50 KHz
15 Km	1 KHz
20 Km	0 KHz

13. We can use Table 7.1 to find the power for different frequencies:

1 KHz	dB = –3	$P_2 = P_1 \times 10^{-3/10}$	= **100.23 mw**
10 KHz	dB = –5	$P_2 = P_1 \times 10^{-5/10}$	= **63.25 mw**
100 KHz	dB = –7	$P_2 = P_1 \times 10^{-7/10}$	= **39.90 mw**

The table shows that the power is reduced 5 times, which may not be acceptable for some applications.

14. See Table 7.2 (the values are approximate).

Table 7.2 *Solution to Exercise 14*

Distance	*dB at 1 KHz*	*dB at 10 KHz*	*dB at 100 KHz*
1 Km	**–3**	**–7**	**–20**
10 Km	**–30**	**–70**	**–200**
15 Km	**–45**	**–105**	**–300**
20 Km	**–60**	**–140**	**–400**

15. As Table 7.2 shows, for a specific maximum value of attenuation, the highest frequency decreases with distance. If we consider the bandwidth to start from zero, we can say that the bandwidth decreases with distance. For example, if we can tol-

erate a maximum attenuation of 50 dB (loss), then we can give the following listing of distance versus bandwidth.

Distance	**Bandwidth**
1 Km	100 KHz
10 Km	1 KHz
15 Km	1 KHz
20 Km	0 KHz

16. We can use Table 7.2 to find the power for different frequencies:

1 KHz	dB = –3	$P_2 = P_1 \times 10^{-3/10}$	= **100.23 mw**
10 KHz	dB = –7	$P_2 = P_1 \times 10^{-7/10}$	= **39.90 mw**
100 KHz	dB = –20	$P_2 = P_1 \times 10^{-20/10}$	= **2.00 mw**

The table shows that power is decreased 100 times for 100 KHz, which is unacceptable for most applications.

17. We can use the formula $f = c / \lambda$ to find the corresponding frequency for each wave length as shown below (c is the speed of propagation):
    a. $B = [(2 \times 10^8)/1000 \times 10^{-9}] - [(2 \times 10^8)/ 1200 \times 10^{-9}] =$ **33 THz**
    b. $B = [(2 \times 10^8)/1000 \times 10^{-9}] - [(2 \times 10^8)/ 1400 \times 10^{-9}] =$ **57 THz**
18.
    a. The ***wave length*** is the ***inverse*** of the ***frequency*** if the propagation speed is fixed (based on the formula $\lambda = c / f$). This means all three figures represent the same thing.
    b. We can change the wave length to frequency. For example, the value 1000 nm can be written as 200 THz.
    c. The vertical-axis units may not change because they represent dB/km.
    d. The curve must be flipped horizontally.
19. See Table 7.3 (The values are approximate).

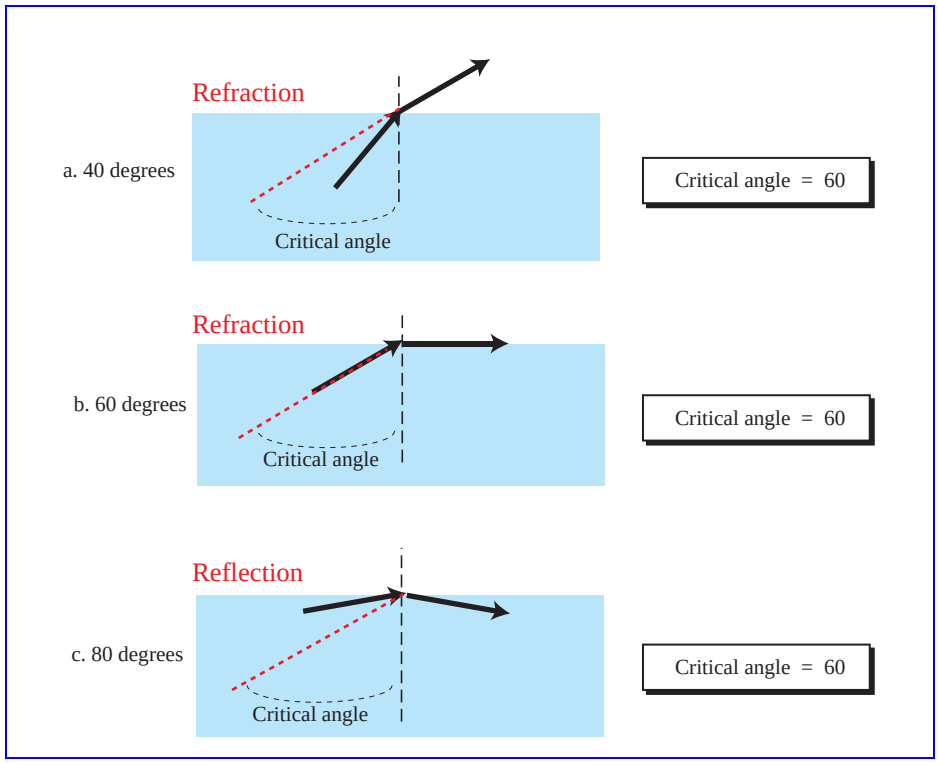
Table 7.3 *Solution to Exercise 19*

Distance	*dB at 800 nm*	*dB at 1000 nm*	*dB at 1200 nm*
1 Km	**–3**	**–1.1**	**–0.5**
10 Km	**–30**	**–11**	**–5**
15 Km	**–45**	**–16.5**	**–7.5**
20 Km	**–60**	**–22**	**–10**

20. The delay = distance / (propagation speed). Therefore, we have:
    a. Delay = $10/(2 \times 10^8) =$ **0.05 ms**
    b. Delay = $100/(2 \times 10^8) =$ **0.5 ms**
    c. Delay = $1000/(2 \times 10^8) =$ **5 ms**

21. See Figure 7.1.

Figure 7.1 *Solution to Exercise 21*



a. The incident angle (40 degrees) is smaller than the critical angle (60 degrees). We have ***refraction***.The light ray enters into the less dense medium.

b. The incident angle (60 degrees) is the same as the critical angle (60 degrees). We have ***refraction***. The light ray travels along the interface.

c. The incident angle (80 degrees) is greater than the critical angle (60 degrees). We have ***reflection***. The light ray returns back to the more dense medium.

CHAPTER 8

Switching

Solutions to Review Questions and Exercises

Review Questions

1. ***Switching*** provides a practical solution to the problem of connecting multiple devices in a network. It is more practical than using a bus topology; it is more efficient than using a star topology and a central hub. Switches are devices capable of creating temporary connections between two or more devices linked to the switch.
2. The three traditional switching methods are ***circuit switching***, ***packet switching***, and ***message switching***. The most common today are ***circuit switching*** and ***packet switching***.
3. There are two approaches to packet switching: ***datagram approach*** and ***virtual-circuit approach***.
4. In a ***circuit-switched network***, data are not packetized; data flow is somehow a continuation of bits that travel the same channel during the data transfer phase. In a ***packet-switched network*** data are packetized; each packet is somehow an independent entity with its local or global addressing information.
5. The address field defines the ***end-to-end*** (source to destination) addressing.
6. The address field defines the ***virtual circuit number*** (local) addressing.
7. In a ***space-division*** switch, the path from one device to another is spatially separate from other paths. The inputs and the outputs are connected using a grid of electronic microswitches. In a ***time-division*** switch, the inputs are divided in time using TDM. A control unit sends the input to the correct output device.
8. ***TSI*** (time-slot interchange) is the most popular technology in a time-division switch. It used random access memory (RAM) with several memory locations. The RAM fills up with incoming data from time slots in the order received. Slots are then sent out in an order based on the decisions of a control unit.
9. In multistage switching, ***blocking*** refers to times when one input cannot be connected to an output because there is no path available between them—all the possible intermediate switches are occupied. One solution to blocking is to increase the number of intermediate switches based on the Clos criteria.

10. A packet switch has four components: ***input ports***, ***output ports***, the ***routing processor***, and the ***switching fabric***. An input port performs the physical and data link functions of the packet switch. The output port performs the same functions as the input port, but in the reverse order. The routing processor performs the function of table lookup in the network layer. The switching fabric is responsible for moving the packet from the input queue to the output queue.

Exercises

11. We assume that the setup phase is a two-way communication and the teardown phase is a one-way communication. These two phases are common for all three cases. The delay for these two phases can be calculated as three propagation delays and three transmission delays or

$$3\ [(5000\ \text{km})/\ (2 \times 10^8\ \text{m/s})] + 3\ [(1000\ \text{bits}/1\ \text{Mbps})] = 75\ \text{ms} + 3\ \text{ms} = \mathbf{78\ ms}$$

We assume that the data transfer is in one direction; the total delay is then

delay for setup and teardown + propagation delay + transmission delay

a. 78 + 25 + 1 = ***104 ms***
b. 78 + 25 + 100 = ***203 ms***
c. 78 + 25 + 1000 = ***1103 ms***
d. In case a, we have 104 ms. In case b we have 203/100 = 2.03 ms. In case c, we have 1103/1000 = 1.101 ms. The ratio for case c is the smallest because we use one setup and teardown phase to send more data.

12. We assume that the transmission time is negligible in this case. This means that we suppose all datagrams start at time 0. The arrival timed are calculated as:

First:	(3200 Km) / (2×10^8 m/s)	+ (3 + 20 + 20)	= **59.0 ms**
Second:	(11700 Km) / (2×10^8 m/s)	+ (3 + 10 + 20)	= **91.5 ms**
Third:	(12200 Km) / (2×10^8 m/s)	+ (3 + 10+ 20 + 20)	= **114.0 ms**
Fourth:	(10200 Km) / (2×10^8 m/s)	+ (3 + 7 + 20)	= **81.0 ms**
Fifth:	(10700 Km) / (2×10^8 m/s)	+ (3 + 7 + 20 + 20)	= **103.5 ms**

The order of arrival is: **3** → **5** → **2** → **4** → **1**

13.

a. In a ***circuit-switched*** network, end-to-end addressing is needed during the setup and teardown phase to create a connection for the whole data transfer phase. After the connection is made, the data flow travels through the already-reserved resources. The switches remain connected for the entire duration of the data transfer; there is no need for further addressing.
b. In a ***datagram network***, each packet is independent. The routing of a packet is done for each individual packet. Each packet, therefore, needs to carry an end-to-end address. There is no setup and teardown phases in a datagram network (connectionless transmission). The entries in the routing table are somehow permanent and made by other processes such as routing protocols.

c. In a ***virtual-circuit*** network, there is a need for end-to-end addressing during the setup and teardown phases to make the corresponding entry in the switching table. The entry is made for each request for connection. During the data transfer phase, each packet needs to carry a virtual-circuit identifier to show which virtual-circuit that particular packet follows.

14. A ***datagram*** or ***virtual-circuit*** network handles packetized data. For each packet, the switch needs to consult its table to find the output port in the case of a datagram network, and to find the combination of the output port and the virtual circuit identifier in the case of a virtual-circuit network. In a ***circuit-switched*** network, data are not packetized; no routing information is carried with the data. The whole path is established during the setup phase.

15. In ***circuit-switched*** and ***virtual-circuit*** networks, we are dealing with connections. A connection needs to be made before the data transfer can take place. In the case of a circuit-switched network, a physical connection is established during the setup phase and the is broken during the teardown phase. In the case of a virtual-circuit network, a virtual connection is made during setup and is broken during the teardown phase; the connection is virtual, because it is an entry in the table. These two types of networks are considered ***connection-oriented***. In the case of a ***datagram*** network no connection is made. Any time a switch in this type of network receives a packet, it consults its table for routing information. This type of network is considered a ***connectionless*** network.

16. The switching or routing in a ***datagram network*** is based on the final destination address, which is global. The minimum number of entries is two; one for the final destination and one for the output port. Here the input port, from which the packet has arrived is irrelevant. The switching or routing in a ***virtual-circuit*** network is based on the virtual circuit identifier, which has a local jurisdiction. This means that two different input or output ports may use the same virtual circuit number. Therefore, four pieces of information are required: input port, input virtual circuit number, output port, and output virtual circuit number.

17.

Packet 1: **2**
Packet 2: **3**
Packet 3: **3**
Packet 4: **2**

18.

Packet 1: **2, 70**
Packet 2: **1, 45**
Packet 3: **3, 11**
Packet 4: **4, 41**

19.

a. In a ***datagram*** network, the destination addresses are unique. They cannot be duplicated in the routing table.

b. In a ***virtual-circuit*** network, the VCIs are local. A VCI is unique only in relationship to a port. In other words, the (port, VCI) combination is unique. This means that we can have two entries with the same input or output ports. We can

have two entries with the same VCIs. However, we cannot have two entries with the same (port, VCI) pair.

20. When a packet arrives at a router in a ***datagram*** network, the only information in the packet that can help the router in its routing is the ***destination address*** of the packet. The table then is sorted to make the searching faster. Today's routers use some sophisticated searching techniques. When a packet arrives at a switch in a ***virtual-circuit*** network, the pair ***(input port, input VCI)*** can uniquely determined how the packet is to be routed; the pair is the only two pieces of information in the packet that is used for routing. The table in the virtual-circuit switch is sorted based on the this pair. However, since the number of port numbers is normally much smaller than the number of virtual circuits assigned to each port, sorting is done in two steps: first according to the input port number and second according to the input VCI.

21.

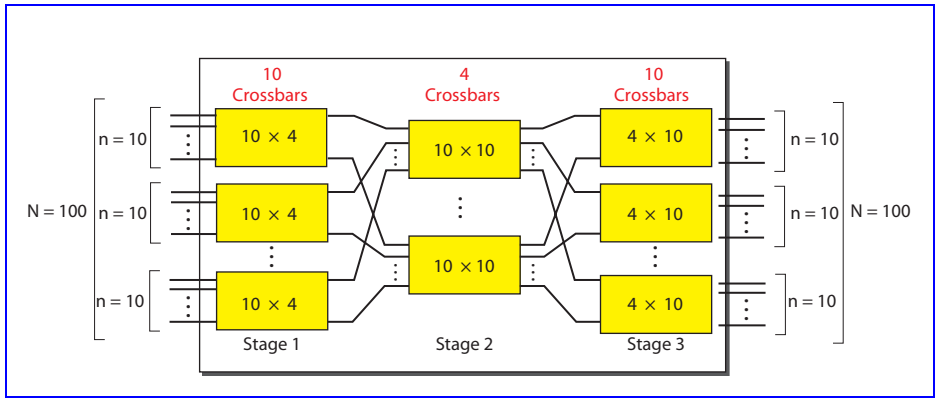
a. If $n > k$, an $n \times k$ crossbar is like a ***multiplexer*** that combines n inputs into k outputs. However, we need to know that a regular multiplexer discussed in Chapter 6 is $n \times 1$.

b. If $n < k$, an $n \times k$ crossbar is like a ***demultiplexer*** that divides n inputs into k outputs. However, we need to know that a regular demultiplexer discussed in Chapter 6 is $1 \times n$.

22.

a. See Figure 8.1.

Figure 8.1 *Solution to Exercise 22 Part a*



b. The total number of crosspoints are

$$\text{Number of crosspoints} = 10\ (10 \times 4) + 4\ (10 \times 10) + 10\ (4 \times 10) = \mathbf{1200}$$

c. Only four simultaneous connections are possible for each crossbar at the first stage. This means that the total number of simultaneous connections is **40.**

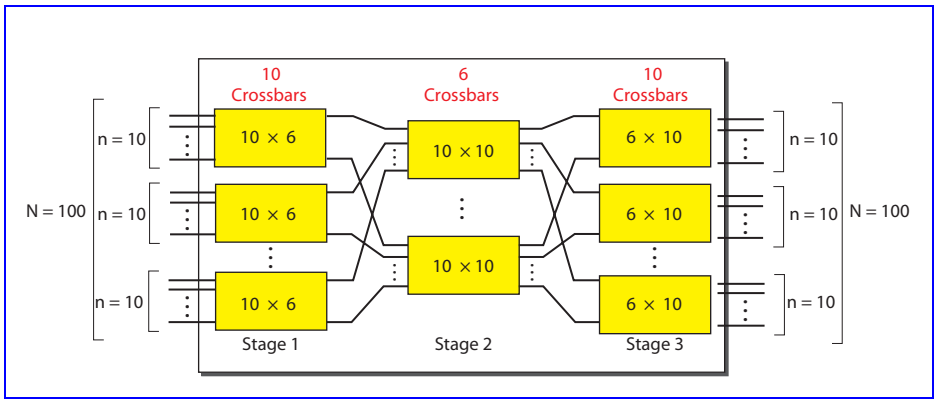
d. If we use one crossbar (100×100), all input lines can have a connection at the same time, which means **100** simultaneous connections.

e. The blocking factor is 40/100 or ***40 percent***.

23.

a. See Figure 8.2.

Figure 8.2 *Solution to Exercise 23 Part a*



b. The total number of crosspoints are

$$\text{Number of crosspoints} = 10\ (10 \times 6) + 6\ (10 \times 10) + 10\ (6 \times 10) = \mathbf{1800}$$

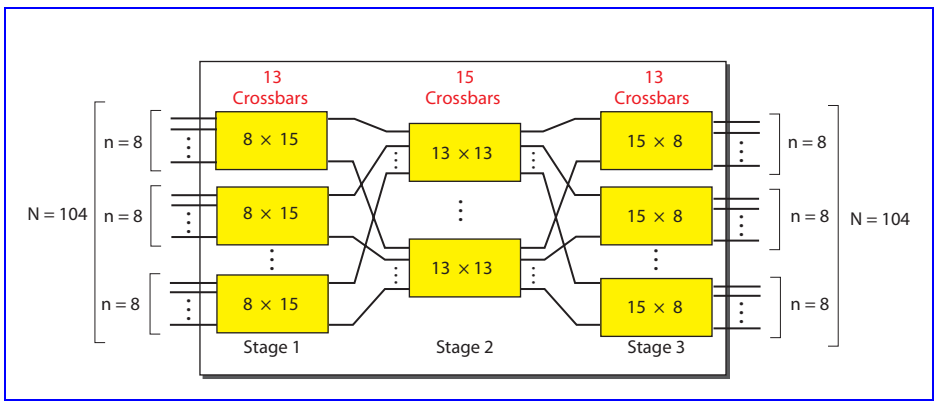
c. Only six simultaneous connections are possible for each crossbar at the first stage. This means that the total number of simultaneous connections is **60**.

d. If we use one crossbar (100 × 100), all input lines can have a connection at the same time, which means **100** simultaneous connections.

e. The blocking factor is 60/100 or ***60 percent.***

24. According to Clos, $n = (N/2)^{1/2} = 7.07$. We can choose n = 8. The number of crossbars in the first stage can be 13 (to have similar crossbars). Some of the input lines can be left unused. We then have $k = 2n - 1 = 15$. Figure 8.3 shows the configuration.

Figure 8.3 *Solution to Exercise 24 Part a*



We can calculate the total number of crosspoints as

$$13\ (8 \times 15) + 15\ (13 \times 13) + 13\ (15 \times 8) = \textbf{\textit{5655}}$$

The number of crosspoints is still much less than the case with one crossbar (10,000). We can see that there is no blocking involved because each 8 input line has 15 intermediate crossbars. The total number of crosspoints here is a little greater than the minimum number of crosspoints according to Clos using the formula $4N[(2N)^{1/2} - 1]$, which is 5257.

25.

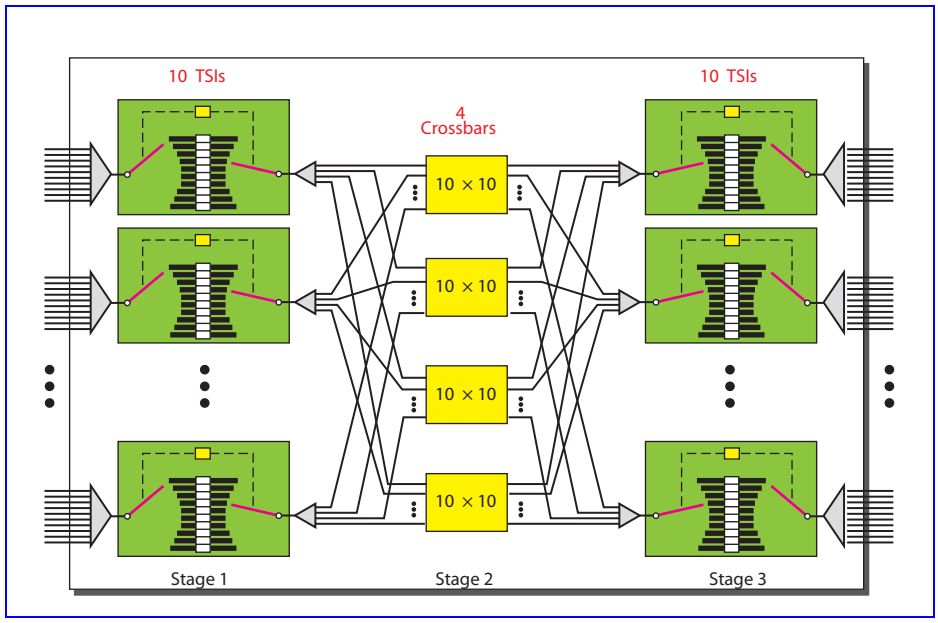
a. Total crosspoints = $N^2 = 1000^2 =$ ***1,000,000***

b. Total crosspoints $\geq 4N[(2N)^{1/2} - 1] \geq$ ***174,886.*** With less than 200,000 crosspoints we can design a three-stage switch. We can use $n = (N/2)^{1/2} = 23$ and choose k = 45. The total number of crosspoints is ***178,200***.

26. We give two solutions.

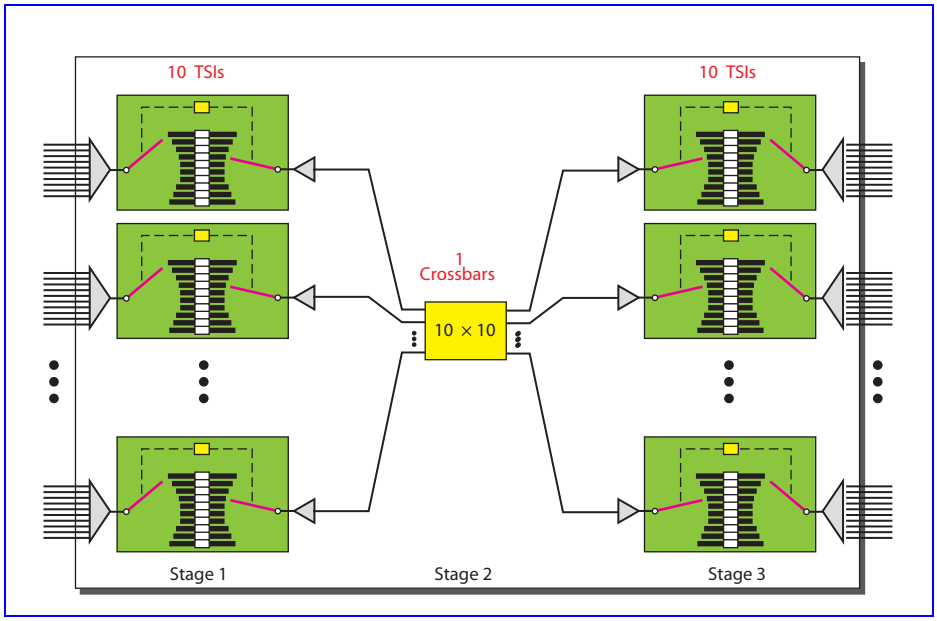
a. We first solve the problem using only crossbars and then we replace the crossbars at the first and the last stage with TSIs. Figure 8.1 shows the solution using only crossbars. We can replace the crossbar at the first and third stages with TSIs as shown in Figure 8.4. The total number of crosspoints is ***400*** and the total number of memory locations is ***200***. Each TSI at the first stage needs one TDM multiplexer and one TDM demultiplexer. The multiplexer is 10×1, but the demultiplexer is 1×4. In other words, the input frame has 10 slots and the output frame has only 4 slots. The data in the first slot of all input TSIs are directed to the first switch, the output in the second slot are directed to the second switch, and so on.

Figure 8.4 *First solution to Exercise 26*



b. We can see the inefficiency in the first solution. Since the slots are separated in time, only one of the switches at the middle stage is active at each moment. This means that, instead of 4 crossbars, we could have used only one with the same result. Figure 8.5 shows the new design. In this case we still need ***200*** memory locations but only ***100*** crosspoints.

Figure 8.5 *Second solution to Exercise 26*



CHAPTER 9

Using Telephone and Cable Networks

Solutions to Review Questions and Exercises

Review Questions

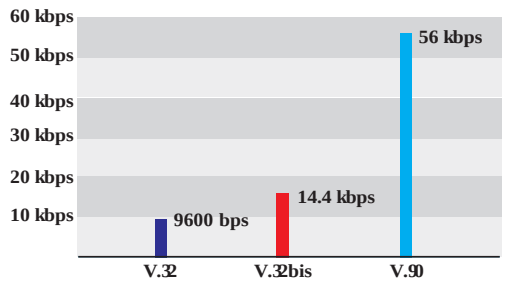
1. The telephone network is made of three major components: ***local loops***, ***trunks***, and ***switching offices***.
2. The telephone network has several levels of switching offices such as ***end offices***, ***tandem offices***, and ***regional offices***.
3. A ***LATA*** is a small or large metropolitan area that according to the divestiture of 1984 was under the control of a single telephone-service provider. The services offered by the common carriers inside a LATA are called intra-LATA services. The services between LATAs are handled by interexchange carriers (IXCs). These carriers, sometimes called long-distance companies, provide communication services between two customers in different LATAs.
4. ***Signaling System Seven (SS7)*** is the protocol used to provide signaling services in the telephone network. It is very similar to the five-layer Internet model.
5. Telephone companies provide two types of services: ***analog*** and ***digital***.
6. ***Dial-up modems*** use part of the bandwidth of the local loop to transfer data. The latest dial-up modems use the V-series standards such as V.32 and V.32bis (9600 bps), V.34bis (28,800 or 33,600 bps), V.90 (56 kbps for downloading and 33.6 kbps for uploading), and V.92. (56 kbps for downloading and 48 kbps for uploading.
7. Telephone companies developed ***digital subscriber line (DSL)*** technology to provide higher-speed access to the Internet. DSL technology is a set of technologies, each differing in the first letter (ADSL, VDSL, HDSL, and SDSL). The set is often referred to as xDSL, where x can be replaced by A, V, H, or S. DSL uses a device called ***ADSL modem*** at the customer site. It uses a device called a ***digital subscriber line access multiplexer (DSLAM)*** at the telephone company site.
8. The ***traditional cable networks*** use only coaxial cables to distribute video information to the customers. The ***hybrid fiber-coaxial (HFC) networks*** use a combination of fiber-optic and coaxial cable to do so.

9. To provide Internet access, the cable company has divided the available bandwidth of the coaxial cable into three bands: video, downstream data, and upstream data. The ***downstream-only video band*** occupies frequencies from 54 to 550 MHz. ***The downstream data*** occupies the upper band, from 550 to 750 MHz. The ***upstream data*** occupies the lower band, from 5 to 42 MHz.
10. The ***cable modem (CM)*** is installed on the subscriber premises. The ***cable modem transmission system (CMTS)*** is installed inside the distribution hub by the cable company. It receives data from the Internet and passes them to the combiner, which sends them to the subscriber. The CMTS also receives data from the subscriber and passes them to the Internet.

Exercises

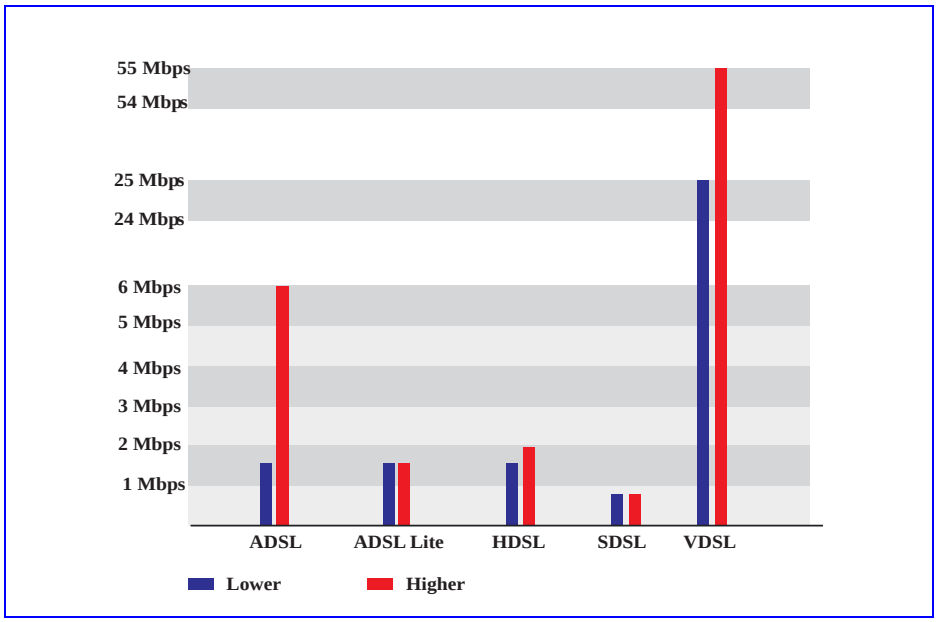
11. ***Packet-switched*** networks are well suited for carrying data in packets. The end-to-end addressing or local addressing (VCI) occupies a field in each packet. Telephone networks were designed to carry voice, which was not packetized. A ***circuit-switched*** network, which dedicates resources for the whole duration of the conversation, is more suitable for this type of communication.
12. ***The setup phase*** can be matched to the dialing process. After the callee responds, the ***data transfer phase*** (here voice transfer phase) starts. When any of the parties hangs up, the data transfer is terminated and the ***teardown phase*** starts. It takes a while before all resources are released.
13. In a telephone network, the ***telephone numbers*** of the caller and callee are serving as source and destination addresses. These are used only during the setup (dialing) and teardown (hanging up) phases.
14. The ***delay*** can be attributed to the fact that some telephone companies use ***satellite*** networks for overseas communication. In these case, the signals need to travel several thousands miles (earth station to satellite and satellite to earth station).
15. See Figure 9.1.

Figure 9.1 *Solution to Exercise 15*



16. See Figure 9.2.

Figure 9.2 *Solution to Exercise 16*



17.

a. V.32 → **Time = (1,000,000 × 8) /9600** ≈ **834 s**
b. V.32bis → **Time = (1,000,000 × 8) / 14400** ≈ **556 s**
c. V.90 → **Time = (1,000,000 × 8) / 56000** ≈ **143 s**

18.

a. ADSL → **Time = (1,000,000 × 8) / 1,500,000** ≈ **5.3 s**
b. ADSL Lite → **Time = (1,000,000 × 8) / 1,500,000** ≈ **5.3 s**
c. HDSL → **Time = (1,000,000 × 8) / 1,500,000** ≈ **5.3 s**
d. SDSL → **Time = (1,000,000 × 8) / 768,000** ≈ **10.42 s**
e. VDSL → **Time = (1,000,000 × 8) / 25,000,000** ≈ **0.32 s**

19. We can calculate time based on the assumption of 10 Mbps data rate:

$$\text{Time} = (1{,}000{,}000 \times 8) / 10{,}000{,}000 \approx 0.8 \text{ seconds}$$

20. The ***DSL*** technology is based on ***star*** topology with the hub at the telephone office. The local loop connects each customer to the end office. This means that there is no sharing; the allocated bandwidth for each customer is not shared with neighbors. The data rate does not depend on how many people in the area are transferring data at the same time.

