



Homework Problems and Questions

Chapter 1 Review Questions

SECTION 1.1

- R1. What is the difference between a host and an end system? List several different types of end systems. Is a Web server an end system?
- R2. The word *protocol* is often used to describe diplomatic relations. How does Wikipedia describe diplomatic protocol?
- R3. Why are standards important for protocols?

SECTION 1.2

- R4. List six access technologies. Classify each one as home access, enterprise access, or wide-area wireless access.
- R5. Is HFC transmission rate dedicated or shared among users? Are collisions possible in a downstream HFC channel? Why or why not?
- R6. List the available residential access technologies in your city. For each type of access, provide the advertised downstream rate, upstream rate, and monthly price.
- R7. What is the transmission rate of Ethernet LANs?
- R8. What are some of the physical media that Ethernet can run over?
- R9. Dial-up modems, HFC, DSL and FTTH are all used for residential access. For each of these access technologies, provide a range of transmission rates and comment on whether the transmission rate is shared or dedicated.
- R10. Describe the most popular wireless Internet access technologies today. Compare and contrast them.

SECTION 1.3

- R11. Suppose there is exactly one packet switch between a sending host and a receiving host. The transmission rates between the sending host and the switch and between the switch and the receiving host are R_1 and R_2 , respectively. Assuming that the switch uses store-and-forward packet switching, what is the total end-to-end delay to send a packet of length L ? (Ignore queuing, propagation delay, and processing delay.)
- R12. What advantage does a circuit-switched network have over a packet-switched network? What advantages does TDM have over FDM in a circuit-switched network?
- R13. Suppose users share a 2 Mbps link. Also suppose each user transmits continuously at 1 Mbps when transmitting, but each user transmits only 20 percent of the time. (See the discussion of statistical multiplexing in Section 1.3.)

- a. When circuit switching is used, how many users can be supported?
- b. For the remainder of this problem, suppose packet switching is used. Why will there be essentially no queuing delay before the link if two or fewer users transmit at the same time? Why will there be a queuing delay if three users transmit at the same time?
- c. Find the probability that a given user is transmitting.
- d. Suppose now there are three users. Find the probability that at any given time, all three users are transmitting simultaneously. Find the fraction of time during which the queue grows.

R14. Why will two ISPs at the same level of the hierarchy often peer with each other? How does an IXP earn money?

R15. Some content providers have created their own networks. Describe Google's network. What motivates content providers to create these networks?

SECTION 1.4

R16. Consider sending a packet from a source host to a destination host over a fixed route. List the delay components in the end-to-end delay. Which of these delays are constant and which are variable?

R17. Visit the Transmission Versus Propagation Delay applet at the companion Web site. Among the rates, propagation delay, and packet sizes available, find a combination for which the sender finishes transmitting before the first bit of the packet reaches the receiver. Find another combination for which the first bit of the packet reaches the receiver before the sender finishes transmitting.

R18. How long does it take a packet of length 1,000 bytes to propagate over a link of distance 2,500 km, propagation speed $2.5 \cdot 10^8$ m/s, and transmission rate 2 Mbps? More generally, how long does it take a packet of length L to propagate over a link of distance d , propagation speed s , and transmission rate R bps? Does this delay depend on packet length? Does this delay depend on transmission rate?

R19. Suppose Host A wants to send a large file to Host B. The path from Host A to Host B has three links, of rates $R_1 = 500$ kbps, $R_2 = 2$ Mbps, and $R_3 = 1$ Mbps.

- a. Assuming no other traffic in the network, what is the throughput for the file transfer?
- b. Suppose the file is 4 million bytes. Dividing the file size by the throughput, roughly how long will it take to transfer the file to Host B?
- c. Repeat (a) and (b), but now with R_2 reduced to 100 kbps.

R20. Suppose end system A wants to send a large file to end system B. At a very high level, describe how end system A creates packets from the file. When

one of these packets arrives to a packet switch, what information in the packet does the switch use to determine the link onto which the packet is forwarded? Why is packet switching in the Internet analogous to driving from one city to another and asking directions along the way?

- R21. Visit the Queuing and Loss applet at the companion Web site. What is the maximum emission rate and the minimum transmission rate? With those rates, what is the traffic intensity? Run the applet with these rates and determine how long it takes for packet loss to occur. Then repeat the experiment a second time and determine again how long it takes for packet loss to occur. Are the values different? Why or why not?

SECTION 1.5

- R22. List five tasks that a layer can perform. Is it possible that one (or more) of these tasks could be performed by two (or more) layers?
- R23. What are the five layers in the Internet protocol stack? What are the principal responsibilities of each of these layers?
- R24. What is an application-layer message? A transport-layer segment? A network-layer datagram? A link-layer frame?
- R25. Which layers in the Internet protocol stack does a router process? Which layers does a link-layer switch process? Which layers does a host process?

SECTION 1.6

- R26. What is the difference between a virus and a worm?
- R27. Describe how a botnet can be created, and how it can be used for a DDoS attack.
- R28. Suppose Alice and Bob are sending packets to each other over a computer network. Suppose Trudy positions herself in the network so that she can capture all the packets sent by Alice and send whatever she wants to Bob; she can also capture all the packets sent by Bob and send whatever she wants to Alice. List some of the malicious things Trudy can do from this position.



Problems

- P1. Design and describe an application-level protocol to be used between an automatic teller machine and a bank's centralized computer. Your protocol should allow a user's card and password to be verified, the account balance (which is maintained at the centralized computer) to be queried, and an account withdrawal to be made (that is, money disbursed to the user). Your

protocol entities should be able to handle the all-too-common case in which there is not enough money in the account to cover the withdrawal. Specify your protocol by listing the messages exchanged and the action taken by the automatic teller machine or the bank's centralized computer on transmission and receipt of messages. Sketch the operation of your protocol for the case of a simple withdrawal with no errors, using a diagram similar to that in Figure 1.2. Explicitly state the assumptions made by your protocol about the underlying end-to-end transport service.

- P2. Equation 1.1 gives a formula for the end-to-end delay of sending one packet of length L over N links of transmission rate R . Generalize this formula for sending P such packets back-to-back over the N links.
- P3. Consider an application that transmits data at a steady rate (for example, the sender generates an N -bit unit of data every k time units, where k is small and fixed). Also, when such an application starts, it will continue running for a relatively long period of time. Answer the following questions, briefly justifying your answer:
- Would a packet-switched network or a circuit-switched network be more appropriate for this application? Why?
 - Suppose that a packet-switched network is used and the only traffic in this network comes from such applications as described above. Furthermore, assume that the sum of the application data rates is less than the capacities of each and every link. Is some form of congestion control needed? Why?
- P4. Consider the circuit-switched network in Figure 1.13. Recall that there are 4 circuits on each link. Label the four switches A, B, C and D, going in the clockwise direction.
- What is the maximum number of simultaneous connections that can be in progress at any one time in this network?
 - Suppose that all connections are between switches A and C. What is the maximum number of simultaneous connections that can be in progress?
 - Suppose we want to make four connections between switches A and C, and another four connections between switches B and D. Can we route these calls through the four links to accommodate all eight connections?
- P5. Review the car-caravan analogy in Section 1.4. Assume a propagation speed of 100 km/hour.
- Suppose the caravan travels 150 km, beginning in front of one tollbooth, passing through a second tollbooth, and finishing just after a third tollbooth. What is the end-to-end delay?
 - Repeat (a), now assuming that there are eight cars in the caravan instead of ten.



VideoNote
Exploring propagation
delay and transmission
delay

- P6. This elementary problem begins to explore propagation delay and transmission delay, two central concepts in data networking. Consider two hosts, A and B, connected by a single link of rate R bps. Suppose that the two hosts are separated by m meters, and suppose the propagation speed along the link is s meters/sec. Host A is to send a packet of size L bits to Host B.
- Express the propagation delay, d_{prop} , in terms of m and s .
 - Determine the transmission time of the packet, d_{trans} , in terms of L and R .
 - Ignoring processing and queuing delays, obtain an expression for the end-to-end delay.
 - Suppose Host A begins to transmit the packet at time $t = 0$. At time $t = d_{\text{trans}}$, where is the last bit of the packet?
 - Suppose d_{prop} is greater than d_{trans} . At time $t = d_{\text{trans}}$, where is the first bit of the packet?
 - Suppose d_{prop} is less than d_{trans} . At time $t = d_{\text{trans}}$, where is the first bit of the packet?
 - Suppose $s = 2.5 \cdot 10^8$, $L = 120$ bits, and $R = 56$ kbps. Find the distance m so that d_{prop} equals d_{trans} .
- P7. In this problem, we consider sending real-time voice from Host A to Host B over a packet-switched network (VoIP). Host A converts analog voice to a digital 64 kbps bit stream on the fly. Host A then groups the bits into 56-byte packets. There is one link between Hosts A and B; its transmission rate is 2 Mbps and its propagation delay is 10 msec. As soon as Host A gathers a packet, it sends it to Host B. As soon as Host B receives an entire packet, it converts the packet's bits to an analog signal. How much time elapses from the time a bit is created (from the original analog signal at Host A) until the bit is decoded (as part of the analog signal at Host B)?
- P8. Suppose users share a 3 Mbps link. Also suppose each user requires 150 kbps when transmitting, but each user transmits only 10 percent of the time. (See the discussion of packet switching versus circuit switching in Section 1.3.)
- When circuit switching is used, how many users can be supported?
 - For the remainder of this problem, suppose packet switching is used. Find the probability that a given user is transmitting.
 - Suppose there are 120 users. Find the probability that at any given time, exactly n users are transmitting simultaneously. (*Hint*: Use the binomial distribution.)
 - Find the probability that there are 21 or more users transmitting simultaneously.

- P9. Consider the discussion in Section 1.3 of packet switching versus circuit switching in which an example is provided with a 1 Mbps link. Users are generating data at a rate of 100 kbps when busy, but are busy generating data only with probability $p = 0.1$. Suppose that the 1 Mbps link is replaced by a 1 Gbps link.
- What is N , the maximum number of users that can be supported simultaneously under circuit switching?
 - Now consider packet switching and a user population of M users. Give a formula (in terms of p , M , N) for the probability that more than N users are sending data.
- P10. Consider a packet of length L which begins at end system A and travels over three links to a destination end system. These three links are connected by two packet switches. Let d_i , s_i , and R_i denote the length, propagation speed, and the transmission rate of link i , for $i = 1, 2, 3$. The packet switch delays each packet by d_{proc} . Assuming no queuing delays, in terms of d_i , s_i , R_i ($i = 1, 2, 3$), and L , what is the total end-to-end delay for the packet? Suppose now the packet is 1,500 bytes, the propagation speed on all three links is $2.5 \cdot 10^8$ m/s, the transmission rates of all three links are 2 Mbps, the packet switch processing delay is 3 msec, the length of the first link is 5,000 km, the length of the second link is 4,000 km, and the length of the last link is 1,000 km. For these values, what is the end-to-end delay?
- P11. In the above problem, suppose $R_1 = R_2 = R_3 = R$ and $d_{proc} = 0$. Further suppose the packet switch does not store-and-forward packets but instead immediately transmits each bit it receives before waiting for the entire packet to arrive. What is the end-to-end delay?
- P12. A packet switch receives a packet and determines the outbound link to which the packet should be forwarded. When the packet arrives, one other packet is halfway done being transmitted on this outbound link and four other packets are waiting to be transmitted. Packets are transmitted in order of arrival. Suppose all packets are 1,500 bytes and the link rate is 2 Mbps. What is the queuing delay for the packet? More generally, what is the queuing delay when all packets have length L , the transmission rate is R , x bits of the currently-being-transmitted packet have been transmitted, and n packets are already in the queue?
- P13. (a) Suppose N packets arrive simultaneously to a link at which no packets are currently being transmitted or queued. Each packet is of length L and the link has transmission rate R . What is the average queuing delay for the N packets?
- (b) Now suppose that N such packets arrive to the link every LN/R seconds. What is the average queuing delay of a packet?

- P14. Consider the queuing delay in a router buffer. Let I denote traffic intensity; that is, $I = \lambda a / R$. Suppose that the queuing delay takes the form $IL/R (1 - I)$ for $I < 1$.
- Provide a formula for the total delay, that is, the queuing delay plus the transmission delay.
 - Plot the total delay as a function of L/R .
- P15. Let a denote the rate of packets arriving at a link in packets/sec, and let μ denote the link's transmission rate in packets/sec. Based on the formula for the total delay (i.e., the queuing delay plus the transmission delay) derived in the previous problem, derive a formula for the total delay in terms of a and μ .
- P16. Consider a router buffer preceding an outbound link. In this problem, you will use Little's formula, a famous formula from queuing theory. Let N denote the average number of packets in the buffer plus the packet being transmitted. Let a denote the rate of packets arriving at the link. Let d denote the average total delay (i.e., the queuing delay plus the transmission delay) experienced by a packet. Little's formula is $N = a \cdot d$. Suppose that on average, the buffer contains 10 packets, and the average packet queuing delay is 10 msec. The link's transmission rate is 100 packets/sec. Using Little's formula, what is the average packet arrival rate, assuming there is no packet loss?
- P17. a. Generalize Equation 1.2 in Section 1.4.3 for heterogeneous processing rates, transmission rates, and propagation delays.
- Repeat (a), but now also suppose that there is an average queuing delay of d_{queue} at each node.
- P18. Perform a Traceroute between source and destination on the same continent at three different hours of the day.
- Find the average and standard deviation of the round-trip delays at each of the three hours.
 - Find the number of routers in the path at each of the three hours. Did the paths change during any of the hours?
 - Try to identify the number of ISP networks that the Traceroute packets pass through from source to destination. Routers with similar names and/or similar IP addresses should be considered as part of the same ISP. In your experiments, do the largest delays occur at the peering interfaces between adjacent ISPs?
 - Repeat the above for a source and destination on different continents. Compare the intra-continent and inter-continent results.
- P19. (a) Visit the site www.traceroute.org and perform traceroutes from two different cities in France to the same destination host in the United States. How many links are the same in the two traceroutes? Is the transatlantic link the same?



VideoNote
Using Traceroute to
discover network
paths and measure
network delay

- (b) Repeat (a) but this time choose one city in France and another city in Germany.
- (c) Pick a city in the United States, and perform traceroutes to two hosts, each in a different city in China. How many links are common in the two traceroutes? Do the two traceroutes diverge before reaching China?
- P20. Consider the throughput example corresponding to Figure 1.20(b). Now suppose that there are M client-server pairs rather than 10. Denote R_s , R_c , and R for the rates of the server links, client links, and network link. Assume all other links have abundant capacity and that there is no other traffic in the network besides the traffic generated by the M client-server pairs. Derive a general expression for throughput in terms of R_s , R_c , R , and M .
- P21. Consider Figure 1.19(b). Now suppose that there are M paths between the server and the client. No two paths share any link. Path k ($k = 1, \dots, M$) consists of N links with transmission rates $R_1^k, R_2^k, \dots, R_N^k$. If the server can only use one path to send data to the client, what is the maximum throughput that the server can achieve? If the server can use all M paths to send data, what is the maximum throughput that the server can achieve?
- P22. Consider Figure 1.19(b). Suppose that each link between the server and the client has a packet loss probability p , and the packet loss probabilities for these links are independent. What is the probability that a packet (sent by the server) is successfully received by the receiver? If a packet is lost in the path from the server to the client, then the server will re-transmit the packet. On average, how many times will the server re-transmit the packet in order for the client to successfully receive the packet?
- P23. Consider Figure 1.19(a). Assume that we know the bottleneck link along the path from the server to the client is the first link with rate R_s bits/sec. Suppose we send a pair of packets back to back from the server to the client, and there is no other traffic on this path. Assume each packet of size L bits, and both links have the same propagation delay d_{prop} .
- What is the packet inter-arrival time at the destination? That is, how much time elapses from when the last bit of the first packet arrives until the last bit of the second packet arrives?
 - Now assume that the second link is the bottleneck link (i.e., $R_c < R_s$). Is it possible that the second packet queues at the input queue of the second link? Explain. Now suppose that the server sends the second packet T seconds after sending the first packet. How large must T be to ensure no queuing before the second link? Explain.
- P24. Suppose you would like to urgently deliver 40 terabytes data from Boston to Los Angeles. You have available a 100 Mbps dedicated link for data transfer. Would you prefer to transmit the data via this link or instead use FedEx overnight delivery? Explain.

- P25. Suppose two hosts, A and B, are separated by 20,000 kilometers and are connected by a direct link of $R = 2$ Mbps. Suppose the propagation speed over the link is $2.5 \cdot 10^8$ meters/sec.
- Calculate the bandwidth-delay product, $R \cdot d_{\text{prop}}$.
 - Consider sending a file of 800,000 bits from Host A to Host B. Suppose the file is sent continuously as one large message. What is the maximum number of bits that will be in the link at any given time?
 - Provide an interpretation of the bandwidth-delay product.
 - What is the width (in meters) of a bit in the link? Is it longer than a football field?
 - Derive a general expression for the width of a bit in terms of the propagation speed s , the transmission rate R , and the length of the link m .
- P26. Referring to problem P25, suppose we can modify R . For what value of R is the width of a bit as long as the length of the link?
- P27. Consider problem P25 but now with a link of $R = 1$ Gbps.
- Calculate the bandwidth-delay product, $R \cdot d_{\text{prop}}$.
 - Consider sending a file of 800,000 bits from Host A to Host B. Suppose the file is sent continuously as one big message. What is the maximum number of bits that will be in the link at any given time?
 - What is the width (in meters) of a bit in the link?
- P28. Refer again to problem P25.
- How long does it take to send the file, assuming it is sent continuously?
 - Suppose now the file is broken up into 20 packets with each packet containing 40,000 bits. Suppose that each packet is acknowledged by the receiver and the transmission time of an acknowledgment packet is negligible. Finally, assume that the sender cannot send a packet until the preceding one is acknowledged. How long does it take to send the file?
 - Compare the results from (a) and (b).
- P29. Suppose there is a 10 Mbps microwave link between a geostationary satellite and its base station on Earth. Every minute the satellite takes a digital photo and sends it to the base station. Assume a propagation speed of $2.4 \cdot 10^8$ meters/sec.
- What is the propagation delay of the link?
 - What is the bandwidth-delay product, $R \cdot d_{\text{prop}}$?
 - Let x denote the size of the photo. What is the minimum value of x for the microwave link to be continuously transmitting?
- P30. Consider the airline travel analogy in our discussion of layering in Section 1.5, and the addition of headers to protocol data units as they flow down

the protocol stack. Is there an equivalent notion of header information that is added to passengers and baggage as they move down the airline protocol stack?

- P31. In modern packet-switched networks, including the Internet, the source host segments long, application-layer messages (for example, an image or a music file) into smaller packets and sends the packets into the network. The receiver then reassembles the packets back into the original message. We refer to this process as *message segmentation*. Figure 1.27 illustrates the end-to-end transport of a message with and without message segmentation. Consider a message that is $8 \cdot 10^6$ bits long that is to be sent from source to destination in Figure 1.27. Suppose each link in the figure is 2 Mbps. Ignore propagation, queuing, and processing delays.
- Consider sending the message from source to destination *without* message segmentation. How long does it take to move the message from the source host to the first packet switch? Keeping in mind that each switch uses store-and-forward packet switching, what is the total time to move the message from source host to destination host?
 - Now suppose that the message is segmented into 800 packets, with each packet being 10,000 bits long. How long does it take to move the first packet from source host to the first switch? When the first packet is being sent from the first switch to the second switch, the second packet is being sent from the source host to the first switch. At what time will the second packet be fully received at the first switch?
 - How long does it take to move the file from source host to destination host when message segmentation is used? Compare this result with your answer in part (a) and comment.

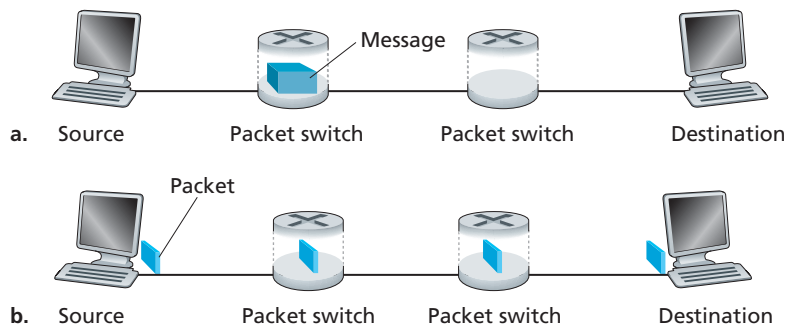


Figure 1.27 ♦ End-to-end message transport: (a) without message segmentation; (b) with message segmentation

- d. In addition to reducing delay, what are reasons to use message segmentation?
 - e. Discuss the drawbacks of message segmentation.
- P32. Experiment with the Message Segmentation applet at the book's Web site. Do the delays in the applet correspond to the delays in the previous problem? How do link propagation delays affect the overall end-to-end delay for packet switching (with message segmentation) and for message switching?
- P33. Consider sending a large file of F bits from Host A to Host B. There are three links (and two switches) between A and B, and the links are uncongested (that is, no queuing delays). Host A segments the file into segments of S bits each and adds 80 bits of header to each segment, forming packets of $L = 80 + S$ bits. Each link has a transmission rate of R bps. Find the value of S that minimizes the delay of moving the file from Host A to Host B. Disregard propagation delay.
- P34. Skype offers a service that allows you to make a phone call from a PC to an ordinary phone. This means that the voice call must pass through both the Internet and through a telephone network. Discuss how this might be done.



Wireshark Lab

“Tell me and I forget. Show me and I remember. Involve me and I understand.”
Chinese proverb

One's understanding of network protocols can often be greatly deepened by seeing them in action and by playing around with them—observing the sequence of messages exchanged between two protocol entities, delving into the details of protocol operation, causing protocols to perform certain actions, and observing these actions and their consequences. This can be done in simulated scenarios or in a real network environment such as the Internet. The Java applets at the textbook Web site take the first approach. In the Wireshark labs, we'll take the latter approach. You'll run network applications in various scenarios using a computer on your desk, at home, or in a lab. You'll observe the network protocols in your computer, interacting and exchanging messages with protocol entities executing elsewhere in the Internet. Thus, you and your computer will be an integral part of these live labs. You'll observe—and you'll learn—by doing.

The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. As the name suggests, a packet sniffer passively copies (sniffs) messages being sent from and received by your computer; it also displays the contents of the various protocol fields of these captured messages. A screenshot of the Wireshark packet sniffer is shown in Figure 1.28. Wireshark is a free packet sniffer that runs on Windows, Linux/Unix, and Mac

Chapter 1 Review Questions

1. There is no difference. Throughout this text, the words “host” and “end system” are used interchangeably. End systems include PCs, workstations, Web servers, mail servers, PDAs, Internet-connected game consoles, etc.
2. From Wikipedia: Diplomatic protocol is commonly described as a set of international courtesy rules. These well-established and time-honored rules have made it easier for nations and people to live and work together. Part of protocol has always been the acknowledgment of the hierarchical standing of all present. Protocol rules are based on the principles of civility.
3. Standards are important for protocols so that people can create networking systems and products that interoperate.
4. 1. Dial-up modem over telephone line: home; 2. DSL over telephone line: home or small office; 3. Cable to HFC: home; 4. 100 Mbps switched Ethernet: enterprise; 5. Wifi (802.11): home and enterprise; 6. 3G and 4G: wide-area wireless.
5. HFC bandwidth is shared among the users. On the downstream channel, all packets emanate from a single source, namely, the head end. Thus, there are no collisions in the downstream channel.
6. In most American cities, the current possibilities include: dial-up; DSL; cable modem; fiber-to-the-home.
7. Ethernet LANs have transmission rates of 10 Mbps, 100 Mbps, 1 Gbps and 10 Gbps.
8. Today, Ethernet most commonly runs over twisted-pair copper wire. It also can run over fibers optic links.
9. Dial up modems: up to 56 Kbps, bandwidth is dedicated; ADSL: up to 24 Mbps downstream and 2.5 Mbps upstream, bandwidth is dedicated; HFC, rates up to 42.8 Mbps and upstream rates of up to 30.7 Mbps, bandwidth is shared. FTTH: 2-10Mbps upload; 10-20 Mbps download; bandwidth is not shared.
10. There are two popular wireless Internet access technologies today:
 - a) Wifi (802.11) In a wireless LAN, wireless users transmit/receive packets to/from an base station (i.e., wireless access point) within a radius of few tens of meters. The base station is typically connected to the wired Internet and thus serves to connect wireless users to the wired network.
 - b) 3G and 4G wide-area wireless access networks. In these systems, packets are transmitted over the same wireless infrastructure used for cellular telephony, with the base station thus being managed by a telecommunications provider. This provides wireless access to users within a radius of tens of kilometers of the base station.

11. At time t_0 the sending host begins to transmit. At time $t_1 = L/R_1$, the sending host completes transmission and the entire packet is received at the router (no propagation delay). Because the router has the entire packet at time t_1 , it can begin to transmit the packet to the receiving host at time t_1 . At time $t_2 = t_1 + L/R_2$, the router completes transmission and the entire packet is received at the receiving host (again, no propagation delay). Thus, the end-to-end delay is $L/R_1 + L/R_2$.
12. A circuit-switched network can guarantee a certain amount of end-to-end bandwidth for the duration of a call. Most packet-switched networks today (including the Internet) cannot make any end-to-end guarantees for bandwidth. FDM requires sophisticated analog hardware to shift signal into appropriate frequency bands.
13. a) 2 users can be supported because each user requires half of the link bandwidth.
 b) Since each user requires 1Mbps when transmitting, if two or fewer users transmit simultaneously, a maximum of 2Mbps will be required. Since the available bandwidth of the shared link is 2Mbps, there will be no queuing delay before the link. Whereas, if three users transmit simultaneously, the bandwidth required will be 3Mbps which is more than the available bandwidth of the shared link. In this case, there will be queuing delay before the link.
 c) Probability that a given user is transmitting = 0.2
 d) Probability that all three users are transmitting simultaneously = $\binom{3}{3} p^3 (1-p)^{3-3}$
 $= (0.2)^3 = 0.008$. Since the queue grows when all the users are transmitting, the fraction of time during which the queue grows (which is equal to the probability that all three users are transmitting simultaneously) is 0.008.
14. If the two ISPs do not peer with each other, then when they send traffic to each other they have to send the traffic through a provider ISP (intermediary), to which they have to pay for carrying the traffic. By peering with each other directly, the two ISPs can reduce their payments to their provider ISPs. An Internet Exchange Points (IXP) (typically in a standalone building with its own switches) is a meeting point where multiple ISPs can connect and/or peer together. An ISP earns its money by charging each of the the ISPs that connect to the IXP a relatively small fee, which may depend on the amount of traffic sent to or received from the IXP.
15. Google's private network connects together all its data centers, big and small. Traffic between the Google data centers passes over its private network rather than over the public Internet. Many of these data centers are located in, or close to, lower tier ISPs. Therefore, when Google delivers content to a user, it often can bypass higher tier ISPs. What motivates content providers to create these networks? First, the content provider has more control over the user experience, since it has to use few intermediary ISPs. Second, it can save money by sending less traffic into provider

networks. Third, if ISPs decide to charge more money to highly profitable content providers (in countries where net neutrality doesn't apply), the content providers can avoid these extra payments.

16. The delay components are processing delays, transmission delays, propagation delays, and queuing delays. All of these delays are fixed, except for the queuing delays, which are variable.
17. a) 1000 km, 1 Mbps, 100 bytes
b) 100 km, 1 Mbps, 100 bytes
18. 10msec; d/s; no; no
19. a) 500 kbps
b) 64 seconds
c) 100kbps; 320 seconds
20. End system A breaks the large file into chunks. It adds header to each chunk, thereby generating multiple packets from the file. The header in each packet includes the IP address of the destination (end system B). The packet switch uses the destination IP address in the packet to determine the outgoing link. Asking which road to take is analogous to a packet asking which outgoing link it should be forwarded on, given the packet's destination address.
21. The maximum emission rate is 500 packets/sec and the maximum transmission rate is 350 packets/sec. The corresponding traffic intensity is $500/350 = 1.43 > 1$. Loss will eventually occur for each experiment; but the time when loss first occurs will be different from one experiment to the next due to the randomness in the emission process.
22. Five generic tasks are error control, flow control, segmentation and reassembly, multiplexing, and connection setup. Yes, these tasks can be duplicated at different layers. For example, error control is often provided at more than one layer.
23. The five layers in the Internet protocol stack are – from top to bottom – the application layer, the transport layer, the network layer, the link layer, and the physical layer. The principal responsibilities are outlined in Section 1.5.1.
24. Application-layer message: data which an application wants to send and passed onto the transport layer; transport-layer segment: generated by the transport layer and encapsulates application-layer message with transport layer header; network-layer datagram: encapsulates transport-layer segment with a network-layer header; link-layer frame: encapsulates network-layer datagram with a link-layer header.
25. Routers process network, link and physical layers (layers 1 through 3). (This is a little bit of a white lie, as modern routers sometimes act as firewalls or caching

components, and process Transport layer as well.) Link layer switches process link and physical layers (layers 1 through 2). Hosts process all five layers.

26. a) Virus

Requires some form of human interaction to spread. Classic example: E-mail viruses.

b) Worms

No user replication needed. Worm in infected host scans IP addresses and port numbers, looking for vulnerable processes to infect.

27. Creation of a botnet requires an attacker to find vulnerability in some application or system (e.g. exploiting the buffer overflow vulnerability that might exist in an application). After finding the vulnerability, the attacker needs to scan for hosts that are vulnerable. The target is basically to compromise a series of systems by exploiting that particular vulnerability. Any system that is part of the botnet can automatically scan its environment and propagate by exploiting the vulnerability. An important property of such botnets is that the originator of the botnet can remotely control and issue commands to all the nodes in the botnet. Hence, it becomes possible for the attacker to issue a command to all the nodes, that target a single node (for example, all nodes in the botnet might be commanded by the attacker to send a TCP SYN message to the target, which might result in a TCP SYN flood attack at the target).

28. Trudy can pretend to be Bob to Alice (and vice-versa) and partially or completely modify the message(s) being sent from Bob to Alice. For example, she can easily change the phrase "Alice, I owe you \$1000" to "Alice, I owe you \$10,000". Furthermore, Trudy can even drop the packets that are being sent by Bob to Alice (and vice-versa), even if the packets from Bob to Alice are encrypted.

Chapter 1 Problems

Problem 1

There is no single right answer to this question. Many protocols would do the trick. Here's a simple answer below:

Messages from ATM machine to Server

Msg name	purpose
-----	-----
HELLO <userid>	Let server know that there is a card in the ATM machine
	ATM card transmits user ID to Server
PASSWD <passwd>	User enters PIN, which is sent to server
BALANCE	User requests balance
WITHDRAWAL <amount>	User asks to withdraw money

BYE user all done

Messages from Server to ATM machine (display)

Msg name	purpose
-----	-----
PASSWD	Ask user for PIN (password)
OK	last requested operation (PASSWD, WITHDRAWL) OK
ERR	last requested operation (PASSWD, WITHDRAWL) in ERROR
AMOUNT <amt>	sent in response to BALANCE request
BYE	user done, display welcome screen at ATM

Correct operation:

client		server
HELO (userid)	----->	(check if valid userid)
	<-----	PASSWD
PASSWD <passwd>	----->	(check password)
	<-----	OK (password is OK)
BALANCE	----->	
	<-----	AMOUNT <amt>
WITHDRAWL <amt>	----->	check if enough \$ to cover withdrawl
	<-----	OK
ATM dispenses \$		
BYE	----->	
	<-----	BYE

In situation when there's not enough money:

HELO (userid)	----->	(check if valid userid)
	<-----	PASSWD
PASSWD <passwd>	----->	(check password)
	<-----	OK (password is OK)
BALANCE	----->	
	<-----	AMOUNT <amt>
WITHDRAWL <amt>	----->	check if enough \$ to cover withdrawl
	<-----	ERR (not enough funds)
error msg displayed		
no \$ given out		
BYE	----->	
	<-----	BYE

Problem 2

At time $N*(L/R)$ the first packet has reached the destination, the second packet is stored in the last router, the third packet is stored in the next-to-last router, etc. At time $N*(L/R) + L/R$, the second packet has reached the destination, the third packet is stored in the last router, etc. Continuing with this logic, we see that at time $N*(L/R) + (P-1)*(L/R) = (N+P-1)*(L/R)$ all packets have reached the destination.

Problem 3

- a) A circuit-switched network would be well suited to the application, because the application involves long sessions with predictable smooth bandwidth requirements. Since the transmission rate is known and not bursty, bandwidth can be reserved for each application session without significant waste. In addition, the overhead costs of setting up and tearing down connections are amortized over the lengthy duration of a typical application session.
- b) In the worst case, all the applications simultaneously transmit over one or more network links. However, since each link has sufficient bandwidth to handle the sum of all of the applications' data rates, no congestion (very little queuing) will occur. Given such generous link capacities, the network does not need congestion control mechanisms.

Problem 4

- a) Between the switch in the upper left and the switch in the upper right we can have 4 connections. Similarly we can have four connections between each of the 3 other pairs of adjacent switches. Thus, this network can support up to 16 connections.
- b) We can 4 connections passing through the switch in the upper-right-hand corner and another 4 connections passing through the switch in the lower-left-hand corner, giving a total of 8 connections.
- c) Yes. For the connections between A and C, we route two connections through B and two connections through D. For the connections between B and D, we route two connections through A and two connections through C. In this manner, there are at most 4 connections passing through any link.

Problem 5

Tollbooths are 75 km apart, and the cars propagate at 100km/hr. A tollbooth services a car at a rate of one car every 12 seconds.

- a) There are ten cars. It takes 120 seconds, or 2 minutes, for the first tollbooth to service the 10 cars. Each of these cars has a propagation delay of 45 minutes (travel 75 km) before arriving at the second tollbooth. Thus, all the cars are lined up before the second tollbooth after 47 minutes. The whole process repeats itself for traveling between the second and third tollbooths. It also takes 2 minutes for the third tollbooth to service the 10 cars. Thus the total delay is 96 minutes.
- b) Delay between tollbooths is 8×12 seconds plus 45 minutes, i.e., 46 minutes and 36 seconds. The total delay is twice this amount plus 8×12 seconds, i.e., 94 minutes and 48 seconds.

Problem 6

- a) $d_{prop} = m / s$ seconds.
- b) $d_{trans} = L / R$ seconds.
- c) $d_{end-to-end} = (m / s + L / R)$ seconds.
- d) The bit is just leaving Host A.
- e) The first bit is in the link and has not reached Host B.
- f) The first bit has reached Host B.
- g) Want

$$m = \frac{L}{R} s = \frac{120}{56 \times 10^3} (2.5 \times 10^8) = 536 \text{ km.}$$

Problem 7

Consider the first bit in a packet. Before this bit can be transmitted, all of the bits in the packet must be generated. This requires

$$\frac{56 \cdot 8}{64 \times 10^3} \text{ sec} = 7 \text{ msec.}$$

The time required to transmit the packet is

$$\frac{56 \cdot 8}{2 \times 10^6} \text{ sec} = 224 \mu \text{ sec.}$$

Propagation delay = 10 msec.

The delay until decoding is

$$7\text{msec} + 224\mu\text{sec} + 10\text{msec} = 17.224\text{msec}$$

A similar analysis shows that all bits experience a delay of 17.224 msec.

Problem 8

a) 20 users can be supported.

b) $p = 0.1$.

c) $\binom{120}{n} p^n (1-p)^{120-n}$.

d) $1 - \sum_{n=0}^{20} \binom{120}{n} p^n (1-p)^{120-n}$.

We use the central limit theorem to approximate this probability. Let X_j be independent random variables such that $P(X_j = 1) = p$.

$$P(\text{"21 or more users"}) = 1 - P\left(\sum_{j=1}^{120} X_j \leq 21\right)$$

$$\begin{aligned} P\left(\sum_{j=1}^{120} X_j \leq 21\right) &= P\left(\frac{\sum_{j=1}^{120} X_j - 12}{\sqrt{120 \cdot 0.1 \cdot 0.9}} \leq \frac{9}{\sqrt{120 \cdot 0.1 \cdot 0.9}}\right) \\ &\approx P\left(Z \leq \frac{9}{3.286}\right) = P(Z \leq 2.74) \\ &= 0.997 \end{aligned}$$

when Z is a standard normal r.v. Thus $P(\text{"21 or more users"}) \approx 0.003$.

Problem 9

a) 10,000

b) $\sum_{n=N+1}^M \binom{M}{n} p^n (1-p)^{M-n}$

Problem 10

The first end system requires L/R_I to transmit the packet onto the first link; the packet propagates over the first link in d_I/s_I ; the packet switch adds a processing delay of d_{proc} ; after receiving the entire packet, the packet switch connecting the first and the second

link requires L/R_2 to transmit the packet onto the second link; the packet propagates over the second link in d_2/s_2 . Similarly, we can find the delay caused by the second switch and the third link: L/R_3 , d_{proc} , and d_3/s_3 .

Adding these five delays gives

$$d_{end-end} = L/R_1 + L/R_2 + L/R_3 + d_1/s_1 + d_2/s_2 + d_3/s_3 + d_{proc} + d_{proc}$$

To answer the second question, we simply plug the values into the equation to get $6 + 6 + 6 + 20 + 16 + 4 + 3 + 3 = 64$ msec.

Problem 11

Because bits are immediately transmitted, the packet switch does not introduce any delay; in particular, it does not introduce a transmission delay. Thus,

$$d_{end-end} = L/R + d_1/s_1 + d_2/s_2 + d_3/s_3$$

For the values in Problem 10, we get $6 + 20 + 16 + 4 = 46$ msec.

Problem 12

The arriving packet must first wait for the link to transmit $4.5 * 1,500$ bytes = 6,750 bytes or 54,000 bits. Since these bits are transmitted at 2 Mbps, the queuing delay is 27 msec. Generally, the queuing delay is $(nL + (L - x))/R$.

Problem 13

- a) The queuing delay is 0 for the first transmitted packet, L/R for the second transmitted packet, and generally, $(n-1)L/R$ for the n^{th} transmitted packet. Thus, the average delay for the N packets is:

$$\begin{aligned} & (L/R + 2L/R + \dots + (N-1)L/R)/N \\ &= L/(RN) * (1 + 2 + \dots + (N-1)) \\ &= L/(RN) * N(N-1)/2 \\ &= LN(N-1)/(2RN) \\ &= (N-1)L/(2R) \end{aligned}$$

Note that here we used the well-known fact:

$$1 + 2 + \dots + N = N(N+1)/2$$

- b) It takes LN/R seconds to transmit the N packets. Thus, the buffer is empty when a each batch of N packets arrive. Thus, the average delay of a packet across all batches is the average delay within one batch, i.e., $(N-1)L/2R$.

Problem 14

- a) The transmission delay is L/R . The total delay is

$$\frac{IL}{R(1-I)} + \frac{L}{R} = \frac{L/R}{1-I}$$

- b) Let $x = L/R$.

$$\text{Total delay} = \frac{x}{1-ax}$$

For $x=0$, the total delay $=0$; as we increase x , total delay increases, approaching infinity as x approaches $1/a$.