21. The ***cable modem*** technology is based on the ***bus*** (or rather tree) topology. The cable is distributed in the area and customers have to share the available bandwidth. This means if all neighbors try to transfer data, the effective data rate will be decreased.

CHAPTER 10

Error Detection and Correction

Solutions to Review Questions and Exercises

Review Questions

1. In a ***single bit error*** only one bit of a data unit is corrupted; in a **burst error** more than one bit is corrupted (not necessarily contiguous).
2. ***Redundancy*** is a technique of adding extra bits to each data unit to determine the accuracy of transmission.
3. In ***forward error correction***, the receiver tries to correct the corrupted codeword; in ***error detection by retransmission***, the corrupted message is discarded (the sender needs to retransmit the message).
4. A ***linear block code*** is a block code in which the exclusive-or of any two codewords results in another codeword. A ***cyclic code*** is a linear block code in which the rotation of any codeword results in another codeword.
5. The ***Hamming distance*** between two words (of the same size) is the number of differences between the corresponding bits. The Hamming distance can easily be found if we apply the XOR operation on the two words and count the number of 1s in the result. The ***minimum Hamming distance*** is the smallest Hamming distance between all possible pairs in a set of words.
6. The ***single parity check*** uses one redundant bit for the whole data unit. In a ***two-dimensional parity check***, original data bits are organized in a table of rows and columns. The parity bit is then calculated for each column and each row.
7.
    a. The only relationship between the size of the codeword and dataword is the one based on the definition: **n = k + r**., where ***n*** is the size of the codeword, ***k*** is the size of the dataword, and ***r*** is the size of the remainder.
    b. The ***remainder*** is always ***one bit smaller*** than the ***divisor***.
    c. The ***degree*** of the generator polynomial is ***one less than*** the size of the ***divisor***. For example, the CRC-32 generator (with the polynomial of degree 32) uses a 33-bit divisor.

d. The ***degree*** of the generator polynomial is the ***same as*** the size of the remainder (length of checkbits). For example, CRC-32 (with the polynomial of degree 32) creates a remainder of 32 bits.

8. ***One's complement arithmetic*** is used to add data items in checksum calculation. In this arithmetic, when a number needs more than *n* bits, the extra bits are wrapped and added to the number. In this arithmetic, the complement of a number is made by inverting all bits.
9. ***At least three types of error*** cannot be detected by the current checksum calculation. First, if two data items are swapped during transmission, the sum and the checksum values will not change. Second, if the value of one data item is increased (intentionally or maliciously) and the value of another one is decreased (intentionally or maliciously) the same amount, the sum and the checksum cannot detect these changes. Third, if one or more data items is changed in such a way that the change is a multiple of $2^{16} - 1$, the sum or the checksum cannot detect the changes.
10. ***The value of a checksum can be all 0s*** (in binary). This happens when the value of the sum (after wrapping) becomes all 1s (in binary). ***It is almost impossible for the value of a checksum to be all 1s***. For this to happen, the value of the sum (after wrapping) must be all 0s which means all data units must be 0s.

Exercises

11. We can say that **(vulnerable bits) = (data rate) × (burst duration)**

a.	vulnerable bits	$= (1{,}500) \times (2 \times 10^{-3})$	**= 3 bits**
b.	vulnerable bits	$= (12 \times 10^{3}) \times (2 \times 10^{-3})$	**= 24 bits**
c.	vulnerable bits	$= (100 \times 10^{3}) \times (2 \times 10^{-3})$	**= 200 bits**
d.	vulnerable bits	$= (100 \times 10^{6}) \times (2 \times 10^{-3})$	**= 200,000 bits**

Comment: The last example shows how a noise of small duration can affect so many bits if the data rate is high.

12.

a.	(10001)	⊕	(10000)	= **00001**
b.	(10001)	⊕	(10001)	= **00000**
c.	(11100)	⊕	(00000)	= **11100**
d.	(10011)	⊕	(11111)	= **01100**

Comment: The above shows three properties of the exclusive-or operation. First, the result of XORing two equal patterns is an all-zero pattern (part b). Second, the result of XORing of any pattern with an all-zero pattern is the original non-zero pattern (part c). Third, the result of XORing of any pattern with an all-one pattern is the complement of the original non-one pattern.

13. The codeword for dataword **10** is **101**. This codeword will be changed to **010** if a 3-bit burst error occurs. This pattern is not one of the valid codewords, so the receiver detects the error and discards the received pattern.

14. The codeword for dataword **10** is **10101**. This codeword will be changed to **01001** if a 3-bit burst error occurs. This pattern is not one of the valid codewords, so the receiver discards the received pattern.

15.

a. d (10000, 00000) = **1**

b. d (10101, 10000) = **2**

c. d (1111, 1111) = **0**

d. d (000, 000) = **0**

Comment: Part c and d show that the distance between a codeword and itself is 0.

16.

a. For error detection $\rightarrow$ $d_{min} = s + 1 = 2 + 1 = \mathbf{3}$

b. For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = \mathbf{5}$

c.

For error section $\rightarrow$ $d_{min} = s + 1 = 3 + 1 = \mathbf{4}$
For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = \mathbf{5}$
Therefore d_{min} should be **5**.

d.

For error detection $\rightarrow$ $d_{min} = s + 1 = 6 + 1 = \mathbf{7}$
For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = \mathbf{5}$
Therefore d_{min} should be **7**.

17.

a. **01**

b. **error**

c. **00**

d. **error**

18. We show that the exclusive-or of the second and the third code word

$$(01011) \oplus (10111) = \mathbf{11100}$$

is not in the code. The code is not linear.

19. We check five random cases. All are in the code.

I.	(1st)	$\oplus$	(2nd)	=	(2nd)
II.	(2nd)	$\oplus$	(3th)	=	(4th)
III.	(3rd)	$\oplus$	(4th)	=	(2nd)
IV.	(4th)	$\oplus$	(5th)	=	(8th)
V.	(5th)	$\oplus$	(6th)	=	(2nd)

20. We show the dataword, the codeword, the corrupted codeword, and the interpretation of the receiver for each case:

a. Dataword: **0100** $\rightarrow$ Codeword: **0100011** $\rightarrow$ Corrupted: **0010011**
This pattern is not in the table. $\rightarrow$ **Correctly discarded**.

b. Dataword: **0111** $\rightarrow$ Codeword: **0111001** $\rightarrow$ Corrupted: **1111000**
This pattern is not in the table. $\rightarrow$ **Correctly discarded.**

c. Dataword: **1111** → Codeword: **1111111** → Corrupted: **0101110**
This pattern is in the table. → **Erroneously accepted as 0101.**

d. Dataword: **0000** → Codeword: **0000000** → Corrupted: **1101000**
This pattern is in the table. → **Erroneously accepted as 1101.**

Comment: The above result does not mean that the code can never detect three errors. The last two cases show that it may happen that three errors remain undetected.

21. We show the dataword, codeword, the corrupted codeword, the syndrome, and the interpretation of each case:

a. Dataword: 0100 → Codeword: 0100011 → Corrupted: **1**100011 → $s_2s_1s_0 = 110$
Change b_3 (Table 10.5) → Corrected codeword: **0100011** → dataword: 0100
The dataword is correctly found.

b. Dataword: 0111 → Codeword: 0111001 → Corrupted: **00**11001 → $s_2s_1s_0 = 011$
Change b_2 (Table 10.5) → Corrected codeword: **0111001**→ dataword: 0111
The dataword is correctly found.

c. Dataword: 1111 → Codeword: 1111111 → Corrupted: **0111110** → $s_2s_1s_0 = 111$
Change b_1 (Table 10.5) → Corrected codeword: **0101110**→ dataword: 0101
The dataword is found, but it is **incorrect**. C(7,4) cannot correct two errors.

d. Dataword: 0000 → Codeword: 0000000 → Corrupted: **1**10000**1** → $s_2s_1s_0 = 100$
Change q_2 (Table 10.5) → Corrected codeword: **1100101**→ dataword: 1100
The dataword is found, but it is **incorrect**. C(7,4) cannot correct three errors.

22.

a. If we rotate **0101100** one bit, the result is **0010110**, which is in the code. If we rotate **0101100** two bits, the result is **0001011**, which is in the code. And so on.

b. The XORing of the two codewords (0010110) ⊕ (1111111) = 1101001, which is in the code.

23. We need to find $k = 2^m - 1 - m \geq 11$. We use *trial and error* to find the right answer:

a. Let m = 1 $k = 2^m - 1 - m = 2^1 - 1 - 1 = 0$ (not acceptable)

b. Let m = 2 $k = 2^m - 1 - m = 2^2 - 1 - 2 = 1$ (not acceptable)

c. Let m = 3 $k = 2^m - 1 - m = 2^3 - 1 - 3 = 4$ (not acceptable)

d. Let m = 4 $k = 2^m - 1 - m = 2^4 - 1 - 4 = 11$ (acceptable)

Comment: The code is **C(15, 11)** with $\mathbf{d_{min} = 3}$.

24.

a. $(x^3 + x^2 + x + 1) + (x^4 + x^2 + x + 1) = x^4 + x^3$

b. $(x^3 + x^2 + x + 1) - (x^4 + x^2 + x + 1) = x^4 + x^3$

c. $(x^3 + x^2) \times (x^4 + x^2 + x + 1) = x^7 + x^6 + x^5 + x^2$

d. $(x^3 + x^2 + x + 1) / (x^2 + 1) = x + 1$ (remainder is 0)

25.

a. 101110 → $x^5 + x^3 + x^2 + x$

b. 101110 → 101110**000** (Three 0s are added to the right)

c. $x^3 \times (x^5 + x^3 + x^2 + x) = x^8 + x^6 + x^5 + x^4$

d. 101110 → 10 (The four rightmost bits are deleted)

e. $x^{-4} \times (x^5 + x^3 + x^2 + x) = x$ (Note that negative powers are deleted)

26. To detect single bit errors, a CRC generator must have at least two terms and the coefficient of x^0 must be nonzero.

a. $x^3 + x + 1 \rightarrow$ It meets both critic.

b. $x^4 + x^2 \rightarrow$ It meets the first criteria, but not the second.

c. $1 \rightarrow$ It meets the second criteria, but not the first.

d. $x^2 + 1 \rightarrow$ It meets both criteria.

27. CRC-8 generator is $x^8 + x^2 + x + 1$.

a. It has more than one term and the coefficient of x^0 is 1. It can detect a single-bit error.

b. The polynomial is of degree 8, which means that the number of checkbits (remainder) r = 8. It will detect all burst errors of size 8 or less.

c. Burst errors of size 9 are detected most of the time, but they slip by with probability $(1/2)^{r-1}$ or $(1/2)^{8-1} \approx 0.008$. This means **8 out of 1000** burst errors of size 9 are left undetected.

d. Burst errors of size 15 are detected most of the time, but they slip by with probability $(1/2)^r$ or $(1/2)^8 \approx 0.004$. This means **4 out of 1000** burst errors of size 15 are left undetected.

28. This generator is $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$.

a. It has more than one term and the coefficient of x^0 is 1. It detects all single-bit error.

b. The polynomial is of degree 32, which means that the number of checkbits (remainder) r = 32. It will detect all burst errors of size 32 or less.

c. Burst errors of size 33 are detected most of the time, but they are slip by with probability $(1/2)^{r-1}$ or $(1/2)^{32-1} \approx 465 \times 10^{-12}$. This means **465 out of 10^{12}** burst errors of size 33 are left undetected.

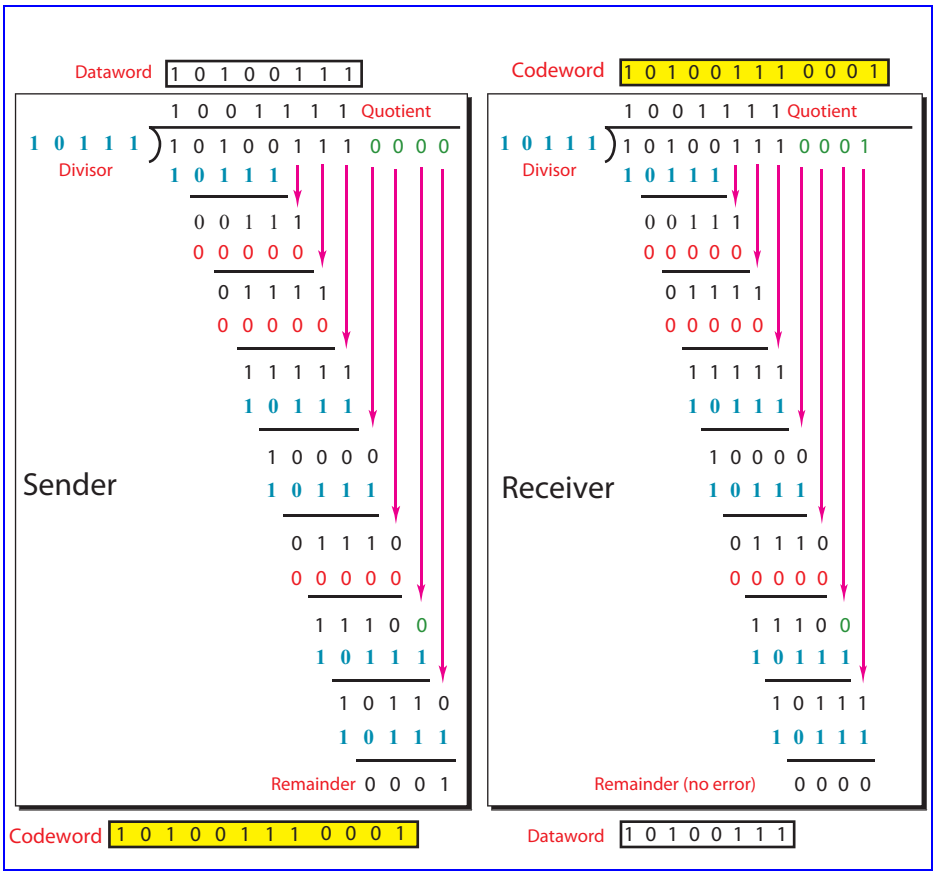
d. Burst errors of size 55 are detected most of the time, but they are slipped with probability $(1/2)^r$ or $(1/2)^{32} \approx 233 \times 10^{-12}$. This means **233 out of 10^{12}** burst errors of size 55 are left undetected.

29. We need to add all bits modulo-2 (XORing). However, it is simpler to count the number of 1s and make them even by adding a 0 or a 1. We have shown the parity bit in the codeword in color and separate for emphasis.

	Dataword		Number of 1s		Parity	Codeword
a.	1001011	→	4 (even)	→	**0**	**0** 1001011
b.	0001100	→	2 (even)	→	**0**	**0** 0001100
c.	1000000	→	1 (odd)	→	**1**	**1** 1000000
d.	1110111	→	6 (even)	→	**0**	**0** 1110111

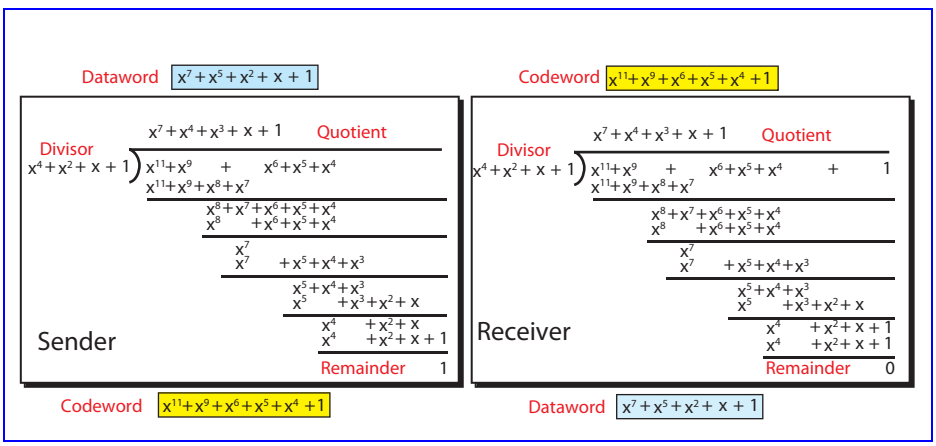
30. Figure 10.1 shows the calculation of the checksum both at the sender and receiver site using binary division.

Figure 10.1 *Solution to Exercise 30*



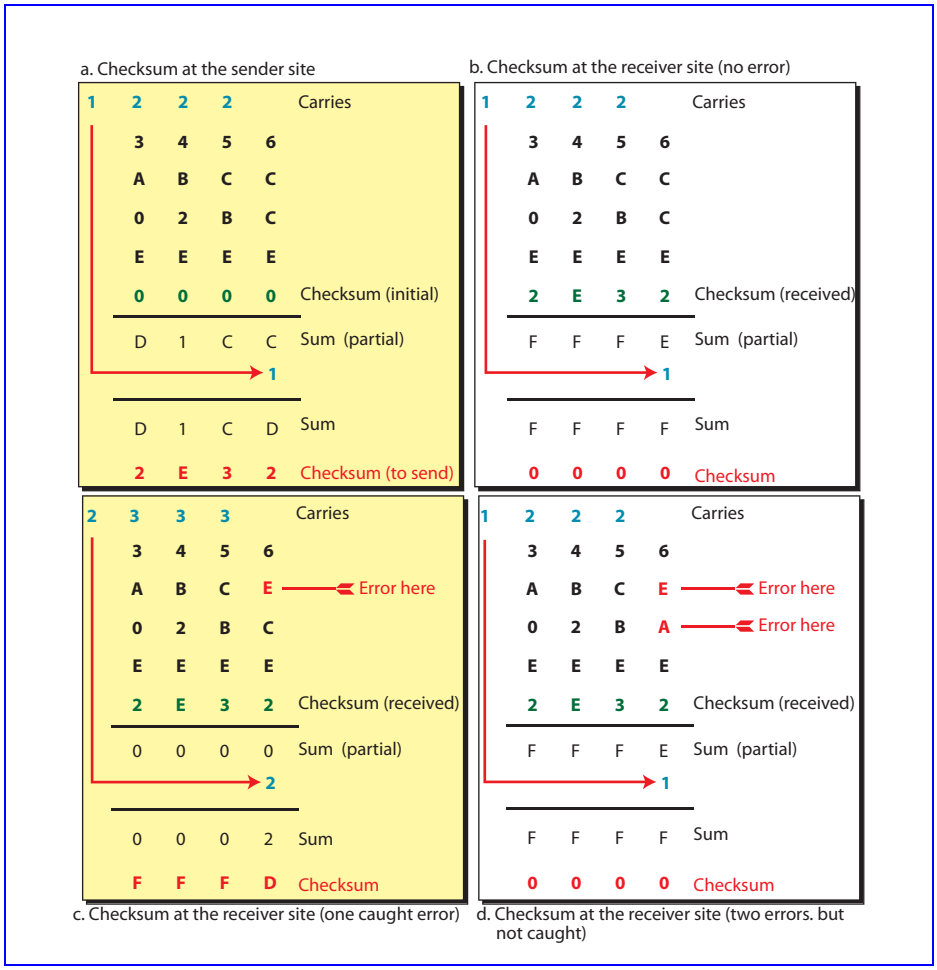
31. Figure 10.2 shows the generation of the codeword at the sender and the checking of the received codeword at the receiver using polynomial division.

Figure 10.2 *Solution to Exercise 31*



32. Figure 10.3 shows the four situations.

Figure 10.3 *Solution to Exercise 32*



a. In part a, we calculate the checksum to be sent (0x2E32)

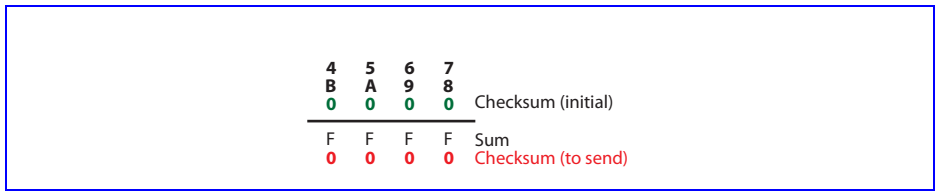
b. In part b, there is no error in transition. The receiver recalculates the checksum to be all 0x0000. The receiver correctly assumes that there is no error.

c. In part c, there is one single error in transition. The receiver calculates the checksum to be 0FFFD. The receiver correctly assumes that there is some error and discards the packet.

d. In part d, there are two errors that cancel the effect of each other. The receiver calculates the checksum to be 0x0000. The receiver erroneously assumes that there is no error and accepts the packet. This is an example that shows that the checksum may slip in finding some types of errors.

33. Figure 10.4 shows the checksum to send (0x0000). This example shows that the checksum ***can be all 0s***. ***It can be all 1s only if all data items are all 0, which means no data at all***.

Figure 10.4 *Solution to Exercise 33*



CHAPTER 11

Data Link Control

Solutions to Review Questions and Exercises

Review Questions

1. The two main functions of the data link layer are ***data link control*** and ***media access control***. Data link control deals with the design and procedures for communication between two adjacent nodes: node-to-node communication. Media access control deals with procedures for sharing the link.
2. The data link layer needs to pack bits into ***frames***. Framing divides a message into smaller entities to make flow and error control more manageable.
3. In a ***byte-oriented protocol***, data to be carried are 8-bit characters from a coding system. Character-oriented protocols were popular when only text was exchanged by the data link layers. In a ***bit-oriented protocol***, the data section of a frame is a sequence of bits. Bit-oriented protocols are more popular today because we need to send text, graphic, audio, and video which can be better represented by a bit pattern than a sequence of characters.
4. Character-oriented protocols use ***byte-stuffing*** to be able to carry an 8-bit pattern that is the same as the flag. Byte-stuffing adds an extra character to the data section of the frame to escape the flag-like pattern. Bit-oriented protocols use ***bit-stuffing*** to be able to carry patterns similar to the flag. Bit-stuffing adds an extra bit to the data section of the frame whenever a sequence of bits is similar to the flag.
5. ***Flow control*** refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment. ***Error control*** refers to a set of procedures used to detect and correct errors.
6. In this chapter, we discussed two protocols for noiseless channels: the ***Simplest*** and the ***Stop-and-Wait***.
7. In this chapter, we discussed three protocols for noisy channels: the ***Stop-and-Wait ARQ***, the ***Go-Back-N ARQ***, and the ***Selective-Repeat ARQ***.
8. Go-Back-N ARQ is more efficient than Stop-and-Wait ARQ. The second uses ***pipelining***, the first does not. In the first, we need to wait for an acknowledgment for each frame before sending the next one. In the second we can send several frames before receiving an acknowledgment.

9. In the ***Go-Back-N ARQ Protocol***, we can send several frames before receiving acknowledgments. If a frame is lost or damaged, all outstanding frames sent before that frame are resent. In the ***Selective- Repeat ARQ protocol*** we avoid unnecessary transmission by sending only the frames that are corrupted or missing. Both Go-Back-*N* and Selective-Repeat Protocols use ***sliding windows***. In Go-Back-*N* ARQ, if m is the number of bits for the sequence number, then the size of the send window must be at most $2^m - 1$; the size of the receiver window is always 1. In Selective-Repeat ARQ, the size of the sender and receiver window must be at most 2^{m-1}.
10. ***HDLC*** is a *bit-oriented protocol* for communication over point-to-point and multipoint links. ***PPP*** is a byte-oriented protocol used for point-to-point links.
11. ***Piggybacking*** is used to improve the efficiency of bidirectional transmission. When a frame is carrying data from A to B, it can also carry control information about frames from B; when a frame is carrying data from B to A, it can also carry control information about frames from A.
12. Only ***Go-Back-N*** and ***Selective-Repeat*** protocols use ***pipelining***.

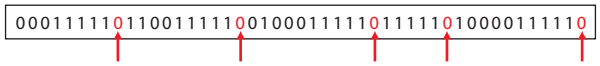
Exercises

13. We give a very simple solution. Every time we encounter an ESC or flag character, we insert an extra ESC character in the data part of the frame (see Figure 11.1).

Figure 11.1 *Solution to Exercise 13*

14. Figure 11.2 shows data to be encapsulated in the frame.

Figure 11.2 *Solution to exercise 14*



15. We write two very simple algorithms. We assume that a frame is made of a one-byte beginning flag, variable-length data (possibly byte-stuffed), and a one-byte ending flag; we ignore the header and trailer. We also assume that there is no error during the transmission.
    a. Algorithm 11.1 can be used at the sender site. It inserts one ESC character whenever a flag or ESC character is encountered.

Algorithm 11.1 *Sender's site solution to Exercise 15*

```
InsertFrame (one-byte flag);       // Insert beginning flag
while (more characters in data buffer)
{
```

Algorithm 11.1 *Sender's site solution to Exercise 15*

```
    ExtractBuffer (character);
    if (character is flag or ESC)    InsertFrame (ESC);    // Byte stuff
    InsertFrame (character);
}
InsertFrame (one-byte flag);     // Insert ending flag
```

b. Algorithm 11.2 can be used at the receiver site.

Algorithm 11.2 *Receiver's site solution to Exercise 15*

```
ExtractFrame (character);    // Extract beginning flag
Discard (character);         // Discard beginning flag
while (more characters in the frame)
{
    ExtractFrame (character);
    if (character = = flag)   exit();          // Ending flag is extracted

    if (character = = ESC)
    {
        Discard (character);       // Un-stuff
        ExtractFrame (character);       // Extract flag or ESC as data
    }
    InsertBuffer (character);
}
Discard (character);          // Discard ending flag
```

16. We write two very simple algorithms. We assume that a frame is made of an 8-bit flag (01111110), variable-length data (possibly bit-stuffed), and an 8-bit ending flag (0111110); we ignore header and trailer. We also assume that there is no error during the transmission.

    a. Algorithm 11.3 can be used at the sender site.

Algorithm 11.3 *Sender's site solution to Exercise 16*

```
InsertFrame (8-bit flag);   // Insert beginning flag
counter = 0;
while (more bits in data buffer)
{
    ExtractBuffer (bit);
    InsertFrame (bit);
    if (bit = = 1)      counter = counter + 1;
    else                counter = 0;

    if (counter = = 5)
    {
        InsertFrame (bit 0);   // Bit stuff
        counter = 0;
    }
}
InsertFrame (8-bit flag);   // Insert ending flag
```

b. Algorithm 11.4 can be used at the receiver's site. Note that when the algorithm exits from the loop, there are six bits of the ending flag in the buffer, which need to be removed after the loop.

Algorithm 11.4 *Receiver's site solution to Exercise 16*

```
ExtractFrame (8 bits);    // Extract beginning flag
counter = 0;
while (more bits in frame)
{
    ExtractFrame (bit);
    if (counter = = 5)
    {
        if (bit is 0)    Discard (bit);      counter = 0;   // Un-stuff
        if (bit is 1)    exit ();    // Flag is encountered
    }

    if (counter < 5)
    {
        if (bit is 0)   counter = 0;
        if (bit is 1)   counter = counter + 1;
        InsertBuffer (bit);
    }
}
ExtractBuffer (last 6 bits);    // These bits are part of flag
Discard (6 bits);
```

17. A five-bit sequence number can create sequence numbers from 0 to 31. The sequence number in the Nth packet is (N mod 32). This means that the 101th packet has the sequence number (101 mod 32) or **5**.

18.

Stop-And-Wait ARQ	send window = **1**	receive window = **1**
Go-Back-N ARQ	send window = $2^5 - 1$ = **31**	receive window = **1**
Selective-Repeat ARQ	send window = 2^4 = **16**	receive window = **16**

19. See Algorithm 11.5. Note that we have assumed that both events (request and arrival) have the same priority.

Algorithm 11.5 *Algorithm for bidirectional Simplest Protocol*

```
while (true)    // Repeat forever
{
    WaitForEvent ();    // Sleep until an event occurs
    if (Event (RequestToSend))    // There is a packet to send
    {
        GetData ();
        MakeFrame ();
        SendFrame ();  // Send the frame
    }

    if (Event (ArrivalNotification))    // Data frame arrived
    {
        ReceiveFrame ();
```

Algorithm 11.5 *Algorithm for bidirectional Simplest Protocol*

```
        ExtractData ();
        DeliverData ();     // Deliver data to network layer
    }
}  // End Repeat forever
```

20. See Algorithm 11.6. Note that in this algorithm, we assume that the arrival of a frame by a site also means the acknowledgment of the previous frame sent by the same site.

Algorithm 11.6

```
while (true)     // Repeat forever
{
    canSend = true;
    WaitForEvent ();    // Sleep until an event occurs
    if (Event (RequestToSend) AND canSend)    // A packet can be sent
    {
        GetData ();
        MakeFrame ();
        SendFrame ();     // Send the frame
        canSend = false;
    }

    if (Event (ArrivalNotification))     // Data frame arrived
    {
        ReceiveFrame ();
        ExtractData ();
        DeliverData ();    // Deliver data to network layer
        canSend = true;
    }
}  // End Repeat forever
```

21. Algorithm 11.7 shows one design. This is a very simple implementation in which we assume that both sites always have data to send.

Algorithm 11.7 *A bidirectional algorithm for Stop-And-Wait ARQ*

```
Sn = 0;    // Frame 0 should be sent first
Rn = 0;   // Frame 0 expected to arrive first
canSend = true;    // Allow the first request to go
while (true)     // Repeat forever
{
    WaitForEvent ();    // Sleep until an event occurs
    if (Event (RequestToSend) AND canSend)   // Packet to send
    {
        GetData ();
        MakeFrame (Sn , Rn);   // The seqNo of frame is Sn
        StoreFrame (Sn , Rn);    //Keep copy for possible resending
        SendFrame (Sn , Rn);
        StartTimer ();
        Sn = (Sn + 1) mod 2;
        canSend = false;
    }
```

Algorithm 11.7 *A bidirectional algorithm for Stop-And-Wait ARQ*

```
    if (Event (ArrivalNotification))    // Data frame arrives
    {
        ReceiveFrame ();
        if (corrupted (frame))    sleep();
        if (seqNo = = Rn)     // Valid data frame
        {
            ExtractData ();
            DeliverData ();     // Deliver data
            Rn = (Rn + 1) mod 2;
        }
        if (ackNo = = Sn)     // Valid ACK
        {
            StopTimer ();
            PurgeFrame (Sn–1 , Rn–1);    //Copy is not needed
            canSend = true;
        }
    }

    if (Event(TimeOut))   //  The timer expired
    {
      StartTimer ();
      ResendFrame (Sn-1 , Rn-1);     // Resend a copy
    }
}    // End Repeat forever
```

22. Algorithm 11.8 shows one design. This is a very simple implementation in which we assume that both sites always have data to send.