Problem 15

$$\text{Total delay} = \frac{L/R}{1-I} = \frac{L/R}{1-aL/R} = \frac{1/\mu}{1-a/\mu} = \frac{1}{\mu-a}.$$

Problem 16

The total number of packets in the system includes those in the buffer and the packet that is being transmitted. So, $N=10+1$.

Because $N = a \cdot d$, so $(10+1)=a \cdot (\text{queuing delay} + \text{transmission delay})$. That is, $11=a \cdot (0.01+1/100)=a \cdot (0.01+0.01)$. Thus, $a=550$ packets/sec.

Problem 17

- a) There are Q nodes (the source host and the $Q-1$ routers). Let d_{proc}^q denote the processing delay at the q th node. Let R^q be the transmission rate of the q th link and let

$d_{trans}^q = L/R^q$. Let d_{prop}^q be the propagation delay across the q th link. Then

$$d_{end-to-end} = \sum_{q=1}^Q [d_{proc}^q + d_{trans}^q + d_{prop}^q].$$

- b) Let d_{queue}^q denote the average queuing delay at node q . Then

$$d_{end-to-end} = \sum_{q=1}^Q [d_{proc}^q + d_{trans}^q + d_{prop}^q + d_{queue}^q].$$

Problem 18

On linux you can use the command

```
traceroute www.targethost.com
```

and in the Windows command prompt you can use

```
tracert www.targethost.com
```

In either case, you will get three delay measurements. For those three measurements you can calculate the mean and standard deviation. Repeat the experiment at different times of the day and comment on any changes.

Here is an example solution:

```
traceroute to www.poly.edu (128.238.24.40), 30 hops max, 40 byte packets
 1 thunder.sdsc.edu (132.249.20.5)  2.802 ms  0.645 ms  0.484 ms
 2 dolphin.sdsc.edu (132.249.31.17)  0.227 ms  0.248 ms  0.239 ms
 3 dc-sdg-aggr1--sdsc-1.cenic.net (137.164.23.129)  0.360 ms  0.260 ms  0.240 ms
 4 dc-riv-core1--sdg-aggr1-10ge-2.cenic.net (137.164.47.14)  8.847 ms  8.497 ms  8.230 ms
 5 dc-lax-core1--lax-core2-10ge-2.cenic.net (137.164.46.64)  9.969 ms  9.920 ms  9.846 ms
 6 dc-lax-px1--lax-core1-10ge-2.cenic.net (137.164.46.151)  9.845 ms  9.729 ms  9.724 ms
 7 hurricane--lax-px1-ge.cenic.net (198.32.251.86)  9.971 ms  16.981 ms  9.850 ms
 8 10gigabitethernet4-3.core1.nyc4.he.net (72.52.92.225)  72.796 ms  80.278 ms  72.346 ms
 9 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218)  71.126 ms  71.442 ms  73.623 ms
10 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106)  70.924 ms  70.959 ms  71.072 ms
11 ae0.nycmnyzrj91.lighttower.net (72.22.160.156)  70.870 ms  71.089 ms  70.957 ms
12 72.22.188.102 (72.22.188.102)  71.242 ms  71.228 ms  71.102 ms
```

```
traceroute to www.poly.edu (128.238.24.40), 30 hops max, 40 byte packets
 1 thunder.sdsc.edu (132.249.20.5)  0.478 ms  0.353 ms  0.308 ms
 2 dolphin.sdsc.edu (132.249.31.17)  0.212 ms  0.251 ms  0.238 ms
 3 dc-sdg-aggr1--sdsc-1.cenic.net (137.164.23.129)  0.237 ms  0.246 ms  0.240 ms
 4 dc-riv-core1--sdg-aggr1-10ge-2.cenic.net (137.164.47.14)  8.628 ms  8.348 ms  8.357 ms
 5 dc-lax-core1--lax-core2-10ge-2.cenic.net (137.164.46.64)  9.934 ms  9.963 ms  9.852 ms
 6 dc-lax-px1--lax-core1-10ge-2.cenic.net (137.164.46.151)  9.831 ms  9.814 ms  9.676 ms
 7 hurricane--lax-px1-ge.cenic.net (198.32.251.86)  10.194 ms  10.012 ms  16.722 ms
 8 10gigabitethernet4-3.core1.nyc4.he.net (72.52.92.225)  73.856 ms  73.196 ms  73.979 ms
 9 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218)  71.247 ms  71.199 ms  71.646 ms
10 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106)  70.987 ms  71.073 ms  70.985 ms
11 ae0.nycmnyzrj91.lighttower.net (72.22.160.156)  71.075 ms  71.042 ms  71.328 ms
12 72.22.188.102 (72.22.188.102)  71.626 ms  71.299 ms  72.236 ms
```

```

1 thunder.sdsc.edu (132.249.20.5) 0.403 ms 0.347 ms 0.358 ms
2 dolphin.sdsc.edu (132.249.31.17) 0.225 ms 0.244 ms 0.237 ms
3 dc-sdg-aggr1--sdsc-1.cenic.net (137.164.23.129) 0.362 ms 0.256 ms 0.239 ms
4 dc-riv-core1--sdg-aggr1-10ge-2.cenic.net (137.164.47.14) 8.850 ms 8.358 ms 8.227 ms
5 dc-lax-core1--lax-core2-10ge-2.cenic.net (137.164.46.64) 10.096 ms 9.869 ms 10.351 ms
6 dc-lax-px1--lax-core1-10ge-2.cenic.net (137.164.46.151) 9.721 ms 9.621 ms 9.725 ms
7 hurricane--lax-px1-ge.cenic.net (198.32.251.86) 11.345 ms 10.048 ms 13.844 ms
8 10gigabitethernet4-3.core1.nyc4.he.net (72.52.92.225) 71.920 ms 72.977 ms 77.264 ms
9 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218) 71.273 ms 71.247 ms 71.291 ms
10 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106) 71.114 ms 82.516 ms 71.136 ms
11 ae0.nycmnyzrj91.lighttower.net (72.22.160.156) 71.232 ms 71.071 ms 71.039 ms
12 72.22.188.102 (72.22.188.102) 71.585 ms 71.608 ms 71.493 ms

```

Traceroutes between San Diego Super Computer Center and www.poly.edu

- The average (mean) of the round-trip delays at each of the three hours is 71.18 ms, 71.38 ms and 71.55 ms, respectively. The standard deviations are 0.075 ms, 0.21 ms, 0.05 ms, respectively.
- In this example, the traceroutes have 12 routers in the path at each of the three hours. No, the paths didn't change during any of the hours.
- Traceroute packets passed through four ISP networks from source to destination. Yes, in this experiment the largest delays occurred at peering interfaces between adjacent ISPs.

```

traceroute to www.poly.edu (128.238.24.40), 30 hops max, 60 byte packets
1 62-193-36-1.stella-net.net (62.193.36.1) 0.500 ms 0.415 ms 0.440 ms
2 62.193.33.29 (62.193.33.29) 0.910 ms 1.065 ms 1.026 ms
3 bg1.stella-net.net (62.193.32.254) 0.972 ms 1.026 ms 1.078 ms
4 62.193.32.66 (62.193.32.66) 1.021 ms 0.988 ms 0.947 ms
5 10gigabitethernet-2-2.par2.he.net (195.42.144.104) 1.537 ms 1.752 ms 1.714 ms
6 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93) 80.273 ms 80.103 ms 79.971 ms
7 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85) 86.494 ms 85.872 ms 86.223 ms
8 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218) 85.248 ms 85.424 ms 85.388 ms
9 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106) 86.194 ms 85.864 ms 86.116 ms
10 ae0.nycmnyzrj91.lighttower.net (72.22.160.156) 85.796 ms 85.823 ms 85.766 ms
11 72.22.188.102 (72.22.188.102) 87.717 ms 86.817 ms 86.774 ms

```



```

traceroute to www.poly.edu (128.238.24.40), 30 hops max, 60 byte packets
 1 62-193-36-1.stella-net.net (62.193.36.1)  0.375 ms  0.397 ms  0.355 ms
 2 62.193.33.29 (62.193.33.29)  0.810 ms  0.877 ms  0.836 ms
 3 bg1.stella-net.net (62.193.32.254)  1.098 ms  0.991 ms  1.055 ms
 4 62.193.32.66 (62.193.32.66)  0.994 ms  0.960 ms  1.157 ms
 5 10gigabitethernet-2-2.par2.he.net (195.42.144.104)  1.679 ms  1.816 ms  1.768 ms
 6 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93)  80.416 ms  90.573 ms  90.659 ms
 7 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85)  85.933 ms  95.987 ms  96.087 ms
 8 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218)  90.268 ms  90.229 ms  90.030 ms
 9 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106)  85.833 ms  85.448 ms  85.418 ms
10 ae0.nycmnyzrj91.lighttower.net (72.22.160.156)  87.067 ms  86.025 ms  85.962 ms
11 72.22.188.102 (72.22.188.102)  86.542 ms  86.369 ms  86.170 ms

```

```

traceroute to 128.238.24.40 (128.238.24.40), 30 hops max, 60 byte packets
 1 62-193-36-1.stella-net.net (62.193.36.1)  0.396 ms  0.284 ms  0.239 ms
 2 62.193.33.29 (62.193.33.29)  0.817 ms  0.786 ms  0.848 ms
 3 bg1.stella-net.net (62.193.32.254)  1.150 ms  1.216 ms  1.265 ms
 4 62.193.32.66 (62.193.32.66)  1.002 ms  0.963 ms  0.923 ms
 5 10gigabitethernet-2-2.par2.he.net (195.42.144.104)  1.573 ms  1.534 ms  1.643 ms
 6 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93)  88.738 ms  82.866 ms  82.783 ms
 7 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85)  94.888 ms  90.936 ms  90.877 ms
 8 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218)  90.498 ms  90.543 ms  90.482 ms
 9 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106)  85.716 ms  85.408 ms  85.637 ms
10 ae0.nycmnyzrj91.lighttower.net (72.22.160.156)  85.779 ms  85.290 ms  85.252 ms
11 72.22.188.102 (72.22.188.102)  86.217 ms  86.652 ms  86.588 ms

```

Traceroutes from www.stella-net.net (France) to www.poly.edu (USA).

- d) The average round-trip delays at each of the three hours are 87.09 ms, 86.35 ms and 86.48 ms, respectively. The standard deviations are 0.53 ms, 0.18 ms, 0.23 ms, respectively. In this example, there are 11 routers in the path at each of the three hours. No, the paths didn't change during any of the hours. Traceroute packets passed three ISP networks from source to destination. Yes, in this experiment the largest delays occurred at peering interfaces between adjacent ISPs.

Problem 19

An example solution:


```

traceroute to www.poly.edu (128.238.24.30), 30 hops max, 60 byte packets
 1 62-193-36-1.stella-net.net (62.193.36.1) 0.426 ms 0.329 ms 0.284 ms
 2 62.193.33.25 (62.193.33.25) 0.810 ms 0.771 ms 0.878 ms
 3 62.193.32.66 (62.193.32.66) 0.815 ms 0.840 ms 0.801 ms
 4 10gigabitethernet-2-2.par2.he.net (195.42.144.104) 1.387 ms 1.506 ms 1.467 ms
 5 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93) 85.402 ms 85.553 ms 85.353 ms
 6 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85) 94.360 ms 96.220 ms 96.355 ms
 7 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218) 90.279 ms 87.459 ms 87.709 ms
 8 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106) 85.474 ms 85.450 ms 85.983 ms
 9 ae0.nycmnyzrj91.lighttower.net (72.22.160.156) 86.160 ms 85.768 ms 86.016 ms
10 72.22.188.102 (72.22.188.102) 124.111 ms 89.340 ms 89.556 ms

```

```

 1 vl200.hs01.mar01.jaguar-network.net (85.31.192.253) 0.552 ms 0.414 ms
 2 ae1.cr01.mar01.jaguar-network.net (85.31.194.9) 0.340 ms 0.213 ms
 3 xe2-0-0.cr01.par02.jaguar-network.net (78.153.231.201) 9.933 ms 9.841 ms
 4 te1-3.er01.par02.jaguar-network.net (85.31.194.14) 9.828 ms 9.962 ms
 5 10gigabitethernet-2-2.par2.he.net (195.42.144.104) 10.456 ms 10.332 ms
 6 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93) 88.793 ms 96.781 ms
 7 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85) 94.651 ms 99.654 ms
 8 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218) 94.786 ms 94.755 ms
 9 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106) 91.935 ms 91.776 ms
10 ae0.nycmnyzrj91.lighttower.net (72.22.160.156) 91.909 ms 91.784 ms
11 72.22.188.102 (72.22.188.102) 93.791 ms 93.515 ms

```

Traceroutes from two different cities in France to New York City in United States

- a) In these traceroutes from two different cities in France to the same destination host in United States, seven links are in common including the transatlantic link.

```

traceroute to www.poly.edu (128.238.24.30), 30 hops max, 60 byte packets
 1 * * *
 2 hos-tr3.juniper2.rz10.hetzner.de 213.239.224.65 de 0.224 ms
   hos-tr2.juniper1.rz10.hetzner.de 213.239.224.33 de 0.174 ms 0.176 ms
 3 hos-bb1.juniper1.ffm.hetzner.de 213.239.240.224 de 4.746 ms 4.780 ms
   hos-bb1.juniper4.ffm.hetzner.de 213.239.240.230 de 4.823 ms
 4 20gigabitethernet4-3.core1.fra1.he.net 80.81.192.172 de 5.462 ms 5.461 ms 5.456 ms
 5 10gigabitethernet1-4.core1.ams1.he.net 72.52.92.94 us 12.899 ms
   10gigabitethernet5-3.core1.ams1.he.net 72.52.92.77 us 13.197 ms
   10gigabitethernet5-3.core1.lon1.he.net 184.105.213.145 us 26.110 ms
 6 10gigabitethernet1-4.core1.lon1.he.net 72.52.92.81 us 18.720 ms 18.871 ms 18.862 ms
 7 10gigabitethernet7-4.core1.nyc4.he.net 72.52.92.241 us 86.677 ms 85.580 ms 86.560 ms
 8 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net 216.66.50.106 us 118.500 ms
   10gigabitethernet3-4.core1.nyc5.he.net 184.105.213.218 us 90.346 ms
   lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net 216.66.50.106 us 118.500 ms
 9 ae0.nycmnyzrj91.lighttower.net 72.22.160.156 us 85.289 ms 85.552 ms 85.283 ms

```

```

traceroute to www.poly.edu (128.238.24.30), 30 hops max, 60 byte packets
 1 62-193-36-1.stella-net.net (62.193.36.1) 0.426 ms 0.329 ms 0.284 ms
 2 62.193.33.25 (62.193.33.25) 0.810 ms 0.771 ms 0.878 ms
 3 62.193.32.66 (62.193.32.66) 0.815 ms 0.840 ms 0.801 ms
 4 10gigabitethernet-2-2.par2.he.net (195.42.144.104) 1.387 ms 1.506 ms 1.467 ms
 5 10gigabitethernet7-1.core1.ash1.he.net (184.105.213.93) 85.402 ms 85.553 ms 85.353 ms
 6 10gigabitethernet1-2.core1.nyc4.he.net (72.52.92.85) 94.360 ms 96.220 ms 96.355 ms
 7 10gigabitethernet3-4.core1.nyc5.he.net (184.105.213.218) 90.279 ms 87.459 ms 87.709 ms
 8 lighttower-fiber-networks.10gigabitethernet3-2.core1.nyc5.he.net (216.66.50.106) 85.474 ms 85.450 ms 85.983 ms
 9 ae0.nycmnyzrj91.lighttower.net (72.22.160.156) 86.160 ms 85.768 ms 86.016 ms
10 72.22.188.102 (72.22.188.102) 124.111 ms 89.340 ms 89.556 ms

```

- b) In this example of traceroutes from one city in France and from another city in Germany to the same host in United States, three links are in common including the transatlantic link.

```

Tracing route to www.autoisp.shu.edu.cn [27.115.83.251]
over a maximum of 30 hops:
 1      9 ms      8 ms      10 ms  10.40.32.1
 2      12 ms     12 ms      9 ms  gig-3-0-4-nycmnyj-rtr1.nyc.rr.com [24.29.119.189]
 3      21 ms     20 ms     22 ms  tenge-0-6-0-0-nyquny91-rtr001.nyc.rr.com [24.29.100.122]
 4      19 ms     21 ms     22 ms  bun6-nyquny91-rtr002.nyc.rr.com [24.29.148.254]
 5      11 ms     11 ms     19 ms  ae-3-0-cr0.nyc20.tbone.rr.com [66.109.6.76]
 6      14 ms     18 ms     14 ms  ae-0-0-pr0.nyc30.tbone.rr.com [66.109.6.159]
 7      14 ms     11 ms     10 ms  xe-9-0-0.edge2.Newark1.Level3.net [4.59.20.29]
 8      12 ms     10 ms     13 ms  ae-31-51.ebr1.Newark1.Level3.net [4.69.156.30]
 9      10 ms     15 ms     13 ms  ae-2-2.ebr1.NewYork1.Level3.net [4.69.132.97]
10      11 ms     17 ms     14 ms  ae-81-81.csw3.NewYork1.Level3.net [4.69.134.74]
11      12 ms     14 ms     11 ms  ae-82-82.ebr2.NewYork1.Level3.net [4.69.148.41]
12      83 ms     83 ms     88 ms  ae-2-2.ebr4.SanJose1.Level3.net [4.69.135.185]
13      91 ms     87 ms     84 ms  ae-71-71.csw2.SanJose1.Level3.net [4.69.153.6]
14      83 ms     83 ms     88 ms  ae-2-70.edge3.SanJose1.Level3.net [4.69.152.82]
15     595 ms     593 ms     600 ms  CHINA-NETCO.edge3.SanJose1.Level3.net [4.79.54.6]
16     594 ms     591 ms     592 ms  219.158.96.213
17     539 ms     540 ms     540 ms  219.158.11.173
18     593 ms     586 ms     585 ms  219.158.19.93
19     585 ms     585 ms     584 ms  219.158.21.246
20     568 ms     587 ms     569 ms  112.64.243.62
21     570 ms     566 ms     568 ms  112.64.243.146
22     342 ms     341 ms     347 ms  112.65.183.106
23     574 ms     571 ms     573 ms  27.115.83.251

Trace complete.