Algorithm 11.8 *Bidirectional algorithm for Go-Back-And-N algorithm*

```
Sw = 2m – 1;
Sf = 0;
Sn = 0;
Rn = 0;
while (true)   // Repeat forever
{
    WaitForEvent ();
    if (Event (RequestToSend))     // There is a packet to send
    {
        if (Sn – Sf >= Sw)    Sleep();    // If window is full
        GetData();
        MakeFrame (Sn , Rn);
        StoreFrame (Sn, Rn);
        SendFrame (Sn, Rn);
        Sn = Sn + 1;
        if (timer not running)    StartTimer ();
    }

    if (Event (ArrivalNotification))
    {
        Receive (Frame);
        if (corrupted (ACK))      Sleep();
        if ((ackNo > Sf) AND (ackNo <= Sn))   // If a valid ACK
        {
```

Algorithm 11.8 *Bidirectional algorithm for Go-Back-And-N algorithm*

```
            while (Sf <= ackNo)
            {
                PurgeFrame (Sf);
                Sf = Sf + 1;
            }
            StopTimer ();
        }
        if (seqNo = = Rn)   // If expected frame
        {
            DeliverData ();    // Deliver data
            Rn = Rn + 1;    // Slide window one slot
            SendACK (Rn);
        }
    }

    if (Event (TimeOut))    // The timer expires
    {
        StartTimer ();
        Temp = Sf;
        while (Temp < Sn);
        {
            SendFrame (Sf);
            Sf = Sf + 1;
        }
    }
}   // End Repeat forever
```

23. Algorithm 11.9 shows one design. This is a very simple implementation in which we assume that both sites always have data to send.

Algorithm 11.9 *A bidirectional algorithm for Selective-Repeat ARQ*

```
Sw = 2m–1;
Sf = 0;
Sn = 0;
Rn = 0;
NakSent = false;
AckNeeded = false;
Repeat (for all slots);
Marked (slot) = false;
 while (true)    // Repeat forever
 {
    WaitForEvent ();
    if (Event (RequestToSend))     // There is a packet to send
    {
        if (Sn–Sf >= Sw)    Sleep ();    // If window is full
        GetData ();
        MakeFrame (Sn , Rn);
        StoreFrame (Sn , Rn);
        SendFrame (Sn , Rn);
        Sn = Sn + 1;
        StartTimer (Sn);
    }
```

Algorithm 11.9 *A bidirectional algorithm for Selective-Repeat ARQ*

```
if (Event (ArrivalNotification))
{
    Receive (frame);   // Receive Data or NAK
    if (FrameType is NAK)
    {
        if (corrupted (frame))      Sleep();
        if (nakNo between Sf and Sn)
        {
            resend (nakNo);
            StartTimer (nakNo);
        }
    }

    if (FrameType is Data)
    {
        if (corrupted (Frame)) AND (NOT NakSent)
        {
            SendNAK (Rn);
            NakSent = true;
            Sleep();
        }

        if (ackNo between Sf and Sn)
        {
            while (Sf < ackNo)
            {
                Purge (Sf);
                StopTimer (Sf);
                Sf = Sf + 1;
            }
        }

        if ((seqNo <> Rn) AND (NOT NakSent))
        {
            SendNAK (Rn);
            NakSent = true;
        }

        if ((seqNo in window) AND (NOT Marked (seqNo))
        {
            StoreFrame (seqNo);
            Marked (seqNo) = true;
            while (Marked (Rn))
            {
                DeliverData (Rn);
                Purge (Rn);
                Rn = Rn + 1;
                AckNeeded = true;
            }
        }
    }   // End if (FrameType is Data)
}   // End if (arrival event)

if (Event (TimeOut (t)))    // The timer expires
```

Algorithm 11.9 *A bidirectional algorithm for Selective-Repeat ARQ*

```
    {
        StartTimer (t);
        SendFrame (t);
    }
} // End Repeat forever
```

24. State $S_n = 0$ means the sender has sent Frame 1, but is waiting for the acknowledgment. State $S_n = 1$ means the sender has sent Frame 0, but is waiting for the acknowledgment. We can then say

Event A:	**Sender Site:**	ACK 0 received.
Event B:	**Sender Site:**	ACK 1 received.

25. State $R_n = 0$ means the receiver is waiting for Frame 0. State $R_n = 1$ means the receiver is waiting for Frame 1. We can then say

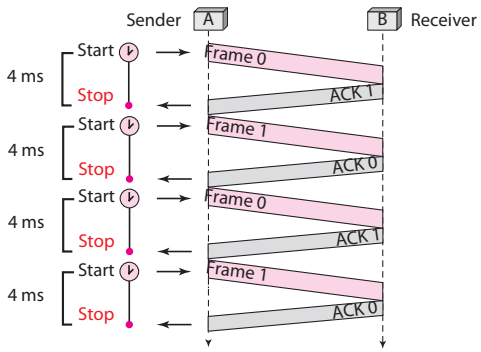
Event A:	**Receiver Site:**	Frame 0 received.
Event B:	**Receiver Site:**	Frame 1 received.

26. We can say that in this case, each state defines that a frame or an acknowledgment in transit. In other words,
$(1 , 0) \rightarrow$ Frame 0 is in transit $\quad (1 , 1) \rightarrow$ ACK 1 is in transit
$(0 , 1) \rightarrow$ Frame 1 is in transit $\quad (0 , 0) \rightarrow$ ACK 0 is in transit

Event A:	**Receiver Site:**	Frame 0 arrives and ACK 1 is sent.
Event B:	**Sender Site:**	ACK 1 arrives and Frame 1 is sent.
Event C:	**Receiver Site:**	Frame 1 arrives and ACK 0 is sent.
Event D:	**Sender Site:**	ACK 0 arrives and Frame 0 is sent.

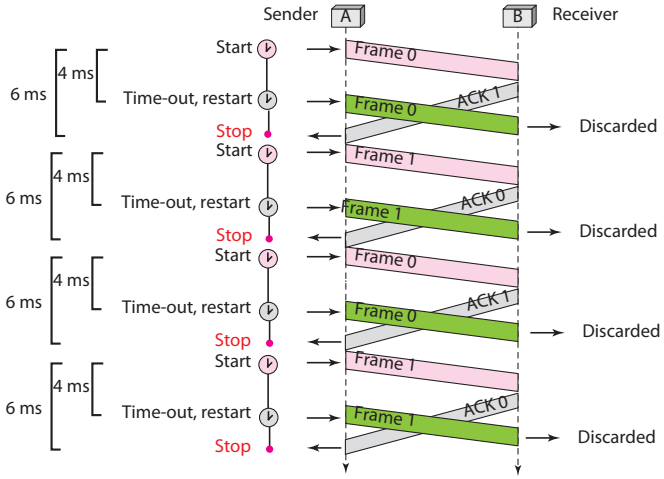
27. Figure 11.3 shows the situation. Since there are no lost or damaged frames and the round trip time is less than the time-out, each frame is sent only once.

Figure 11.3 *Solution to Exercise 27*



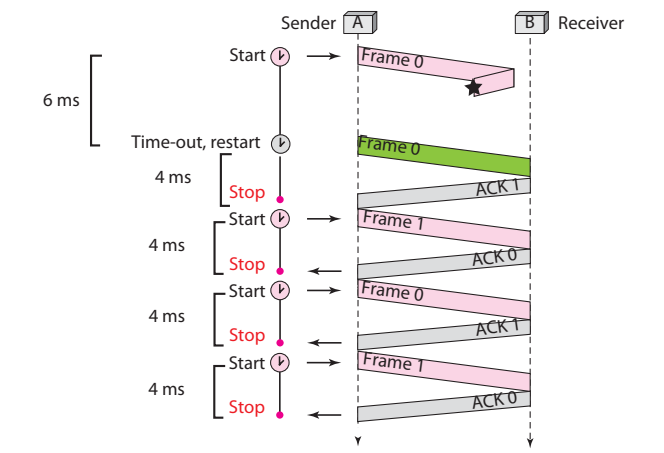
28. Figure 11.4 shows the situation. Here, we have a special situation. Although no frame is damaged or lost, the sender sends each frame twice. The reason is that the the acknowledgement for each frame reaches the sender after its timer expires. The sender thinks that the frame is lost.

Figure 11.4 *Solution to Exercise 28*



29. Figure 11.5 shows the situation. In this case, only the first frame is resent; the acknowledgment for other frames arrived on time.

Figure 11.5 *Solution to Exercise 29*



30. We need to send 1000 frames. We ignore the overhead due to the header and trailer.

    Data frame Transmission time = 1000 bits / 1,000,000 bits = 1 ms
    Data frame trip time = 5000 km / 200,000 km = 25 ms
    ACK transmission time = 0 (It is usually negligible)
    ACK trip time = 5000 km / 200,000 km = 25 ms
    Delay for 1 frame = 1 + 25 + 25 = 51 ms.
    Total delay = 1000 × 51 = 51 s

31. In the worst case, we send the a full window of size 7 and then wait for the acknowledgment of the whole window. We need to send 1000/7 ≈ 143 windows. We ignore the overhead due to the header and trailer.

    Transmission time for one window = 7000 bits / 1,000,000 bits = 7 ms
    Data frame trip time = 5000 km / 200,000 km = 25 ms
    ACK transmission time = 0 (It is usually negligible)
    ACK trip time = 5000 km / 200,000 km = 25 ms
    Delay for 1 window = 7 + 25 + 25 = 57 ms.
    Total delay = 143 × 57 ms = **8.151 s**

32. In the worst case, we send the a full window of size 4 and then wait for the acknowledgment of the whole window. We need to send 1000/4 = 250 windows. We ignore the overhead due to the header and trailer.

    Transmission time for one window = 4000 bits / 1,000,000 bits = 4 ms
    Data frame trip time = 5000 km / 200,000 km = 25 ms
    ACK transmission time = 0 (It is usually negligible)
    ACK trip time = 5000 km / 200,000 km = 25 ms
    Delay for 1 window = 4 + 25 + 25 = 54 ms.
    Total delay = 250 × 54 ms = **13.5 s**

CHAPTER 12

Multiple Access

Solutions to Review Questions and Exercises

Review Questions

1. The three categories of multiple access protocols discussed in this chapter are ***random access***, ***controlled access***, and ***channelization***.
2. In ***random access*** methods, no station is superior to another station and none is assigned the control over another. Each station can transmit when it desires on the condition that it follows the predefined procedure. Three common protocols in this category are ***ALOHA***, ***CSMA/CD***, and ***CSMA/CA***.
3. In ***controlled access methods***, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by other stations. We discuss three popular controlled-access methods: ***reservation***, ***polling***, and ***token passing***.
4. ***Channelization*** is a multiple-access method in which the available bandwidth of a link is shared in time, frequency, or through code, between different stations. The common protocols in this category are ***FDMA***, ***TDMA***, and ***CDMA***.
5. In ***random access*** methods, there is no access control (as there is in controlled access methods) and there is no predefined channels (as in channelization). Each station can transmit when it desires. This liberty may create ***collision***.
6. In a ***random access*** method, there is no control; access is based on ***contention***. In a ***controlled access*** method, either a central authority (in polling) or other stations (in reservation and token passing) control the access. Random access methods have less administration overhead. On the other hand, controlled access method are collision free.
7. In a ***random access*** method, the whole available bandwidth belongs to the station that wins the contention; the other stations needs to wait. In a ***channelization*** method, the available bandwidth is divided between the stations. If a station does not have data to send, the allocated channel remains idle.
8. In a ***controlled access*** method, the whole available bandwidth belongs to the station that is granted permission either by a central authority or by other stations. In a ***channelization*** method, the available bandwidth is divided between the stations. If a station does not have data to send the allocated channel remains idle.

9. We do not need a multiple access method in this case. The local loop provides a dedicated ***point-to-point*** connection to the telephone office.
10. We do need a multiple access, because a channel in the CATV band is normally shared between several neighboring customers. The cable company uses the ***random access*** method to share the bandwidth between neighbors.

Exercises

11. To achieve the maximum efficiency in pure ALOHA, G = 1/2. If we let ***ns*** to be the number of stations and ***nfs*** to be the number of frames a station can send per second.

$$G = \mathbf{ns} \times \mathbf{nfs} \times T_{fr} = 100 \times \mathbf{nfs} \times 1\ \mu s = 1/2 \rightarrow \mathbf{nfs} = \mathbf{5000\ frames/s}$$

The reader may have noticed that the T_{fr} is very small in this problem. This means that either the data rate must be very high or the frames must be very small.

12. To achieve the maximum efficiency in slotted ALOHA, G = 1. If we let ***ns*** to be the number of stations and ***nfs*** to be the number of frames a station can send per second.

$$G = \mathbf{ns} \times \mathbf{nfs} \times T_{fr} = 100 \times \mathbf{nfs} \times 1\ \mu s = 1 \rightarrow \mathbf{nfs} = \mathbf{10{,}000\ frames/s}$$

The reader may have noticed that the T_{fr} is very small in this problem. This means that either the data rate must be very high or the frames must be very small.

13. We can first calculate T_{fr} and G, and then the throughput.

$$T_{fr} = (1000\ bits) / 1\ Mbps = 1\ ms$$

$$G = \mathbf{ns} \times \mathbf{nfs} \times T_{fr} = 100 \times 10 \times 1\ ms = 1$$

$$\text{For } \mathbf{pure\ ALOHA} \rightarrow S = G \times e^{-2G} \approx \mathbf{13.53\ percent}$$

This means that each station can successfully send only 1.35 frames per second.

14. We can first calculate T_{fr} and G, and then the throughput.

$$T_{fr} = (1000\ bits) / 1\ Mbps = 1\ ms$$

$$G = \mathbf{ns} \times \mathbf{nfs} \times T_{fr} = 100 \times 10 \times 1\ ms = 1$$

$$\text{For } \mathbf{slotted\ ALOHA} \rightarrow S = G \times e^{-G} \approx \mathbf{36.7\ percent}$$

This means that each station can successfully send only 3.67 frames per second.

15. Let us find the relationship between the minimum frame size and the data rate. We know that

$$T_{fr} = (\text{frame size}) / (\text{data rate}) = 2 \times T_p = 2 \times \text{distance} / (\text{propagation speed})$$

or

$$(\text{frame size}) = [2 \times (\text{distance}) / (\text{propagation speed})] \times (\text{data rate})$$

or

$$\mathbf{(frame\ size) = K \times (data\ rate)}$$

This means that minimum frame size is proportional to the data rate (K is a constant). When the data rate is increased, the frame size must be increased in a network with a fixed length to continue the proper operation of the CSMA/CD. In Example 12.5, we mentioned that the minimum frame size for a data rate of 10 Mbps is 512 bits. We calculate the minimum frame size based on the above proportionality relationship

Data rate =	10 Mbps	→	minimum frame size = **512 bits**
Data rate =	100 Mbps	→	minimum frame size = **5120 bits**
Data rate =	1 Gbps	→	minimum frame size = **51,200 bits**
Data rate =	10 Gbps	→	minimum frame size = **512,000 bits**

16. Let us find the relationship between the collision domain (maximum length of the network) and the data rate. We know that

$$T_{fr} = (\text{frame size}) / (\text{data rate}) = 2 \times T_p = 2 \times \text{distance} / (\text{propagation speed})$$

or

$$\text{distance} = [(\text{frame size})(\text{propagation speed})] / [2 \times (\text{data rate})]$$

or

distance = K / (data rate)

This means that distance is inversely proportional to the data rate (K is a constant). When the data rate is increased, the distance or maximum length of network or collision domain is decreased proportionally. In Example 12.5, we mentioned that the maximum distance for a data rate of 10 Mbps is 2500 meters. We calculate the maximum distance based on the above proportionality relationship.

Data rate =	10 Mbps	→	maximum distance = **2500 m**
Data rate =	100 Mbps	→	maximum distance = **250 m**
Data rate =	1 Gbps	→	maximum distance = **25 m**
Data rate =	10 Gbps	→	maximum distance = **2.5 m**

This means that when the data rate is very high, it is almost impossible to have a network using CSMA/CD.

17. We have $t_1 = 0$ and $t_2 = 3\ \mu s$
    a. $t_3 - t_1 = (2000\ m) / (2 \times 10^8\ m/s) = 10\ \mu s \rightarrow t_3 = 10\ \mu s + t_1 =$ **10 μs**
    b. $t_4 - t_2 = (2000\ m) / (2 \times 10^8\ m/s) = 10\ \mu s \rightarrow t_4 = 10\ \mu s + t_2 =$ **13 μs**
    c. $T_{fr(A)} = t_4 - t_1 = 13 - 0 = 13\ \mu s \rightarrow Bits_A = 10\ Mbps \times 13\ \mu s =$ **130 bits**
    d. $T_{fr(C)} = t_3 - t_2 = 10 - 3 = 07 \mu s \rightarrow Bits_C = 10\ Mbps \times 07\ \mu s =$ **70 bits**
18. We have $t_1 = 0$ and $t_2 = 3\ \mu s$
    a. $t_3 - t_1 = (2000\ m) / (2 \times 10^8\ m/s) = 10\ \mu s \rightarrow t_3 = 10\ \mu s + t_1 =$ **10 μs**
    b. $t_4 - t_2 = (2000\ m) / (2 \times 10^8\ m/s) = 10\ \mu s \rightarrow t_4 = 10\ \mu s + t_2 =$ **13 μs**
    c. $T_{fr(A)} = t_4 - t_1 = 13 - 0 = 13\ \mu s \rightarrow Bits_A = 100\ Mbps \times 13\ \mu s =$ **1300 bits**
    d. $T_{fr(C)} = t_3 - t_2 = 10 - 3 = 07\ \mu s \rightarrow Bits_C = 100\ Mbps \times 07\ \mu s =$ **700 bits**

Note that in this case, both stations have already sent more bits than the minimum number of bits required for detection of collision. The reason is that with the 100 Mbps, the minimum number of bits requirement is feasible only when the maximum distance between stations is less than or equal to 250 meters as we will see in Chapter 13.

19. See Figure 12.1.

Figure 12.1 *Solution to Exercise 19*

$$W_8 = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & +1 & -1 & -1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$$

20. See Figure 12.2.

Figure 12.2 *Solution to Exercise 20*

$$W_1 = \begin{bmatrix} -1 \end{bmatrix} \qquad W_2 = \begin{bmatrix} -1 & -1 \\ -1 & +1 \end{bmatrix} \qquad W_4 = \begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & +1 & -1 & +1 \\ -1 & -1 & +1 & +1 \\ -1 & +1 & +1 & -1 \end{bmatrix}$$

21.

Third Property: we calculate the inner product of each row with itself:

$$\begin{array}{llll} \text{Row 1} \bullet \text{Row 1} & [+1\ +1\ +1\ +1] & \bullet\ [+1\ +1\ +1\ +1] & = +1+1+1+1 = \mathbf{4} \\ \text{Row 2} \bullet \text{Row 2} & [+1\ -1\ +1\ -1] & \bullet\ [+1\ -1\ +1\ -1] & = +1+1+1+1 = \mathbf{4} \\ \text{Row 3} \bullet \text{Row 1} & [+1\ +1\ -1\ -1] & \bullet\ [+1\ +1\ -1\ -1] & = +1+1+1+1 = \mathbf{4} \\ \text{Row 4} \bullet \text{Row 4} & [+1\ -1\ -1\ +1] & \bullet\ [+1\ -1\ -1\ +1] & = +1+1+1+1 = \mathbf{4} \end{array}$$

Fourth Property: we need to prove 6 relations:

$$\begin{array}{llll} \text{Row 1} \bullet \text{Row 2} & [+1\ +1\ +1\ +1] & \bullet\ [+1\ -1\ +1\ -1] & = +1-1+1-1 = \mathbf{0} \\ \text{Row 1} \bullet \text{Row 3} & [+1\ +1\ +1\ +1] & \bullet\ [+1\ +1\ -1\ -1] & = +1+1-1-1 = \mathbf{0} \\ \text{Row 1} \bullet \text{Row 4} & [+1\ +1\ +1\ +1] & \bullet\ [+1\ -1\ -1\ +1] & = +1-1-1+1 = \mathbf{0} \\ \text{Row 2} \bullet \text{Row 3} & [+1\ -1\ +1\ -1] & \bullet\ [+1\ +1\ -1\ -1] & = +1-1-1+1 = \mathbf{0} \\ \text{Row 2} \bullet \text{Row 4} & [+1\ -1\ +1\ -1] & \bullet\ [+1\ -1\ -1\ +1] & = +1+1-1-1 = \mathbf{0} \\ \text{Row 3} \bullet \text{Row 4} & [+1\ +1\ -1\ -1] & \bullet\ [+1\ -1\ -1\ +1] & = +1-1+1-1 = \mathbf{0} \end{array}$$

22.

Third Property: we calculate the inner product of each row with itself:

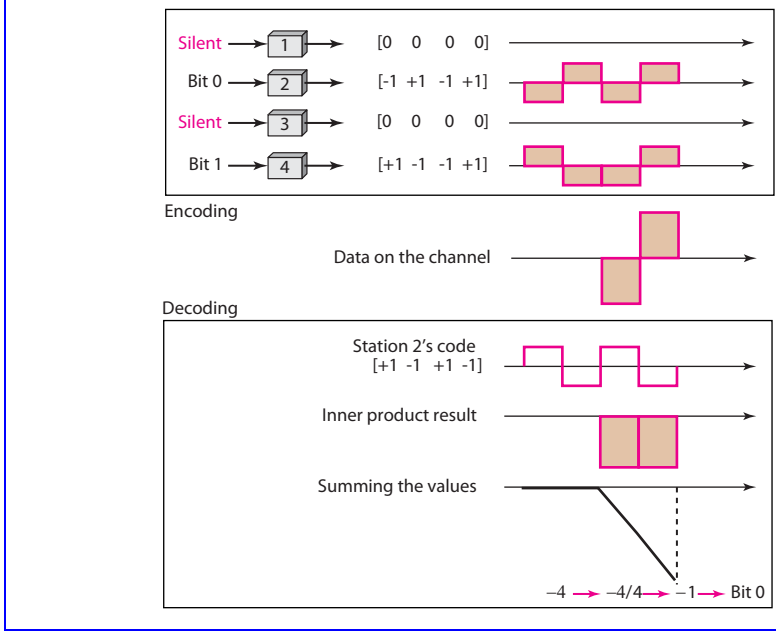
Row 1 • Row 1	[–1 –1 –1 –1]	•	[–1 –1 –1 –1]	= +1 + 1 + 1 + 1 = **4**
Row 2 • Row 2	[–1 +1 –1 +1]	•	[–1 +1 –1 +1]	= +1 + 1 + 1 + 1 = **4**
Row 3 • Row 1	[–1 –1 +1 +1]	•	[–1 –1 +1 +1]	= +1 + 1 + 1 + 1 = **4**
Row 4 • Row 4	[–1 +1 +1 –1]	•	[–1 +1 +1 –1]	= +1 + 1 + 1 + 1 = **4**

Fourth Property: we neede to prove 6 relations:

Row 1 • Row 2	[–1 –1 –1 –1]	•	[–1 +1 –1 +1]	= +1 – 1 + 1 – 1 = **0**
Row 1 • Row 3	[–1 –1 –1 –1]	•	[–1 –1 +1 +1]	= +1 + 1 – 1 – 1 = **0**
Row 1 • Row 4	[–1 –1 –1 –1]	•	[–1 +1 +1 –1]	= +1 – 1 – 1 + 1 = **0**
Row 2 • Row 3	[–1 +1 –1 +1]	•	[–1 –1 +1 +1]	= +1 – 1 – 1 + 1 = **0**
Row 2 • Row 4	[–1 +1 –1 +1]	•	[–1 +1 +1 –1]	= +1 + 1 – 1 – 1 = **0**
Row 3 • Row 4	[–1 –1 +1 +1]	•	[–1 +1 +1 –1]	= +1 – 1 + 1 – 1 = **0**

23. Figure 12.3 shows the encoding, the data on the channel, and the decoding.

Figure 12.3 *Solution to Exercise 23*



24. We can say:

Polling and Data Transfer

Station 1: [poll + 5 × (frame + ACK)]
Station 2: [poll + 5 × (frame + ACK)]
Station 3: [poll + 5 × (frame + ACK)]
Station 4: [poll + 5 × (frame + ACK)]

Polling and Sending NAKs

Station 1: [poll + NAK]
Station 2: [poll + NAK]
Station 3: [poll + NAK]
Station 4: [poll + NAK]

Total Activity:

8 polls + 20 frames + 20 ACKs + 4 NAKs = **21024 bytes**

We have 1024 bytes of overhead.

25. We can say:

Polling and Data Transfer

Frame 1 for all four stations: 4 × [poll + frame + ACK)]
Frame 2 for all four stations: 4 × [poll + frame + ACK)]
Frame 3 for all four stations: 4 × [poll + frame + ACK)]
Frame 4 for all four stations: 4 × [poll + frame + ACK)]
Frame 5 for all four stations: 4 × [poll + frame + ACK)]

Polling and Sending NAKs

Station 1: [poll + NAK]
Station 2: [poll + NAK]
Station 3: [poll + NAK]
Station 4: [poll + NAK]

Total Activity:

24 polls + 20 frames + 20 ACKs + 4 NAKs = **21536 bytes**

We have 1536 bytes of overhead which is 512 bytes more than the case in Exercise 23. The reason is that we need to send 16 extra polls.

CHAPTER 13

Local Area Networks: Ethernet

Solutions to Review Questions and Exercises

Review Questions

1. The ***preamble*** is a 56-bit field that provides an alert and timing pulse. It is added to the frame at the physical layer and is not formally part of the frame. SFD is a one-byte field that serves as a flag.
2. An ***NIC*** provides an Ethernet station with a 6-byte physical address. Most of the physical and data-link layer duties are done by the NIC.
3. A ***multicast address*** identifies a group of stations; a ***broadcast address*** identifies all stations on the network. A ***unicast*** address identifies one of the addresses in a group.
4. A ***bridge*** can raise the bandwidth and separate collision domains.
5. A ***layer-2 switch*** is an N-port bridge with additional sophistication that allows faster handling of packets.
6. In a ***full-duplex Ethernet***, each station can send data without the need to sense the line.
7. The rates are as follows:

Standard Ethernet:	**10 Mbps**
Fast Ethernet:	**100 Mbps**
Gigabit Ethernet:	**1 Gbps**
Ten-Gigabit Ethernet:	**10 Gbps**

8. The common traditional Ethernet implementations are ***10Base5***, ***10Base2***, ***10-Base-T***, and ***10Base-F***.
9. The common Fast Ethernet implementations are ***100Base-TX***, ***100Base-FX***, and ***100Base-T4***.
10. The common Gigabit Ethernet implementations are ***1000Base-SX***, 1000Base-LX, ***1000Base-CX***, and ***1000Base-T***.
11. The common Ten-Gigabit Ethernet implementations are ***10GBase-S***, ***10GBase-L***, and ***10GBase-E***.

Exercises

12. We interpret each four-bit pattern as a hexadecimal digit. We then group the hexadecimal digits with a colon between the pairs:

    5A:11:55:18:AA:0F

13. The bytes are sent from left to right. However, the bits in each byte are sent from the least significant (rightmost) to the most significant (leftmost). We have shown the bits with spaces between bytes for readability, but we should remember that that bits are sent without gaps. The arrow shows the direction of movement.

    ← **01011000 11010100 00111100 11010010 01111010 11110110**

14. The first byte in binary is 0000011**1**. The least significant bit is 1. This means that the pattern defines a **multicast address.**

15. The first byte in binary is 0100000**1**. The least significant bit is 1. This means that the pattern defines a multicast address. ***A multicast address can be a destination address, but not a source address.*** Therefore, the receiver knows that there is an error, and discards the packet.

16. The minimum data size in the Standard Ethernet is 46 bytes. Therefore, we need to add **4 bytes of padding** to the data (46 – 42 = 4)

17. The maximum data size in the Standard Ethernet is 1500 bytes. The data of 1510 bytes, therefore, must be split between two frames. The standard dictates that the first frame must carry the maximum possible number of bytes (1500); the second frame then needs to carry only 10 bytes of data (it requires padding). The following shows the breakdown:

    Data size for the first frame: **1500 bytes**
    Data size for the second frame: **46 bytes** (with padding)

18. The smallest Ethernet frame is 64 bytes and carries 46 bytes of data (and possible padding). The largest Ethernet frame is 1518 bytes and carries 1500 bytes of data. The ratio is (data size) / (frame size) in percent. We can then answer the question as follows:

Smallest Frame	Frame size = 64	Data size ≤ 46	**Ratio ≤ 71.9%**
Largest Frame	Frame size =1518	Data size = 1500	**Ratio = 98.8%**

19. We can calculate the propagation time as t = (2500 m) / (200,000.000) = **12.5 μs**. To get the total delay, we need to add propagation delay in the equipment (10 μs). This results in **T = 22.5 μs.**

20. The smallest frame is 64 bytes or 512 bits. With a data rate of 10 Mbps, we have

    $$T_{fr} = (512 \text{ bits}) / (10 \text{ Mbps}) = \mathbf{51.2\ \mu s}$$

    This means that the time required to send the smallest frame is the same at the maximum time required to detect the collision.

CHAPTER 14

Wireless LANs

Solutions to Review Questions and Exercises

Review Questions

1. The ***basic service set (BSS)*** is the building block of a wireless LAN. A BSS without an AP is called an ad hoc architecture; a BSS with an AP is sometimes referred to as an infrastructure network. An ***extended service set (ESS)*** is made up of two or more BSSs with APs. In this case, the BSSs are connected through a distribution system, which is usually a wired LAN.
2. A station with ***no-transition*** mobility is either stationary or moving only inside a BSS. A station with ***BSS-transition*** mobility can move from one BSS to anther, but the movement is confined inside one ESS. A station with ***ESS-transition*** mobility can move from one ESS to another.
3. The ***orthogonal frequency-division multiplexing (OFDM)*** method for signal generation in a 5-GHz ISM band is similar to ***frequency division multiplexing (FDM)***, with one major difference: All the subbands are used by one source at a given time. Sources contend with one another at the data link layer for access.
4. Stations on wireless LANs normally use ***CSMA/CA***.
5. ***Network Allocation Vector (NAV)*** forces other stations to defer sending their data if one station acquires access. In other words, it provides the collision avoidance aspect. When a station sends an RTS frame, it includes the duration of time that it needs to occupy the channel. The stations that are affected by this transmission create a timer called a NAV.
6. A Bluetooth network is called a ***piconet***. A ***scatternet*** is two or more piconets.
7. The following shows the relationship:

Radio layer	→	**Internet physical layer**
Baseband layer	→	**MAC sublayer of Internet data link layer**
L2CAP layer	→	**LLC sublayer of Internet data link layer**

8. A Bluetooth primary and secondary can be connected by a ***synchronous connection-oriented (SCO)*** link or an ***asynchronous connectionless (ACL)*** link. An SCO link is used when avoiding latency (delay in data delivery) is more important than

integrity (error-free delivery). An ACL link is used when data integrity is more important than avoiding latency.

9. The primary sends on the ***even-numbered*** slots; the secondary sends on the ***odd-numbered*** slots.
10. In all types of frames, a duration of **259 μs** is used for hopping.

Exercises

11. In ***CSMA/CD***, the protocol allows collisions to happen. If there is a collision, it will be detected, destroyed, and the frame will be resent. ***CSMA/CA*** uses a technique that prevents collision.
12. See Table 14.1.

Table 14.1 *Exercise 12*

Fields	*802.3 field size (bytes)*	*802.11 field size (bytes)*
Destination Address	6	
Source Address	6	
Address 1		6
Address 2		6
Address 3		6
Address 4		6
FC		2
D/ID		2
SC		2
PDU Length	2	
Data and Padding	46 to 1500	
Frame Body	64-1518	0 to 2312
FCS (CRC)	4	4

CHAPTER 15

Connecting LANs, Backbone Networks, and Virtual Networks

Solutions to Review Questions and Exercises

Review Questions

1. An ***amplifier*** amplifies the signal, as well as noise that may come with the signal, whereas a ***repeater*** regenerates the signal, bit for bit, at the original strength.
2. ***Bridges*** have access to station ***physical addresses*** and can forward a packet to the appropriate segment of the network. In this way, they ***filter*** traffic and help control congestion.
3. A ***transparent bridge*** is a bridge in which the stations are completely unaware of the bridge's existence. If a bridge is added or deleted from the system, reconfiguration of the stations is unnecessary.
4. A signal can only travel so far before it becomes corrupted. A ***repeater*** regenerates the original signal; the signal can continue to travel and the LAN length is thus extended.
5. A ***hub*** is a ***multiport repeater***.
6. A ***forwarding port*** forwards a frame that it receives; a ***blocking port*** does not.
7. In a ***bus backbone***, the topology of the backbone is a ***bus***; in a ***star backbone***, the topology is a ***star***.
8. A ***VLAN*** saves time and money because reconfiguration is done through software. Physical reconfiguration is not necessary.
9. Members of a ***VLAN*** can send broadcast messages with the assurance that users in other groups will not receive these messages.
10. A ***VLAN*** creates virtual workgroups. Each workgroup member can send broadcast messages to others in the workgroup. This eliminates the need for multicasting and all the overhead messages associated with it.
11. Stations can be grouped by ***port number***, ***MAC address***, ***IP address***, or by a combination of these characteristics.

Exercises

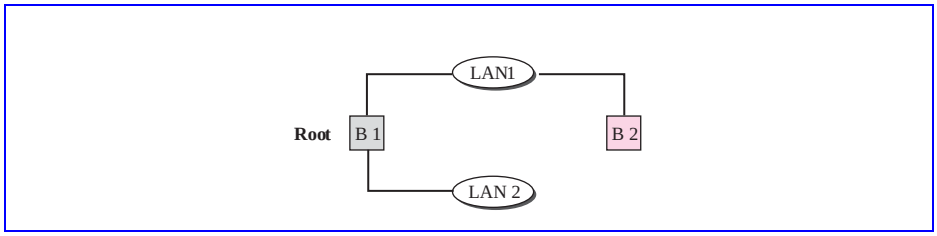
12. Table 15.1 shows one possibility. We have sorted the table based on the physical address to make the searching faster.

Table 15.1 *Solution to Exercise 12*

Address	*Port*
A	1
B	1
C	2
D	2
E	3
F	3

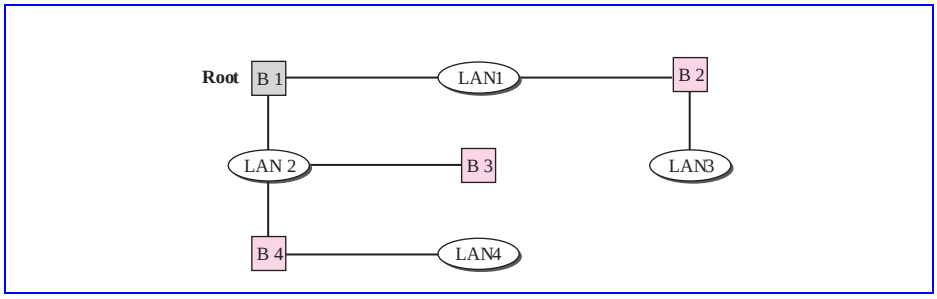
13. Figure 15.1 shows one possible solution. We made bridge B1 the root.

Figure 15.1 *Solution to Exercise 13*



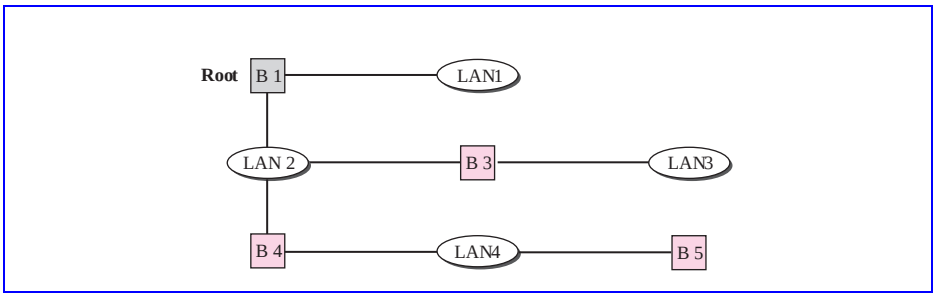
14. Figure 15.2 shows one possible solution.

Figure 15.2 *Solution to Exercise 14*



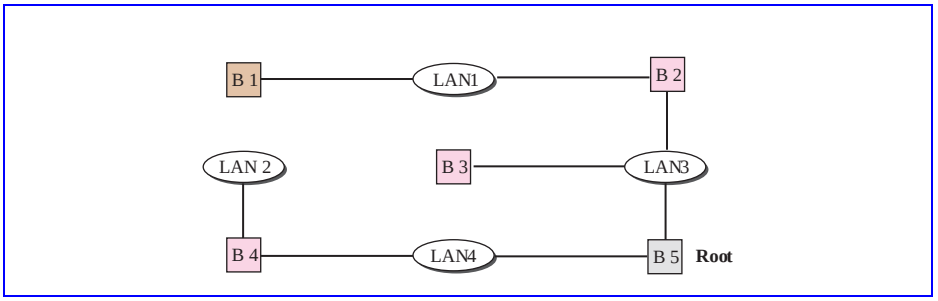
15. Figure 15.3 shows one possible solution.

Figure 15.3 *Solution to Exercise 15*



16. Figure 15.4 shows one possible solution.

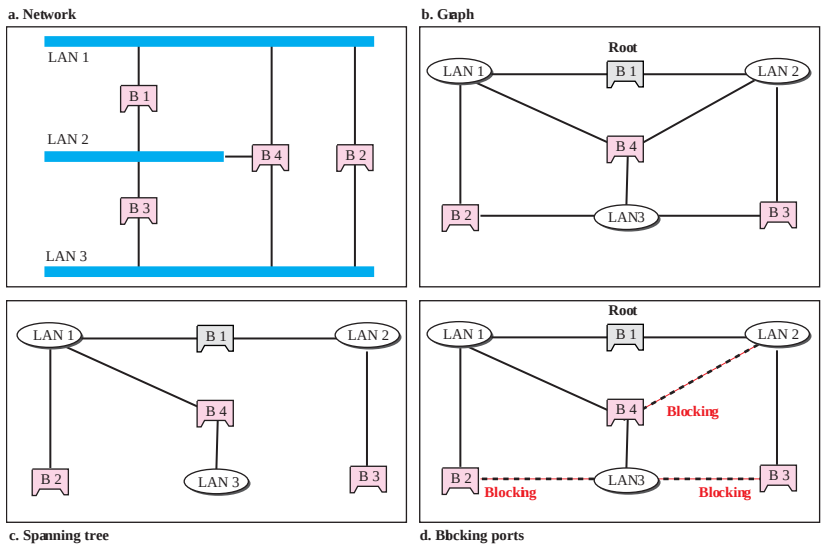
Figure 15.4 *Solution to Exercise 16*



17. Although any ***router*** is also a ***bridge***, replacing bridges with routers has the following consequences:
    a. Routers are more expensive than bridges.
    b. Routers operate at the first three-layers; bridges operates at the first two layers. Routers are not designed to provide direct filtering the way the bridges do. A router needs to search a routing table which is normally longer and more time consuming than a filtering table.
    c. A router needs to decapsulate and encapsulate the frame and change physical addresses in the frame because the physical addresses in the arriving frame define the previous node and the current router; they must be changed to the physical addresses of the current router and the next hop. A bridge does not change the physical addresses. Changing addresses, and other fields, in the frame means much unnecessary overhead.
18. A ***filtering table*** is based on ***physical addresses***; a ***routing table*** is based on the ***logical addresses***.

19. Figure 15.5 shows one possible solution. We have shown the network, the graph, the spanning tree, and the blocking ports.

Figure 15.5 *Solution to Exercise 19*



20. A ***router*** has more ***overhead*** than a bridge. A router process the packet at ***three layers***; a bridge processes a frame at only ***two layers***. A router needs to search a routing table for finding the output port based on the best route to the final destination; A bridge needs only to consult a filtering table based on the location of stations in a local network. A routing table is normally longer than a filtering table; searching a routing table needs more time than searching a filtering table. A router changes the physical addresses; a bridge does not.
21. A ***bridge*** has more overhead than a ***repeater***. A ***bridge*** processes the packet at ***two layers***; a ***repeater*** processes a frame at ***only one layer***. A bridge needs to search a table and find the forwarding port as well as to regenerate the signal; a repeater only regenerates the signal. In other words, a bridge is also a repeater (and more); a repeater is not a bridge.
22. A ***gateway*** has more overhead than a ***router***. A ***gateway*** processes the packet at ***five layers***; a ***router*** processes a packet at ***only three layers***. A gateway needs to worry about the format of the packet at the transport and application layers; a router does not. In other words, a gateway is also a router (but more); a router is not a gateway. A gateway may need to change the port addresses and application addresses if the gateway connects two different systems together; a router does not change these addresses.

CHAPTER 16

Cellular Telephone and Satellite Networks

Solutions to Review Questions and Exercises

Review Questions

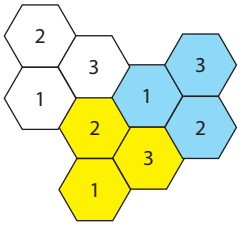
1. A ***mobile switching center*** coordinates communications between a ***base station*** and a ***telephone central office***.
2. A ***mobile switching*** center connects cells, records call information, and is responsible for billing.
3. A ***high reuse factor*** is better because the cells that use the same set of frequencies are farther apart (separated by more cells).
4. In a ***hard handoff***, a mobile station communicates with only one base station. In a ***soft handoff***, a mobile station communicates with two base stations at the same time.
5. ***AMPS*** is an analog cellular phone system using FDMA.
6. ***D-AMPS*** is a digital cellular phone system that is backward compatible with AMPS.
7. ***GSM*** is a European standard that provides a common second-generation technology for all of Europe.
8. ***CDMA*** encodes each traffic channel using one of the rows in the Walsh-64 table.
9. The three orbit types are ***equatorial***, ***inclined***, and ***polar***.
10. A ***GEO*** satellite has an equatorial orbit since the satellite needs to remain fixed at a certain spot above the earth.
11. A ***footprint*** is the area on earth at which the satellite aims its signal.
12. A satellite orbiting in a ***Van Allen belt*** would be destroyed by the charged particles. Therefore, satellites need to orbit either above or below these belts.
13. Transmission from the earth to the satellite is called the ***uplink***. Transmission from the satellite to the earth is called the ***downlink***.
14. ***GPS*** is a satellite system that provides land and sea navigation data for vehicles and ships. The system is also used for clock synchronization.

15. The main difference between ***Iridium*** and ***Globalstar*** is the relaying mechanism. Iridium requires relaying between satellites. Globalstar requires relaying between satellites and earth stations.

16.1 EXERCISES

16. Figure 16.1 shows one possibility.

Figure 16.1 *Exercise 16*



17. ***In AMPS***, there are two separate bands for each direction in communication. In each band, we have 416 analog channels. Out of this number, 21 channels are reserved for control. With a reuse factor of 7, the maximum number of simultaneous calls in each cell is

$$\text{Maximum number of simultaneous calls} = (416 - 21) / 7 = 56.4 \approx 56$$

18. In ***D-AMPS***, there is only one band with 832 channels. Since duplexing is provided at the digital level, this means that 832 analog channels are available in each cell (assuming no control channels). With a reuse factor of 7 and the fact that each analog channel combines three duplex digital channels, the maximum number of simultaneous calls in each cell is

$$\text{Maximum number of simultaneous calls} = [(832) \times 3] / 7 = 356.57 \approx \mathbf{356}$$

19. In ***GSM***, separate bands are assigned for each direction in communication. This means 124 analog channels are available in each cell (assuming no control channels). Each analog channel carries 1 multiframe. Each multiframe carries 26 frames (2 frames are for control). Each frame allows 8 calls. With a reuse factor of 3, we have

$$\text{Maximum number of simultaneous calls} = [(124) \times 24 \times 8] / 3 = \mathbf{7936}$$

20. In ***IS-95***, separate bands are assigned for each direction in communication. This means 20 analog channels are available in each cell (assuming no control channels). Each analog channel carries 64 digital traffic channel (9 channels are for control). With a reuse factor of 1, we have

$$\text{Maximum number of simultaneous calls} = [(20) \times 55] / 1 = \mathbf{1100}$$

21. In Exercise 17, we showed that the maximum simultaneous calls per cell for ***APMS*** is 56. Using the total bandwidth of 50 MHz (for both directions), we have

$$\text{Efficiency} = 56 / 50 = \mathbf{1.12\ calls/MHz}$$

22. In Exercise 18, we showed that the maximum simultaneous calls per cell for D-***APMS*** is 356. Using the total bandwidth of 50 MHz (for both directions), we have

$$\text{Efficiency} = 356 / 50 = \mathbf{7.12\ calls/MHz}$$

23. In Exercise 19, we showed that the maximum simultaneous calls per cell for ***GSM*** is 7936. Using the total bandwidth of 50 MHz (for both directions), we have

$$\text{Efficiency} = 7936 / 50 = \mathbf{158.72\ calls/MHz}$$

24. In Exercise 20, we showed that the maximum simultaneous calls per cell for ***IS-95*** is 1100. Using the total bandwidth of 50 MHz (for both directions), we have

$$\text{Efficiency} = 1100 / 50 = \mathbf{22\ calls/MHz}$$

25. A 3-KHz voice signal is modulated using FM to create a 30-KHz analog signal. As we learned in Chapter 5, the bandwidth required for FM can be determined from the bandwidth of the audio signal using the formula $B_{FM} = 2(1 + \beta)B$. ***AMPS*** uses $\beta = 5$. This means $\mathbf{B_{FM} = 10 \times B}$.
26. ***D-AMPS*** sends 25 frames per seconds in each channel. Each frame carries 6 slots. This means that the total number of slots in each channel is **150 slots/s**. Each frame is shared by three users, which means each user sends **50 slots/s.**
27. ***GPS*** satellites are orbiting at 18,000 km above the earth surface. Considering the radius of the earth, the radius of the orbit is then (18,000 km + 6378 km) = 24,378 km. Using the Kepler formula, we have

$$\text{Period} = (1/100)\ (\text{distance})^{1.5} = (1/100)\ (24{,}378)^{1.5} = 38062\ \text{s} = \mathbf{10.58\ hours}$$

28. ***Iridium*** satellites are orbiting at 750 km above the earth surface. Considering the radius of the earth, the radius of the orbit is then (750 km + 6378 km) = 7128 km. Using the Kepler formula, we have

$$\text{Period} = (1/100)\ (\text{distance})^{1.5} = (1/100)\ (7128)^{1.5} = 6017\ \text{s} = \mathbf{1.67\ hours}$$

29. ***Globalstar*** satellites are orbiting at 1400 km above the earth surface. Considering the radius of the earth, the radius of the orbit is then (1400 km + 6378 km) = 7778 km. Using the Kepler formula, we have

$$\text{Period} = (1/100)\ (\text{distance})^{1.5} = (1/100)\ (7778)^{1.5} = 6860\ \text{s} = \mathbf{1.9\ hours}$$

CHAPTER 17

SONET/SDH

Solutions to Review Questions and Exercises

Review Questions

1. The ANSI standard is called ***SONET*** and the ITU-T standard is called ***SDH***. The standards are nearly identical.
2. SONET defines a hierarchy of electrical signaling levels called ***synchronous transport signals (STSs)***. SDH specifies a similar system called a ***synchronous transport module (STM).***
3. ***STS multiplexers/demultiplexers*** mark the beginning points and endpoints of a SONET link. An STS multiplexer multiplexes signals from multiple electrical sources and creates the corresponding optical signal. An STS demultiplexer demultiplexes an optical signal into corresponding electric signals. ***Add/drop multiplexers*** allow insertion and extraction of signals in an STS. An add/drop multiplexer can add an electrical signals into a given path or can remove a desired signal from a path.
4. ***STSs*** are the hierarchy of electrical signals defined by the SONET standards. ***OCs*** are the corresponding optical signals.
5. ***Pointers*** are used to show the ***offset*** of the SPE in the frame or for ***justification***. SONET uses two pointers show the position of an SPE with respect to an STS. SONET use the third pointer for rate adjustment between SPE and STS.
6. A ***single clock*** handles the timing of transmission and equipment across the entire network.
7. A ***regenerator*** takes a received optical signal and regenerates it. The SONET regenerator also replaces some of the existing overhead information with new information.
8. SONET defines four layers: ***path***, ***line***, ***section***, and ***photonic***.
9. The ***path layer*** is responsible for the movement of a signal from its source to its destination. The ***line layer*** is responsible for the movement of a signal across a physical line. The ***section layer*** is responsible for the movement of a signal across a physical section. The ***photonic layer*** corresponds to the physical layer of the OSI model. It includes physical specifications for the optical fiber channel. SONET

uses NRZ encoding with the presence of light representing 1 and the absence of light representing 0.

10. ***A virtual tributary*** is a partial payload that can be inserted into an STS-1 and combined with other partial payloads to fill out the frame. Instead of using all 86 payload columns of an STS-1 frame for data from one source, we can subdivide the SPE and call each component a VT.

Exercises

11. Each STS-*n* frame carries $(9 \times \boldsymbol{n} \times 86)$ bytes of bytes. SONET sends 8000 frames in each second. We can then calculate the user data rate as follows:

$$\begin{array}{lcll} \text{STS-}\mathbf{3} & \rightarrow & 8000 \times (9 \times \mathbf{3} \times 86) \times 8 & = \mathbf{148.608\ Mbps} \\ \text{STS-}\mathbf{9} & \rightarrow & 8000 \times (9 \times \mathbf{9} \times 86) \times 8 & = \mathbf{445.824\ Mbps} \\ \text{STS-}\mathbf{12} & \rightarrow & 8000 \times (9 \times \mathbf{12} \times 86) \times 8 & = \mathbf{594.432\ Mbps} \end{array}$$

12. To create one STS-36 from four STS-9s, we first need to demultiplex each STS-9 into nine STS-1s. We can then multiplex thirty-six STS-1s into one STS-36. However, there is no extra overhead involved in the process of demultiplexing or multiplexing. Demultiplexing is done byte by byte; multiplexing is also done byte by byte.
13. The user data rate of STS-1 is $(8000 \times 9 \times 86 \times 8) = 49.536$ Mbps. To carry a load with a data rate 49.540, we need another 4 kbps. This means that we need to insert 4000 / 8 = 500 bytes into every 8000 frames. In other words, ***500 out of every 8000*** frames need to allow the H3 byte to carry data. For example, we can have sequences of 16 frames in which the first frame is an overloaded frame and then 15 frames are normal.
14. The user data rate of STS-1 is $(8000 \times 9 \times 86 \times 8) = 49.536$ Mbps. To carry a load with a data rate 49.526, we need 10 kbps worth of dummy data. This means that we need 10000 / 8 = 750 bytes of dummy data in 8000 frames. In other words, ***750 out of every 8000*** frames need to allow the next byte after H3 to be empty (dummy). For example, we can have sequences of 32 frames in which the first three frames are underloaded and the next 29 are normal.
15. In answering this question, we need to think about the three upper layers in SONET. The path layer is responsible for end-to-end communication. The line layer is responsible between multiplexers. The section layer is responsible between devices.
    a. ***A1*** and ***A2*** are used as ***aligners*** (synchronizers). They perform the same job as a preamble or flag field in other networks. We can call them ***framing bytes***. These bytes are set and renewed at each device to synchronize the two adjacent devices. There is no need for these bytes at the line or path layer.
    b. ***C1*** is used at the section layer to identify multiplexed STSs. This idea can be compared to statistical TDM in which each slot needs an address. In other words, C1 is the address of each STS-1 in an STS-n. C2 is like the port numbers in other protocols. When different processes need to communicate over the

same network, we need port addresses to distinguish between them. There is no need for C byte at the line layer.

c. ***D*** bytes are used for SONET administration. SONET requires two separate channels at the section (device-to-device) and line (multiplexer-to-multiplexer) layers.No administration is provided at the line layer.

d. ***E*** byte creates a voice communication channel between two devices at the ends of a section.

e. ***F*** bytes also create a voice communication. F1 is used between two devices at the end of a section; F2 is used between two ends. No bytes are assigned at the line layer.

f. The only ***G*** bytes are used for status reporting. A device at the end of the path reports its status to a device at the beginning of the path. No other layer needs this byte.

g. ***H*** bytes are the pointers. H1 and H2 are used to show the offsetting of the SPE with respect to STS-1. H3 is used to compensate for a faster or slower user data. All three are used in the line layer because add/drop multiplexing is done at this layer. H4 is used at the path layer to show a multiframe payload. Obviously we do not need an H byte in the section layer because no multiplexing or demultiplexing happens at this layer.

h. The only ***J*** byte is at the path layer to show the continuous stream of data at the path layer (end-to-end). The user uses a pattern that must be repeated to show the stream is going at the right destination. There is no need for this byte at the other layers.

i. As we discussed, ***K*** bytes are used for automatic protection switching, which happens at the line layer (multiplexing). Other layers do not need these bytes.

j. Z bytes are unused bytes. All of the bytes in SOH are assigned, but in LOH and POH some bytes are still unused.

16. The ***B*** bytes are ***error-detection*** bytes. They are used at all layers. B1 is used at the section layer (over the whole frame). Each bit of this byte is calculated over the corresponding bit of all bytes in the previous frame. B2 is used at the line layer. B3 is used at the path layer calculated over all bits of previous SPE.

CHAPTER 18

Virtual Circuit Switching: Frame Relay and ATM

Solutions to Review Questions and Exercises

Review Questions

1. ***Frame Relay*** does not use *flow* or *error control*, which means it does not use the sliding window protocol. Therefore, there is no need for ***sequence numbers***.
2. ***DLCIs*** are unique only for a particular interface. A switch assigns a DCLI to each virtual connection in an interface. This way two different connections belonging to two different interfaces may have the same DLCI.
3. ***T-lines*** provide point-to-point connections, not many-to-many. In order to connect several LANs together using T-lines, we need a mesh with many lines. Using Frame Relay we need only one line for each LAN to get connected to the Frame Relay network.
4. In a ***PVC***, two end systems are connected permanently through a virtual connection. In a ***SVC***, a virtual circuit needs to be established each time an end system wants to be connected with another end system.
5. ***Frame Relay*** does not define a specific protocol for the physical layer. Any protocol recognized by ANSI is acceptable.
6. If data packets are different sizes there might be variable delays in delivery.
7. A ***UNI*** (user network interface) connects a user access device to a switch inside the ATM network, while an ***NNI*** (network to network interface) connects two switches or two ATM networks.
8. A ***TP*** (transmission path) is the physical connection between a user and a switch or between two switches. It is divided into several ***VPs*** (virtual paths), which provide a connection or a set of connections between two switches. VPs in turn consist of several ***VCs*** (virtual circuits) that logically connect two points together.
9. An ATM virtual connection is defined by two numbers: a ***virtual path identifier (VPI)*** and a ***virtual circuit identifier (VCI).***
10. The ***Application Adaptation Layer (AAL)*** allows existing networks to connect to ATM facilities by mapping packet data into fixed-sized ATM cells. The ***ATM layer*** provides routing, traffic management, switching, and multiplexing services.

11. In an UNI, the total length of VPI+VCI is 24 bits. This means that we can define 2^{24} virtual circuits in an UNI. In an NNI, the total length of VPI+VCI is 28 bits. This means that we can define 2^{28} virtual circuits in an NNI.
12. We can briefly summarize the most important issues:
    a. Traditional LANs are ***connectionless*** protocols; ATM is a ***connection-oriented*** protocol.
    b. Traditional LANs define the route of a packet through ***source and destination addresses***; ATM defines the route of a cell through ***virtual connection identifiers***.
    c. Traditional LANs can do ***unicast***, ***multicast***, and ***broadcast*** transmission; ATM is designed only for ***unicast*** transmission.

Exercises

13. We first need to look at the EA bits. In each byte, the EA bit is the last bit (the eight bit from the left). If EA bit is 0, the address ends at the current byte; if it 1, the address continues to the next byte.

    Address → **10110000 00010111**

    The first EA bit is 0 and the second is 1. This means that the address is only two bytes (no address extension). DLCI is only 10 bits, bits 1 to 6 and 9 to 12 (from left).

    Address → **101100**00 **0001**0111
    DLCI → **1011000001** → **705**

14. The address field in Frame Relay is 16 bits. The address given is only 15 bits. It is **not valid**.
15. We first need to look at the EA bits. In each byte, the EA bit is the last bit (the eight bit from the left). If the EA bit is 0, the address ends at the current byte; if it 1, the address continues to the next byte.

    Address → **0x7C74E1** → **01111100 01110100 11100001**

    The first two EA bit are 0s and the last is 1. This means that the address is three bytes (address extension). DLCI is 16 bits, bits 1 to 6, 9 to 12, and 17 to 22.

    Address → **011111**00 **0111**0100 **111000**
    DLCI → **0111110111111000** → **32248**

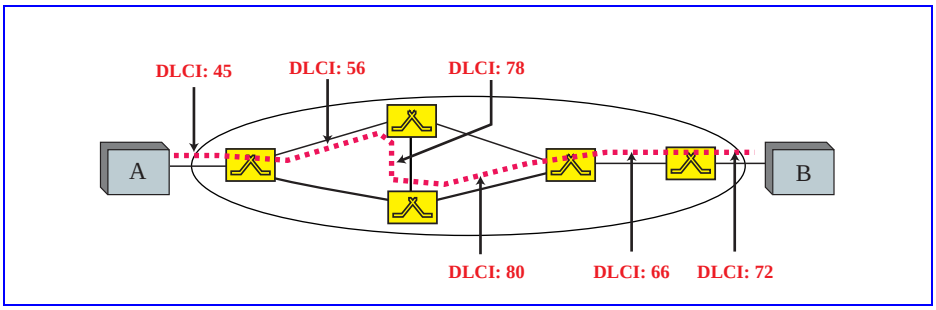
16. We first change the number 178 to 10-bit binary **0010110010**. We then add separate DLCI into a 6-bit and a 4-bit and add extra bits. Note that the first EA bit is 0; the second is 1.

    DLCI → **0010110010**

    Address → **001011**00 **0010**0001 → **0x2C21**

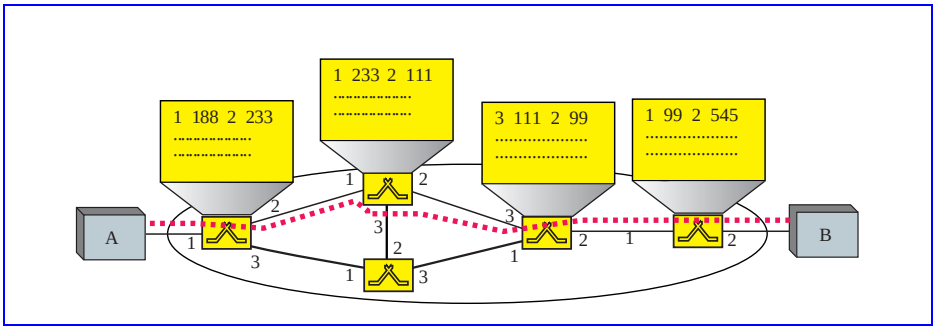
17. See Figure 18.1.

Figure 18.1 *Solution to Exercise 17*



18. See Figure 18.2.

Figure 18.2 *Solution to Exercise 18*



19. In AAL1, each cell carries only 47 bytes of user data. This means the number of cells sent per second can be calculated as [(2,000,000/8)/47] ≈ **5319.15**.
20. In AAL1, each 53-byte cell carries only 47 bytes of user data. There are 6 bytes of overhead. The efficiency can be calculated as 47/ 53 ≈ **89%**.
21.
    a. In AAL3/4, the CS layer needs to pass 44-byte data units to SAR layer. This means that the total length of the packet in the CS layer should be a multiple of 44. We can find the smallest value for padding as follows:

    H + Data + Padding + T = 0 mod 44
    4 + 47,787 + Padding + 4 = 0 mod 44
    Padding = **33 bytes**

    b. The number of data unit in the SAR layer is

    $$(4 + 47787 + 33 + 4) / 44 = \mathbf{1087}$$

    c. In AAL3/4, the number of cells in the ATM layer is the same as the number of data unit in the SAR layer. This means we have **1087 cells**.

22. If we assume that there is no need for padding, the efficiency of the AAL3/4 depends on the size of the packet because of the 8 bytes of overhead in the CS layer. A larger packet is more efficient than a smaller packet. A packet of size 8 bytes has an efficiency of 8/16 = 50% while a packet of size 1000 bytes has an efficiency of 1000/1008 ≈ 99%.

23.

    a. The minimum number of cells is **1**. ***This happens when the data size ≤ 36 bytes***. Padding is added to make it exactly 36 bytes. Then 8 bytes of header creates a data unit of 44 bytes at the SAR layer.

    b. The maximum number of cells can be determined from the maximum number of data units at the CS sublayer. If we assume no padding, the maximum size of the packet is 65535 + 8 = 65543. This needs 65543 / 44 ≈ 1489.61. The maximum number of cells is **1490**. ***This happens when the data size is between 65,509 and 65,535 (inclusive) bytes.*** We need to add between 17 to 43 (inclusive) bytes of padding to make the size 65552 bytes. The 8 byte overhead at the CS layer makes the total size 65560 which means 1490 data units of size 44.

24.

    a. The minimum number of cells is **1**. ***This happens when the data size ≤ 40 bytes***. Padding is added to make it exactly 40 bytes. Then 8 bytes of header creates a data unit of 48 bytes at the SAR layer.

    b. The maximum number of cells is **1366**. It can be determined from the maximum number of data units at the CS sublayer. If we assume no padding, the maximum size of the packet is 65535 + 8 = 65543. This needs 65543 / 48 ≈ 1365.48 or 1366 cells. ***This happens when the data size is between 65,513 and 65,535 (inclusive) bytes***. We need to add between 25 to 47 (inclusive) bytes of padding to make the size 65560 bytes. The 8 byte overhead at the CS layer makes the total size 65568 which means 1366 data unit of size 48.