```

```

Tracing route to www.lb.pku.edu.cn [162.105.131.113]
over a maximum of 30 hops:
 1      8 ms      8 ms      8 ms  10.40.32.1
 2      14 ms      9 ms     10 ms  gig-0-3-0-18-nycmnyj-rtr1.nyc.rr.com [24.168.138.85]
 3      21 ms     10 ms     11 ms  tenge-0-6-0-0-nyquny91-rtr001.nyc.rr.com [24.29.100.122]
 4      13 ms     22 ms     22 ms  bun6-nyquny91-rtr002.nyc.rr.com [24.29.148.254]
 5      11 ms     18 ms     12 ms  ae-3-0-cr0.nyc20.tbone.rr.com [66.109.6.76]
 6      43 ms     38 ms     41 ms  ae-8-0-cr0.chi10.tbone.rr.com [66.109.6.25]
 7      86 ms     88 ms     88 ms  ae-6-0-cr0.sjc30.tbone.rr.com [66.109.6.14]
 8      86 ms     89 ms     91 ms  ae-1-0-pr0.sjc10.tbone.rr.com [66.109.6.137]
 9      87 ms     86 ms     86 ms  66.109.10.210
10     257 ms     258 ms     258 ms  ge3-0-0.gw4.hkg3.asianetcom.net [61.14.157.250]
11     298 ms     296 ms     295 ms  CER-0002.gw4.hkg3.asianetcom.net [203.192.137.198]
12     297 ms     305 ms     305 ms  202.112.61.13
13     295 ms     296 ms     296 ms  202.112.61.157
14      *          *          *  Request timed out.
15     298 ms     302 ms     298 ms  202.112.41.178
16     308 ms     300 ms     300 ms  202.112.41.182

```

Traceroutes to two different cities in China from same host in United States

- c) Five links are common in the two traceroutes. The two traceroutes diverge before reaching China

Problem 20

$$\text{Throughput} = \min\{R_s, R_c, R/M\}$$

Problem 21

If only use one path, the max throughput is given by:

$$\max\{\min\{R_1^1, R_2^1, \dots, R_N^1\}, \min\{R_1^2, R_2^2, \dots, R_N^2\}, \dots, \min\{R_1^M, R_2^M, \dots, R_N^M\}\}.$$

If use all paths, the max throughput is given by $\sum_{k=1}^M \min\{R_1^k, R_2^k, \dots, R_N^k\}.$

Problem 22

Probability of successfully receiving a packet is: $p_s = (1-p)^N.$

The number of transmissions needed to be performed until the packet is successfully received by the client is a geometric random variable with success probability p_s . Thus, the average number of transmissions needed is given by: $1/p_s$. Then, the average number of re-transmissions needed is given by: $1/p_s - 1$.

Problem 23

Let's call the first packet A and call the second packet B.

- a) If the bottleneck link is the first link, then packet B is queued at the first link waiting for the transmission of packet A. So the packet inter-arrival time at the destination is simply L/R_s .
- b) If the second link is the bottleneck link and both packets are sent back to back, it must be true that the second packet arrives at the input queue of the second link before the second link finishes the transmission of the first packet. That is,

$$L/R_s + L/R_s + d_{prop} < L/R_s + d_{prop} + L/R_c$$

The left hand side of the above inequality represents the time needed by the second packet to *arrive at* the input queue of the second link (the second link has not started transmitting the second packet yet). The right hand side represents the time needed by the first packet to finish its transmission onto the second link.

If we send the second packet T seconds later, we will ensure that there is no queuing delay for the second packet at the second link if we have:

$$L/R_s + L/R_s + d_{prop} + T \geq L/R_s + d_{prop} + L/R_c$$

Thus, the minimum value of T is $L/R_c - L/R_s$.

Problem 24

40 terabytes = $40 * 10^{12} * 8$ bits. So, if using the dedicated link, it will take $40 * 10^{12} * 8 / (100 * 10^6) = 3200000$ seconds = 37 days. But with FedEx overnight delivery, you can guarantee the data arrives in one day, and it should cost less than \$100.

Problem 25

- a) 160,000 bits
- b) 160,000 bits
- c) The bandwidth-delay product of a link is the maximum number of bits that can be in the link.
- d) the width of a bit = length of link / bandwidth-delay product, so 1 bit is 125 meters long, which is longer than a football field
- e) s/R

Problem 26

$$s/R = 20000 \text{ km}, \text{ then } R = s/20000 \text{ km} = 2.5 * 10^8 / (2 * 10^7) = 12.5 \text{ bps}$$

Problem 27

- a) 80,000,000 bits
- b) 800,000 bits, this is because that the maximum number of bits that will be in the link at any given time = min(bandwidth delay product, packet size) = 800,000 bits.
- c) .25 meters

Problem 28

- a) $t_{trans} + t_{prop} = 400 \text{ msec} + 80 \text{ msec} = 480 \text{ msec}$.
- b) $20 * (t_{trans} + 2 t_{prop}) = 20 * (20 \text{ msec} + 80 \text{ msec}) = 2 \text{ sec}$.
- c) Breaking up a file takes longer to transmit because each data packet and its corresponding acknowledgement packet add their own propagation delays.

Problem 29

Recall geostationary satellite is 36,000 kilometers away from earth surface.

- a) 150 msec
- b) 1,500,000 bits
- c) 600,000,000 bits

Problem 30

Let's suppose the passenger and his/her bags correspond to the data unit arriving to the top of the protocol stack. When the passenger checks in, his/her bags are checked, and a tag is attached to the bags and ticket. This is additional information added in the Baggage layer in Figure 1.20 that allows the Baggage layer to implement the service of separating the passengers and baggage on the sending side, and then reuniting them (hopefully!) on the destination side. When a passenger then passes through security and additional stamp is often added to his/her ticket, indicating that the passenger has passed through a security check. This information is used to ensure (e.g., by later checks for the security information) secure transfer of people.

Problem 31

- a) Time to send message from source host to first packet switch = $\frac{8 \times 10^6}{2 \times 10^6} \text{ sec} = 4 \text{ sec}$

With store-and-forward switching, the total time to move message from source host to destination host = $4 \text{ sec} \times 3 \text{ hops} = 12 \text{ sec}$

- b) Time to send 1st packet from source host to first packet switch = $\frac{1 \times 10^4}{2 \times 10^6} \text{ sec} = 5 \text{ msec}$. Time at which 2nd packet is received at the first switch = time

at which 1st packet is received at the second switch = $2 \times 5 \text{ msec} = 10 \text{ msec}$

- c) Time at which 1st packet is received at the destination host = $5 \text{ msec} \times 3 \text{ hops} = 15 \text{ msec}$. After this, every 5msec one packet will be received; thus time at which last (800th) packet is received = $15 \text{ msec} + 799 * 5 \text{ msec} = 4.01 \text{ sec}$. It can be seen that delay in using message segmentation is significantly less (almost 1/3rd).

d)

- i. Without message segmentation, if bit errors are not tolerated, if there is a single bit error, the whole message has to be retransmitted (rather than a single packet).
- ii. Without message segmentation, huge packets (containing HD videos, for example) are sent into the network. Routers have to accommodate these huge packets. Smaller packets have to queue behind enormous packets and suffer unfair delays.

e)

- i. Packets have to be put in sequence at the destination.
- ii. Message segmentation results in many smaller packets. Since header size is usually the same for all packets regardless of their size, with message segmentation the total amount of header bytes is more.

Problem 32

Yes, the delays in the applet correspond to the delays in the Problem 31. The propagation delays affect the overall end-to-end delays both for packet switching and message switching equally.

Problem 33

There are F/S packets. Each packet is $S=80$ bits. Time at which the last packet is received at the first router is $\frac{S+80}{R} \times \frac{F}{S}$ sec. At this time, the first $F/S-2$ packets are at the destination, and the $F/S-1$ packet is at the second router. The last packet must then be transmitted by the first router and the second router, with each transmission taking $\frac{S+80}{R}$ sec. Thus delay in sending the whole file is $delay = \frac{S+80}{R} \times (\frac{F}{S} + 2)$

To calculate the value of S which leads to the minimum delay,

$$\frac{d}{dS} delay = 0 \Rightarrow S = \sqrt{40F}$$

Problem 34

The circuit-switched telephone networks and the Internet are connected together at "gateways". When a Skype user (connected to the Internet) calls an ordinary telephone, a circuit is established between a gateway and the telephone user over the circuit switched network. The skype user's voice is sent in packets over the Internet to the gateway. At the gateway, the voice signal is reconstructed and then sent over the circuit. In the other direction, the voice signal is sent over the circuit switched network to the gateway. The gateway packetizes the voice signal and sends the voice packets to the Skype user.

Equipped with knowledge about Internet application structure and application-level protocols, we're now ready to head further down the protocol stack and examine the transport layer in Chapter 3.



Homework Problems and Questions

Chapter 2 Review Questions

SECTION 2.1

- R1. List five nonproprietary Internet applications and the application-layer protocols that they use.
- R2. What is the difference between network architecture and application architecture?
- R3. For a communication session between a pair of processes, which process is the client and which is the server?
- R4. For a P2P file-sharing application, do you agree with the statement, “There is no notion of client and server sides of a communication session”? Why or why not?
- R5. What information is used by a process running on one host to identify a process running on another host?
- R6. Suppose you wanted to do a transaction from a remote client to a server as fast as possible. Would you use UDP or TCP? Why?
- R7. Referring to Figure 2.4, we see that none of the applications listed in Figure 2.4 requires both no data loss and timing. Can you conceive of an application that requires no data loss and that is also highly time-sensitive?
- R8. List the four broad classes of services that a transport protocol can provide. For each of the service classes, indicate if either UDP or TCP (or both) provides such a service.
- R9. Recall that TCP can be enhanced with SSL to provide process-to-process security services, including encryption. Does SSL operate at the transport layer or the application layer? If the application developer wants TCP to be enhanced with SSL, what does the developer have to do?

SECTIONS 2.2–2.5

- R10. What is meant by a handshaking protocol?
- R11. Why do HTTP, FTP, SMTP, and POP3 run on top of TCP rather than on UDP?
- R12. Consider an e-commerce site that wants to keep a purchase record for each of its customers. Describe how this can be done with cookies.

- R13. Describe how Web caching can reduce the delay in receiving a requested object. Will Web caching reduce the delay for all objects requested by a user or for only some of the objects? Why?
- R14. Telnet into a Web server and send a multiline request message. Include in the request message the `If-modified-since:` header line to force a response message with the `304 Not Modified` status code.
- R15. Why is it said that FTP sends control information “out-of-band”?
- R16. Suppose Alice, with a Web-based e-mail account (such as Hotmail or gmail), sends a message to Bob, who accesses his mail from his mail server using POP3. Discuss how the message gets from Alice’s host to Bob’s host. Be sure to list the series of application-layer protocols that are used to move the message between the two hosts.
- R17. Print out the header of an e-mail message you have recently received. How many `Received:` header lines are there? Analyze each of the header lines in the message.
- R18. From a user’s perspective, what is the difference between the download-and-delete mode and the download-and-keep mode in POP3?
- R19. Is it possible for an organization’s Web server and mail server to have exactly the same alias for a hostname (for example, `foo.com`)? What would be the type for the RR that contains the hostname of the mail server?
- R20. Look over your received emails, and examine the header of a message sent from a user with an `.edu` email address. Is it possible to determine from the header the IP address of the host from which the message was sent? Do the same for a message sent from a gmail account.

SECTION 2.6

- R21. In BitTorrent, suppose Alice provides chunks to Bob throughout a 30-second interval. Will Bob necessarily return the favor and provide chunks to Alice in this same interval? Why or why not?
- R22. Consider a new peer Alice that joins BitTorrent without possessing any chunks. Without any chunks, she cannot become a top-four uploader for any of the other peers, since she has nothing to upload. How then will Alice get her first chunk?
- R23. What is an overlay network? Does it include routers? What are the edges in the overlay network?
- R24. Consider a DHT with a mesh overlay topology (that is, every peer tracks all peers in the system). What are the advantages and disadvantages of such a design? What are the advantages and disadvantages of a circular DHT (with no shortcuts)?

- R25. List at least four different applications that are naturally suitable for P2P architectures. (*Hint*: File distribution and instant messaging are two.)

SECTION 2.7

- R26. In Section 2.7, the UDP server described needed only one socket, whereas the TCP server needed two sockets. Why? If the TCP server were to support n simultaneous connections, each from a different client host, how many sockets would the TCP server need?
- R27. For the client-server application over TCP described in Section 2.7, why must the server program be executed before the client program? For the client-server application over UDP, why may the client program be executed before the server program?



Problems

- P1. True or false?
- a. A user requests a Web page that consists of some text and three images. For this page, the client will send one request message and receive four response messages.
 - b. Two distinct Web pages (for example, `www.mit.edu/research.html` and `www.mit.edu/students.html`) can be sent over the same persistent connection.
 - c. With nonpersistent connections between browser and origin server, it is possible for a single TCP segment to carry two distinct HTTP request messages.
 - d. The `Date:` header in the HTTP response message indicates when the object in the response was last modified.
 - e. HTTP response messages never have an empty message body.
- P2. Read RFC 959 for FTP. List all of the client commands that are supported by the RFC.
- P3. Consider an HTTP client that wants to retrieve a Web document at a given URL. The IP address of the HTTP server is initially unknown. What transport and application-layer protocols besides HTTP are needed in this scenario?
- P4. Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters `<cr><lf>` are carriage return and line-feed characters (that is, the italicized character string `<cr>` in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.

```
GET /cs453/index.html HTTP/1.1<cr><lf>Host: gai
a.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8,image/png,*/*;q=0.5
<cr><lf>Accept-Language: en-us,en;q=0.5<cr><lf>Accept-
Encoding: zip,deflate<cr><lf>Accept-Charset: ISO
-8859-1,utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```

- a. What is the URL of the document requested by the browser?
 - b. What version of HTTP is the browser running?
 - c. Does the browser request a non-persistent or a persistent connection?
 - d. What is the IP address of the host on which the browser is running?
 - e. What type of browser initiates this message? Why is the browser type needed in an HTTP request message?
- P5. The text below shows the reply sent from the server in response to the HTTP GET message in the question above. Answer the following questions, indicating where in the message below you find the answer.

```
HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008
12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)
<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46
GMT<cr><lf>ETag: "526c3-f22-a88a4c80"<cr><lf>Accept-
Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>
Keep-Alive: timeout=max=100<cr><lf>Connection:
Keep-Alive<cr><lf>Content-Type: text/html; charset=
ISO-8859-1<cr><lf><cr><lf><!doctype html public "-
//w3c//dtd html 4.0 transitional//en"><lf><html><lf>
<head><lf> <meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1"><lf> <meta
name="GENERATOR" content="Mozilla/4.79 [en] (Windows NT
5.0; U) Netscape]"><lf> <title>CMPSCI 453 / 591 /
NTU-ST550A Spring 2005 homepage</title><lf></head><lf>
<much more document text following here (not shown)>
```

- a. Was the server able to successfully find the document or not? What time was the document reply provided?
- b. When was the document last modified?
- c. How many bytes are there in the document being returned?
- d. What are the first 5 bytes of the document being returned? Did the server agree to a persistent connection?

- P6. Obtain the HTTP/1.1 specification (RFC 2616). Answer the following questions:
- Explain the mechanism used for signaling between the client and server to indicate that a persistent connection is being closed. Can the client, the server, or both signal the close of a connection?
 - What encryption services are provided by HTTP?
 - Can a client open three or more simultaneous connections with a given server?
 - Either a server or a client may close a transport connection between them if either one detects the connection has been idle for some time. Is it possible that one side starts closing a connection while the other side is transmitting data via this connection? Explain.
- P7. Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that n DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT_1, \dots, RTT_n . Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the object. Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?
- P8. Referring to Problem P7, suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with
- Non-persistent HTTP with no parallel TCP connections?
 - Non-persistent HTTP with the browser configured for 5 parallel connections?
 - Persistent HTTP?
- P9. Consider Figure 2.12, for which there is an institutional network connected to the Internet. Suppose that the average object size is 850,000 bits and that the average request rate from the institution's browsers to the origin servers is 16 requests per second. Also suppose that the amount of time it takes from when the router on the Internet side of the access link forwards an HTTP request until it receives the response is three seconds on average (see Section 2.2.5). Model the total average response time as the sum of the average access delay (that is, the delay from Internet router to institution router) and the average Internet delay. For the average access delay, use $\Delta/(1 - \Delta\beta)$, where Δ is the average time required to send an object over the access link and β is the arrival rate of objects to the access link.
- Find the total average response time.
 - Now suppose a cache is installed in the institutional LAN. Suppose the miss rate is 0.4. Find the total response time.

- P10. Consider a short, 10-meter link, over which a sender can transmit at a rate of 150 bits/sec in both directions. Suppose that packets containing data are 100,000 bits long, and packets containing only control (e.g., ACK or handshaking) are 200 bits long. Assume that N parallel connections each get $1/N$ of the link bandwidth. Now consider the HTTP protocol, and suppose that each downloaded object is 100 Kbits long, and that the initial downloaded object contains 10 referenced objects from the same sender. Would parallel downloads via parallel instances of non-persistent HTTP make sense in this case? Now consider persistent HTTP. Do you expect significant gains over the non-persistent case? Justify and explain your answer.
- P11. Consider the scenario introduced in the previous problem. Now suppose that the link is shared by Bob with four other users. Bob uses parallel instances of non-persistent HTTP, and the other four users use non-persistent HTTP without parallel downloads.
- Do Bob's parallel connections help him get Web pages more quickly? Why or why not?
 - If all five users open five parallel instances of non-persistent HTTP, then would Bob's parallel connections still be beneficial? Why or why not?
- P12. Write a simple TCP program for a server that accepts lines of input from a client and prints the lines onto the server's standard output. (You can do this by modifying the `TCPServer.py` program in the text.) Compile and execute your program. On any other machine that contains a Web browser, set the proxy server in the browser to the host that is running your server program; also configure the port number appropriately. Your browser should now send its GET request messages to your server, and your server should display the messages on its standard output. Use this platform to determine whether your browser generates conditional GET messages for objects that are locally cached.
- P13. What is the difference between `MAIL FROM:` in SMTP and `From:` in the mail message itself?
- P14. How does SMTP mark the end of a message body? How about HTTP? Can HTTP use the same method as SMTP to mark the end of a message body? Explain.
- P15. Read RFC 5321 for SMTP. What does MTA stand for? Consider the following received spam email (modified from a real spam email). Assuming only the originator of this spam email is malicious and all other hosts are honest, identify the malicious host that has generated this spam email.