25. AAL1 takes a ***continuous stream*** of bits from the user without any boundaries. There are always bits to fill the data unit; there is no need for padding. The other AALs take a bounded packet from the upper layer.

26. In AAL3/4 the number of bytes in CS, after adding header, padding, and trailer must be multiple of 44.

    a. When user (**4 + user data + 4**) mod 44 = 0.

    b. When user (**4 + user data + 40 + 4**) mod 44 = 0.

    c. When user (**4 + user data + 43 + 4**) mod 44 = 0.

27. In AAL5 the number of bytes in CS, after adding padding and trailer must be multiple of 48.

    a. When user (**user data** + **8**) mod 48 = 0.

    b. When user (**user data** + **40** + **8**) mod 48 = 0.

    c. When user (**user data** + **43** + **8**) mod 48 = 0.

28. For AAL1 we can calculate the exact number of bytes; for AAL2, AAL3/4, and AAL5, we cannot calculate the portion of the overhead in CS sublayer.

    a. AAL1 → 53 – **5** – 1 = **47**

b. AAL2 → 53 – **5** – 1 – (CS header) < **47**
c. AAL3/4 → 53 – **5** – 4 – (CS header) < **44**
d. AAL3/4 → 53 – **5** – (CS header) < **48**

CHAPTER 19

Network Layer: Logical Addressing

Solutions to Review Questions and Exercises

Review Questions

1. An ***IPv4*** address is **32** bits long. An ***IPv6*** address is **128** bits long.
2. ***IPv4 addresses*** are usually written in decimal form with a decimal point (dot) separating the bytes. This is called ***dotted-decimal notation***. Each address is **4** bytes. ***IPv6*** addresses are usually written in hexadecimal form with a colon separating the bytes. This is called ***hexadecimal notation***. Each address is **16** bytes or **32** hexadecimal digits.
3. ***Classful addressing*** assigns an organization a Class A, Class B, or Class C block of addresses. ***Classless addressing*** assigns an organization a block of contiguous addresses based on its needs.
4. ***Classes A***, ***B***, and ***C*** are used for **unicast** communication. ***Class D*** is for **multicast** communication and ***Class E*** addresses are **reserved** for special purposes.
5. A ***block in class A*** address is ***too large*** for almost any organization. This means most of the addresses in class A are wasted and not used. ***A block in class C*** is probably ***too small*** for many organizations.
6. A ***mask*** in classful addressing is used to find the first address in the block when one of the addresses is given. The ***default mask*** refers to the mask when there is no subnetting or supernetting.
7. The ***network address*** in a block of addresses is the first address. The ***mask*** can be **ANDed** with any address in the block to find the network address.
8. In ***subnetting***, a large address block could be divide into several contiguous groups and each group be assigned to smaller networks called subnets. In ***supernetting***, several small address blocks can be combined to create a larger range of addresses. The new set of addresses can be assigned to a large network called a supernet. ***A subnet mask*** has ***more*** consecutive 1s than the corresponding default mask. ***A supernet mask*** has ***less*** consecutive 1s than the corresponding default mask.
9. Multicast addresses in ***IPv4*** are those that start with the **1110** pattern. Multicast addresses in ***IPv6*** are those that start with the **11111111** pattern.

10. Home users and small businesses may have created small networks with several hosts and need an IP address for each host. With the shortage of addresses, this is a serious problem. A quick solution to this problem is called ***network address translation (NAT***). NAT enables a user to have a large set of addresses internally and one address, or a small set of addresses, externally. The traffic inside can use the large set; the traffic outside, the small set.

Exercises

11.
    a. $2^8 = \mathbf{256}$
    b. $2^{16} = \mathbf{65536}$
    c. $2^{64} = \mathbf{1.846744737 \times 10^{19}}$
12. $2^x = 1024 \rightarrow x = \log_2 1024 = \mathbf{10}$
13. $3^{10} = \mathbf{59{,}049}$
14.

a.	01110010	00100010	00000010	00001000
b.	10000001	00001110	00000110	00001000
c.	11010000	00100010	00110110	00001100
d.	11101110	00100010	00000010	00000001

15.
    a. **127.240.103.125**
    b. **175.192.240.29**
    c. **223.176.31.93**
    d. **239.247.199.29**
16.
    a. **Class C** (first byte is between 192 and 223)
    b. **Class D** (first byte is between 224 and 239)
    c. **Class A** (first byte is between 0 and 127)
    d. **Class B** (first byte is between 128 and 191)
17.
    a. **Class E** (first four bits are 1s)
    b. **Class B** (first bit is 1 and second bit is 0)
    c. **Class C** (first two bits are 1s and the third bit is 0)
    d. **Class D** (first three bits are 1s and the fourth bit is 0)
18.

a.	netid: 114	**hostid: 34.2.8**
b.	netid: 132.56	**hostid: 8.6**
c.	netid: 208.34.54	**hostid: 12**

19. With the information given, the first address is found by ANDing the host address with the mask 255.255.0.0 (/16).

Host Address:	**25**	.	**34**	.	**12**	.	**56**
Mask (ANDed):	**255**	.	**255**	.	**0**	.	**0**
Network Address (First):	**25**	.	**34**	.	**0**	.	**0**

The last address can be found by ORing the host address with the mask complement 0.0.255.255.

Host Address:	**25**	.	**34**	.	**12**	.	**56**
Mask Complement (ORed):	**0**	.	**0**	.	**255**	.	**255**
Last Address:	**25**	.	**34**	.	**255**	.	**255**

However, we need to mention that this is the largest possible block with 2^{16} addresses. We can have many small blocks as long as the number of addresses divides this number.

20. With the information given, the first address is found by ANDing the host address with the mask 255.255.255.192 (/26).

Host Address:	**182**	.	**44**	.	**82**	.	**16**
Mask (ANDed):	**255**	.	**255**	.	**255**	.	**192**
Network Address (First):	**182**	.	**44**	.	**82**	.	**0**

The last address can be found by ORing the host address with the mask complement 0.0.0.63.

Host Address:	**182**	.	**44**	.	**82**	.	**16**
Mask Complement (ORed):	**0**	.	**0**	.	**0**	.	**63**
Last Address:	**182**	.	**44**	.	**82**	.	**63**

However, we need to mention that this is the largest possible block with 2^6 addresses. We can have several small blocks as long as the number of addresses divides this number.

21.

a. $\log_2 500 = 8.95$ Extra 1s = 9 Possible subnets: **512** Mask: **/17** (8+9)

b. $2^{32-17} = 2^{15} =$ **32,768** Addresses per subnet

c. **Subnet 1:** The first address in the this address is the beginning address of the block or **16.0.0.0**. To find the last address, we need to write 32,767 (one less than the number of addresses in each subnet) in base 256 (0.0.127.255) and add it to the first address (in base 256).

First address in subnet 1:	**16**	.	**0**	.	**0**	.	**0**
Number of addresses:	**0**	.	**0**	.	**127**	.	**255**
Last address in subnet 1:	**16**	.	**0**	.	**127**	.	**255**

d. **Subnet 500:**
Note that the subnet 500 is not the last possible subnet; it is the last subnet used by the organization. To find the first address in subnet 500, we need to add 16,351,232 (499 × 32678) in base 256 (0. 249.128.0) to the first address in subnet 1. We have 16.0.0.0 + 0.249.128.0 = **16.249.128.0**. Now we can calculate the last address in subnet 500.

First address in subnet 500:	**16**	.	**249**	.	**128**	.	**0**
Number of addresses:	**0**	.	**0**	.	**127**	.	**255**
Last address in subnet 500:	**16**	.	**249**	.	**255**	.	**255**

22.

a. $\log_2 1024 = 10$ Extra 1s = **10** Possible subnets: **1024** Mask: **/26**

b. $2^{32-26} =$ **64** Addresses per subnet

c. **Subnet 1:**
The first address is the beginning address of the block or **130.56.0.0.** To find the last address, we need to write 63 (one less than the number of addresses in each subnet) in base 256 (0.0.0.63) and add it to the first address (in base 256).

First address in subnet 1:	**130**	.	**56**	.	**0**	.	**0**
Number of addresses:	**0**	.	**0**	.	**0**	.	**63**
Last address in subnet 1:	**130**	.	**56**	.	**0**	.	**63**

d. **Subnet 1024:**
To find the first address in subnet 1024, we need to add 65,472 (1023 × 64) in base 256 (0.0.255.92) to the first address in subnet 1. We have 130.56.0.0. + 0.0.255.192 = **130.56.255.192**. Now we can calculate the last address in subnet 500 as we did for the first address.

First address in subnet 1024:	**130**	.	**56**	.	**255**	.	**192**
Number of addresses:	**0**	.	**0**	.	**0**	.	**63**
Last address in subnet 1024:	**130**	.	**56**	.	**255**	.	**255**

23.

a. $\log_2 32 = 5$ Extra 1s = **5** Possible subnets: **32** Mask: **/29** (24 + 5)

b. $2^{32-29} =$ **8** Addresses per subnet

c. **Subnet 1:**
The first address is the beginning address of the block or **211.17.180.0**. To find the last address, we need to write 7 (one less than the number of addresses in each subnet) in base 256 (0.0.0.7) and add it to the first address (in base 256).

First address in subnet 1:	**211**	.	**17**	.	**180**	.	**0**
Number of addresses:	**0**	.	**0**	.	**0**	.	**7**
Last address in subnet 1:	**211**	.	**17**	.	**180**	.	**7**

d. **Subnet 32:**
To find the first address in subnet 32, we need to add 248 (31 × 8) in base 256 (0.0.0.248) to the first address in subnet 1. We have 211.17.180.0 + 0.0.0.248 or **211.17.180.248.** Now we can calculate the last address in subnet 32 as we did for the first address.

First address in subnet 32:	**211**	.	**17**	.	**180**	.	**248**
Number of addresses:	**0**	.	**0**	.	**0**	.	**7**
Last address in subnet 32:	**211**	.	**17**	.	**180**	.	**255**

24.

a. The mask 255.255.255.0 has **24** consecutive 1s → slash notation: /24
b. The mask 255.0.0.0 has **8** consecutive 1s → slash notation:**/8**
c. The mask 255.255.224.0 has **19** consecutive 1s → slash notation:**/19**
d. The mask 255.255.240.0 has **20** consecutive 1s → slash notation:**/20**

25.

a. The number of address in this block is 2^{32-29} = **8**. We need to add 7 (one less) addresses (0.0.0.7 in base 256) to the first address to find the last address.

From:	**123**	.	**56**	.	**77**	.	**32**
	0	.	**0**	.	**0**	.	**7**
To:	**123**	.	**56**	.	**77**	.	**39**

b. The number of address in this block is 2^{32-27} = **32**. We need to add 31 (one less) addresses (0.0.0.31 in base 256) to the first address to find the last address.

From:	**200**	.	**17**	.	**21**	.	**128**
	0	.	**0**	.	**0**	.	**31**
To:	**200**	.	**17**	.	**21**	.	**159**

c. The number of address in this block is 2^{32-23} = 512. We need to add 511 (one less) addresses (0.0.1.255 in base 256) to the first address to find the last address.

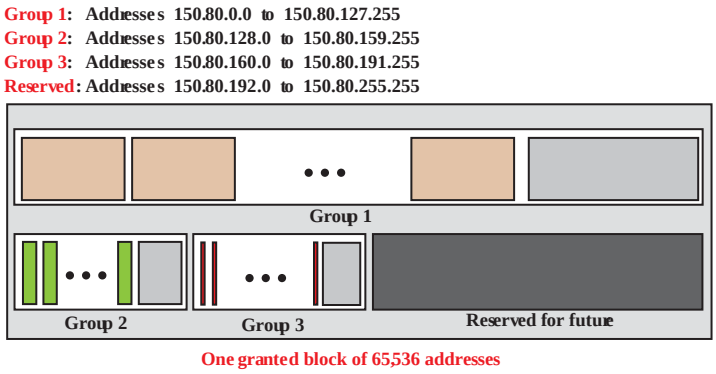
From:	**17**	.	**34**	.	**16**	.	**0**
	0	.	**0**	.	**1**	.	**255**
To:	**17**	.	**34**	.	**17**	.	**255**

d. The number of address in this block is 2^{32-30} = 4. We need to add 3 (one less) addresses (0.0.0.3 in base 256) to the first address to find the last address.

From:	180	.	34	.	64	.	64
	0	.	**0**	.	**0**	.	**3**
To:	180	.	34	.	64	.	67

26. The total number of addresses in this block is 2^{32-16} = **65536**. The ISP can divide this large block in several ways depending on the predicted needs of its customers in the future. We assume that the future needs follow the present pattern. In other words, we assume that the ISP will have customers that belong to one of the present groups. We design four ranges: group 1, group 2, group 3, and one reserved range of addresses as shown in Figure 19.1.

Figure 19.1 *Solution to Exercise 26*



Group 1

In the first group, we have 200 businesses. We augment this number to **256** (the next number after 200 that is a power of 2) to let 56 more customers of this kind in the future. The total number of addresses is 256 × 128 = **32768**. For this group, each customer needs 128 addresses. This means the suffix length is $\log_2 128 = 7$. The prefix length is then 32 – 7 = 25. The addresses are:

1st customer:	**150.80.0.0/25**	**to**	**150.80.0.127/25**
2nd customer:	**150.80.0.128/25**	**to**	**150.80.0.255/25**
...	**...**		**...**
200th customer:	**150.80.99.128/25**	**to**	**150.80.99.255/25**
Unused addresses	**150.80.100.0**	**to**	**150.80.127.255**

Total Addresses in group 1 = 256 × 128 = **32768** Used = 200 × 128 = **25600.**
Reserved: **7168**, which can be assigned to 56 businesses of this size.

Group 2

In the second group, we have 400 business. We augment this number to 512 (the next number after 400 that is a power of 2) to let 112 more customer of this kind in the future. The total number of addresses is = 512 × 16 = **8192**. For this group, each customer needs 16 addresses. This means the suffix length is 4 $\log_2 16 = 4$.

The prefix length is then 32 – 4 = 28. The addresses are:

1st customer:	**150.80.128.0/28**	**to**	**150.80.128.15/28**
2nd customer:	**150.80.128.16/28**	**to**	**150.80.128.31/28**
...	**...**		**...**
400th customer:	**150.80.152.240/28**	**to**	**150.80.152.255/28**
Unused addresses	**150.80.153.0**	**to**	**150.80.159.255**

Total Addresses in group 2 = 512 × 16 = **8192** Used = 400 × 16 = **6400**
Reserved: **1792**, which can be assigned to 112 businesses of this size.

Group 3

In the third group, we have 2000 households. We augment this number to 2048 (the next number after 2000 that is a power of 2) to let 48 more customer of this kind in the future. The total number of addresses is = 2048 × 4 = **8192**. For this group, each customer needs 4 addresses. This means the suffix length is 2 $\log_2 4$ = 2. The prefix length is then 32 – 2 = 30. The addresses are:

1st customer:	**150.80.160.0/30**	**to**	**150.80.160.3/30**
2nd customer:	**150.80.160.4/30**	**to**	**150.80.160.7/30**
...	**...**		**...**
2000th customer:	**150.80.191.60/30**	**to**	**150.80.191.63/30**
Unused addresses	**150.80.191.64**	**to**	**150.80.191.255**

Total Addresses in group 3 = 2048 × 4 = **8192** Used = 2000 × 4 = **8000**
Reserved: **192**, which can be assigned to 48 households.

Reserved Range

In the reserved range, we have 16384 address that are totally unused.

Note that we have unused addresses in each group and a large range of unused addresses in the reserved range.

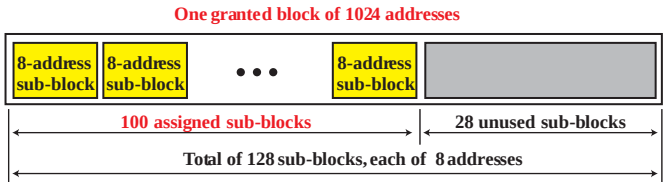
Summary:

The following shows the summary of used and unused addresses:

Group Number	*Total Addresses*	*Used Addresses*	*Unused Addresses*
1	32,768	**25,600**	7168
1	8192	**6400**	1792
1	8192	**8000**	192
Reserved	16,384	**0**	16384
Sum	**65,536**	**40,000**	**25536**

27. The site has $2^{32-22} = 2^{10} =$ **1024** from **120.60.4.0/22** to **120.60.7.255/22** addresses. One solution would be to divide this block into **128** 8-address sub-blocks as shown in Figure 19.2. The ISP can assign the first 100 sub-blocks to the current customers and keep the remaining 28 sub-blocks. Of course, this does not mean the future customer have to use 8-address subblocks. The remaining addresses can later be divided into different-size sub-blocks (as long as the three restrictions mentioned in this chapter are followed). Each sub-block has 8 addresses. The mask for each sub-block is **/29** ($32 - \log_2 8$). Note that the mask has changed from /22 (for the whole block) to /29 for each subblock because we have 128 sub-blocks ($2^7 = 128$).

Figure 19.2 *Solution to Exercise 27*



Sub-blocks:

1st subnet:	**120.60.4.0/29**	to	**120.60.4.7/29**
2nd subnet:	**120.60.4.8/29**	to	**120.60.4.15/29**
...	**...**		**...**
32nd subnet:	**120.60.4.248/29**	to	**120.60.4.255/29**
33rd subnet:	**120.60.5.0/29**	to	**120.60.5.7/29**
...	**...**		**...**
64th subnet:	**120.60.5.248/29**	to	**120.60.5.255/29**
...	**...**		**...**
99th subnet:	**120.60.7.16/29**	to	**120.60.7.23/29**
100th subnet:	**120.60.7.24/29**	to	**120.60.7.31/29**

1024 – 800 = **224** addresses left (from **120.60.7.31** to 120.60.7.155)

28. Each customer has only 1 address and, therefore, only one device. Since we defined a network as 2 or more connected devices, this is not a network

29.

a. **2340:1ABC:119A:A000::0**
b. **0:AA::119A:A231**
c. **2340::119A:A001:0**
d. **0:0:0:2340::0**

30.
   a. 0000:0000:0000:0000:0000:0000:0000:0000
   b. 0000:00AA:0000:0000:0000:0000:0000:0000
   c. 0000:1234:0000:0000:0000:0000:0000:0003
   d. 0123:0000:0000:0000:0000:0000:0001:0002

31.
   a. ***Link local address***
   b. ***Site local address***
   c. ***Multicast address*** (permanent, link local)
   d. ***Loopback address***

32.
   a. ***Unspecified address***
   b. ***Mapped address***
   c. ***Provider based address*** with the address registered through ***INTERNIC*** (North American registry).
   d. ***Provider based address*** with the address registered through ***RIPNIC*** (European registry).
   e. ***Provider based address*** with the address registered through ***APNIC*** (Asian/ Pacific registry).

33. **58ABC1**

34.
   a. **0000:0000:0000:0000:0000:0000:8106:0C22** or **0::8106:C22**
   b. **0000:0000:0000:0000:0000:FFFF:8106:0C22** or **0::FFFF:8106:C22**

35.
   a. **FE80:0000:0000:0000:0000:0000:0000:0123** or **FE80::123**
   b. **FEC0:0000:0000:0000:0000:0000:0000:0123** or **FEC0::123**

36. **FF02**: **< Group ID >**

37. The node identifier is **0000:0000:1211**. Assuming a 32-bit subnet identifier, the subnet address is **581E:1456:2314:ABCD:0000** where **ABCD:0000** is the subnet identifier.

38.
   from: **581E:1456:2314:0000:ABCD:0000:0001:XXXX**
   to: **581E:1456:2314:0000:ABCD:0000:00C8:XXXX**
   where **XXXX** is the node identifier.

CHAPTER 20

Network Layer: Internet Protocol

Solutions to Review Questions and Exercises

Review Questions

1. The delivery of a frame in the data link layer is ***node-to-node.*** The delivery of a packet at the network layer is ***host-to-host***.
2. In a ***connectionless service***, there is no setup and teardown phases. Each packet is independent from every other packet. ***Communication has only one phase: data transfer.*** In connection-oriented service, a virtual connection is established between the sender and the receiver before transferring data. ***Communication has three phases: setup, data transfer, and teardown.*** IPv4 provides a connectionless service; IPv6 normally provides a connectionless service, but it can provide a connection-oriented service if ***flow label*** field is used.
3. Each data link layer protocol has a limit on the size of the packet it can carry. When a datagram is encapsulated in a frame, the total size of the datagram must be less than this limit. Otherwise, the datagram must be ***fragmented***. IPv4 allows fragmentation at the host and any router; IPv6 allows fragmentation only at the host.
4. First, the value of the checksum field is set to 0. Then the entire header is divided into16-bit sections and added together. The result (sum) is complemented and inserted into the checksum field. ***The checksum in the IPv4 packet covers only the header, not the data***. There are two good reasons for this. First, all higher-level protocols that encapsulate data in the IPv4 datagram have a checksum field that covers the whole packet. Second, the header of the IPv4 packet changes with each visited router, but the data do not. ***The options, if present, are included in the checksum field.***
5. ***Options*** can be used for network testing and debugging. We mentioned six options: no-operation, end-of-option, record-route, strict-source-route, loose-source-route, and timestamp. A ***no-operation*** option is a 1-byte option used as a filler between options. An ***end-of-option*** option is a 1-byte option used for padding at the end of the option field. A ***record-route*** option is used to record the Internet routers that handle the datagram. A ***strict-source-route*** option is used by the source to predetermine a route for the datagram. A ***loose-source-route*** option is

similar to the strict source route, but it is less rigid. Each router in the list must be visited, but the datagram can visit other routers as well. A ***timestamp*** option is used to record the time of datagram processing by a router.

6. See Table 20.1.

Table 20.1 *Answer to Review Question 6*

IPv4 Fields	*IPv6 Fields*	*Explanation*
Version	***Version***	Value 4 for IPv4: value 6 for IPv6
Header Length		Header length is fixed in IPv6.
Service Type	***Priority***	Name and format changed in IPv6.
Total length	***Payload length***	Name changed in IPv6.
Identification		Handled by extension headers in IPv6.
Flags		Handled by extension headers in IPv6.
Fragment offset		Handled by extension headers in IPv6.
Time to live	***Hop limit***	Name changed in IPv6.
Protocol	***Next header***	Name changed in IPv6.
Checksum		Eliminated in IPv6.
Source address	***Source address***	32 bits in IPv4; 128 bits in IPv6
Destination address	***Destination address***	32 bits in IPv4; 128 bits in IPv6
	Flow label	New in IPv6

7. In IPv4, priority is handled by a field called ***service type*** (in the early interpretation) or ***differential services*** (in the latest interpretation). In the former interpretation, the three leftmost bits of this field define the priority or precedence; in the latter interpretation, the four leftmost bits of this field define the priority. In IPv6, the four-bit ***priority*** field handles two categories of traffic: *congestion-controlled* and *noncongestion-controlled*.

8. See Table 20.2.

Table 20.2 *Answer to Review Question 8*

Options in IPv4	*Extension Headers in IPv6*	*Explanation*
No-operation and ***end-of-option***	***Hop-by-hop, Pad-1 and Pad-N***	Redesigned in IPv6
	Hop-by-hop, jumbo payload	New in IPv6
Record route		Eliminated in IPv6
Strict and ***loose source route***	***Source route***	Combined in IPv6
Timestamp		Eliminated in IPv6
	Fragmentation	Base header in IPv4
	Authentication	New in IPv6
	Encrypted security payload	New in IPv6
	Destination	New in IPv6

9. The ***checksum*** is eliminated in IPv6 because it is provided by upper-layer protocols; it is therefore not needed at this level.
10. The three transition strategies are ***dual stack***, ***tunneling***, and ***header translation***. In *tunneling* the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it exits the region. *Tunneling* is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. In *header translation*, the header format must be totally changed from IPv4 to IPv6. *Header translation* is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4.

Exercises

11. If no fragmentation occurs at the router, then the only field to change in the base header is the ***time to live*** field. If any of the multiple-byte options are present, then there will be changes in the option headers as well (to record the route and/or timestamp). If fragmentation does occur, the ***total length*** field will change to reflect the total length of each datagram. The ***more*** fragment bit of the flags field and the fragmentation ***offset*** field may also change to reflect the fragmentation. If options are present and fragmentation occurs, the ***header length*** field of the base header may also change to reflect whether or not the option was included in the fragments.
12.

    Header Length = Total Length - Data Length = 1200 – 1176 = 24
    HLEN = 24/4 = **6** (in decimal) $\rightarrow$ **0110** (in binary)

13.

    Advantages of a large MTU:

    - Good for transferring large amounts of data over long distances
    - No fragmentation necessary; faster delivery and no reassembly
    - Fewer lost datagrams
    - More efficient (less overhead)

    Advantages of a small MTU:

    - Good for transferring time-sensitive data such as audio or video
    - Better suited for multiplexing

14. The first byte number can be calculated from the ***offset*** itself. If the offset is 120, that means that 120×8 or 960 bytes (bytes 0 through 959) were sent before this fragment. The first byte number is therefore 960. The last byte number can be calculated by adding the total length field and subtracting one.
15. The value of the header length field of an IP packet can never be less than 5 because every IP datagram must have at least a base header that has a fixed size of 20 bytes. The value of HLEN field, when multiplied by 4, gives the number of

bytes contained in the header. Therefore the minimum value of this field is 5. This field has a value of exactly 5 when there are no options included in the header.

16. If the value of the HLEN field is 7, there are 28 (since $7 \times 4 = 28$) bytes included in the header. There are 20 bytes in the base header, so the total number of option bytes must be **8**.
17. If the size of the option field is 20 bytes, then the total length of the header is 40 bytes (20 byte base header plus 20 bytes of options). The HLEN field will be the total number of bytes in the header divided by 4, in this case ten (1010 in binary).
18. The datagram must contain 16 bytes of data:

36 byte total length – (5 HLEN field $\times$ 4) = 36 – 20 = 16 bytes

19. Since there is no option information, the header length is 20, which means that the value of HLEN field is **5** or **0101** in binary. The value of total length is 1024 + 20 or **1044** (**00000100 00010100** in binary).
20. The identification field is incremented for each non-fragmented datagram. If the first is 1024, then the last is 1024 + 99 = **1123**
21. If the M (*more*) bit is zero, this means that the datagram is either the last fragment or the it is not fragmented at all. Since the *offset* is 0, it cannot be the last fragment of a fragmented datagram. ***The datagram is not fragmented***.
22. Since the *offset* field shows the offset from the beginning of the original datagram in multiples of 8 bytes, an offset of 100 indicates that there were **800** bytes of data sent before the data in this fragment.
23. Let us first find the value of header fields before answering the questions:

    VER = 0x4 = **4**
    HLEN =0x5 = 5 $\rightarrow 5 \times 4 =$ **20**
    Service =0x00 = **0**
    Total Length = 0x0054 = **84**
    Identification = 0x0003 = **3**
    Flags and Fragmentation = 0x0000 $\rightarrow$ D = **0** M= **0** offset = **0**
    Time to live = 0x20 = **32**
    Protocol = 0x06 = **6**
    Checksum = 0x5850
    Source Address: 0x7C4E0302 = **124.78.3.2**
    Destination Address: 0xB40E0F02 = **180.14.15.2**

    We can then answer the questions:

    a. If we calculate the checksum, we get 0x0000. ***The packet is not corrupted***.
    b. Since the length of the header is 20 bytes, ***there are no options.***
    c. Since *M* = 0 and *offset* = 0, ***the packet is not fragmented.***
    d. The total length is 84. ***Data size is 64 bytes (84 –20)***.
    e. Since the value of *time to live* = 32, ***the packet may visit up to 32 more routers***.
    f. ***The identification number of the packet is 3***.
    g. ***The type of service is normal***.

24.

Data size = 200 – (5 × 4) = **180 bytes**
Offset = 200 × 8 = **1600**
The number of the first byte = offset value = **1600**
The number of the last byte = offset value + data size – 1 = **1779**
M = 0, offset ≠ 0 → **last fragment**

CHAPTER 21

Network Layer: Address Mapping, Error Reporting, and Multiplexing

Solutions to Review Questions and Exercises

Review Questions

1. The size of an ARP packet is ***variable***, depending on the length of the logical and physical addresses used.
2. ARP Packet Size = 2 + 2 + 1 + 1 + 2 + 6 + 4 + 6 + 4 = **28 bytes**
3. The size of the ARP packet in Question 2 is 28 bytes. We need to pad the data to have the minimum size of **46**. The size of the packet in the Ethernet frame is then calculated as 6 + 6 + 2 + **46** + 4 = **64 bytes** (without preamble and SFD).
4. The broadcast for Ethernet is all 1s or **0xFFFFFFFFFFFF**.
5. This restriction prevents ICMP packets from ***flooding*** the network. Without this restriction an endless flow of ICMP packets could be created.
6. The ***IP header*** is included because it contains the IP address of the original source. ***The first 8 bytes of the data*** are included because they contain the first section of the TCP or UDP header which contains information about the port numbers (TCP and UDP) and sequence number (TCP). This information allows the source to direct the ICMP message to the correct application.
7. A host would never receive a redirection message if there is only ***one router*** that connects the local network to the outside world.
8. The minimum size of an ICMP packet is **8 bytes** (router solicitation packet). The largest of the ICMP packets is the router advertisement packet with up to 255 listings. The maximum size is then:

   255 listings × 8 bytes/listing + 8 bytes for the ICMP header = **2048 bytes**

9. The minimum size of an IP packet that carries an ICMP packet would be **28 bytes** (a 20 byte IP header + an 8 byte router solicitation packet). The maximum size would be **2068 bytes** (a 20 byte IP header + a 2048 byte router advertisement packet).
10. The value of the ***protocol field*** of an IP packet carrying an ICMP packet is **1**.
11. The minimum size would be **64 bytes** if we do not consider the preamble and SFD fields, which are added at the physical layer. The maximum size would be **1518**

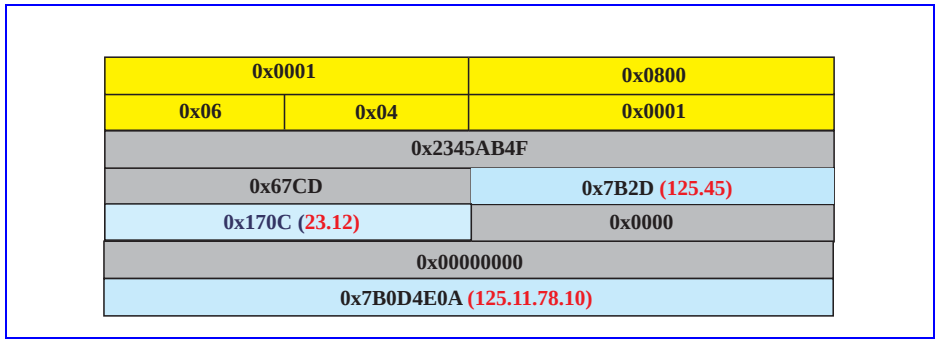
bytes, again not considering the preamble and SFD fields. Although the maximum size of an ICMP packet can be much more than 1500 bytes (for a router advertisement packet), Ethernet can carry only 1500 bytes of it.

12. There is no need for a report message to travel outside of its own network because its only purpose is to ***inform the next router in the spanning tree of group membership***. There is no need for a query message to travel outside of the local network because its only purpose is ***to poll the local network for membership in any groups***.

Exercises

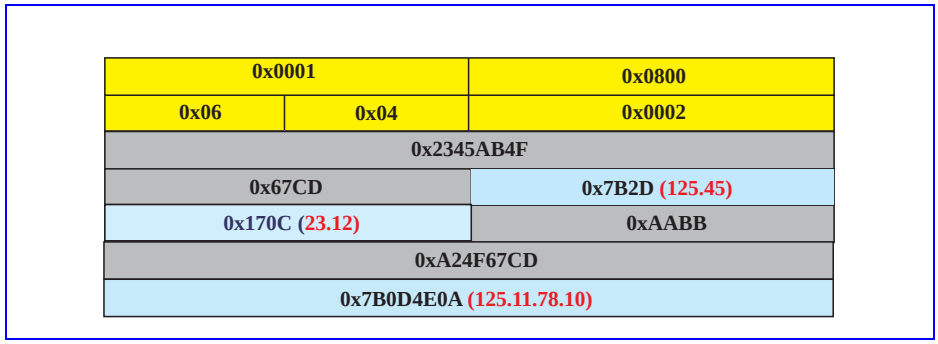
13. See Figure 21.1. Note that all values are in hexadecimal. Note also that the hardware addresses does not fit in the 4-byte word boundaries. We have also shown the IP address in parentheses.

Figure 21.1 *Solution to Exercise 13*



14. See Figure 21.2.

Figure 21.2 *Solution to Exercise 14*



15. See Figure 21.3. We have not shown the preamble and SFD fields, which are added in the physical layer.
16. See Figure 21.4. We have not shown the preamble and SFD fields, which are added in the physical layer.

Figure 21.3 *Solution to Exercise 15*

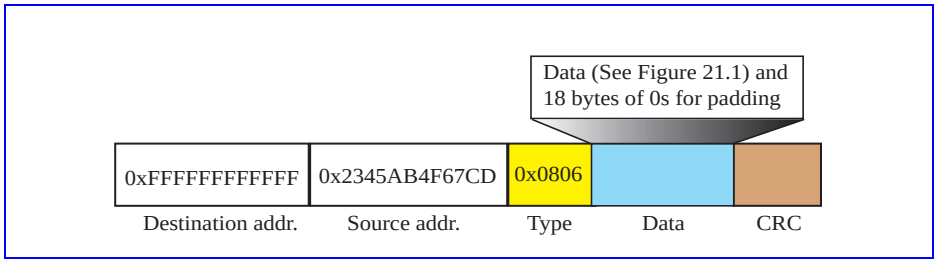
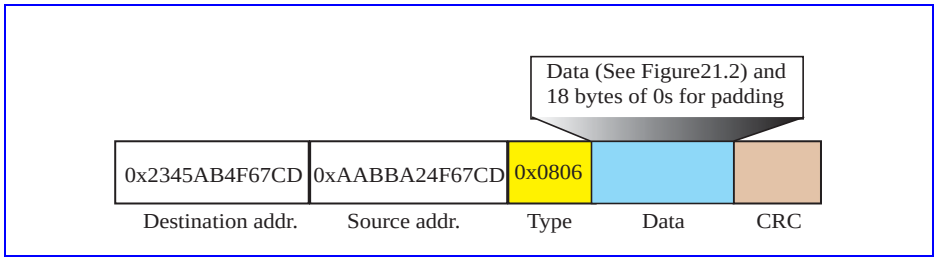


Figure 21.4 *Solution to Exercise 16*



17. It could happen that host B is unreachable, for some reason. The error message generated by an intermediate router could then be lost on its way back to host A. Or perhaps the datagram was dropped due to congestion and the error message generated by an intermediate router was lost.
18. The checksum is **0xD399** or **1101 0011 1001 1001** as calculated below:

		1	2	1	2	← Carry
8 and 0	→	**0**	**8**	**0**	**0**	
0	→	**0**	**0**	**0**	**0**	
123	→	**0**	**0**	**7**	**B**	
25	→	**0**	**0**	**1**	**9**	
H and e	→	**4**	**8**	**6**	**5**	
l and l	→	**6**	**C**	**6**	**C**	
o and pad	→	**6**	**F**	**0**	**0**	
		----	----	----	----	
Partial Sum		**2**	**C**	**6**	**5**	
Carry from last column					**1**	
Sum		**2**	**C**	**6**	**6**	
Checksum		**D**	**3**	**9**	**9**	

19. The appropriate ICMP message is ***destination unreachable*** message. This type of message has different types of codes to declare what is unreachable. In this case, the code is **0**, which means the network is unreachable (The codes are not discussed in the chapter; consult references for more information).

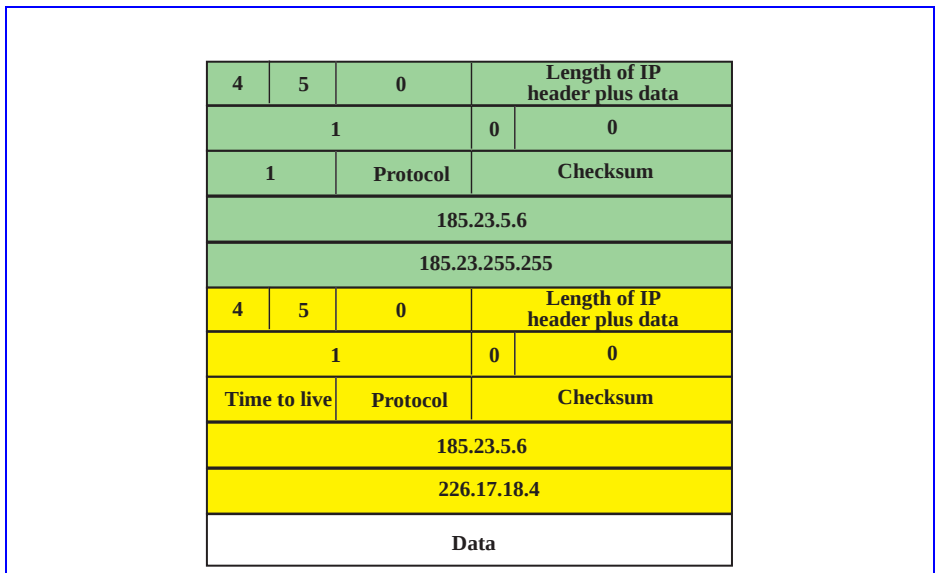
20. The appropriate ICMP message is ***destination unreachable*** message. This type of message has different types of codes to declare what is unreachable. In this case, the code is **3**, which means the port is unreachable (The codes are not discussed in the chapter; consult references for more information).
21. See the transformation process below:

IP:			**11100111**	**0**	**0011000**	**00111100**	**00001001**
Ethernet:	**00000001**	**00000000**	**01011110**	**0**	**0011000**	**00111100**	**00001001**

The Ethernet address in hexadecimal is **0x01005E183C09**

22. A router should send only **1** query message no matter how many entries it has in its group table. The message will be broadcast to all of the local nodes that are below it in the spanning tree.
23. The host must send as many as **five different report messages** at random times in order to preserve membership in five different groups.
24. The router will not need the services of ARP because the frame is broadcast at the physical address level. See Figure 21.5.

Figure 21.5 *Solution to Exercise 24*



25. No action should be taken.
26. It should set the state of the 2 entries to ***Delaying*** and start a timer for each with a random time. As each timer expires, a membership report message is sent twice for each group to the router that sent the query.

27.

Ethernet:
Supported number of groups using 23 bits = 2^{23} = **8,388,608 groups**
IP:
Supported number of groups using 28 bits =2^{28} = **268,435,456 groups**
Address space lost:
268,435,456 – 8,388,608 = **260,046,848 groups**

28. See below. We have shown the process in the solution to Exercise 21.

a.	**IP: 234.18.72.8**	→	**Ethernet:**	**0x01005E124808**
b.	**IP: 235.18.72.8**	→	**Ethernet:**	**0x01005E124808**
c.	**IP: 237.18.6.88**	→	**Ethernet:**	**0x01005E120658**
d.	**IP: 224.88.12.8**	→	**Ethernet:**	**0x01005E580C08**

Note that *a* and *b* represent the same Ethernet address.

CHAPTER 22

Network Layer: Delivery, Forwarding, and Routing

Solutions to Review Questions and Exercises

Review Questions

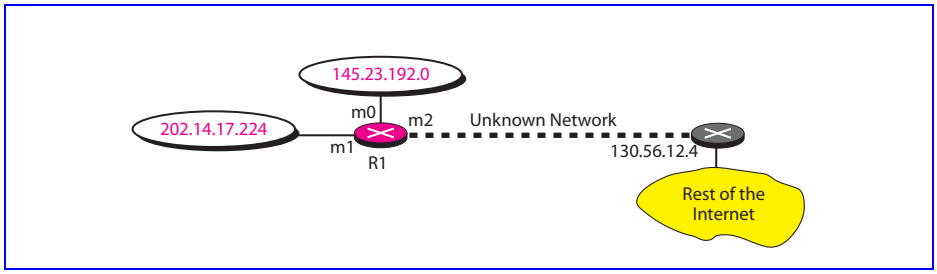
1. We discussed two different methods of delivery: direct and indirect. In a ***direct delivery***, the final destination of the packet is a host connected to the same physical network as the deliverer. In an ***indirect delivery*** the packet goes from router to router until it reaches the one connected to the same physical network as its final destination.
2. The three common forwarding methods used today are: next-hop, network-specific, and default methods. In the ***next-hop method***, the routing table holds only the address of the next hop for each destination. In the ***network-specific*** method, the routing table holds only one entry that defines the address of the destination network instead of all hosts on that network. In the ***default method***, a host sends all packets that are going out of the network to a specific router called the default router.
3. A routing table can be either static or dynamic. A ***static routing*** table contains information entered manually. A ***dynamic routing table*** is updated periodically by using one of the dynamic routing protocols such as RIP, OSPF, or BGP.
4. RIP is an ***intradomain routing protocol*** that enables routers to update their routing tables within an ***autonomous*** system.
5. A RIP ***message*** is used by a router to request and receive routing information about an autonomous system or to periodically share its knowledge with its neighbors.
6. The time-out for the ***expiration time*** is **6** times that of the ***periodic timer*** to allow for some missed communication between routers.
7. The ***hop count limit*** helps RIP instability by limiting the number of times a message can be sent through the routers, thereby limiting the back and forth updating that may occur if part of a network goes down.
8. The two major shortcomings are ***two-node instability*** and ***three-node instability***. For the former, infinity can be re-defined as a number such as 20. Another solution is the split horizon strategy or split horizon combined with poison reverse. These methods do not work for three-node instability.

9. In OSPF, four types of links have been defined: point-to-point, transient, stub, and virtual. A ***point-to-point*** link connects two routers without any other host or router in between. A ***transient*** link is a network with several routers attached to it. The packets can enter and leave through any of the routers. A ***stub*** link is a network that is connected to only one router. The data packets enter the network through this single router and leave the network through this same router. This is a special case of the transient network. When the link between two routers is broken, the administrator may create a ***virtual*** link between them, using a longer path that probably goes through several routers.
10. OSPF messages are propagated immediately because a router using OSPF will immediately ***flood*** the network with news of any changes to its neighborhood. RIP messages are distributed slowly because a network using RIP relies on the ***periodic updates*** that occur every 30 seconds to carry any news from one router to the next and to the next.
11. BGP is an ***interdomain*** routing protocol using path vector routing.
12. We mentioned two groups of multicast routing protocols: the source-based tree and the group-shared tree. In a ***source-based tree*** protocol, each router needs to have one shortest path tree for each group. The shortest path tree for a group defines the next hop for each network that has loyal member(s) for that group. In a ***group-shared tree*** protocol, only one designated router takes the responsibility of distributing multicast traffic. The designated router has *m* shortest path trees in its routing table. The rest of the routers in the domain have none.

Exercises

13. A host that is totally isolated needs no routing information. ***The routing table has no entries***.
14. A routing table for a LAN not connected to the Internet and with no subnets can have a routing table with ***host-specific addresses***. There is no next-hop address since all packets remain within the network.
15. See Figure 22.1.

Figure 22.1 *Solution to Exercise 15*



16. If the packet with destination address 140.24.7.194 arrives at R3, it gets sent to interface ***m0***. If it arrives at R2, it gets sent to interface ***m1*** and then to router R3.

The only way R1 can receive the packet is if the packet comes from organization 1, 2, or 3; it goes to R1 and is sent out from interface ***m3***.

17. R1 cannot receive a packet with this destination from ***m0*** because if any host in Organization 1 sends a packet with this destination address, the delivery is direct and does not go through R1. R1 can receive a packet with this destination from interfaces ***m1*** or ***m2***. This can happen when any host in Organization 2 or 3 sends a packet with this destination address. The packet arrives at R1 and is sent out through ***m0***. R1 can also receive a packet with this destination from interface **m3**. This happens in two cases. First, if R2 receives such a packet, the /24 is applied The packet is sent out from interface m0, which arrives at interface ***m3*** of R1. Second, if R3 receives such a packet, it applies the default mask and sends the packet from its interface ***m2*** to R2, which, in turn, applies the mask (/24) and sends it out from its interface ***m0*** to the interface ***m3*** of R1.
18. See Table 22.1.

Table 22.1 *Solution to Exercise 18: Routing table for regional ISP*

Mask	Network address	Next-hop address	Interface
/20	120.14.64.0	---	**m0**
/20	120.14.96.0	---	**m2**
/20	120.14.112.0	---	**m3**
/0	0.0.0.0	**default router**	**m4**

19. See Table 22.2.

Table 22.2 *Solution to Exercise 19: Routing table for local ISP 1*

Mask	Network address	Next-hop address	Interface
/23	120.14.64.0	---	**m0**
/23	120.14.66.0	---	**m1**
/23	120.14.68.0	---	**m2**
/23	120.14.70.0	---	**m3**
/23	120.14.72.0	---	**m4**
/23	120.14.74.0	---	**m5**
/23	120.14.76.0	---	**m6**
/23	120.14.78.0	---	**m7**
/0	0.0.0.0	**default router**	**m8**

20. See Table 22.3.

Table 22.3 *Solution to Exercise 20: Routing table for local ISP 2*

Mask	Network address	Next-hop address	Interface
/22	120.14.96.0	---	**m0**
/22	120.14.100.0	---	**m1**

Table 22.3 *Solution to Exercise 20: Routing table for local ISP 2*

Mask	Network address	Next-hop address	Interface
/22	120.14.104.0	---	**m2**
/22	120.14.108.0	---	**m3**
/0	0.0.0.0	**default router**	**m4**

21. See Table 22.4.

Table 22.4 *Solution to Exercise 21: Routing table for local ISP 3*

Mask	Network address	Next-hop address	Interface
/24	120.14.112.0	---	**m0**
/24	120.14.113.0	---	**m1**
/24	120.14.114.0	---	**m2**
/24	120.14.115.0	---	**m3**
/24	120.14.116.0	---	**m4**
/24	120.14.117.0	---	**m5**
/24	120.14.118.0	---	**m6**
/24	120.14.119.0	---	**m7**
/24	120.14.120.0	---	**m8**
/24	120.14.121.0	---	**m9**
/24	120.14.122.0	---	**m10**
/24	120.14.123.0	---	**m11**
/24	120.14.124.0	---	**m12**
/24	120.14.125.0	---	**m13**
/24	120.14.126.0	---	**m14**
/24	120.14.127.0	---	**m15**
/0	0.0.0.0	**default router**	**m16**

22. See Table 22.5.

Table 22.5 *Solution to Exercise 22: Routing table for small ISP 1*

Mask	Network address	Next-hop address	Interface
/30	120.14.64.0	----	**m0**
/30	120.14.64.4	----	**m1**
/30	120.14.64.8	----	**m2**
/30	120.14.64.12	----	**m3**
. . .	. . .	. . .	. . .
/30	120.14.65.252	----	**m127**
/0	0.0.0.0	**default router**	**m128**

23. In distance vector routing each router ***sends all of its knowledge about an autonomous system to all of the routers on its neighboring networks at regular intervals***. It uses a fairly simple algorithm to update the routing tables but results in a lot of unneeded network traffic. In link state routing a router ***floods an autonomous system with information about changes in a network only when changes occur***. It uses less network resources than distance vector routing in that it sends less traffic over the network but it uses the much more complex Dijkstra Algorithm to calculate routing tables from the link state database.
24. We assume that router C is one hop away. Then the modified table from C is Table 22.6:

Table 22.6 *Solution to Exercise 24*

Network	*Hops*
Net1	3
Net2	2
Net3	4
Net4	8

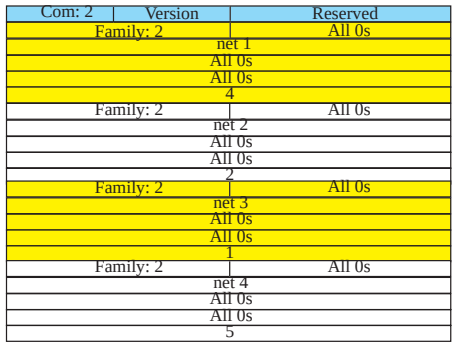
Comparing this to the old table, we get Table 22.7:

Table 22.7 *Solution to Exercise 24*

Network	*Hops*	
Net1	3	**C**
Net2	2	**C**
Net3	1	**F**
Net4	5	**G**

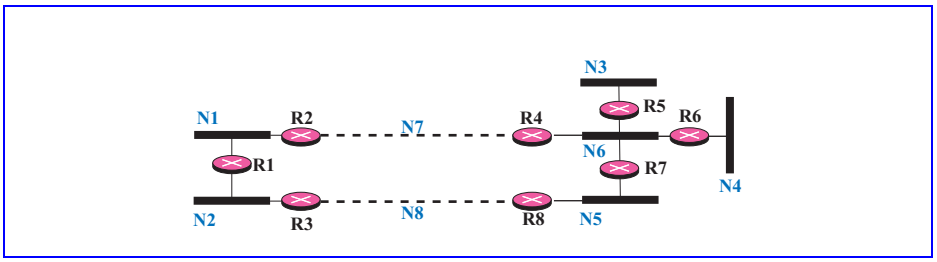
25. There are **2 + (10 × N)** = Empty bytes in a message advertising N networks
26. See Figure 22.2.

Figure 22.2 *Solution to Exercise 26*



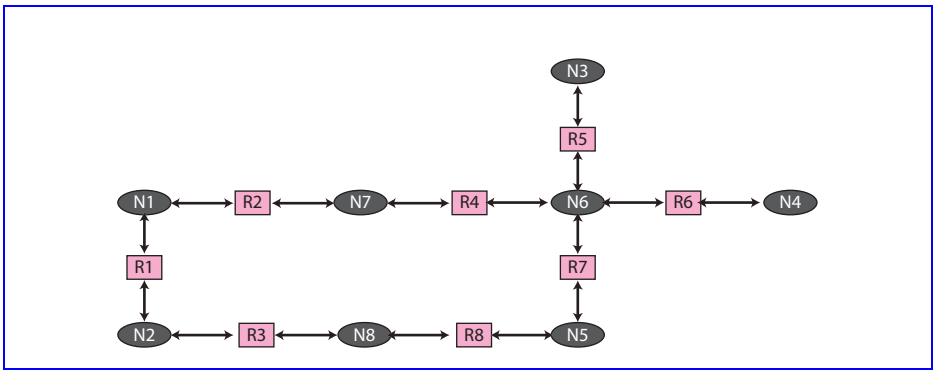
27. See Figure 22.3.

Figure 22.3 *Solution to Exercise 27*



28. See Figure 22.4.

Figure 22.4 *Solution to Exercise 28*



29. **Transient networks**: **N1**, **N2**, **N5**, and **N6**. **Stub networks**: **N3** and **N4**
30. See Table 22.8.

Table 22.8 *Solution to Exercise 30*

Destination	*Interface*
---	---
10.0.0.0	2
---	---

31. No, ***RPF*** does not create a shortest path tree because a network can receive more than one copy of the same multicast packet. RPF creates a graph instead of a tree.
32. Yes, ***RPB*** creates a shortest path tree and its leaves are networks. However, the delivery of the packets are based on broadcasting instead of multicasting.
33. Yes, ***RPM*** creates a shortest path tree because it is actually RPB (see previous answer) with pruning and grafting features. The leaves of the tree are the networks.

CHAPTER 23

Process-to-Process Delivery:

Solutions to Review Questions and Exercises

Review Questions

1. ***Reliability*** is not of primary importance in applications such as echo, daytime, BOOTP, TFTP and SNMP. In custom software, reliability can be built into the client/server applications to provide a more reliable, low overhead service.
2. IP and UDP are both ***connectionless*** and ***unreliable protocols***. The main difference in their reliability is that IP only calculates a checksum for the IP header and not for the data while UDP calculates a checksum for the entire datagram.
3. ***Port addresses*** do not need to be universally unique as long as each IP address/port address pair uniquely identify a particular process running on a particular host. A good example would be a network consisting of 50 hosts, each running echo server software. Each server uses the well known port number 7, but the IP address, together with the port number of 7, uniquely identify a particular server program on a particular host. Port addresses are ***shorter*** than IP addresses because their domain, a single system, is smaller than the domain of IP addresses, all systems on the Internet.
4. ***Ephemeral*** is defined as short-lived or transitory. Ephemeral port numbers are only used for the duration of a single communication between client and server, so they are indeed short-lived.
5. The minimum size of a UDP datagram is **8** bytes at the transport layer and **28** bytes at the IP layer. This size datagram would contain no data–only an IP header with no options and a UDP header. The implementation may require padding.
6. Since the length of a datagram must be contained in a 2 byte field, the maximum size of a UDP datagram is **65,535** bytes (header plus data). However, given that the IP layer must also store the total length of the packet in a 2 byte field, the maximum length would be 20 bytes less than this, or **65,515** bytes, to leave room for the IP header. The implementation may impose a smaller limit than this.
7. The smallest amount of process data that can be encapsulated in a UDP datagram is **0** bytes.

8. The largest amount of process data that can be encapsulated in a UDP datagram is **65,507** bytes. (65,535 minus 8 bytes for the UDP header minus 20 bytes for the IP header). The implementation may impose a smaller limit than this.
9. See Table 23.1.

Table 23.1 *Answer to the Question 9.*

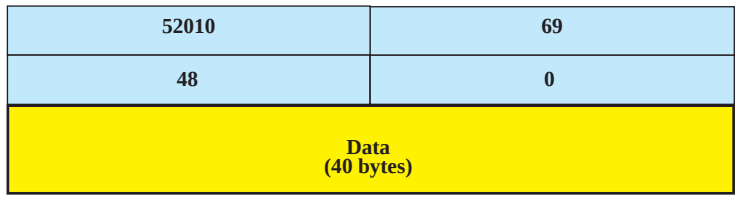
Fields in UDP	*Fields in TCP*	*Explanation*
Source Port Address	**Source Port Address**	
Destination Port Address	**Destination Port Address**	
Total Length		**There is no need for total length.**
Checksum	**Checksum**	
	Sequence Number	**UDP has no flow and error control.**
	Acknowledge Number	**UDP has no flow and error control.**
	Header Length	**UDP has no flow and error control.**
	Reserved	**UDP has no flow and error control.**
	Control	**UDP has no flow and error control.**
	Window Size	**UDP has no flow and error control.**
	Urgent Pointer	**UDP cannot handle urgent data.**
	Options and Padding	**UDP uses no options.**

10. ***UDP*** is preferred because each user datagram can be used for each chunk of data. However, a better solution is ***SCTP***.
11.
    a. None of the control bits are set. The segment is part of a data transmission without piggybacked acknowledgment.
    b. The ***FIN*** bit is set. This is a FIN segment request to terminate the connection.
    c. The ***ACK*** and the ***FIN*** bits are set. This is a ***FIN+ACK*** in response to a received ***FIN*** segment.
12. The ***maximum size*** of the TCP header is **60** bytes. The ***minimum size*** of the TCP header is **20** bytes.

Exercises

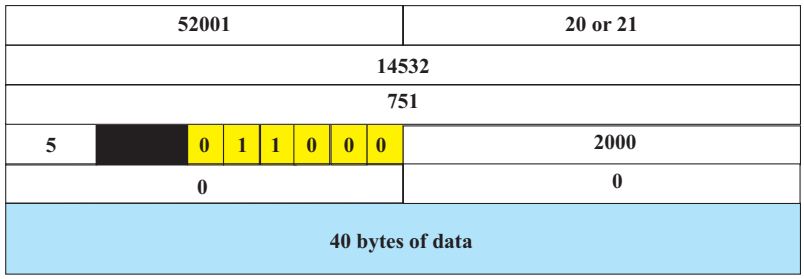
13. See Figure 23.1.
14. The client would use the IP address **122.45.12.7**, combined with an ephemeral port number, for its source socket address and the IP address **200.112.45.90**, combined with the well- known port number **161**, as the destination socket address.
15. The server would use the IP address **130.45.12.7**, combined with the well-known port number **69** for its source socket address and the IP address **14.90.90.33**, combined with an ephemeral port number as the destination socket address.
16. This datagram ***cannot be transferred*** using a single user datagram.

Figure 23.1 *Solution to Exercise 13*



17. 16 bytes of data / 24 bytes of total length = **0.666**
18. 16 bytes of data / 44 bytes of total length = **0.364**
19. 16 bytes of data / 72 byte minimum frame size = **0.222**
20.
    a. Port number **1586**
    b. Port number **13**
    c. **28** bytes
    d. **20** bytes (28 – 8 byte header)
    e. ***From a client to a server***
    f. ***Daytime***
21. It looks as if both the destination IP address and the destination port number are corrupted. ***TCP calculates the checksum and drops the segment***.
22. 0111 in decimal is 7. The total length of the header is 7 × 4 or **28**. The base header is **20** bytes. The segment has **8** bytes of options.
23. See Figure 23.2.

Figure 23.2 *Solution to Exercise 23*



24.
    a. The source port number is **0x0532** (**1330** in decimal).
    b. The destination port number is **0x0017** (**23** in decimal).
    c. The sequence number is **0x00000001** (**1** in decimal).

d. The acknowledgment number is **0x00000000** (**0** in decimal).

e. The header length is **0x5** (**5** in decimal). There are 5 × 4 or **20** bytes of header.

f. The control field is **0x002**. This indicates a SYN segment used for connection establishment.

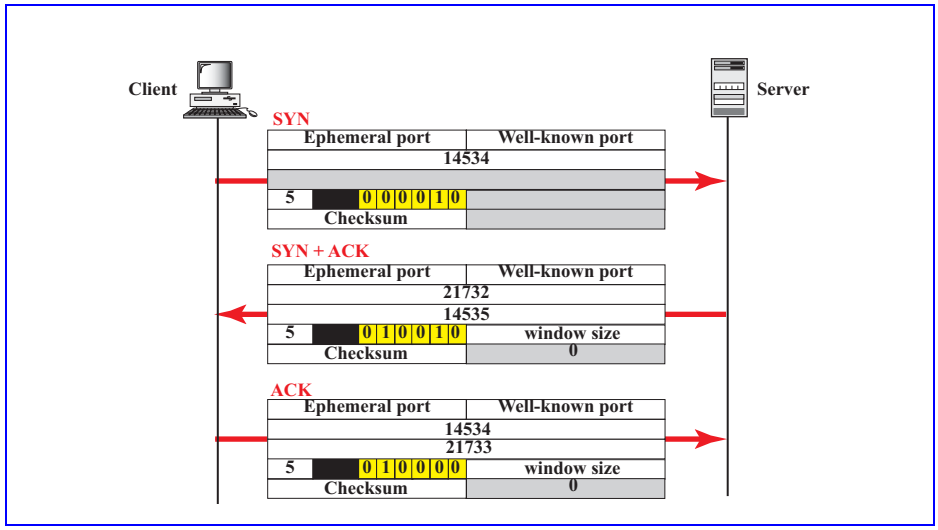
g. The window size field is **0x07FF** (**2047** in decimal).

25. Every second the counter is incremented by 64,000 × 2 = **128,000**. The sequence number field is 32 bits long and can hold only 2^{32}–1. So it takes (2^{32}–1)/(128,000) seconds or **33,554** seconds.

26. 3000 – 2000 = **1000** bytes

27. See Figure 23.3.

Figure 23.3 *Solution to Exercise 27*



28.

a. **At TCP level:**

16 bytes of data / (16 bytes of data + 20 bytes of TCP header) ≈ **0.44** or **44** percent

b. **At IP level**

16 bytes of data / (16 bytes of data + 20 bytes of TCP header + 20 bytes of IP header) ≈ **0.29** or **29** percent

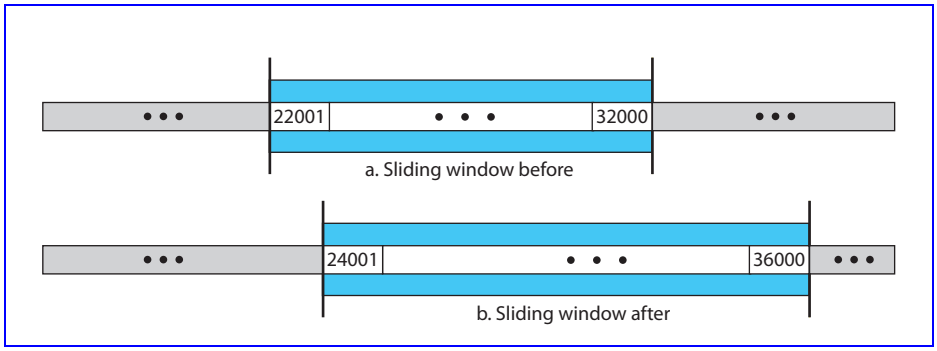
c. **At data link level (**assuming no preamble or flag**):**

16 bytes of data / (16 bytes of data + 20 bytes of TCP header + 20 bytes of IP header + 18 bytes of Ethernet header and trailer ≈ **0.22** or **22** percent,

29. The largest number in the sequence number field is 2^{32} –1. If we start at 7000, it takes [(2^{32} – 1) –7000] / 1,000,000 = **4295** seconds.