```
From - Fri Nov 07 13:41:30 2008
Return-Path: <tennis5@pp33head.com>
Received: from barmail.cs.umass.edu
(barmail.cs.umass.edu [128.119.240.3]) by cs.umass.edu
(8.13.1/8.12.6) for <hg@cs.umass.edu>; Fri, 7 Nov 2008
13:27:10 -0500
```

Received: from asusus-4b96 (localhost [127.0.0.1]) by
 barmail.cs.umass.edu (Spam Firewall) for
 <hg@cs.umass.edu>; Fri, 7 Nov 2008 13:27:07 -0500
 (EST)

Received: from asusus-4b96 ([58.88.21.177]) by
 barmail.cs.umass.edu for <hg@cs.umass.edu>; Fri,
 07 Nov 2008 13:27:07 -0500 (EST)

Received: from [58.88.21.177] by
 inbnd55.exchangeddd.com; Sat, 8 Nov 2008 01:27:07 +0700

From: "Jonny" <tennis5@pp33head.com>
 To: <hg@cs.umass.edu>
 Subject: How to secure your savings

P16. Read the POP3 RFC, RFC 1939. What is the purpose of the UIDL POP3 command?

P17. Consider accessing your e-mail with POP3.

- a. Suppose you have configured your POP mail client to operate in the download-and-delete mode. Complete the following transaction:

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: blah blah ...
S: .....blah
S: .
?
?
```

- b. Suppose you have configured your POP mail client to operate in the download-and-keep mode. Complete the following transaction:

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: blah blah ...
S: .....blah
S: .
?
?
```

- c. Suppose you have configured your POP mail client to operate in the download-and-keep mode. Using your transcript in part (b), suppose you retrieve messages 1 and 2, exit POP, and then five minutes later you again access POP to retrieve new e-mail. Suppose that in the five-minute interval no new messages have been sent to you. Provide a transcript of this second POP session.
- P18. a. What is a *whois* database?
- b. Use various whois databases on the Internet to obtain the names of two DNS servers. Indicate which whois databases you used.
 - c. Use nslookup on your local host to send DNS queries to three DNS servers: your local DNS server and the two DNS servers you found in part (b). Try querying for Type A, NS, and MX reports. Summarize your findings.
 - d. Use nslookup to find a Web server that has multiple IP addresses. Does the Web server of your institution (school or company) have multiple IP addresses?
 - e. Use the ARIN whois database to determine the IP address range used by your university.
 - f. Describe how an attacker can use whois databases and the nslookup tool to perform reconnaissance on an institution before launching an attack.
 - g. Discuss why whois databases should be publicly available.
- P19. In this problem, we use the useful *dig* tool available on Unix and Linux hosts to explore the hierarchy of DNS servers. Recall that in Figure 2.21, a DNS server higher in the DNS hierarchy delegates a DNS query to a DNS server lower in the hierarchy, by sending back to the DNS client the name of that lower-level DNS server. First read the man page for *dig*, and then answer the following questions.
- a. Starting with a root DNS server (from one of the root servers [a-m].root-servers.net), initiate a sequence of queries for the IP address for your department's Web server by using *dig*. Show the list of the names of DNS servers in the delegation chain in answering your query.
 - b. Repeat part a) for several popular Web sites, such as google.com, yahoo.com, or amazon.com.
- P20. Suppose you can access the caches in the local DNS servers of your department. Can you propose a way to roughly determine the Web servers (outside your department) that are most popular among the users in your department? Explain.
- P21. Suppose that your department has a local DNS server for all computers in the department. You are an ordinary user (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

- P22. Consider distributing a file of $F = 15$ Gbits to N peers. The server has an upload rate of $u_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps and an upload rate of u . For $N = 10, 100$, and $1,000$ and $u = 300$ Kbps, 700 Kbps, and 2 Mbps, prepare a chart giving the minimum distribution time for each of the combinations of N and u for both client-server distribution and P2P distribution.
- P23. Consider distributing a file of F bits to N peers using a client-server architecture. Assume a fluid model where the server can simultaneously transmit to multiple peers, transmitting to each peer at different rates, as long as the combined rate does not exceed u_s .
- Suppose that $u_s/N \leq d_{\min}$. Specify a distribution scheme that has a distribution time of NF/u_s .
 - Suppose that $u_s/N \geq d_{\min}$. Specify a distribution scheme that has a distribution time of F/d_{\min} .
 - Conclude that the minimum distribution time is in general given by $\max\{NF/u_s, F/d_{\min}\}$.
- P24. Consider distributing a file of F bits to N peers using a P2P architecture. Assume a fluid model. For simplicity assume that d_{\min} is very large, so that peer download bandwidth is never a bottleneck.
- Suppose that $u_s \leq (u_s + u_1 + \dots + u_N)/N$. Specify a distribution scheme that has a distribution time of F/u_s .
 - Suppose that $u_s \geq (u_s + u_1 + \dots + u_N)/N$. Specify a distribution scheme that has a distribution time of $NF/(u_s + u_1 + \dots + u_N)$.
 - Conclude that the minimum distribution time is in general given by $\max\{F/u_s, NF/(u_s + u_1 + \dots + u_N)\}$.
- P25. Consider an overlay network with N active peers, with each pair of peers having an active TCP connection. Additionally, suppose that the TCP connections pass through a total of M routers. How many nodes and edges are there in the corresponding overlay network?
- P26. Suppose Bob joins a BitTorrent torrent, but he does not want to upload any data to any other peers (so called free-riding).
- Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is Bob's claim possible? Why or why not?
 - Bob further claims that he can further make his "free-riding" more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?
- P27. In the circular DHT example in Section 2.6.2, suppose that peer 3 learns that peer 5 has left. How does peer 3 update its successor state information? Which peer is now its first successor? Its second successor?



- P28. In the circular DHT example in Section 2.6.2, suppose that a new peer 6 wants to join the DHT and peer 6 initially only knows peer 15's IP address. What steps are taken?
- P29. Because an integer in $[0, 2^n - 1]$ can be expressed as an n -bit binary number in a DHT, each key can be expressed as $k = (k_0, k_1, \dots, k_{n-1})$, and each peer identifier can be expressed $p = (p_0, p_1, \dots, p_{n-1})$. Let's now define the XOR distance between a key k and peer p as

$$d(k, p) = \sum_{j=0}^{n-1} |k_j - p_j| 2^j$$

Describe how this metric can be used to assign (key, value) pairs to peers. (To learn about how to build an efficient DHT using this natural metric, see [Maymounkov 2002] in which the Kademlia DHT is described.)

- P30. As DHTs are overlay networks, they may not necessarily match the underlay physical network well in the sense that two neighboring peers might be physically very far away; for example, one peer could be in Asia and its neighbor could be in North America. If we randomly and uniformly assign identifiers to newly joined peers, would this assignment scheme cause such a mismatch? Explain. And how would such a mismatch affect the DHT's performance?
- P31. Install and compile the Python programs TCPClient and UDPClient on one host and TCPServer and UDPServer on another host.
- Suppose you run TCPClient before you run TCPServer. What happens? Why?
 - Suppose you run UDPClient before you run UDPServer. What happens? Why?
 - What happens if you use different port numbers for the client and server sides?
- P32. Suppose that in UDPClient.py, after we create the socket, we add the line:

```
clientSocket.bind(('', 5432))
```

Will it become necessary to change UDPServer.py? What are the port numbers for the sockets in UDPClient and UDPServer? What were they before making this change?

- P33. Can you configure your browser to open multiple simultaneous connections to a Web site? What are the advantages and disadvantages of having a large number of simultaneous TCP connections?
- P34 We have seen that Internet TCP sockets treat the data being sent as a byte stream but UDP sockets recognize message boundaries. What are one

advantage and one disadvantage of byte-oriented API versus having the API explicitly recognize and preserve application-defined message boundaries?

- P35. What is the Apache Web server? How much does it cost? What functionality does it currently have? You may want to look at Wikipedia to answer this question.
- P36. Many BitTorrent clients use DHTs to create a distributed tracker. For these DHTs, what is the “key” and what is the “value”?



Socket Programming Assignments

The companion Web site includes six socket programming assignments. The first four assignments are summarized below. The fifth assignment makes use of the ICMP protocol and is summarized at the end of Chapter 4. The sixth assignment employs multimedia protocols and is summarized at the end of Chapter 7. It is highly recommended that students complete several, if not all, of these assignments. Students can find full details of these assignments, as well as important snippets of the Python code, at the Web site <http://www.awl.com/kurose-ross>.

Assignment 1: Web Server

In this assignment, you will develop a simple Web server in Python that is capable of processing only one request. Specifically, your Web server will (i) create a connection socket when contacted by a client (browser); (ii) receive the HTTP request from this connection; (iii) parse the request to determine the specific file being requested; (iv) get the requested file from the server’s file system; (v) create an HTTP response message consisting of the requested file preceded by header lines; and (vi) send the response over the TCP connection to the requesting browser. If a browser requests a file that is not present in your server, your server should return a “404 Not Found” error message.

In the companion Web site, we provide the skeleton code for your server. Your job is to complete the code, run your server, and then test your server by sending requests from browsers running on different hosts. If you run your server on a host that already has a Web server running on it, then you should use a different port than port 80 for your Web server.

Assignment 2: UDP Pinger

In this programming assignment, you will write a client ping program in Python. Your client will send a simple ping message to a server, receive a corresponding pong message back from the server, and determine the delay between when the client sent the ping message and received the pong message. This delay is called the Round Trip Time (RTT). The functionality provided by the client and server is

Chapter 2 Review Questions

1. The Web: HTTP; file transfer: FTP; remote login: Telnet; e-mail: SMTP; BitTorrent file sharing: BitTorrent protocol
2. Network architecture refers to the organization of the communication process into layers (e.g., the five-layer Internet architecture). Application architecture, on the other hand, is designed by an application developer and dictates the broad structure of the application (e.g., client-server or P2P).
3. The process which initiates the communication is the client; the process that waits to be contacted is the server.
4. No. In a P2P file-sharing application, the peer that is receiving a file is typically the client and the peer that is sending the file is typically the server.
5. The IP address of the destination host and the port number of the socket in the destination process.
6. You would use UDP. With UDP, the transaction can be completed in one roundtrip time (RTT) - the client sends the transaction request into a UDP socket, and the server sends the reply back to the client's UDP socket. With TCP, a minimum of two RTTs are needed - one to set-up the TCP connection, and another for the client to send the request, and for the server to send back the reply.
7. One such example is remote word processing, for example, with Google docs. However, because Google docs runs over the Internet (using TCP), timing guarantees are not provided.
8. a) Reliable data transfer
TCP provides a reliable byte-stream between client and server but UDP does not.

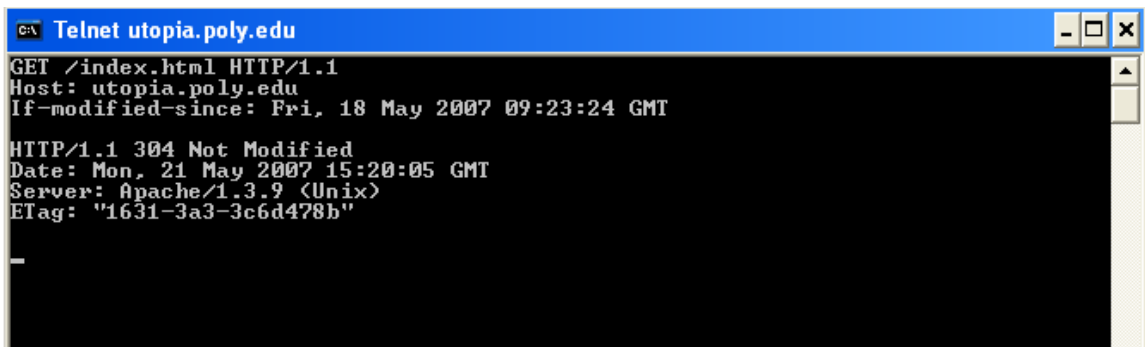
b) A guarantee that a certain value for throughput will be maintained
Neither

c) A guarantee that data will be delivered within a specified amount of time
Neither

d) Confidentiality (via encryption)
Neither
9. SSL operates at the application layer. The SSL socket takes unencrypted data from the application layer, encrypts it and then passes it to the TCP socket. If the application developer wants TCP to be enhanced with SSL, she has to include the SSL code in the application.

10. A protocol uses handshaking if the two communicating entities first exchange control packets before sending data to each other. SMTP uses handshaking at the application layer whereas HTTP does not.
11. The applications associated with those protocols require that all application data be received in the correct order and without gaps. TCP provides this service whereas UDP does not.
12. When the user first visits the site, the server creates a unique identification number, creates an entry in its back-end database, and returns this identification number as a cookie number. This cookie number is stored on the user's host and is managed by the browser. During each subsequent visit (and purchase), the browser sends the cookie number back to the site. Thus the site knows when this user (more precisely, this browser) is visiting the site.
13. Web caching can bring the desired content "closer" to the user, possibly to the same LAN to which the user's host is connected. Web caching can reduce the delay for all objects, even objects that are not cached, since caching reduces the traffic on links.
14. Telnet is not available in Windows 7 by default. to make it available, go to Control Panel, Programs and Features, Turn Windows Features On or Off, Check Telnet client. To start Telnet, in Windows command prompt, issue the following command
> telnet webserver 80

where "webserver" is some webserver. After issuing the command, you have established a TCP connection between your client telnet program and the web server. Then type in an HTTP GET message. An example is given below:



```
C:\> Telnet utopia.poly.edu
GET /index.html HTTP/1.1
Host: utopia.poly.edu
If-modified-since: Fri, 18 May 2007 09:23:24 GMT

HTTP/1.1 304 Not Modified
Date: Mon, 21 May 2007 15:20:05 GMT
Server: Apache/1.3.9 (Unix)
ETag: "1631-3a3-3c6d478b"
```

Since the index.html page in this web server was not modified since Fri, 18 May 2007 09:23:34 GMT, and the above commands were issued on Sat, 19 May 2007, the server returned "304 Not Modified". Note that the first 4 lines are the GET message and header lines inputted by the user, and the next 4 lines (starting from HTTP/1.1 304 Not Modified) is the response from the web server.

15. FTP uses two parallel TCP connections, one connection for sending control information (such as a request to transfer a file) and another connection for actually transferring the file. Because the control information is not sent over the same connection that the file is sent over, FTP sends control information out of band.
16. The message is first sent from Alice's host to her mail server over HTTP. Alice's mail server then sends the message to Bob's mail server over SMTP. Bob then transfers the message from his mail server to his host over POP3.
- 17.

Received: from 65.54.246.203 (EHLO bay0-omc3-s3.bay0.hotmail.com) (65.54.246.203) by mta419.mail.mud.yahoo.com with SMTP; Sat, 19 May 2007 16:53:51 -0700

Received: from hotmail.com ([65.55.135.106]) by bay0-omc3-s3.bay0.hotmail.com with Microsoft SMTPSVC(6.0.3790.2668); Sat, 19 May 2007 16:52:42 -0700

Received: from mail pickup service by hotmail.com with Microsoft SMTPSVC; Sat, 19 May 2007 16:52:41 -0700

Message-ID: <BAY130-F26D9E35BF59E0D18A819AFB9310@phx.gbl>

Received: from 65.55.135.123 by by130fd.bay130.hotmail.msn.com with HTTP; Sat, 19 May 2007 23:52:36 GMT

From: "prithula dhungel" <prithuladhungel@hotmail.com>

To: prithula@yahoo.com

Bcc:

Subject: Test mail

Date: Sat, 19 May 2007 23:52:36 +0000

Mime-Version: 1.0

Content-Type: Text/html; format=flowed

Return-Path: prithuladhungel@hotmail.com

Figure: A sample mail message header

Received: This header field indicates the sequence in which the SMTP servers send and receive the mail message including the respective timestamps.

In this example there are 4 "Received:" header lines. This means the mail message passed through 5 different SMTP servers before being delivered to the receiver's mail box. The last (forth) "Received:" header indicates the mail message flow from the SMTP server of the sender to the second SMTP server in the chain of servers. The sender's SMTP server is at address 65.55.135.123 and the second SMTP server in the chain is by130fd.bay130.hotmail.msn.com.

The third "Received:" header indicates the mail message flow from the second SMTP server in the chain to the third server, and so on.

Finally, the first "Received:" header indicates the flow of the mail messages from the forth SMTP server to the last SMTP server (i.e. the receiver's mail server) in the chain.

Message-id: The message has been given this number BAY130-F26D9E35BF59E0D18A819AFB9310@phx.gbl (by bay0-omc3-s3.bay0.hotmail.com. Message-id is a unique string assigned by the mail system when the message is first created.

From: This indicates the email address of the sender of the mail. In the given example, the sender is "prithuladhungel@hotmail.com"

To: This field indicates the email address of the receiver of the mail. In the example, the receiver is "prithula@yahoo.com"

Subject: This gives the subject of the mail (if any specified by the sender). In the example, the subject specified by the sender is "Test mail"

Date: The date and time when the mail was sent by the sender. In the example, the sender sent the mail on 19th May 2007, at time 23:52:36 GMT.

Mime-version: MIME version used for the mail. In the example, it is 1.0.

Content-type: The type of content in the body of the mail message. In the example, it is "text/html".

Return-Path: This specifies the email address to which the mail will be sent if the receiver of this mail wants to reply to the sender. This is also used by the sender's mail server for bouncing back undeliverable mail messages of mailer-daemon error messages. In the example, the return path is "prithuladhungel@hotmail.com".

18. With download and delete, after a user retrieves its messages from a POP server, the messages are deleted. This poses a problem for the nomadic user, who may want to access the messages from many different machines (office PC, home PC, etc.). In the download and keep configuration, messages are not deleted after the user retrieves the messages. This can also be inconvenient, as each time the user retrieves the stored messages from a new machine, all of non-deleted messages will be transferred to the new machine (including very old messages).
19. Yes an organization's mail server and Web server can have the same alias for a host name. The MX record is used to map the mail server's host name to its IP address.
20. You should be able to see the sender's IP address for a user with an .edu email address. But you will not be able to see the sender's IP address if the user uses a gmail account.
21. It is not necessary that Bob will also provide chunks to Alice. Alice has to be in the top 4 neighbors of Bob for Bob to send out chunks to her; this might not occur even if Alice provides chunks to Bob throughout a 30-second interval.

22. Recall that in BitTorrent, a peer picks a random peer and optimistically unchokes the peer for a short period of time. Therefore, Alice will eventually be optimistically unchoked by one of her neighbors, during which time she will receive chunks from that neighbor.
23. The overlay network in a P2P file sharing system consists of the nodes participating in the file sharing system and the logical links between the nodes. There is a logical link (an “edge” in graph theory terms) from node A to node B if there is a semi-permanent TCP connection between A and B. An overlay network does not include routers.
24. Mesh DHT: The advantage is in order to route a message to the peer (with ID) that is closest to the key, only one hop is required; the disadvantage is that each peer must track all other peers in the DHT. Circular DHT: the advantage is that each peer needs to track only a few other peers; the disadvantage is that $O(N)$ hops are needed to route a message to the peer that is closest to the key.
- 25.
- a) File Distribution
 - b) Instant Messaging
 - c) Video Streaming
 - d) Distributed Computing
26. With the UDP server, there is no welcoming socket, and all data from different clients enters the server through this one socket. With the TCP server, there is a welcoming socket, and each time a client initiates a connection to the server, a new socket is created. Thus, to support n simultaneous connections, the server would need $n+1$ sockets.
27. For the TCP application, as soon as the client is executed, it attempts to initiate a TCP connection with the server. If the TCP server is not running, then the client will fail to make a connection. For the UDP application, the client does not initiate connections (or attempt to communicate with the UDP server) immediately upon execution

Chapter 2 Problems

Problem 1

- a) F
- b) T
- c) F
- d) F
- e) F

Problem 2

Access control commands:

USER, PASS, ACT, CWD, CDUP, SMNT, REIN, QUIT.

Transfer parameter commands:

PORT, PASV, TYPE STRU, MODE.

Service commands:

RETR, STOR, STOU, APPE, ALLO, REST, RNFR, RNT0, ABOR, DELE, RMD, MRD, PWD, LIST, NLST, SITE, SYST, STAT, HELP, NOOP.

Problem 3

Application layer protocols: DNS and HTTP

Transport layer protocols: UDP for DNS; TCP for HTTP

Problem 4

- a) The document request was `http://gaia.cs.umass.edu/cs453/index.html`. The Host : field indicates the server's name and `/cs453/index.html` indicates the file name.
- b) The browser is running HTTP version 1.1, as indicated just before the first `<cr><lf>` pair.
- c) The browser is requesting a persistent connection, as indicated by the Connection: keep-alive.
- d) This is a trick question. This information is not contained in an HTTP message anywhere. So there is no way to tell this from looking at the exchange of HTTP messages alone. One would need information from the IP datagrams (that carried the TCP segment that carried the HTTP GET request) to answer this question.

- e) Mozilla/5.0. The browser type information is needed by the server to send different versions of the same object to different types of browsers.

Problem 5

- a) The status code of 200 and the phrase OK indicate that the server was able to locate the document successfully. The reply was provided on Tuesday, 07 Mar 2008 12:39:45 Greenwich Mean Time.
- b) The document index.html was last modified on Saturday 10 Dec 2005 18:27:46 GMT.
- c) There are 3874 bytes in the document being returned.
- d) The first five bytes of the returned document are : <!doc. The server agreed to a persistent connection, as indicated by the Connection: Keep-Alive field

Problem 6

- a) Persistent connections are discussed in section 8 of RFC 2616 (the real goal of this question was to get you to retrieve and read an RFC). Sections 8.1.2 and 8.1.2.1 of the RFC indicate that either the client or the server can indicate to the other that it is going to close the persistent connection. It does so by including the connection-token "close" in the Connection-header field of the http request/reply.
- b) HTTP does not provide any encryption services.
- c) (From RFC 2616) "Clients that use persistent connections should limit the number of simultaneous connections that they maintain to a given server. A single-user client SHOULD NOT maintain more than 2 connections with any server or proxy."
- d) Yes. (From RFC 2616) "A client might have started to send a new request at the same time that the server has decided to close the "idle" connection. From the server's point of view, the connection is being closed while it was idle, but from the client's point of view, a request is in progress."

Problem 7

The total amount of time to get the IP address is

$$RTT_1 + RTT_2 + \dots + RTT_n.$$

Once the IP address is known, RTT_o elapses to set up the TCP connection and another

RTT_o elapses to request and receive the small object. The total response time is

$$2RTT_o + RTT_1 + RTT_2 + \dots + RTT_n$$

Problem 8

a)

$$RTT_1 + \dots + RTT_n + 2RTT_o + 8 \cdot 2RTT_o \\ = 18RTT_o + RTT_1 + \dots + RTT_n.$$

b)

$$RTT_1 + \dots + RTT_n + 2RTT_o + 2 \cdot 2RTT_o \\ = 6RTT_o + RTT_1 + \dots + RTT_n$$

c)

$$RTT_1 + \dots + RTT_n + 2RTT_o + RTT_o \\ = 3RTT_o + RTT_1 + \dots + RTT_n.$$

Problem 9

- a) The time to transmit an object of size L over a link of rate R is L/R . The average time is the average size of the object divided by R :

$$\Delta = (850,000 \text{ bits}) / (15,000,000 \text{ bits/sec}) = .0567 \text{ sec}$$

The traffic intensity on the link is given by $\beta\Delta = (16 \text{ requests/sec})(.0567 \text{ sec/request}) = 0.907$. Thus, the average access delay is $(.0567 \text{ sec}) / (1 - .907) \approx .6 \text{ seconds}$. The total average response time is therefore $.6 \text{ sec} + 3 \text{ sec} = 3.6 \text{ sec}$.

- b) The traffic intensity on the access link is reduced by 60% since the 60% of the requests are satisfied within the institutional network. Thus the average access delay is $(.0567 \text{ sec}) / [1 - (.4)(.907)] = .089 \text{ seconds}$. The response time is approximately zero if the request is satisfied by the cache (which happens with probability .6); the average response time is $.089 \text{ sec} + 3 \text{ sec} = 3.089 \text{ sec}$ for cache misses (which happens 40% of the time). So the average response time is $(.6)(0 \text{ sec}) + (.4)(3.089 \text{ sec}) = 1.24 \text{ seconds}$. Thus the average response time is reduced from 3.6 sec to 1.24 sec.

Problem 10

Note that each downloaded object can be completely put into one data packet. Let T_p denote the one-way propagation delay between the client and the server.

First consider parallel downloads using non-persistent connections. Parallel downloads would allow 10 connections to share the 150 bits/sec bandwidth, giving each just 15 bits/sec. Thus, the total time needed to receive all objects is given by:

$$\begin{aligned}
& (200/150+T_p + 200/150 +T_p + 200/150+T_p + 100,000/150+ T_p) \\
& + (200/(150/10)+T_p + 200/(150/10) +T_p + 200/(150/10)+T_p + 100,000/(150/10)+ T_p) \\
& = 7377 + 8*T_p \text{ (seconds)}
\end{aligned}$$

Now consider a persistent HTTP connection. The total time needed is given by:

$$\begin{aligned}
& (200/150+T_p + 200/150 +T_p + 200/150+T_p + 100,000/150+ T_p) \\
& + 10*(200/150+T_p + 100,000/150+ T_p) \\
& = 7351 + 24*T_p \text{ (seconds)}
\end{aligned}$$

Assuming the speed of light is $300*10^6$ m/sec, then $T_p=10/(300*10^6)=0.03$ microsec. T_p is therefore negligible compared with transmission delay.

Thus, we see that persistent HTTP is not significantly faster (less than 1 percent) than the non-persistent case with parallel download.

Problem 11

- Yes, because Bob has more connections, he can get a larger share of the link bandwidth.
- Yes, Bob still needs to perform parallel downloads; otherwise he will get less bandwidth than the other four users.

Problem 12

Server.py