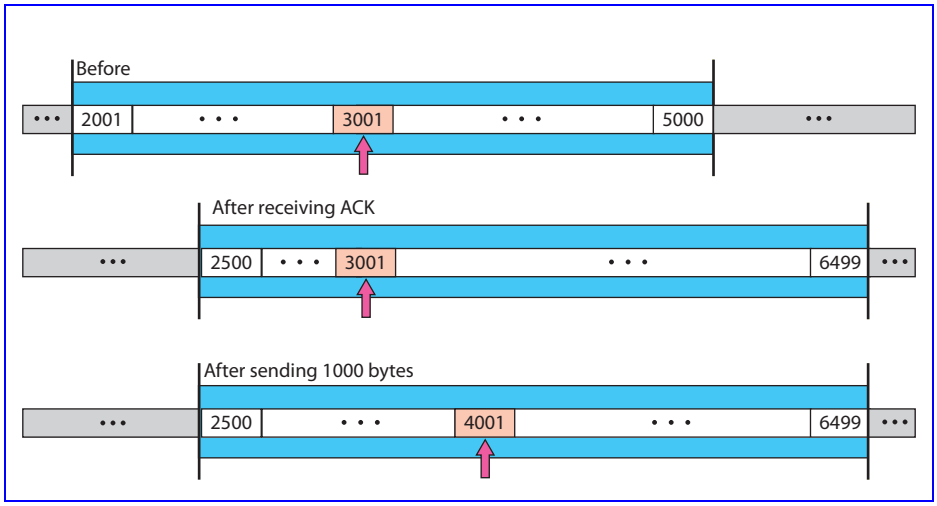
30. See Figure 23.4.

Figure 23.4 *Solution to Exercise 30*



31. See Figure 23.5.

Figure 23.5 *Solution to Exercise 31*



32. The SACK chunk with a cumTSN of **23** was delayed.
33. See Figure 23.6.
    Note that the value of cumTSN must be updated to 8.

34. See Figure 23.7. Chunks 18 and 19 are sent but not acknowledged (200 bytes of data). 18 DATA chunks (1800 bytes) can be sent, but only 4 chunks are in the queue. Chunk **20** is the next chunk to be sent.

Figure 23.6 *Solution to Exercise 33*

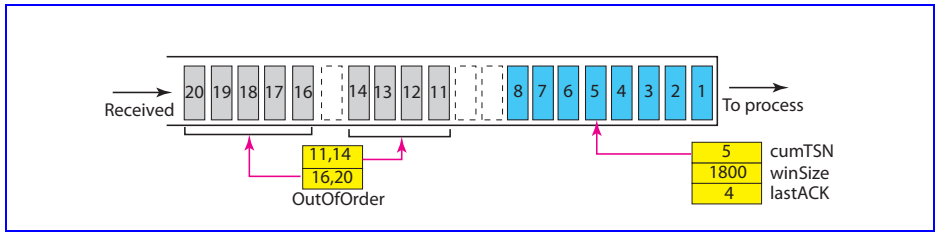
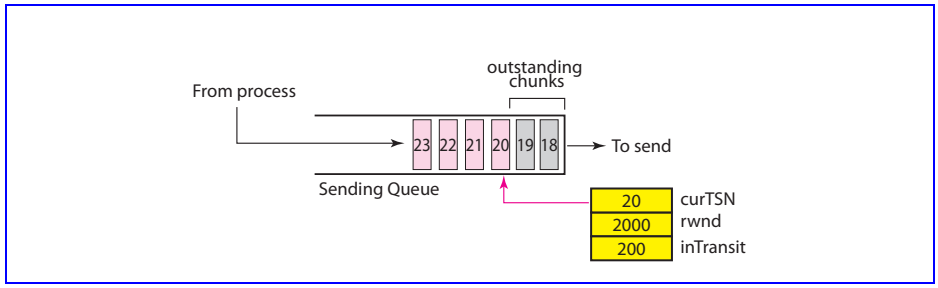


Figure 23.7 *Solution to Exercise 34*



CHAPTER 24

Congestion Control and Quality of Service

Solutions to Review Questions and Exercises

Review Questions

1. In ***congestion control***, the load on a network is prevented from exceeding the capacity. ***Quality of service*** refers to the characteristics that a flow of data seeks to attain. If there is good congestion control, then the QoS is also good and vice versa.
2. A ***traffic descriptor*** is a qualitative value that describes a data flow.
3. The ***average data rate*** is always less than or equal to the ***peak data rate***.
4. The data rate of ***bursty data*** changes suddenly in a very short period of time.
5. ***Open-loop*** congestion control policies try to prevent congestion. ***Closed-loop*** congestion control policies try to alleviate the effects of congestion.
6. The following policies can help to prevent congestion: ***a good retransmission policy***, ***use of the selective-repeat window***, ***a good acknowledgment policy***, ***a good discard policy***, and ***a good admissions policy***.
7. Congestion can be alleviated by ***back pressure***, ***a choke point***, and ***explicit signaling***.
8. The TCP send window size is determined by the ***receiver*** and by the ***congestion on the network***.
9. Frame Relay uses the ***BECN*** bit and the ***FECN*** bit to control congestion.
10. A flow of data can be described by its ***reliability***, ***delay***, ***jitter***, and ***bandwidth***.
11. ***Scheduling***, ***traffic shaping***, ***admission control***, and ***resource reservation*** can improve QoS.
12. ***Traffic shaping*** is a mechanism to control the amount and rate of traffic sent to the network. The ***leaky bucket*** method and the ***token bucket*** method can shape traffic.
13. ***Differentiated Services*** was developed to handle the shortcomings of IntServ. The main processing was moved from the core of the network to the edge of the network. Also, the ***per-flow service*** was changed to ***per-class service***.
14. When ***IntServ*** is used at the IP level, a signaling system is needed to set up the needed virtual circuit. The ***Resource Reservation Protocol*** is this signaling system.

15. The attributes are ***access rate***, ***committed burst size***, ***committed information rate***, and ***excess burst size***.
16. ***User-related*** attributes define how fast the user wants to send data. ***Network-related attributes*** define network characteristics.

Exercises

17. The bit pattern is 10110000 00010**1**1. The ***FECN*** bit is **0** and the ***BECN*** bit is **1**. There is no congestion in the forward direction, but there is congestion in the backward direction.
18. Both ***FECN*** and ***BECN*** bits are set (they are both **1s**).
19. Input: $(100/60) \times 12 + 0 \times 48 = \mathbf{20}$ gallons
    Output: **5** gallons
    Left in the bucket: $20 - 5 = \mathbf{15}$
20.

Second 1:

Initial:	→	n = 8000	
Frame **1** is sent	→	n = 4000	
Frame **2** is sent	→	n = 0	**Stop: n < Frame 3**

Second 2:

Initial:	→	n = 8000	
Frame **3** is sent	→	n = 4000	
Frame **4** is sent	→	n = 0	**Stop: n < Frame 5**

Second 3:

Initial:	→	n = 8000	
Frame **5** is sent	→	n = 4800	
Frame **6** is sent	→	n = 1600	**Stop: n < Frame 7**

Second 4:

Initial:	→	n = 8000	
Frame **7** is sent	→	n = 4800	
Frame **8** is sent	→	n = 4400	
Frame **9** is sent	→	n = 4000	
Frame **10** is sent	→	n = 2000	
Frame **11** is sent	→	n = 0	**Stop: n < Frame 12**

Second 5:

Initial:	→	n = 8000	
Frame **12** is sent	→	n = 6000	**Stop: no more frames**

21.

a. The access rate is the rate of T-1 line (**1.544 Mbps**) that connects the user to the network. Obviously, the user cannot exceed this rate.

b. The user data rate cannot exceed the access rate, the rate of the T-1 line that connects the user to the network. The user should stay below this rate (**1.544 Mbps**).

c. The CIR is **1 Mbps**. This means that the user can send data at this rate all the time without worrying about the discarding of data.

d. The user can send data at the rate of **1.2 Mbps** because it is below the access rate. However, the user sends 6 million bits per 5 seconds, which is above B_c (5 million per 5 seconds), but below B_c+B_e (6 million per 5 seconds). The network will discard no data if there is no congestion, but it may discard data if there is congestion.

e. The user can send data at the rate of **1.4 Mbps** because it is below the access rate. However, the user sends 7 million bits per 5 seconds, which is above B_c and above B_c+B_e (6 million per 5 seconds). In other words, the user rate is beyond its share. The network will discard some data to limit the data rate.

f. To be sure that the network never discard her data, the user should stay at or below CIR rate all the time, which means below or at **1 Mbps**.

g. If the user can accept possible data discarding in case of congestion, she can send at a higher rate if the number of bits is below B_c+B_e (6 million per 5 seconds in this case). This mans that the user can send at **1.2 Mbps** all the time if she accepts this risk.

22. There is no risk of discarding at all because in 5 seconds, the user has sends

$$1.4 \text{ Mbps} \times 2 + 0 \times 3 = \mathbf{2.8} \text{ million bits in 5 seconds,}$$

which is below the B_c.

23. CTD is the average *cell transfer delay*. If each cell takes 10 µs to reach the destination, we can say that CTD = [(10 µs × *n*) / *n*] in which *n* is the total number of cells transmitted in a period of time. This means that CTD = **10 µs**

24.

a. CLR is the average *cell loss ratio*. If the network has lost 5 cells out of 10,000, then CLR = **5 /10,000** = **1/2000**.

b. CER is the average *cell error ratio*. If two cells out of 10,000 are in error, then CLR = **2 /10,000** = **1/5000**.

CHAPTER 25

Domain Name System (DNS)

Solutions to Review Questions and Exercises

Review Questions

1. When the name space is large, searching a name in ***hierarchical*** structure (tree) is much faster that searching it in a ***flat*** structure (linear). The first can use a binary search; the second needs to use a sequential search.
2. A ***primary server*** is a server that stores a file about the zone for which it is an authority. It is responsible for creating, maintaining, and updating the zone file. A ***secondary server*** is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. The secondary server neither creates nor updates the zone files.
3. ***Generic domain***, ***country domain***, and ***inverse domain***.
4. The ***inverse domain*** maps an address to a name.
5. In ***recursive resolution*** the client queries just one server. In ***iterative resolution*** the client queries more than one server.
6. An ***FQDN*** is a domain name consisting of labels beginning with the host and going back through each level to the root node.
7. A ***PQDN*** is a domain name that does not include all the levels between the host and the root node.
8. A ***zone*** is an area for which a server is responsible.
9. ***Caching*** reduces the search time for a name.
10. A ***DNS*** message is either a ***query*** or a ***response***.
11. ***DDNS*** is needed because the many address changes makes manual updating inefficient.

Exercises

12.

    a. **PQDN**
    b. **FQDN**
    c. **PQDN**

d. **FQDN**

13.

a. **FQDN**
b. **FQDN**
c. **PQDN**
d. **PQDN**

14. No specific answer.
15. Remembering a ***name*** is often easier than remembering a ***number.***
16. If computer A needs the IP address of destination B, the answer is in the IP data field. The IP destination field contains the address of computer A.
17. There are ***three labels*** but ***four levels*** of hierarchy since the root is considered a level.
18. It is always shorter by ***at least a dot***.
19. This is a ***generic domain***.
20. The recursive resolution is ***normally faster*** because multiple requests are handled by faster servers.
21. The number of question sections and answer sections must be the same. The relationship is ***one-to-one***.

CHAPTER 26

Remote Log-in, Electronic Mail and File Transfer

Solutions to Review Questions and Exercises

Review Questions

1. In ***local log-in***, the user terminal is directly connected to the target computer; in ***remote log-in***, the user computer is connected to the target computer through the Internet.
2. The **leftmost bit** of a data character is **0**; the **leftmost bit** of a control character is **1**.
3. Options in TELNET are negotiated using four control characters **WILL**, **WONT**, **DO**, and **DONT**.
4. The addressing system has a ***local part*** and a ***domain name*** separated by the **@** symbol. The local part is a file that holds the mail. The domain name refers to a host that receives and sends mail.
5. A ***user agent (UA)*** is a software package that composes, reads, replies to, and forwards messages.
6. ***Multipurpose Internet Mail Extension (MIME)*** is a supplementary protocol that allows non-ASCII data to be sent through SMTP.
7. SMTP is a ***push*** protocol; it pushes the message from the client to the server. In other words, the direction of the bulk data (messages) is from the client to the server. On the other hand, retrieving messages from mail boxes needs a ***pull*** protocol; the client must pull messages from the server. The direction of the bulk data is from the server to the client. The third stage uses a message access agent (MAA) such as POP3 or IMAP4.
8. ***FTP*** copies a file from one host to another.
9. One connection is for ***data transfer***, the other connection is for ***control information***.
10. ***ASCII files***, ***EBCDIC files***, and ***image files***.
11. The three transmission modes in FTP are ***stream***, ***block***, and ***compressed***.
12. ***Storing a file*** means copying a file from the client to the server. ***Retrieving a file*** means copying a file from the server to the client.

13. ***Anonymous FTP*** allows a user to access files without an account or password on a remote server.

Exercises

14. The pattern is:

    11110011 00111100 11111111 11111111

    Note: The last byte is duplicated because it is the same as IAC; it must be repeated to be interpreted as data.

15. There are **15** characters in the command (including the end of line). Each character is sent separately to the server and each is echoed and acknowledged by the server. Each echo from the server is then acknowledged by the client. A total of **45** packets must be sent.

16. To do the task in Exercise 1, we need to send:

    Client to Server: **IAC DO BINARY** (3 bytes)
    Server to Client: **IAC WILL BINARY** (3 bytes)
    Client to Server: **11110011 00111100 11111111 11111111** (4 bytes)

    If each transmission is encapsulated in a single TCP segment with 20 bytes of header, there will be 3 segments of 23, 23, and 24 bytes for the total of **70 bytes** or **560 bits**.

17. Three transmissions, each with a minimum size of 72 bytes, mean a total of **216 bytes** or **1728 bits**.

18. If we assume the useful bits are the 3 bytes of data from Exercise 1:

    3 bytes of data / 216 bytes transmitted = **1:70**

19.
    a. **IAC WILL ECHO**
    b. **IAC DONT ECHO**
    c. **IAC IP** (Interrupt Process)
    d. **IAC GA** (Go Ahead)

20.
    MIME-version: 1.1
    Content-Type: Text/Plain
    Content-Transfer-Encoding: 7bit

21.
    MIME-version: 1.1
    Content-Type: Image/JPEG; name=”something.jpg”
    Content-Transfer-Encoding: base64

22. ***Connection establishment*** is needed for mail transfer because the messages sent relay necessary information about the communication to the client and server software, not just whether the computers have a connection via TCP.

23. There should be limitations on ***anonymous FTP*** because it is unwise to grant the public complete access to a system. If the commands that an anonymous user could use were not limited, that user could do great damage to the file system (e.g., erase it completely).
24. ***FTP*** does not need a message format because there is no need to send additional information back and forth aside from the ***commands*** and ***responses***, which use the control connection.

CHAPTER 27

WWW and HTTP

Solutions to Review Questions and Exercises

Review Questions

1. ***HTTP*** is a file transfer protocol that facilitates access to the ***WWW***.
2. ***HTTP*** is like ***SMTP*** because the data transferred between the client and server are similar in appearance to SMTP messages. Also, the format of the messages is controlled by MIME-like headers.
3. ***HTTP*** is like ***FTP*** because they both transfer files and use the services of TCP.
4. The ***URL*** is a standard that facilitates the access of documents on the Web. A URL defines a method, a host computer, a port, and a path.
5. A ***proxy server*** is a computer that keeps copies of responses to recent requests. When an HTTP client has a request, the cache of the proxy server is checked before the request goes to the regular server.
6. A ***browser*** consists of a controller, client program, and interpreters.
7. A ***Web document*** can be classified as either ***static***, ***active***, or ***dynamic***.
8. ***Hypertext Markup Language (HTML)*** is a language for creating Web pages.
9. A ***dynamic document*** is the product of a program run by a server as requested by a browser. An ***active document*** is the product of a program sent from the server to the client and run at the client site.
10. The ***Common Gateway Interface (CGI)*** is a standard that creates and handles dynamic documents.
11. ***Java*** is one of the languages that is used to write an ***active document***.

Exercises

12. The ***first picture*** will be ***centered*** and the ***second*** will be ***at the bottom***.
13. On the screen you see: The publisher of this book is **McGraw-Hill Publisher**

14.
GET /usr/users/doc/doc.1 HTTP /1.1
Date: Fri, 13-Jan-06 08:45:20 GMT
MIME-version: 1.0
Accept: image/gif
Accept: image/jpeg
Last modified: Mon, 09-Jan-06

15.
HTTP/1.1 200 OK
Date: Fri, 13-Jan-06 08:45:25 GMT
Server: Challenger
MIME-version: 1.0
Content-length: 4623

(Body of document)

16.
HTTP/1.1 302 Moved permanently
Date: Fri, 13-Jan-06 08:45:25 GMT
Server: Challenger
Location: /usr/deads/doc.1

17.
HTTP/1.1 400 Bad Request
Date: Fri, 13-Jan-06 08:45:25 GMT
Server: Challenger

18.
HTTP/1.1 401 Unauthorized
Date: Fri, 13-Jan-06 08:45:25 GMT
Server: Challenger

19.
HEAD /bin/users/file HTTP /1.1
Date: Fri, 13-Jan-06 10:40:22 GMT
MIME-version: 1.0
From: mzzchen@sinonet.cn

20.
HTTP/1.1 200 OK
Date: Fri, 13-Jan-06 10:40:27 GMT
Server: Challenger
Content-type: image/jpeg
Content-length: 4623

21. **COPY /bin/usr/bin/file1 HTTP /1.1**
    Date: Fri, 13-Jan-06 10:52:12 GMT
    MIME-version: 1.0
    Location: /bin/file1
22. **HTTP/1.1 200 OK**
    Date: Fri, 13-Jan-06 10:53:00 GMT
    Server: Challenger
23. **DELETE /bin/file1 HTTP /1.1**
    Date: Fri, 13-Jan-06 11:04:22 GMT
    Server: Challenger
    Authentication: swd22899/3X4ake88rTfh (Not discussed in the book)
24. **HTTP/1.1 200 OK**
    Date: Fri, 13-Jan-06 11:11:02 GMT
    Server: Challenger
25. **GET /bin/etc/file1 HTTP /1.1**
    Date: Fri, 13-Jan-06 11:22:08 GMT
    MIME-version: 1.0
    Accept: */*
    If-modified-since: 23-Jan-1999 00:00:00 GMT
26. **HTTP/1.1 200 OK**
    Date: Fri, 13-Jan-06 11:24:06 GMT
    Server: Challenger
    MIME-version: 1.0
    Content-length: 2686

    (Body of document)
27. **GET /bin/etc/file1 HTTP /1.1**
    Date: Fri, 13-Jan-06 11:41:02 GMT
    MIME-version: 1.0
    Accept: */*
    Host: Mercury
    Authentication: swd22899/3X4ake88rTfh (Not discussed in the book)
28. **HTTP/1.1 200 OK**

Date: Fri, 13-Jan-06 11:45:06 GMT
Server: Challenger
MIME-version: 1.0
Content-type: text/html
Content-length: 4156

(Body of document)

29.

PUT /bin/letter HTTP /1.1
Date: Fri, 13-Jan-06 11:47:00 GMT
MIME-version: 1.0
Accept: text/html
Accept: image/gif
Accept: image/jpeg
Location: /bin/letter

30.

HTTP/1.1 200 OK
Date: Fri, 13-Jan-06 11:51:02 GMT
Server: Challenger
MIME-version: 1.0
Content-type: text/html
Content-length: 6324
Age: 10:06:02
Expires: 09-May-06 00:00:00 GMT

CHAPTER 28

Network Management: SNMP

Solutions to Review Questions and Exercises

Review Questions

1. ***Network management*** is defined as monitoring, testing, configuring, and troubleshooting network components to meet a set of requirements defined by an organization.
2. The functions performed by a network management system can be divided into five broad categories: **configuration management**, **fault management**, **performance management**, ***security management***, and **accounting management**,
3. The ***configuration management*** system updates information about the status of each entity and its relation to other entities.
4. Configuration management can be divided into two subsystems: ***reconfiguration*** and **documentation**.
5. ***Fault management*** supervises the operation of the network, which depends on the proper operation of each individual component and its relation to other components.
6. A fault management system has two subsystems: ***reactive fault management*** and ***proactive fault management.***
7. ***Performance management*** monitors and controls the network to ensure that it is running as efficiently as possible.
8. The four measurable quantities in performance management are ***capacity***, ***traffic***, ***throughput***, and ***response time***.
9. ***Security management*** is responsible for controlling access to the network based on the predefined policy.
10. ***Accounting management*** is the control of users' access to network resources through charges. Under accounting management, individual users, departments, divisions, or even projects are charged for the services they receive from the network.

Exercises

11.

INTEGER tag: **02**
length: **04**
value: **00 00 05 B0**

Answer: **02 04 00 00 05 B0**

12.

OCTET STRING tag:**04**
length: **0C**
value: **48 65 6C 6C 6F 20 57 6F 72 6C 64 2E**
H e l l o space W o r l d .

Answer: **04 0C 48 65 6C 6C 6F 20 57 6F 72 6C 64 2E**

13.

OCTET STRING tag: **04**
length of the length field (2 bytes) (10000010) = **82**
length (1000 bytes) = **03 E8**
value (1000 character)

Answer: **04 82 03 E8** (**Plus 1000 bytes of characters**)

14.

30 16	sequence, length
02 04 00 00 09 29	INTEGER, length, value (2345)
04 08 43 4F 4D 50 55 54 45 52	OCTET STRING, length, value (COMPUTER)
40 04 B9 20 01 05	IP address, length, value (185.32.1.5)

15.

30 15	sequence, length
43 04 00 00 2E E0	TIME TICK, length, value (1200)
02 04 00 00 38 E4	INTEGER, length, value (14564)
06 07 01 03 06 01 02 01 07	Object ID, length, value (1.3.6.2.1.7)

16.

30 18	sequence, length
02 04 00 00 09 29	INTEGER, length, value (2345)
02 04 00 00 04 D4	INTEGER, length, value (1236)
02 04 00 00 00 7A	INTEGER, length, value (122)
02 04 00 00 04 D4	INTEGER, length, value (1236)

17.

30 43	sequence, length
30 41	sequence, length
02 04 00 00 09 29	INTEGER, length, value (2345)
04 08 43 4F 4D 50 55 54 45 52	OCTET STRING, length, value (COMPUTER)
41 04 00 00 01 59	counter, length, value (345)
30 29	sequence, length
02 04 00 00 04 63	INTEGER, length, value (1123)
04 04 44 49 53 4B	OCTET STRING, length, value (DISK)
41 04 00 00 05 96	counter, length, value (1430)
30 15	sequence, length
02 04 00 00 0D 80	INTEGER, length, value (3456)
04 07 4D 4F 4E 49 54 4F 52	OCTET STRING, length, value (MONITOR)
41 04 00 00 09 09	counter, length, value (2313)

18.

a. the integer **16913458**
b. sequence of 2 integers: **17** and **20**
c. sequence of the string: "**ACB**" and the integer **5140**
d. sequence of an IP address **35.81.98.113** and the integer **5138**

CHAPTER 29

Multimedia

Solutions to Review Questions and Exercises

Review Questions

1. In ***streaming stored audio/video***, a client first downloads a compressed file and then listens to or watches it. ***In streaming live audio/video***, a client listens to or watches a file while it is being downloaded.
2. In ***frequency masking***, a loud sound partially or totally masks a softer sound. In ***temporal masking***, a loud sound blocks other sounds for a period of time.
3. A ***metafile*** contains information about a corresponding audio/video file.
4. ***RTSP*** is a control protocol that adds some functionalities to the streaming process. It is an out-of-band controlling protocol that functions like the FTP control connection.
5. ***Jitter*** manifests itself as a gap between what is heard or seen.
6. ***SIP*** is an application layer protocol that establishes, manages, and terminates a multimedia session.
7. ***JPEG*** is used to compress images. ***MPEG*** is used to compress video.
8. ***Blocking*** decreases the number of calculations.
9. The ***DCT*** reveals the number of redundancies of a block.
10. In ***spatial compression***, JPEG compresses each frame. In ***temporal compression***, redundant frames are removed.

Exercises

11.
    a. **9** packets played; **11** packets left
    b. **12** packets played; **8** packets left
    c. **17** packets played; **3** packets left
    d. **22** packets played; **8** packets left
12. ***TCP*** is not suitable for real-time traffic because it has no provision for timestamping, it does not support multicasting, and, most importantly, it retransmits lost or

corrupted packets. ***RTP*** is a protocol designed to handle real-time traffic. RTP handles timestamping, sequencing, and mixing. There is no retransmission when RTP is used with UDP.

13. We can say that ***UDP*** plus ***RTP*** is more suitable than ***TCP*** for multimedia communication. The combination uses the appropriate features of UDP, such as timestamp, multicasting, and lack of retransmission, and appropriate features of ***RTP*** such as error control.
14. ***RTCP*** is a control protocol that handles messages that control the flow and quality of data. It also allows recipient feedback. TCP allows for these types of messages, so it doesn't need RTCP.
15. The ***web server*** and ***media server*** can be two distinct machines since it is the metafile-data file combination that is important.
16. ***SIP*** can be modified to be used for interactive video such as teleconferencing.
17. Both ***SIP*** and ***H.323*** use the Internet as a telephone network. The main difference is that H.323 uses a gateway to transform a telephone network message to an Internet message. See Table 29.1.

Table 29.1 *Solution to Exercise 17*

Issues	***SIP***	***H.323***
Transport layer	UDP or TCP	UDP for data, TCP for control
Address format	IP address, e-mail address, or phone number	IP address
Establishment	3-way handshake	H.225, Q.931, H.245
Data exchange	UDP, TCP	RTP, RTCP, UDP, TCP
Termination	BYE message	Q.931

18. We can mention some of the problems involved in full implementation of ***voice over IP***:
    a. Your computer has to be on all the time as well as connected to the Internet.
    b. If the Internet connection is down, your phone service is also down.
    c. Voice quality can be a problem due to echoes or delays.
    d. There could be potential call degradation if the computer is also doing heavy processing.
19. ***H.323*** can also be used for video, but it requires the use of videophones. Currently most people don't have videophones.

CHAPTER 30

Cryptography

Solutions to Review Questions and Exercises

Review Questions

1. Only ***one key*** (the shared secret key) is needed for two-way communication. However, for more security, it is recommended that a different key be used for each direction.
2. A shared secret key can only be used between two entities. Alice needs a shared secret key to communicate with Bob and a different shared secret key to communicate with John.
3. Each person in the first group needs to have **10** keys to communicate with all people in the second group. This means we need at least $10 \times 10 =$ **100** keys. Note that the same keys can be used for communication in the reverse direction. However, note that we are not considering the communication between the people in the same group. For this purpose, we would need more keys.
4. Each person needs 9 keys to communicate with the other people. At first glance, it looks like we need 10×9 keys. However, if the same key can be used in both directions (from A to B and from B to A), then we need only $(10 \times 9) / 2 =$ **45 keys**.
5. For two-way communication, **4** keys are needed. Alice needs a private key and a public key; Bob needs a private key and a public key.
6. Alice can use the same pair of keys (private and public) to communicate with both Bob and John. However, if there is a need for two-way communication, Bob and John need to have their own pair of keys. In other words, for two-way communication **6** keys are needed, two for each entity.
7. For two-way communication, the people in the first group need 10 pairs of keys, and the people in the second group need a separate 10 pairs of keys. In other words, for two-way communication **40** keys are needed.
8. Each person in the group needs only one pair of keys (private and public). In other words, **20** keys are needed for two-way communication.

Exercises

9. If the two persons have two pairs of asymmetric keys, then they can send messages using these keys to create a ***session symmetric key***, a key which is valid for one session and should not be used again. Another solution is to use a ***trusted center*** that creates and send symmetric keys to both of them using the symmetric key or asymmetric key that has been already established between each person and the trusted center. We will discuss this mechanism in Chapter 31.
10. Each person can create a pair of keys. Each person then keeps the private key and advertises the public key (for example, posting on her web site). Another common solution is to use a ***trusted party*** that accepts the private key of individual and then distributes it using certificates. We will discuss this mechanism in Chapter 31.
11.
    a. We can show the encryption character by character. We encode characters A to Z as 0 to 25. To wrap, we subtract 26.

T	**19 + 20 = 39 – 26 = 13**	→	**N**
H	**07 + 20 = 27 – 26 = 01**	→	**B**
I	**08 + 20 = 28 – 26 = 02**	→	**C**
S	**18 + 20 = 38 – 26 = 12**	→	**M**
I	**08 + 20 = 28 – 26 = 02**	→	**C**
S	**18 + 20 = 38 – 26 = 12**	→	**M**
A	**00 + 20 = 20**	→	**U**
N	**13 + 20 = 33 – 26 = 07**	→	**H**
E	**04 + 20 = 24**	→	**Y**
X	**23 + 20 = 43 – 26 = 17**	→	**R**
E	**04 + 20 = 24**	→	**Y**
R	**17 + 20 = 37 – 26 = 11**	→	**L**
C	**02 + 20 = 22**	→	**W**
I	**08 + 20 = 28 – 26 = 02**	→	**C**
S	**18 + 20 = 38 – 26 = 12**	→	**M**
E	**04 + 20 = 24**	→	**Y**

The encrypted message is ***NBCM CM UH YRYLWCMY***.

b. We can show the decryption character by character. We encode characters A to Z as 0 to 25. To wrap the negative numbers, we add 26.

N	**13 – 20 = –07 + 26 = 19**	→	**T**
B	**01 – 20 = –19 + 26 = 07**	→	**H**
C	**02 – 20 = –18 + 26 = 08**	→	**I**
M	**12 – 20 = –08 + 26 = 18**	→	**S**
C	**02 – 20 = –18 + 26 = 08**	→	**I**
M	**12 – 20 = –08 + 26 = 18**	→	**S**
U	**20 – 20 = 00**	→	**A**
H	**07– 20 = –13 + 26 = 13**	→	**N**
Y	**24 – 20 = 04**	→	**E**
R	**17 – 20 = –03 + 26 = 23**	→	**X**
Y	**24 – 20 = 04**	→	**E**
L	**11 – 20 = –09 + 26 = 17**	→	**R**
W	**22 – 20 = 02**	→	**C**
C	**02 – 20 = –18 + 26 = 08**	→	**I**
M	**12 – 20 = –08 + 26 = 18**	→	**S**
Y	**24 – 20 = 04**	→	**E**

The decrypted message is ***THIS IS AN EXERCISE***.

12. We can, ***but it is not safe at all***. This is a shift cipher with key = 2 and only two symbols. All 0s are changed to 1s and all 1s are changed to 0. The intruder can easily intercept the ciphertext and find the plaintext.
13. We can, ***but it is not safe at all***. The best we can do is to change a 0 sometimes to 0 and sometimes to 1 and to change a 1 sometimes to 0 and sometimes to 1. It can be easily broken using trial and error.
14. We divide the message into five-character blocks and add two bogus characters (Z) at the end. We have:

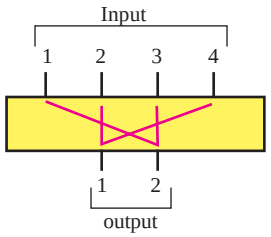
Plaintext:	**INTER**	**NETZZ**
Ciphertext:	**TRNIE**	**TZENZ**

15. Input: 111001 → output: **001111**
16. Input: 100111 → output: **111100**
17.
    a. Input: **1 1 0 0 1 0** → output: **0 1**
    b. Input: **1 0 1 1 0 1** → output: **0 0**
18. The possible number of inputs is $2^6 = \mathbf{32}$ (000000 to 111111). The possible number of output is $2^2 = \mathbf{4}$ (00 to 11).
19.
    a. Input: 1011 (the leftmost bit is 1), the output is: **110**

b. Input: 0110 (the leftmost bit is 0), the output is: **011**

20. See Figure 30.1. It is a ***compression permutation*** because the number of outputs is less than the number of inputs.