```

from socket import *
serverPort=12000
serverSocket=socket(AF_INET,SOCK_STREAM)
serverSocket.bind(('',serverPort))
serverSocket.listen(1)
connectionSocket, addr = serverSocket.accept()
while 1:
    sentence = connectionSocket.recv(1024)
    print 'From Server:', sentence, '\n'
serverSocket.close()

```

Problem 13

The MAIL FROM: in SMTP is a message from the SMTP client that identifies the sender of the mail message to the SMTP server. The From: on the mail message itself is NOT an SMTP message, but rather is just a line in the body of the mail message.

Problem 14

SMTP uses a line containing only a period to mark the end of a message body.

HTTP uses "Content-Length header field" to indicate the length of a message body.

No, HTTP cannot use the method used by SMTP, because HTTP message could be binary data, whereas in SMTP, the message body must be in 7-bit ASCII format.

Problem 15

MTA stands for Mail Transfer Agent. A host sends the message to an MTA. The message then follows a sequence of MTAs to reach the receiver's mail reader. We see that this spam message follows a chain of MTAs. An honest MTA should report where it receives the message. Notice that in this message, "asusus-4b96 ([58.88.21.177])" does not report from where it received the email. Since we assume only the originator is dishonest, so "asusus-4b96 ([58.88.21.177])" must be the originator.

Problem 16

UIDL abbreviates "unique-ID listing". When a POP3 client issues the UIDL command, the server responds with the unique message ID for all of the messages present in the user's mailbox. This command is useful for "download and keep". By maintaining a file that lists the messages retrieved during earlier sessions, the client can use the UIDL command to determine which messages on the server have already been seen.

Problem 17

a) C: dele 1
C: retr 2
S: (blah blah ...
S:blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off

b) C: retr 2
S: blah blah ...
S:blah
S: .
C: quit
S: +OK POP3 server signing off

c) C: list
 S: 1 498
 S: 2 912
 S: .
 C: retr 1
 S: blah
 S:blah
 S: .
 C: retr 2
 S: blah blah ...
 S:blah
 S: .
 C: quit
 S: +OK POP3 server signing off

Problem 18

a) For a given input of domain name (such as ccn.com), IP address or network administrator name, the *whois* database can be used to locate the corresponding registrar, whois server, DNS server, and so on.

b) NS4.YAHOO.COM from www.register.com; NS1.MSFT.NET from ww.register.com

c) *Local Domain: www.mindspring.com*

Web servers : www.mindspring.com

207.69.189.21, 207.69.189.22,
 207.69.189.23, 207.69.189.24,
 207.69.189.25, 207.69.189.26, 207.69.189.27,
 207.69.189.28

Mail Servers : mx1.mindspring.com (207.69.189.217)

mx2.mindspring.com (207.69.189.218)

mx3.mindspring.com (207.69.189.219)

mx4.mindspring.com (207.69.189.220)

Name Servers: itchy.earthlink.net (207.69.188.196)

scratchy.earthlink.net (207.69.188.197)

www.yahoo.com

Web Servers: www.yahoo.com (216.109.112.135, 66.94.234.13)

Mail Servers: a.mx.mail.yahoo.com (209.191.118.103)

b.mx.mail.yahoo.com (66.196.97.250)

c.mx.mail.yahoo.com (68.142.237.182, 216.39.53.3)

d.mx.mail.yahoo.com (216.39.53.2)

e.mx.mail.yahoo.com (216.39.53.1)

f.mx.mail.yahoo.com (209.191.88.247, 68.142.202.247)

g.mx.mail.yahoo.com (209.191.88.239, 206.190.53.191)

Name Servers: ns1.yahoo.com (66.218.71.63)
ns2.yahoo.com (68.142.255.16)
ns3.yahoo.com (217.12.4.104)
ns4.yahoo.com (68.142.196.63)
ns5.yahoo.com (216.109.116.17)
ns8.yahoo.com (202.165.104.22)
ns9.yahoo.com (202.160.176.146)

www.hotmail.com

Web Servers: www.hotmail.com (64.4.33.7, 64.4.32.7)

Mail Servers: mx1.hotmail.com (65.54.245.8, 65.54.244.8, 65.54.244.136)
mx2.hotmail.com (65.54.244.40, 65.54.244.168, 65.54.245.40)
mx3.hotmail.com (65.54.244.72, 65.54.244.200, 65.54.245.72)
mx4.hotmail.com (65.54.244.232, 65.54.245.104, 65.54.244.104)

Name Servers: ns1.msft.net (207.68.160.190)
ns2.msft.net (65.54.240.126)
ns3.msft.net (213.199.161.77)
ns4.msft.net (207.46.66.126)
ns5.msft.net (65.55.238.126)

- d) The yahoo web server has multiple IP addresses
www.yahoo.com (216.109.112.135, 66.94.234.13)
- e) The address range for Polytechnic University: 128.238.0.0 – 128.238.255.255
- f) An attacker can use the *whois* database and nslookup tool to determine the IP address ranges, DNS server addresses, etc., for the target institution.
- g) By analyzing the source address of attack packets, the victim can use whois to obtain information about domain from which the attack is coming and possibly inform the administrators of the origin domain.

Problem 19

- a) The following delegation chain is used for gaia.cs.umass.edu
a.root-servers.net
E.GTLD-SERVERS.NET
ns1.umass.edu(authoritative)

First command:

dig +norecurse @a.root-servers.net any gaia.cs.umass.edu

:: AUTHORITY SECTION:

edu.	172800	IN	NS	E.GTLD-SERVERS.NET.
edu.	172800	IN	NS	A.GTLD-SERVERS.NET.
edu.	172800	IN	NS	G3.NSTLD.COM.
edu.	172800	IN	NS	D.GTLD-SERVERS.NET.
edu.	172800	IN	NS	H3.NSTLD.COM.
edu.	172800	IN	NS	L3.NSTLD.COM.
edu.	172800	IN	NS	M3.NSTLD.COM.
edu.	172800	IN	NS	C.GTLD-SERVERS.NET.

Among all returned edu DNS servers, we send a query to the first one.

dig +norecurse @E.GTLD-SERVERS.NET any gaia.cs.umass.edu

umass.edu.	172800	IN	NS	ns1.umass.edu.
umass.edu.	172800	IN	NS	ns2.umass.edu.
umass.edu.	172800	IN	NS	ns3.umass.edu.

Among all three returned authoritative DNS servers, we send a query to the first one.

dig +norecurse @ns1.umass.edu any gaia.cs.umass.edu

gaia.cs.umass.edu. 21600 IN A 128.119.245.12

b) The answer for google.com could be:

a.root-servers.net

E.GTLD-SERVERS.NET

ns1.google.com(authoritative)

Problem 20

We can periodically take a snapshot of the DNS caches in the local DNS servers. The Web server that appears most frequently in the DNS caches is the most popular server. This is because if more users are interested in a Web server, then DNS requests for that server are more frequently sent by users. Thus, that Web server will appear in the DNS caches more frequently.

For a complete measurement study, see:

Craig E. Wills, Mikhail Mikhailov, Hao Shang

“Inferring Relative Popularity of Internet Applications by Actively Querying DNS Caches”, in IMC'03, October 27-29, 2003, Miami Beach, Florida, USA

Problem 21

Yes, we can use dig to query that Web site in the local DNS server.

For example, “dig cnn.com” will return the query time for finding cnn.com. If cnn.com was just accessed a couple of seconds ago, an entry for cnn.com is cached in the local DNS cache, so the query time is 0 msec. Otherwise, the query time is large.

Problem 22

For calculating the minimum distribution time for client-server distribution, we use the following formula:

$$D_{cs} = \max \{NF/u_s, F/d_{\min}\}$$

Similarly, for calculating the minimum distribution time for P2P distribution, we use the following formula:

$$D_{P2P} = \max \{F/u_s, F/d_{\min}, NF/(u_s + \sum_{i=1}^N u_i)\}$$

Where, $F = 15 \text{ Gbits} = 15 * 1024 \text{ Mbits}$

$u_s = 30 \text{ Mbps}$

$d_{\min} = d_i = 2 \text{ Mbps}$

Note, 300Kbps = 300/1024 Mbps.

Client Server

		N		
		10	100	1000
u	300 Kbps	7680	51200	512000
	700 Kbps	7680	51200	512000
	2 Mbps	7680	51200	512000

Peer to Peer

		N		
		10	100	1000
u	300 Kbps	7680	25904	47559
	700 Kbps	7680	15616	21525
	2 Mbps	7680	7680	7680

Problem 23

- a) Consider a distribution scheme in which the server sends the file to each client, in parallel, at a rate of a rate of u_s/N . Note that this rate is less than each of the client's download rate, since by assumption $u_s/N \leq d_{\min}$. Thus each client can also receive at rate u_s/N . Since each client receives at rate u_s/N , the time for each client to receive the entire file is $F/(u_s/N) = NF/u_s$. Since all the clients receive the file in NF/u_s , the overall distribution time is also NF/u_s .

- b) Consider a distribution scheme in which the server sends the file to each client, in parallel, at a rate of d_{\min} . Note that the aggregate rate, $N d_{\min}$, is less than the server's link rate u_s , since by assumption $u_s/N \geq d_{\min}$. Since each client receives at rate d_{\min} , the time for each client to receive the entire file is F/d_{\min} . Since all the clients receive the file in this time, the overall distribution time is also F/d_{\min} .

- c) From Section 2.6 we know that

$$D_{CS} \geq \max \{NF/u_s, F/d_{\min}\} \quad (\text{Equation 1})$$

Suppose that $u_s/N \leq d_{\min}$. Then from Equation 1 we have $D_{CS} \geq NF/u_s$. But from (a) we have $D_{CS} \leq NF/u_s$. Combining these two gives:

$$D_{CS} = NF/u_s \text{ when } u_s/N \leq d_{\min}. \quad (\text{Equation 2})$$

We can similarly show that:

$$D_{CS} = F/d_{\min} \text{ when } u_s/N \geq d_{\min} \quad (\text{Equation 3}).$$

Combining Equation 2 and Equation 3 gives the desired result.

Problem 24

- a) Define $u = u_1 + u_2 + \dots + u_N$. By assumption

$$u_s \leq (u_s + u)/N \quad \text{Equation 1}$$

Divide the file into N parts, with the i^{th} part having size $(u_i/u)F$. The server transmits the i^{th} part to peer i at rate $r_i = (u_i/u)u_s$. Note that $r_1 + r_2 + \dots + r_N = u_s$, so that the aggregate server rate does not exceed the link rate of the server. Also have each peer i forward the bits it receives to each of the $N-1$ peers at rate r_i . The aggregate forwarding rate by peer i is $(N-1)r_i$. We have

$$(N-1)r_i = (N-1)(u_s u_i)/u \leq u_i,$$

where the last inequality follows from Equation 1. Thus the aggregate forwarding rate of peer i is less than its link rate u_i .

In this distribution scheme, peer i receives bits at an aggregate rate of

$$r_i + \sum_{j \neq i} r_j = u_s$$

Thus each peer receives the file in F/u_s .

- b) Again define $u = u_1 + u_2 + \dots + u_N$. By assumption

$$u_s \geq (u_s + u)/N \quad \text{Equation 2}$$

$$\text{Let } r_i = u_i/(N-1) \text{ and} \\ r_{N+1} = (u_s - u/(N-1))/N$$

In this distribution scheme, the file is broken into $N+1$ parts. The server sends bits from the i^{th} part to the i^{th} peer ($i = 1, \dots, N$) at rate r_i . Each peer i forwards the bits arriving at rate r_i to each of the other $N-1$ peers. Additionally, the server sends bits from the $(N+1)^{\text{st}}$ part at rate r_{N+1} to each of the N peers. The peers do not forward the bits from the $(N+1)^{\text{st}}$ part.

The aggregate send rate of the server is

$$r_1 + \dots + r_N + N r_{N+1} = u/(N-1) + u_s - u/(N-1) = u_s$$

Thus, the server's send rate does not exceed its link rate. The aggregate send rate of peer i is

$$(N-1)r_i = u_i$$

Thus, each peer's send rate does not exceed its link rate.

In this distribution scheme, peer i receives bits at an aggregate rate of

$$r_i + r_{N+1} + \sum_{j < i} r_j = u/(N-1) + (u_s - u/(N-1))/N = (u_s + u)/N$$

Thus each peer receives the file in $NF/(u_s + u)$.

(For simplicity, we neglected to specify the size of the file part for $i = 1, \dots, N+1$. We now provide that here. Let $\Delta = (u_s + u)/N$ be the distribution time. For $i = 1, \dots, N$, the i^{th} file part is $F_i = r_i \Delta$ bits. The $(N+1)^{\text{st}}$ file part is $F_{N+1} = r_{N+1} \Delta$ bits. It is straightforward to show that $F_1 + \dots + F_{N+1} = F$.)

c) The solution to this part is similar to that of 17 (c). We know from section 2.6 that

$$D_{P2P} \geq \max\{F/u_s, NF/(u_s + u)\}$$

Combining this with a) and b) gives the desired result.

Problem 25

There are N nodes in the overlay network. There are $N(N-1)/2$ edges.

Problem 26

Yes. His first claim is possible, as long as there are enough peers staying in the swarm for a long enough time. Bob can always receive data through optimistic unchoking by other peers.

His second claim is also true. He can run a client on each host, let each client “free-ride,” and combine the collected chunks from the different hosts into a single file. He can even write a small scheduling program to make the different hosts ask for different chunks of the file. This is actually a kind of Sybil attack in P2P networks.

Problem 27

Peer 3 learns that peer 5 has just left the system, so Peer 3 asks its first successor (Peer 4) for the identifier of its immediate successor (peer 8). Peer 3 will then make peer 8 its second successor.

Problem 28

Peer 6 would first send peer 15 a message, saying “what will be peer 6’s predecessor and successor?” This message gets forwarded through the DHT until it reaches peer 5, who realizes that it will be 6’s predecessor and that its current successor, peer 8, will become 6’s successor. Next, peer 5 sends this predecessor and successor information back to 6. Peer 6 can now join the DHT by making peer 8 its successor and by notifying peer 5 that it should change its immediate successor to 6.

Problem 29

For each key, we first calculate the distances (using $d(k,p)$) between itself and all peers, and then store the key in the peer that is closest to the key (that is, with smallest distance value).

Problem 30

Yes, randomly assigning keys to peers does not consider the underlying network at all, so it very likely causes mismatches.

Such mismatches may degrade the search performance. For example, consider a logical path p_1 (consisting of only two logical links): $A \rightarrow B \rightarrow C$, where A and B are neighboring peers, and B and C are neighboring peers. Suppose that there is another logical path p_2 from A to C (consisting of 3 logical links): $A \rightarrow D \rightarrow E \rightarrow C$.

It might be the case that A and B are very far away physically (and separated by many routers), and B and C are very far away physically (and separated by many routers). But

it may be the case that A, D, E, and C are all very close physically (and all separated by few routers). In other words, a shorter logical path may correspond to a much longer physical path.

Problem 31

- a) If you run TCPClient first, then the client will attempt to make a TCP connection with a non-existent server process. A TCP connection will not be made.
- b) UDPClient doesn't establish a TCP connection with the server. Thus, everything should work fine if you first run UDPClient, then run UDPServer, and then type some input into the keyboard.
- c) If you use different port numbers, then the client will attempt to establish a TCP connection with the wrong process or a non-existent process. Errors will occur.

Problem 32

In the original program, UDPClient does not specify a port number when it creates the socket. In this case, the code lets the underlying operating system choose a port number. With the additional line, when UDPClient is executed, a UDP socket is created with port number 5432 .

UDPServer needs to know the client port number so that it can send packets back to the correct client socket. Glancing at UDPServer, we see that the client port number is not “hard-wired” into the server code; instead, UDPServer determines the client port number by unraveling the datagram it receives from the client. Thus UDP server will work with any client port number, including 5432. UDPServer therefore does not need to be modified.

Before:

Client socket = x (chosen by OS)
Server socket = 9876

After:

Client socket = 5432

Problem 33

Yes, you can configure many browsers to open multiple simultaneous connections to a Web site. The advantage is that you will potentially download the file faster. The

disadvantage is that you may be hogging the bandwidth, thereby significantly slowing down the downloads of other users who are sharing the same physical links.

Problem 34

For an application such as remote login (telnet and ssh), a byte-stream oriented protocol is very natural since there is no notion of message boundaries in the application. When a user types a character, we simply drop the character into the TCP connection.

In other applications, we may be sending a series of messages that have inherent boundaries between them. For example, when one SMTP mail server sends another SMTP mail server several email messages back to back. Since TCP does not have a mechanism to indicate the boundaries, the application must add the indications itself, so that receiving side of the application can distinguish one message from the next. If each message were instead put into a distinct UDP segment, the receiving end would be able to distinguish the various messages without any indications added by the sending side of the application.

Problem 35

To create a web server, we need to run web server software on a host. Many vendors sell web server software. However, the most popular web server software today is Apache, which is open source and free. Over the years it has been highly optimized by the open-source community.

Problem 36

The key is the infohash, the value is an IP address that currently has the file designated by the infohash.

Only the future will tell whether DCCP, SCTP, or TFRC will see widespread deployment. While these protocols clearly provide enhanced capabilities over TCP and UDP, TCP and UDP have proven themselves “good enough” over the years. Whether “better” wins out over “good enough” will depend on a complex mix of technical, social, and business considerations.

In Chapter 1, we said that a computer network can be partitioned into the “network edge” and the “network core.” The network edge covers everything that happens in the end systems. Having now covered the application layer and the transport layer, our discussion of the network edge is complete. It is time to explore the network core! This journey begins in the next chapter, where we’ll study the network layer, and continues into Chapter 5, where we’ll study the link layer.



Homework Problems and Questions

Chapter 3 Review Questions

SECTIONS 3.1–3.3

- R1. Suppose the network layer provides the following service. The network layer in the source host accepts a segment of maximum size 1,200 bytes and a destination host address from the transport layer. The network layer then guarantees to deliver the segment to the transport layer at the destination host. Suppose many network application processes can be running at the destination host.
- Design the simplest possible transport-layer protocol that will get application data to the desired process at the destination host. Assume the operating system in the destination host has assigned a 4-byte port number to each running application process.
 - Modify this protocol so that it provides a “return address” to the destination process.
 - In your protocols, does the transport layer “have to do anything” in the core of the computer network?
- R2. Consider a planet where everyone belongs to a family of six, every family lives in its own house, each house has a unique address, and each person in a given house has a unique name. Suppose this planet has a mail service that delivers letters from source house to destination house. The mail service requires that (1) the letter be in an envelope, and that (2) the address of the destination house (and nothing more) be clearly written on the envelope. Suppose each family has a delegate family member who collects and distributes letters for the other family members. The letters do not necessarily provide any indication of the recipients of the letters.

- a. Using the solution to Problem R1 above as inspiration, describe a protocol that the delegates can use to deliver letters from a sending family member to a receiving family member.
 - b. In your protocol, does the mail service ever have to open the envelope and examine the letter in order to provide its service?
- R3. Consider a TCP connection between Host A and Host B. Suppose that the TCP segments traveling from Host A to Host B have source port number x and destination port number y . What are the source and destination port numbers for the segments traveling from Host B to Host A?