Figure 30.1 *Solution to Exercise 20*



21. We can follow the process until we find the value of d. For the last step, we need to use an algorithm defined in abstract algebra. We don't expect students know how to do it unless they have taken a course in abstract algebra or cryptography.

    a. $n = p \times q = 19 \times 23 =$ ***437***

    b. $\phi = (p - 1) \times (q - 1) = 18 \times 22 =$ ***396***

    c. $e =$ ***5*** $d =$ ***317***

    We can check that $e \times d = 5 \times 317 = 1 \bmod 396$

22. The main point in the RSA method is that n needs to be a very large number so that an intruder cannot factor it. In our example, n can be easily broken because, the intruder can find that $n = 187 = 17 \times 11$. In other words, p is 17 and q is 11. Now, the intruder can calculate the value of $\phi = (17 - 1) \times (11 - 1) = 160$, When the intruder knows this number and the public value of $e = 17$, the value of d can be found as ***d = 113***. The secret can be broken.

23. Bob knows p and q, so he can calculate $\phi = (p - 1) \times (q - 1)$ and find d such that $d \times e = 1 \bmod \phi$. Eve does not know the value of p or q. She just knows that $n = p \times q$. If n is very large (hundreds of digits), it is very hard to factor it to p and q. Without knowing one of these values, she cannot calculate ϕ. Without ϕ, it is impossible to find d given e. The whole idea of RSA is that n should be so large that it is impossible to factor it.

24.

    a. Encryption:

P1 = "F" = 05	→	**C1 = 05^{13} mod 77 = 26**
P2 = "I" = 08	→	**C2 = 08^{13} mod 77 = 50**
P3 = "N" = 13	→	**C3 = 13^{13} mod 77 = 41**
P4 = "E" = 04	→	**C4= 04^{13} mod 77 = 53**

b. Decryption:

C1 = 26 → P1 = 26^{37} mod 77 = 05 = "F"
C2 = 50 → P2 = 50^{37} mod 77 = 08 = "I"
C3 = 41 → P3 = 41^{37} mod 77 = 13 = "N"
C4= 53 → P4= 53^{37} mod 77 = 04 = "E"

25. The value of $e = 1$ means no encryption at all because **$C = P^e = P$**. The ciphertext is the same as plaintext. Eve can intercept the ciphertext and use it as plaintext.
26. If the value of *e* is so small, and the value of P is not very large, modular arithmetic looses its effect. For example, if **$C = P^2$** is always smaller than *n*, Eve can easily find the value of P by using $P = (C)^{1/2}$. It it therefore recommended not to choose a very small value for *e*.
27. Although Eve can use what is called the *ciphertext attack* to find Bob's key, she could have done it by intercepting the message. In the ciphertext attack, the intruder can get several different ciphertexts (using the same pair of keys) and find the private key of the receiver. If the value of the public key and *n* are very large, this is a very time-consuming and difficult task.
28. $R1 = 7^2 \bmod 23 = \mathbf{3}$
    $R2 = 7^5 \bmod 23 = \mathbf{17}$
    Alice calculates $K = (R2)^2 \bmod 23 = 17^2 \bmod 23 = \mathbf{13}$
    Bob calculates $K = (R1)^2 \bmod 23 = 3^5 \bmod 23 = \mathbf{13}$
29. Nothing happens in particular. Assume both Alice and Bob choose x = y = 9. We have the following situation with $g = 7$ and $p = 23$:
    $R1 = 7^9 \bmod 23 = \mathbf{15}$
    $R2 = 7^9 \bmod 23 = \mathbf{15}$
    Alice calculates $K = (R2)^9 \bmod 23 = 15^9 \bmod 23 = \mathbf{14}$
    Bob calculates $K = (R1)^9 \bmod 23 = 15^9 \bmod 23 = \mathbf{14}$

CHAPTER 31

Network Security

Solutions to Review Questions and Exercises

Review Questions

1. A ***nonce*** is a large random number that is used ***only once*** to help distinguish a fresh authentication request from a repeated one.
2. The ***N^2*** problem refers to the large number of keys needed for symmetric key cryptography. For ***N*** people, ***$(N \times (N-1))/2$*** keys are needed, which is proportional to N^2.
3. Both the ***Needham-Schroeder*** and the ***Otway-Rees*** protocols use a ***KDC*** for user authentication.
4. The ***Kerberos authentication server (AS)*** registers each user and grants each user a user identity and a password. The AS issues a session key for use between the sender and the ticket-granting server (TGS).
5. The ***Kerberos TGS*** issues a ticket for the real server and provides the session key between the sender and the receiver.
6. ***X.509*** is a protocol that describes the certificate in a structural way.
7. A ***certification authority (CA)*** is a federal or state organization that binds a public key to an entity and issues a certificate.
8. A ***long password*** is more immune to guessing than a ***short password***. However, a long password is difficult to remember; it is often written somewhere. This may make it easier for the adversary to steal it.
9. A ***frequently-changed password*** is more secure than a ***fixed password*** but less secure than a ***one-time password.*** However, a one-time password needs more effort from the system and the user. The system needs to check if the password is fresh every time the user tries to use the password. The user needs to be careful not to use the pervious one. A more frequently changed password can be used as an alternative. One solution is that the system initializes the process of changing the password by sending the new password, through a secure channel, and challenging the user to be sure that the right user has received the new password.
10. One way to prevent a ***guessing attack*** on a password is to use long passwords. For example, it is more difficult to guess a 10-digit password than a 4-digit one. Banks

recommend that a customer not use a short PIN (a type of password). In particular, they recommend not using an easily-guessed number such as the birth year. Banks also request a change in the PIN when a stolen bank card is reported and replaced by a new one.

Exercises

11.

a. The algorithm meets the first criteria (***one-wayness***). It is not possible to find the original numbers if the digest is given. For example, if we know the digest is 76, we cannot find the original ten numbers. They can be any set of 10 numbers.

b. The algorithm does not meet the second criteria (***weak collision***). If the digest is given, we can create 10 numbers that hash to the same digest. For example, Eve, without knowing the original set of numbers, can intercept the digest of **51** and create the set {12, 23, 45, 12, 34, 56, 9, 12, 34, 14} and send it with the digest **51** to Bob. Bob is fooled and believes that the set is authentic.

c. The algorithm does not meet the third criteria (***strong collision***). If the digest is given, we can create at least two sets of 10 numbers that hash to the same digest. For example, Alice can create two sets {12, 23, 45, 12, 34, 56, 9, 12, 34, 14} and {12, 23, 45, 16, 34, 56, 9, 12, 34, 10} that both hash to **51**. Alice can send the first set and the digest to Bob, but later she can claimed that she sent the second set.

12.

a. The algorithm meets the first criteria (***one-wayness***). Most of the characters are lost in the process and cannot be reproduced from the digest.

b. The algorithm does not meet the second criteria (***weak collision***). If the digest is given, we can create a message as long as the characters 1, 11, 21,..., 91 are the same as the corresponding characters in the digest. Eve, without knowing the original set of characters, can intercept the digest and create a new set out of the digest and send it with the digest to Bob. Bob is fooled and believes that the set is authentic.

c. The algorithm does not meet the third criteria (***strong collision***). We can easily create two messages in which characters 1, 11, 21,..., 91 are the same but the other characters are different. The digests for both messages are the same. Alice can send the first message and the digest to Bob, but later she can claim that she sent the second set.

13. The possible number of digests is 2^N because each bit can be in one of the two values (0 or 1).

14. It is more probable to find two people with the same birthday than to find a person born on a particular day of the year. For example, in a party of 10 people, we can find the probabilities for the two cases:

a. The probability that a person is born on a particular day (such as February 20) is **0.027** (almost **3** percent)

b. The probability that two or more persons are born in the same day is **0.117** (almost **12** percent)

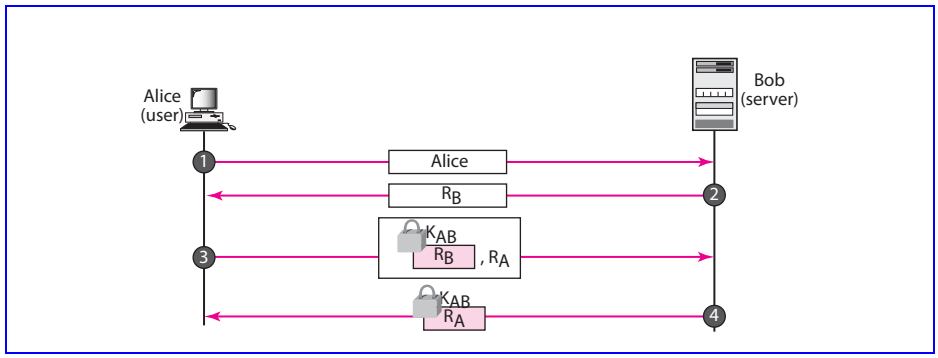
The difference increases sharply when the number of people in a party reaches 20 or more. In the classic birthday probability problem, if there are **23** people in a party, the probability is more than fifty percent that two people will have the same birthday.

15. The second and third criteria for a hashing function are closely related to the solution found in problem 14. In the problem we try to related the number of people at the party to the number of days in a year. In a hashing function, we can relate the number of possible messages to the number of possible digests. To understand the problem assume that there are only 10 possible messages (number of people at the party) but there are 365 possible digests.
    a. If a particular digest is given (a particular birthday), the probability that Eve can find one of the ten messages (one of the ten people in the party) is 0.027 (2.7 percent).This is related to the weak collision. The probability is very weak. That is why it is called ***weak collision***.
    b. The probability that Alice can create two or more messages with the same digests is the probability of finding two or more people with the same birthday in a party. If the number of possible messages is 10 and the number of possible digest is 365, this probability is 0.117 or (11 percent). That is why this criterion is called ***strong collision***. The probability is higher. It is more probable that Alice can find two or messages with the same digest than Eve can find a message with a given digest.

    The above discussion leads us to the point that we should worry more about the second criterion that the first. To decrease the probability of both criteria, we need to increase the number of possible digests and the number of possible messages. We need to increase the number of bits in a digest and impose a minimum number of bits on messages.
16. A ***fixed-size digest*** is more feasible. A ***variable-size digest*** needs to be dependent on the length of the message, which makes applying the criteria more difficult and the function itself more involved.
17. The whole idea of a sophisticated hash function such as ***SHA-1*** is that the partial digest of each block is dependent on the partial digest of the previous block and the message on the current block. Each block mingles and mixes the bits in a such a way that changing even one bit in the last block of the message may changed the whole final digest.
18. We can distinguish between the two:
    a. A signed hash normally means first making a hash and then encrypting it with a secret key.
    b. A MAC normally means first concatenating the secret key with the message and then applying the hash function.
19. It is normally both. The entity authentication (based on the PIN) is needed to protect the person and the bank in case the money card is stolen. The message authentication is normally needed for the entity authentication.

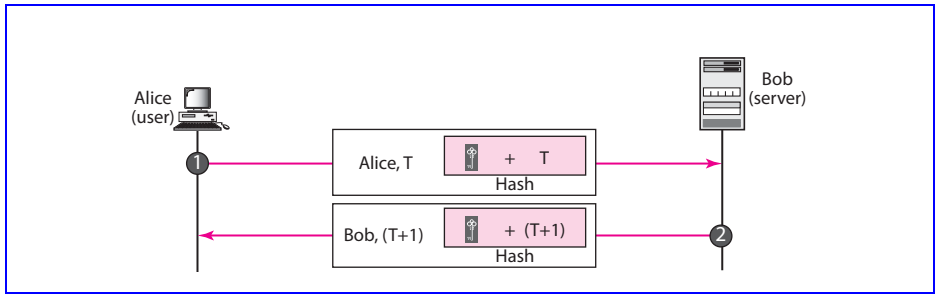
20. Figure 31.1 show one scheme using four messages. In this scheme, Alice, the initiator, needs to authenticate herself before Bob does the same. After the third message, Alice is authenticated for Bob; after the fourth message, Bob is authenticated for Alice. Although, the number of messages can be reduced to three, but (as you can see in textbooks devoted to security) the three-message scheme suffers from some flaws.

Figure 31.1 *Solution to Exercise 20*



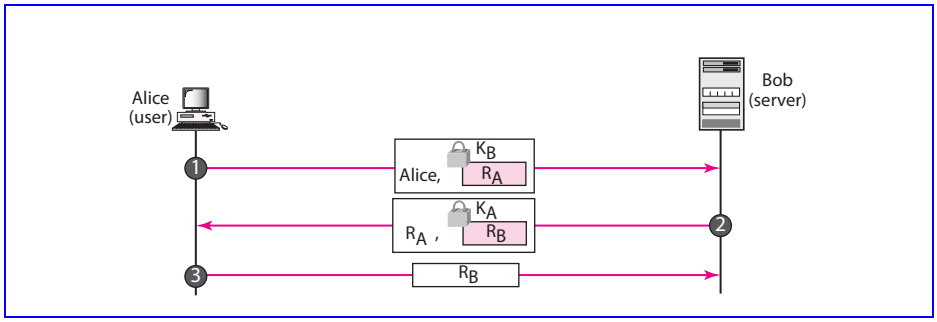
21. Figure 31.2. shows one scheme. Note that the scheme forces Bob to use the timestamp which is related to the timestamp used by Alice (T+1), this ensures that the two messages belongs to the same session.

Figure 31.2 *Solution to Exercise 21*



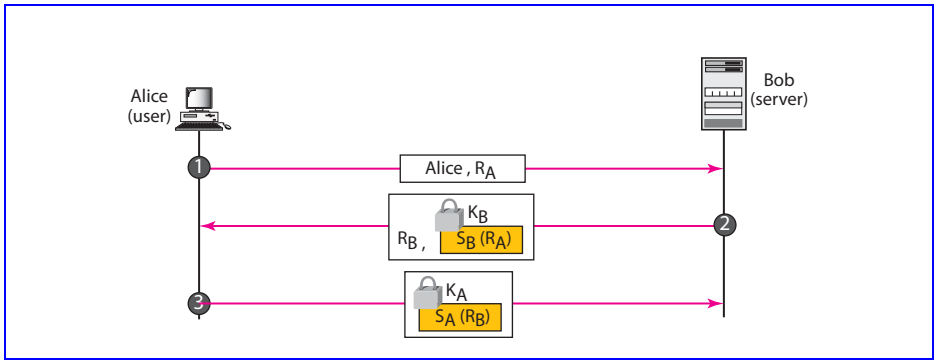
22. Figure 31.3 shows one scheme. In the first message, Alice introduces herself and sends a nonce (R_A) encrypted with Bob's public key. In the second message, Bob decrypts the first message and sends R_A in plain text to authenticate himself. Bob also challenges Alice in the second message by sending his nonce (R_B) encrypted with Alice's public key. In the third message, Alice can authenticate herself by sending Bob's decrypted nonce (R_B). Note that in this scheme, Bob, the server, has been authenticated for Alice, the user, before Alice is authenticated for Bob. However, Bob has not released any information that endangers security.
23. Figure 31.4 shows one simple scheme. Note that in the second message, Bob signs the message with his private key. When Alice verifies the message using Bob's public key, Bob is authenticated for Alice. In the third message, Alice signs the

Figure 31.3 *Solution to Exercise 22*



message with her private key. When Bob verifies the message using Alice's public key, Alice is authenticated for Bob.

Figure 31.4 *Solution to Exercise 23*



24. The ***encryption*** protects the student and the university for the ***first time***. However, the intruder can intercept the encrypted password and replay the process some other times. The intruder does not have to know the password in plaintext; the encrypted password suffices for replaying. The university system cannot determine if the student has encrypted the message again or the intruder is replaying it.
25. The ***timestamp*** definitely helps. If Alice adds a timestamp to the password before encrypting, the university, after decrypting, can check the freshness of the plaintext. In other words, adding a timestamp to a password, is like creating a new password each time.
26. A list of passwords can also help, but the question is how long the list should be. Another problem is that the student must remember to use the next password in the sequence. If she accidentally uses a password out of order, access will be denied.
27. If the ***KDC*** is down, nothing can take place. KDC is needed to create the session key for the two parties.
28.
    a. If the ***AS*** is down, the process cannot start because Alice cannot be authenticated.

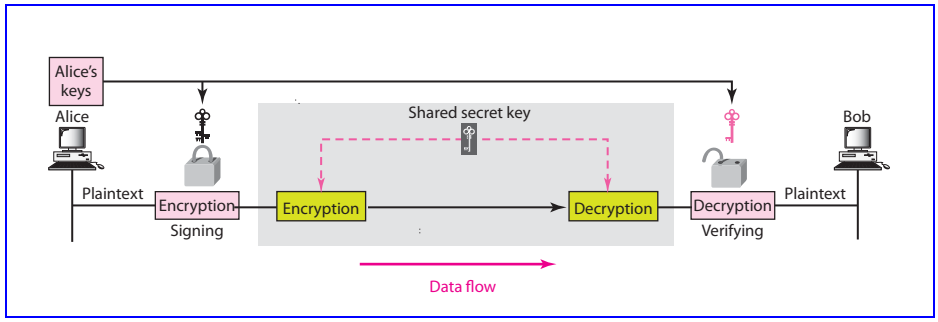
b. If the ***AS*** is running, but the ***TGS*** is down, Alice can be authenticated and get the ticket for TGS, but cannot receive the session key. Alice can apply later and present her tickets to obtain the session key. We can compare the process with air travelling. We need a ticket, but we also need a boarding pass. We can get the ticket if the airline office is open, but we cannot get the boarding pass if the flight is cancelled. We can apply another time, when that particular flight is operational to get the boarding pass.

c. If the ***AS*** and ***TGS*** are running, but the ***main server*** is down, we can get the session key, but we cannot access the main server. Some systems allow the use of a session key in a future time; some do not. The situation is like having the boarding pass to board the air craft. If the flight is delayed, we can wait and apply the boarding pass later. If the flight is cancelled, the boarding passes are probably invalid.

29. If the ***trusted center*** is down, Bob cannot obtain his certificate. Bob still can use his public key if the other party does not ask for a certificate.

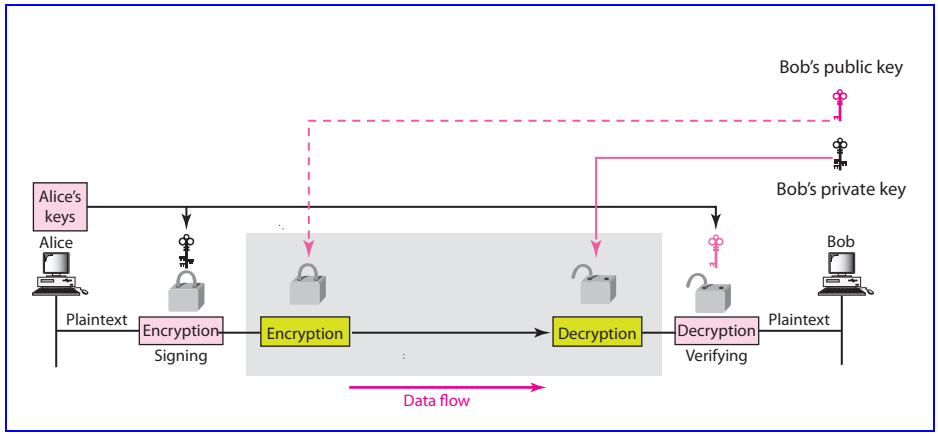
30. See Figure 31.5. The shaded area shows the encryption/decryption layer.

Figure 31.5 *Solution to Exercise 30*



31. See Figure 31.6. The shaded area shows the encryption/decryption layer.

Figure 31.6 *Solution to Exercise 31*



CHAPTER 32

Security In the Internet

Solutions to Review Questions and Exercises

Review Questions

1. ***IPSec*** needs a set of security parameters before it can be operative. In IPSec, the establishment of the security parameters is done via a mechanism called ***security association (SA)***.
2. A set of ***security parameters*** between any two entities is created using the ***security association***. Security association uses three protocols: ***IKE***, ***Oakley***, and ***SKEME*** to create a security association between two parties or a security association database between a group of users.
3. The two protocols defined by IPSec for exchanging datagrams are ***Authentication Header (AH)*** and ***Encapsulating Security Payload*** (ESP).
4. The ***Authentication Header (AH)*** protocol adds an ***AH header*** that contains next header, payload length, security parameter index, sequence number, and digest fields. Note that the ***digest*** is part of the AH header.
5. The ***Encapsulating Security Payload (ESP)*** protocol adds an ***ESP header***, ***ESP trailer***, and the ***digest***. The ESP header contains the security parameter index and the sequence number fields. The ESP trailer contains the padding, the padding length, and the next header fields. Note that the ***digest*** is a field separate from the header or trailer.
6. Either ***AH*** or ***ESP*** is needed for IP security. ESP, with greater functionality than AH, was developed after AH was already in use.
7. The two dominant protocols for providing security at the transport layer are the ***Secure Sockets Layer (SSL)*** Protocol and the ***Transport Layer Security (TLS)*** Protocol. The latter is actually an IETF version of the former.
8. The ***Internet Key Exchange (IKE)*** is a protocol designed to create both inbound and outbound security associations in SADBs. ***IKE*** is a complex protocol based on three other protocols: ***Oakley***, ***SKEME***, and ***ISAKMP***
9. A ***session*** between two systems is an association that can last for a long time; a ***connection*** can be established and broken several times during a session. Some of the security parameters are created during the session establishment and are in

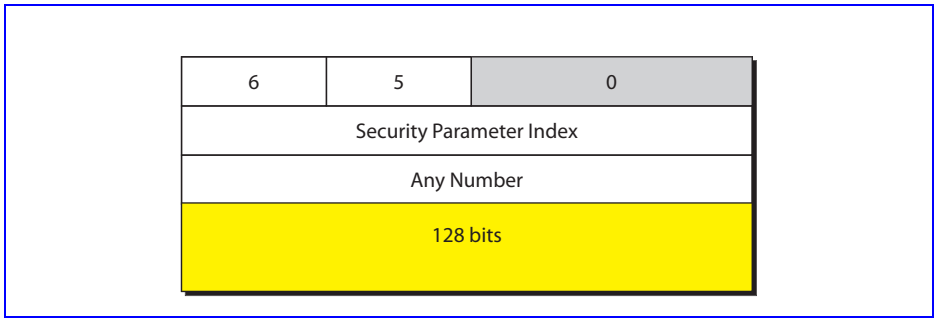
effect until the session is terminated. Some of the security parameters must be recreated (or occasionally resumed) for each connection.

10. ***SSL*** uses two protocols for this purpose: the ***Handshake Protocol*** and ***ChangeCipherSpec Protocol***.
11. One of the protocols designed to provide security for email is ***Pretty Good Privacy (PGP)***. ***PGP*** is designed to create authenticated and confidential e-mails.
12. In ***PGP***, the ***security parameters*** need to be sent with the message because e-mail is a one-time activity, in which the sender and receiver cannot agree on the security parameters to be used before sending the message.
13. The ***Handshake Protocol*** establishes a cipher set and provides keys and security parameters. It also authenticates the server to the client and the client to the server, if needed.
14. The ***Record Protocol*** carries messages from the upper layer. The message is fragmented and optionally compressed; a MAC is added to the compressed message by using the negotiated hash algorithm. The compressed fragment and the MAC are encrypted by using the negotiated encryption algorithm. Finally, the SSL header is added to the encrypted message.
15. A ***firewall*** is a security mechanism that stands between the global Internet and a network. A firewall selectively filters packets.
16. Two types of firewalls discussed in this chapter are ***packet-filter firewall*** and ***proxy-based firewall***.
17. A ***VPN*** is a technology that allows an organization to use the global Internet yet safely maintain private internal communication.
18. ***LANs*** on a fully private internet can communicate through ***routers*** and ***leased lines***.

Exercises

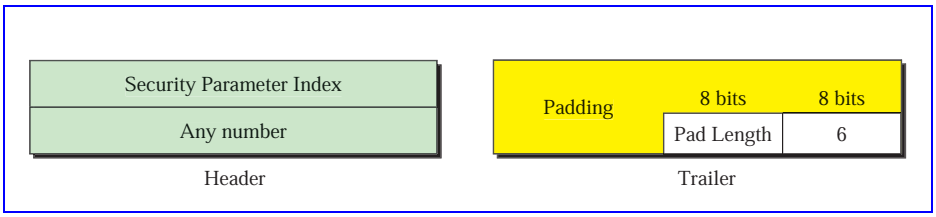
19. The only fields we can fill are the next header (assuming the packet encapsulates TCP) and the length field. The sequence number can be any number. Note that the length field defines the number of 32-bit words minus 2. See Figure 32.1.

Figure 32.1 *Solution to Exercise 19*



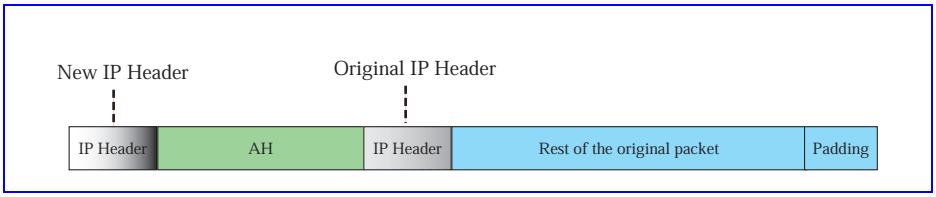
20. The only field we can fill is the next field assuming the packet carries a TCP segment. See Figure 32.2.

Figure 32.2 *Solution to Exercise 20*



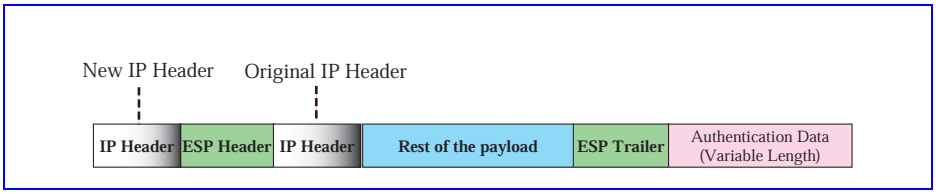
21. See Figure 32.3.

Figure 32.3 *Solution to Exercise 21*



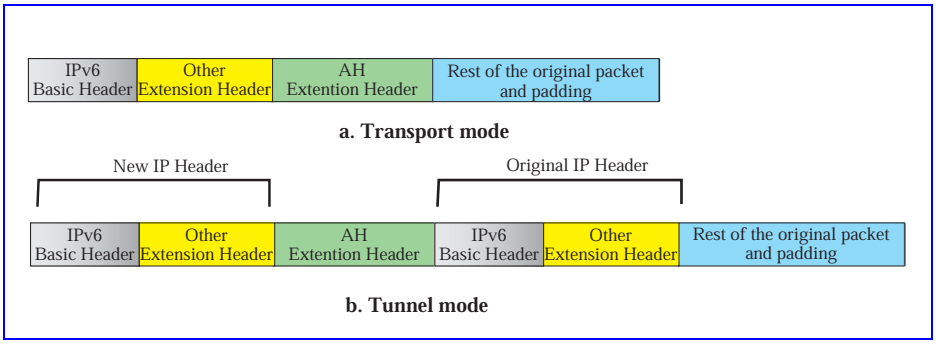
22. See Figure 32.4.

Figure 32.4 *Solution to Exercise 22*



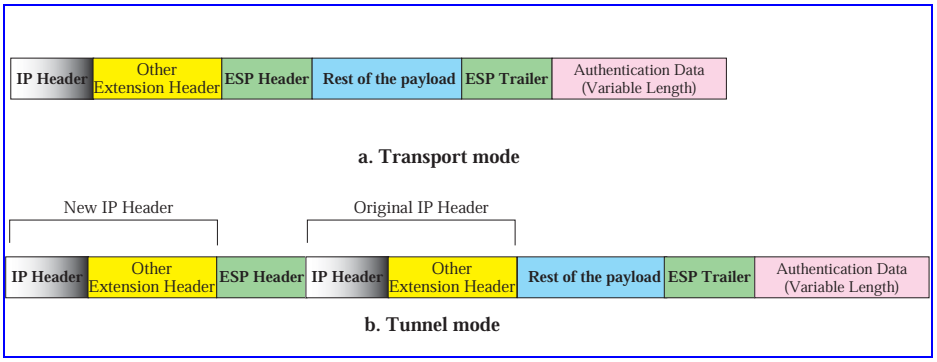
23. See Figure 32.5.

Figure 32.5 *Solution to Exercise 23*



24. See Figure 32.6.

Figure 32.6 *Solution to Exercise 24*



25. ***IPSec*** uses the services of IKE to create a security association that includes session keys. However, this does not start from scratch. Some kind of secret needs to exist between the two parties. In one of the methods used in IKE, the assumption is that there is a ***shared secret key*** between the two parties. In this case, a ***KDC*** can be used to create this shared secret key.
26. ***IPSec*** uses the services of IKE to create a security association that includes session keys. However, this does not start from scratch. Some kind of secret needs to exist between the two parties. In most methods used by IKE, the assumption is that there are some ***public keys*** established between the two parties. In this case, a ***CA*** can be used to create certified public keys.
27. Some ***SSL*** cipher suites need to use shared session keys. However, these session keys are created during hand-shaking. There is no need for a ***KDC***.
28. Some protocols used for key-exchange and authentication require that there should be ***established certified public keys*** between the two parties. An ***AC*** can be used for this purpose.
29. One of the purposes of ***PGP*** is to free the sender of the message from using a ***KDC***. In PGP, the session key is created and encrypted with the public key established between the sender and the receiver.
30. Although ***PGP*** needs to use certified public keys for its operation, it normally does not use the services of a ***CA***. The ***web of trust*** created between the group of people provides the public and private key rings.
31. ***IPSec*** uses IKE to create security parameters. IKE has defined several methods to do so. Each method uses a different set of ciphers to accomplish its task However, the list of ciphers for each method is pre-defined. Although the two parties can choose any of the methods during negotiation, the cipher used for that particular method is predefined. In other words, we can say that IPSec has a list of method suites, but not a cipher suite.
32. ***PGP*** creates security parameters for each message sent. Although the sender of the message can choose an encryption/decryption algorithm from the predefined list of these algorithms and the sender can choose an authentication algorithm from

another predefined list of algorithms, we cannot say that PGP is using a ciphersuite in the sense that SSL uses a cipher suite. In SSL, a suite defines a package that contains all the protocols involved; in PGP a sender can choose any protocol from either list and combine them.