- R4. Describe why an application developer might choose to run an application over UDP rather than TCP.
- R5. Why is it that voice and video traffic is often sent over TCP rather than UDP in today's Internet? (*Hint*: The answer we are looking for has nothing to do with TCP's congestion-control mechanism.)
- R6. Is it possible for an application to enjoy reliable data transfer even when the application runs over UDP? If so, how?
- R7. Suppose a process in Host C has a UDP socket with port number 6789. Suppose both Host A and Host B each send a UDP segment to Host C with destination port number 6789. Will both of these segments be directed to the same socket at Host C? If so, how will the process at Host C know that these two segments originated from two different hosts?
- R8. Suppose that a Web server runs in Host C on port 80. Suppose this Web server uses persistent connections, and is currently receiving requests from two different Hosts, A and B. Are all of the requests being sent through the same socket at Host C? If they are being passed through different sockets, do both of the sockets have port 80? Discuss and explain.

SECTION 3.4

- R9. In our `rdt` protocols, why did we need to introduce sequence numbers?
- R10. In our `rdt` protocols, why did we need to introduce timers?
- R11. Suppose that the roundtrip delay between sender and receiver is constant and known to the sender. Would a timer still be necessary in protocol `rdt 3.0`, assuming that packets can be lost? Explain.
- R12. Visit the Go-Back-N Java applet at the companion Web site.
- a. Have the source send five packets, and then pause the animation before any of the five packets reach the destination. Then kill the first packet and resume the animation. Describe what happens.
 - b. Repeat the experiment, but now let the first packet reach the destination and kill the first acknowledgment. Describe again what happens.
 - c. Finally, try sending six packets. What happens?

R13. Repeat R12, but now with the Selective Repeat Java applet. How are Selective Repeat and Go-Back-N different?

SECTION 3.5

R14. True or false?

- a. Host A is sending Host B a large file over a TCP connection. Assume Host B has no data to send Host A. Host B will not send acknowledgments to Host A because Host B cannot piggyback the acknowledgments on data.
- b. The size of the TCP `rwnd` never changes throughout the duration of the connection.
- c. Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.
- d. Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m , then the sequence number for the subsequent segment will necessarily be $m + 1$.
- e. The TCP segment has a field in its header for `rwnd`.
- f. Suppose that the last `SampleRTT` in a TCP connection is equal to 1 sec. The current value of `TimeoutInterval` for the connection will necessarily be ≥ 1 sec.
- g. Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.

R15. Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number 90; the second has sequence number 110.

- a. How much data is in the first segment?
- b. Suppose that the first segment is lost but the second segment arrives at B. In the acknowledgment that Host B sends to Host A, what will be the acknowledgment number?

R16. Consider the Telnet example discussed in Section 3.5. A few seconds after the user types the letter 'C,' the user types the letter 'R.' After typing the letter 'R,' how many segments are sent, and what is put in the sequence number and acknowledgment fields of the segments?

SECTION 3.7

R17. Suppose two TCP connections are present over some bottleneck link of rate R bps. Both connections have a huge file to send (in the same direction over the

bottleneck link). The transmissions of the files start at the same time. What transmission rate would TCP like to give to each of the connections?

- R18. True or false? Consider congestion control in TCP. When the timer expires at the sender, the value of `ssthresh` is set to one half of its previous value.
- R19. In the discussion of TCP splitting in the sidebar in Section 7.2, it was claimed that the response time with TCP splitting is approximately $4 \cdot \text{RTT}_{\text{FE}} + \text{RTT}_{\text{BE}} + \text{processing time}$. Justify this claim.



Problems

- P1. Suppose Client A initiates a Telnet session with Server S. At about the same time, Client B also initiates a Telnet session with Server S. Provide possible source and destination port numbers for
- The segments sent from A to S.
 - The segments sent from B to S.
 - The segments sent from S to A.
 - The segments sent from S to B.
 - If A and B are different hosts, is it possible that the source port number in the segments from A to S is the same as that from B to S?
 - How about if they are the same host?
- P2. Consider Figure 3.5. What are the source and destination port values in the segments flowing from the server back to the clients' processes? What are the IP addresses in the network-layer datagrams carrying the transport-layer segments?
- P3. UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?
- P4. a. Suppose you have the following 2 bytes: 01011100 and 01100101. What is the 1s complement of the sum of these 2 bytes?
- b. Suppose you have the following 2 bytes: 11011010 and 01100101. What is the 1s complement of the sum of these 2 bytes?
- c. For the bytes in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1s complement doesn't change.

- P5. Suppose that the UDP receiver computes the Internet checksum for the received UDP segment and finds that it matches the value carried in the checksum field. Can the receiver be absolutely certain that no bit errors have occurred? Explain.
- P6. Consider our motivation for correcting protocol `rdt2.1`. Show that the receiver, shown in Figure 3.57, when operating with the sender shown in Figure 3.11, can lead the sender and receiver to enter into a deadlock state, where each is waiting for an event that will never occur.
- P7. In protocol `rdt3.0`, the ACK packets flowing from the receiver to the sender do not have sequence numbers (although they do have an ACK field that contains the sequence number of the packet they are acknowledging). Why is it that our ACK packets do not require sequence numbers?
- P8. Draw the FSM for the receiver side of protocol `rdt3.0`.
- P9. Give a trace of the operation of protocol `rdt3.0` when data packets and acknowledgment packets are garbled. Your trace should be similar to that used in Figure 3.16.
- P10. Consider a channel that can lose packets but has a maximum delay that is known. Modify protocol `rdt2.1` to include sender timeout and retransmit. Informally argue why your protocol can communicate correctly over this channel.

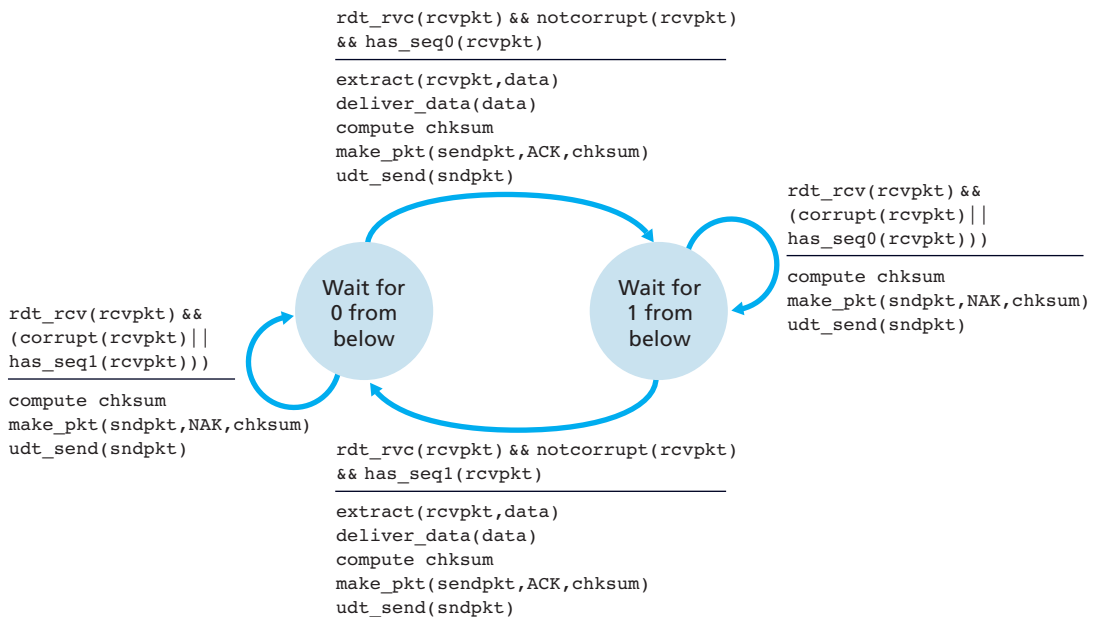


Figure 3.57 ♦ An incorrect receiver for protocol `rdt2.1`

- P11. Consider the `rdt2.2` receiver in Figure 3.14, and the creation of a new packet in the self-transition (i.e., the transition from the state back to itself) in the Wait-for-0-from-below and the Wait-for-1-from-below states: `sndpkt=make_pkt(ACK,0,checksum)` and `sndpkt=make_pkt(ACK,0,checksum)`. Would the protocol work correctly if this action were removed from the self-transition in the Wait-for-1-from-below state? Justify your answer. What if this event were removed from the self-transition in the Wait-for-0-from-below state? [*Hint*: In this latter case, consider what would happen if the first sender-to-receiver packet were corrupted.]
- P12. The sender side of `rdt3.0` simply ignores (that is, takes no action on) all received packets that are either in error or have the wrong value in the `acknum` field of an acknowledgment packet. Suppose that in such circumstances, `rdt3.0` were simply to retransmit the current data packet. Would the protocol still work? (*Hint*: Consider what would happen if there were only bit errors; there are no packet losses but premature timeouts can occur. Consider how many times the n th packet is sent, in the limit as n approaches infinity.)
- P13. Consider the `rdt3.0` protocol. Draw a diagram showing that if the network connection between the sender and receiver can reorder messages (that is, that two messages propagating in the medium between the sender and receiver can be reordered), then the alternating-bit protocol will not work correctly (make sure you clearly identify the sense in which it will not work correctly). Your diagram should have the sender on the left and the receiver on the right, with the time axis running down the page, showing data (D) and acknowledgment (A) message exchange. Make sure you indicate the sequence number associated with any data or acknowledgment segment.
- P14. Consider a reliable data transfer protocol that uses only negative acknowledgments. Suppose the sender sends data only infrequently. Would a NAK-only protocol be preferable to a protocol that uses ACKs? Why? Now suppose the sender has a lot of data to send and the end-to-end connection experiences few losses. In this second case, would a NAK-only protocol be preferable to a protocol that uses ACKs? Why?
- P15. Consider the cross-country example shown in Figure 3.17. How big would the window size have to be for the channel utilization to be greater than 98 percent? Suppose that the size of a packet is 1,500 bytes, including both header fields and data.
- P16. Suppose an application uses `rdt3.0` as its transport layer protocol. As the stop-and-wait protocol has very low channel utilization (shown in the cross-country example), the designers of this application let the receiver keep sending back a number (more than two) of alternating ACK 0 and ACK 1 even if

the corresponding data have not arrived at the receiver. Would this application design increase the channel utilization? Why? Are there any potential problems with this approach? Explain.

- P17. Consider two network entities, A and B, which are connected by a perfect bi-directional channel (i.e., any message sent will be received correctly; the channel will not corrupt, lose, or re-order packets). A and B are to deliver data messages to each other in an alternating manner: First, A must deliver a message to B, then B must deliver a message to A, then A must deliver a message to B and so on. If an entity is in a state where it should not attempt to deliver a message to the other side, and there is an event like `rdt_send(data)` call from above that attempts to pass data down for transmission to the other side, this call from above can simply be ignored with a call to `rdt_unable_to_send(data)`, which informs the higher layer that it is currently not able to send data. [Note: This simplifying assumption is made so you don't have to worry about buffering data.]

Draw a FSM specification for this protocol (one FSM for A, and one FSM for B!). Note that you do not have to worry about a reliability mechanism here; the main point of this question is to create a FSM specification that reflects the synchronized behavior of the two entities. You should use the following events and actions that have the same meaning as protocol rdt1.0 in Figure 3.9: `rdt_send(data)`, `packet = make_pkt(data)`, `udt_send(packet)`, `rdt_rcv(packet)`, `extract(packet, data)`, `deliver_data(data)`. Make sure your protocol reflects the strict alternation of sending between A and B. Also, make sure to indicate the initial states for A and B in your FSM descriptions.

- P18. In the generic SR protocol that we studied in Section 3.4.4, the sender transmits a message as soon as it is available (if it is in the window) without waiting for an acknowledgment. Suppose now that we want an SR protocol that sends messages two at a time. That is, the sender will send a pair of messages and will send the next pair of messages only when it knows that both messages in the first pair have been received correctly.

Suppose that the channel may lose messages but will not corrupt or reorder messages. Design an error-control protocol for the unidirectional reliable transfer of messages. Give an FSM description of the sender and receiver. Describe the format of the packets sent between sender and receiver, and vice versa. If you use any procedure calls other than those in Section 3.4 (for example, `udt_send()`, `start_timer()`, `rdt_rcv()`, and so on), clearly state their actions. Give an example (a timeline trace of sender and receiver) showing how your protocol recovers from a lost packet.

- P19. Consider a scenario in which Host A wants to simultaneously send packets to Hosts B and C. A is connected to B and C via a broadcast channel—a packet

sent by A is carried by the channel to both B and C. Suppose that the broadcast channel connecting A, B, and C can independently lose and corrupt packets (and so, for example, a packet sent from A might be correctly received by B, but not by C). Design a stop-and-wait-like error-control protocol for reliably transferring packets from A to B and C, such that A will not get new data from the upper layer until it knows that both B and C have correctly received the current packet. Give FSM descriptions of A and C. (*Hint:* The FSM for B should be essentially the same as for C.) Also, give a description of the packet format(s) used.

- P20. Consider a scenario in which Host A and Host B want to send messages to Host C. Hosts A and C are connected by a channel that can lose and corrupt (but not reorder) messages. Hosts B and C are connected by another channel (independent of the channel connecting A and C) with the same properties. The transport layer at Host C should alternate in delivering messages from A and B to the layer above (that is, it should first deliver the data from a packet from A, then the data from a packet from B, and so on). Design a stop-and-wait-like error-control protocol for reliably transferring packets from A and B to C, with alternating delivery at C as described above. Give FSM descriptions of A and C. (*Hint:* The FSM for B should be essentially the same as for A.) Also, give a description of the packet format(s) used.
- P21. Suppose we have two network entities, A and B. B has a supply of data messages that will be sent to A according to the following conventions. When A gets a request from the layer above to get the next data (D) message from B, A must send a request (R) message to B on the A-to-B channel. Only when B receives an R message can it send a data (D) message back to A on the B-to-A channel. A should deliver exactly one copy of each D message to the layer above. R messages can be lost (but not corrupted) in the A-to-B channel; D messages, once sent, are always delivered correctly. The delay along both channels is unknown and variable.

Design (give an FSM description of) a protocol that incorporates the appropriate mechanisms to compensate for the loss-prone A-to-B channel and implements message passing to the layer above at entity A, as discussed above. Use only those mechanisms that are absolutely necessary.

- P22. Consider the GBN protocol with a sender window size of 4 and a sequence number range of 1,024. Suppose that at time t , the next in-order packet that the receiver is expecting has a sequence number of k . Assume that the medium does not reorder messages. Answer the following questions:
- What are the possible sets of sequence numbers inside the sender's window at time t ? Justify your answer.
 - What are all possible values of the ACK field in all possible messages currently propagating back to the sender at time t ? Justify your answer.

- P23. Consider the GBN and SR protocols. Suppose the sequence number space is of size k . What is the largest allowable sender window that will avoid the occurrence of problems such as that in Figure 3.27 for each of these protocols?
- P24. Answer true or false to the following questions and briefly justify your answer:
- With the SR protocol, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.
 - With GBN, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.
 - The alternating-bit protocol is the same as the SR protocol with a sender and receiver window size of 1.
 - The alternating-bit protocol is the same as the GBN protocol with a sender and receiver window size of 1.
- P25. We have said that an application may choose UDP for a transport protocol because UDP offers finer application control (than TCP) of what data is sent in a segment and when.
- Why does an application have more control of what data is sent in a segment?
 - Why does an application have more control on when the segment is sent?
- P26. Consider transferring an enormous file of L bytes from Host A to Host B. Assume an MSS of 536 bytes.
- What is the maximum value of L such that TCP sequence numbers are not exhausted? Recall that the TCP sequence number field has 4 bytes.
 - For the L you obtain in (a), find how long it takes to transmit the file. Assume that a total of 66 bytes of transport, network, and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow control and congestion control so A can pump out the segments back to back and continuously.
- P27. Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.
- In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?
 - If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?

- c. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?
 - d. Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost and the second acknowledgment arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgments sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes of data; for each acknowledgment that you add, provide the acknowledgment number.
- P28. Host A and B are directly connected with a 100 Mbps link. There is one TCP connection between the two hosts, and Host A is sending to Host B an enormous file over this connection. Host A can send its application data into its TCP socket at a rate as high as 120 Mbps but Host B can read out of its TCP receive buffer at a maximum rate of 50 Mbps. Describe the effect of TCP flow control.
- P29. SYN cookies were discussed in Section 3.5.6.
- a. Why is it necessary for the server to use a special initial sequence number in the SYNACK?
 - b. Suppose an attacker knows that a target host uses SYN cookies. Can the attacker create half-open or fully open connections by simply sending an ACK packet to the target? Why or why not?
 - c. Suppose an attacker collects a large amount of initial sequence numbers sent by the server. Can the attacker cause the server to create many fully open connections by sending ACKs with those initial sequence numbers? Why?
- P30. Consider the network shown in Scenario 2 in Section 3.6.1. Suppose both sending hosts A and B have some fixed timeout values.
- a. Argue that increasing the size of the finite buffer of the router might possibly decrease the throughput (λ_{out}).
 - b. Now suppose both hosts dynamically adjust their timeout values (like what TCP does) based on the buffering delay at the router. Would increasing the buffer size help to increase the throughput? Why?
- P31. Suppose that the five measured `SampleRTT` values (see Section 3.5.3) are 106 ms, 120 ms, 140 ms, 90 ms, and 115 ms. Compute the `EstimatedRTT` after each of these `SampleRTT` values is obtained, using a value of $\alpha = 0.125$ and assuming that the value of `EstimatedRTT` was 100 ms just before the first of these five samples were obtained. Compute also the `DevRTT` after each sample is obtained, assuming a value of $\beta = 0.25$ and assuming the value of `DevRTT` was 5 ms just before the first of these five samples was obtained. Last, compute the `TCP TimeoutInterval` after each of these samples is obtained.

- P32. Consider the TCP procedure for estimating RTT. Suppose that $\alpha = 0.1$. Let `SampleRTT1` be the most recent sample RTT, let `SampleRTT2` be the next most recent sample RTT, and so on.
- For a given TCP connection, suppose four acknowledgments have been returned with corresponding sample RTTs: `SampleRTT4`, `SampleRTT3`, `SampleRTT2`, and `SampleRTT1`. Express `EstimatedRTT` in terms of the four sample RTTs.
 - Generalize your formula for n sample RTTs.
 - For the formula in part (b) let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.
- P33. In Section 3.5.3, we discussed TCP's estimation of RTT. Why do you think TCP avoids measuring the `SampleRTT` for retransmitted segments?
- P34. What is the relationship between the variable `SendBase` in Section 3.5.4 and the variable `LastByteRcvd` in Section 3.5.5?
- P35. What is the relationship between the variable `LastByteRcvd` in Section 3.5.5 and the variable y in Section 3.5.4?
- P36. In Section 3.5.4, we saw that TCP waits until it has received three duplicate ACKs before performing a fast retransmit. Why do you think the TCP designers chose not to perform a fast retransmit after the first duplicate ACK for a segment is received?
- P37. Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been correctly received by Host B.
- How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.
 - If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in shortest time interval?
- P38. In our description of TCP in Figure 3.53, the value of the threshold, `ssthresh`, is set as `ssthresh=cwnd/2` in several places and `ssthresh` value is referred to as being set to half the window size when a loss event occurred. Must the rate at which the sender is sending when the loss event occurred be approximately equal to `cwnd` segments per RTT? Explain your answer. If your answer is no, can you suggest a different manner in which `ssthresh` should be set?



VideoNote
Examining the
behavior of TCP

- P39. Consider Figure 3.46(b). If λ'_{in} increases beyond $R/2$, can λ_{out} increase beyond $R/3$? Explain. Now consider Figure 3.46(c). If λ'_{in} increases beyond $R/2$, can λ_{out} increase beyond $R/4$ under the assumption that a packet will be forwarded twice on average from the router to the receiver? Explain.
- P40. Consider Figure 3.58. Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions. In all cases, you should provide a short discussion justifying your answer.
- Identify the intervals of time when TCP slow start is operating.
 - Identify the intervals of time when TCP congestion avoidance is operating.
 - After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
 - After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
 - What is the initial value of `ssthresh` at the first transmission round?
 - What is the value of `ssthresh` at the 18th transmission round?
 - What is the value of `ssthresh` at the 24th transmission round?
 - During what transmission round is the 70th segment sent?
 - Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of `ssthresh`?

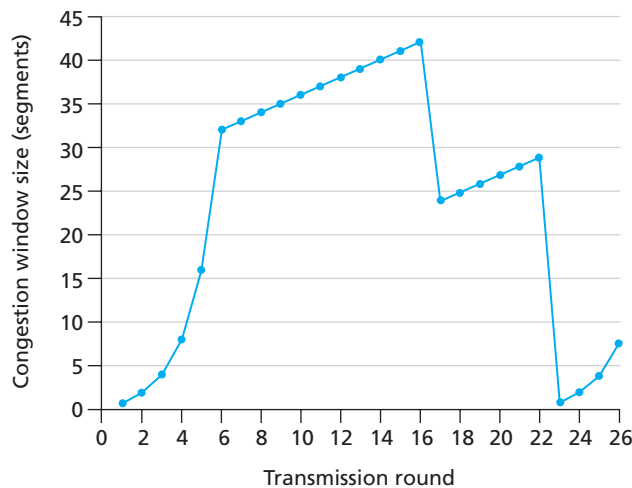


Figure 3.58 ♦ TCP window size as a function of time

- j. Suppose TCP Tahoe is used (instead of TCP Reno), and assume that triple duplicate ACKs are received at the 16th round. What are the `ssthresh` and the congestion window size at the 19th round?
- k. Again suppose TCP Tahoe is used, and there is a timeout event at 22nd round. How many packets have been sent out from 17th round till 22nd round, inclusive?
- P41. Refer to Figure 3.56, which illustrates the convergence of TCP's AIMD algorithm. Suppose that instead of a multiplicative decrease, TCP decreased the window size by a constant amount. Would the resulting AIAD algorithm converge to an equal share algorithm? Justify your answer using a diagram similar to Figure 3.56.
- P42. In Section 3.5.4, we discussed the doubling of the timeout interval after a timeout event. This mechanism is a form of congestion control. Why does TCP need a window-based congestion-control mechanism (as studied in Section 3.7) in addition to this doubling-timeout-interval mechanism?
- P43. Host A is sending an enormous file to Host B over a TCP connection. Over this connection there is never any packet loss and the timers never expire. Denote the transmission rate of the link connecting Host A to the Internet by R bps. Suppose that the process in Host A is capable of sending data into its TCP socket at a rate S bps, where $S = 10 \cdot R$. Further suppose that the TCP receive buffer is large enough to hold the entire file, and the send buffer can hold only one percent of the file. What would prevent the process in Host A from continuously passing data to its TCP socket at rate S bps? TCP flow control? TCP congestion control? Or something else? Elaborate.
- P44. Consider sending a large file from a host to another over a TCP connection that has no loss.
- Suppose TCP uses AIMD for its congestion control without slow start. Assuming `cwnd` increases by 1 MSS every time a batch of ACKs is received and assuming approximately constant round-trip times, how long does it take for `cwnd` increase from 6 MSS to 12 MSS (assuming no loss events)?
 - What is the average throughput (in terms of MSS and RTT) for this connection up through time = 6 RTT?
- P45. Recall the macroscopic description of TCP throughput. In the period of time from when the connection's rate varies from $W/(2 \cdot RTT)$ to W/RTT , only one packet is lost (at the very end of the period).
- Show that the loss rate (fraction of packets lost) is equal to

$$L = \text{loss rate} = \frac{1}{\frac{3}{8} W^2 + \frac{3}{4} W}$$

- b. Use the result above to show that if a connection has loss rate L , then its average rate is approximately given by

$$\approx \frac{1.22 \cdot MSS}{RTT \sqrt{L}}$$

- P46. Consider that only a single TCP (Reno) connection uses one 10Mbps link which does not buffer any data. Suppose that this link is the only congested link between the sending and receiving hosts. Assume that the TCP sender has a huge file to send to the receiver, and the receiver's receive buffer is much larger than the congestion window. We also make the following assumptions: each TCP segment size is 1,500 bytes; the two-way propagation delay of this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is, ignore slow start.
- What is the maximum window size (in segments) that this TCP connection can achieve?
 - What is the average window size (in segments) and average throughput (in bps) of this TCP connection?
 - How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?
- P47. Consider the scenario described in the previous problem. Suppose that the 10Mbps link can buffer a finite number of segments. Argue that in order for the link to always be busy sending data, we would like to choose a buffer size that is at least the product of the link speed C and the two-way propagation delay between the sender and the receiver.
- P48. Repeat Problem 43, but replacing the 10 Mbps link with a 10 Gbps link. Note that in your answer to part c, you will realize that it takes a very long time for the congestion window size to reach its maximum window size after recovering from a packet loss. Sketch a solution to solve this problem.
- P49. Let T (measured by RTT) denote the time interval that a TCP connection takes to increase its congestion window size from $W/2$ to W , where W is the maximum congestion window size. Argue that T is a function of TCP's average throughput.
- P50. Consider a simplified TCP's AIMD algorithm where the congestion window size is measured in number of segments, not in bytes. In additive increase, the congestion window size increases by one segment in each RTT. In multiplicative decrease, the congestion window size decreases by half (if the result is not an integer, round down to the nearest integer). Suppose that two TCP connections, C_1 and C_2 , share a single congested link of speed 30 segments per second. Assume that both C_1 and C_2 are in the congestion avoidance

phase. Connection C_1 's RTT is 50 msec and connection C_2 's RTT is 100 msec. Assume that when the data rate in the link exceeds the link's speed, all TCP connections experience data segment loss.

- a. If both C_1 and C_2 at time t_0 have a congestion window of 10 segments, what are their congestion window sizes after 1000 msec?
 - b. In the long run, will these two connections get the same share of the bandwidth of the congested link? Explain.
- P51. Consider the network described in the previous problem. Now suppose that the two TCP connections, C_1 and C_2 , have the same RTT of 100 msec. Suppose that at time t_0 , C_1 's congestion window size is 15 segments but C_2 's congestion window size is 10 segments.
- a. What are their congestion window sizes after 2200msec?
 - b. In the long run, will these two connections get about the same share of the bandwidth of the congested link?
 - c. We say that two connections are synchronized, if both connections reach their maximum window sizes at the same time and reach their minimum window sizes at the same time. In the long run, will these two connections get synchronized eventually? If so, what are their maximum window sizes?
 - d. Will this synchronization help to improve the utilization of the shared link? Why? Sketch some idea to break this synchronization.
- P52. Consider a modification to TCP's congestion control algorithm. Instead of additive increase, we can use multiplicative increase. A TCP sender increases its window size by a small positive constant a ($0 < a < 1$) whenever it receives a valid ACK. Find the functional relationship between loss rate L and maximum congestion window W . Argue that for this modified TCP, regardless of TCP's average throughput, a TCP connection always spends the same amount of time to increase its congestion window size from $W/2$ to W .
- P53. In our discussion of TCP futures in Section 3.7, we noted that to achieve a throughput of 10 Gbps, TCP could only tolerate a segment loss probability of $2 \cdot 10^{-10}$ (or equivalently, one loss event for every 5,000,000,000 segments). Show the derivation for the values of $2 \cdot 10^{-10}$ (1 out of 5,000,000) for the RTT and MSS values given in Section 3.7. If TCP needed to support a 100 Gbps connection, what would the tolerable loss be?
- P54. In our discussion of TCP congestion control in Section 3.7, we implicitly assumed that the TCP sender always had data to send. Consider now the case that the TCP sender sends a large amount of data and then goes idle (since it has no more data to send) at t_1 . TCP remains idle for a relatively long period of time and then wants to send more data at t_2 . What are the advantages and disadvantages of having TCP use the `cwnd` and `ssthresh` values from t_1 when starting to send data at t_2 ? What alternative would you recommend? Why?

- P55. In this problem we investigate whether either UDP or TCP provides a degree of end-point authentication.
- Consider a server that receives a request within a UDP packet and responds to that request within a UDP packet (for example, as done by a DNS server). If a client with IP address X spoofs its address with address Y , where will the server send its response?
 - Suppose a server receives a SYN with IP source address Y , and after responding with a SYNACK, receives an ACK with IP source address Y with the correct acknowledgment number. Assuming the server chooses a random initial sequence number and there is no “man-in-the-middle,” can the server be certain that the client is indeed at Y (and not at some other address X that is spoofing Y)?
- P56. In this problem, we consider the delay introduced by the TCP slow-start phase. Consider a client and a Web server directly connected by one link of rate R . Suppose the client wants to retrieve an object whose size is exactly equal to $15S$, where S is the maximum segment size (MSS). Denote the round-trip time between client and server as RTT (assumed to be constant). Ignoring protocol headers, determine the time to retrieve the object (including TCP connection establishment) when
- $4S/R > S/R + RTT > 2S/R$
 - $S/R + RTT > 4S/R$
 - $S/R > RTT$.



Programming Assignments

Implementing a Reliable Transport Protocol

In this laboratory programming assignment, you will be writing the sending and receiving transport-level code for implementing a simple reliable data transfer protocol. There are two versions of this lab, the alternating-bit-protocol version and the GBN version. This lab should be fun—your implementation will differ very little from what would be required in a real-world situation.

Since you probably don’t have standalone machines (with an OS that you can modify), your code will have to execute in a simulated hardware/software environment. However, the programming interface provided to your routines—the code that would call your entities from above and from below—is very close to what is done in an actual UNIX environment. (Indeed, the software interfaces described in this programming assignment are much more realistic than the infinite loop senders and receivers that many texts describe.) Stopping and starting

Chapter 3 Review Questions

1.
 - a) Call this protocol Simple Transport Protocol (STP). At the sender side, STP accepts from the sending process a chunk of data not exceeding 1196 bytes, a destination host address, and a destination port number. STP adds a four-byte header to each chunk and puts the port number of the destination process in this header. STP then gives the destination host address and the resulting segment to the network layer. The network layer delivers the segment to STP at the destination host. STP then examines the port number in the segment, extracts the data from the segment, and passes the data to the process identified by the port number.
 - b) The segment now has two header fields: a source port field and destination port field. At the sender side, STP accepts a chunk of data not exceeding 1192 bytes, a destination host address, a source port number, and a destination port number. STP creates a segment which contains the application data, source port number, and destination port number. It then gives the segment and the destination host address to the network layer. After receiving the segment, STP at the receiving host gives the application process the application data and the source port number.
 - c) No, the transport layer does not have to do anything in the core; the transport layer “lives” in the end systems.
2.
 1. For sending a letter, the family member is required to give the delegate the letter itself, the address of the destination house, and the name of the recipient. The delegate clearly writes the recipient’s name on the top of the letter. The delegate then puts the letter in an envelope and writes the address of the destination house on the envelope. The delegate then gives the letter to the planet’s mail service. At the receiving side, the delegate receives the letter from the mail service, takes the letter out of the envelope, and takes note of the recipient name written at the top of the letter. The delegate then gives the letter to the family member with this name.
 2. No, the mail service does not have to open the envelope; it only examines the address on the envelope.
3. Source port number y and destination port number x .
4. An application developer may not want its application to use TCP’s congestion control, which can throttle the application’s sending rate at times of congestion. Often, designers of IP telephony and IP videoconference applications choose to run their applications over UDP because they want to avoid TCP’s congestion control. Also, some applications do not need the reliable data transfer provided by TCP.
5. Since most firewalls are configured to block UDP traffic, using TCP for video and

voice traffic lets the traffic through the firewalls.

6. Yes. The application developer can put reliable data transfer into the application layer protocol. This would require a significant amount of work and debugging, however.
7. Yes, both segments will be directed to the same socket. For each received segment, at the socket interface, the operating system will provide the process with the IP addresses to determine the origins of the individual segments.
8. For each persistent connection, the Web server creates a separate “connection socket”. Each connection socket is identified with a four-tuple: (source IP address, source port number, destination IP address, destination port number). When host C receives an IP datagram, it examines these four fields in the datagram/segment to determine to which socket it should pass the payload of the TCP segment. Thus, the requests from A and B pass through different sockets. The identifier for both of these sockets has 80 for the destination port; however, the identifiers for these sockets have different values for source IP addresses. Unlike UDP, when the transport layer passes a TCP segment’s payload to the application process, it does not specify the source IP address, as this is implicitly specified by the socket identifier.
9. Sequence numbers are required for a receiver to find out whether an arriving packet contains new data or is a retransmission.
10. To handle losses in the channel. If the ACK for a transmitted packet is not received within the duration of the timer for the packet, the packet (or its ACK or NACK) is assumed to have been lost. Hence, the packet is retransmitted.
11. A timer would still be necessary in the protocol rdt 3.0. If the round trip time is known then the only advantage will be that, the sender knows for sure that either the packet or the ACK (or NACK) for the packet has been lost, as compared to the real scenario, where the ACK (or NACK) might still be on the way to the sender, after the timer expires. However, to detect the loss, for each packet, a timer of constant duration will still be necessary at the sender.
12.
 - a) The packet loss caused a time out after which all the five packets were retransmitted.
 - b) Loss of an ACK didn’t trigger any retransmission as Go-Back-N uses cumulative acknowledgements.
 - c) The sender was unable to send sixth packet as the send window size is fixed to 5.

13.

- a) When the packet was lost, the received four packets were buffered the receiver. After the timeout, sender retransmitted the lost packet and receiver delivered the buffered packets to application in correct order.
- b) Duplicate ACK was sent by the receiver for the lost ACK.
- c) The sender was unable to send sixth packet as the send window size is fixed to 5

When a packet was lost, GO-Back-N retransmitted all the packets whereas Selective Repeat retransmitted the lost packet only. In case of lost acknowledgement, selective repeat sent a duplicate ACK and as GO-Back-N used cumulative acknowledgment, so that duplicate ACK was unnecessary.

14. a) false b) false c) true d) false e) true f) false g) false

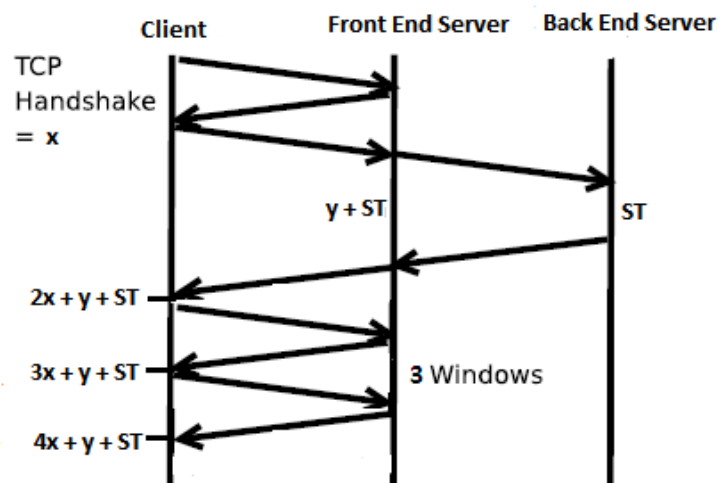
15. a) 20 bytes b) ack number = 90

16. 3 segments. First segment: seq = 43, ack = 80; Second segment: seq = 80, ack = 44; Third segment; seq = 44, ack = 81

17. $R/2$

18. False, it is set to half of the current value of the congestion window.

19. Let $X = RTT_{FE}$, $Y = RTT_{BE}$ and $ST =$ Search time. Consider the following timing diagram.



TCP packet exchange diagram between a client and a server (Back End) with a proxy (Front End) between them.

From this diagram we see that the total time is $4X + Y + ST = 4 \cdot \text{RTT}_{FE} + \text{RTT}_{BE} + \text{Search time}$

Chapter 3 Problems

Problem 1

	source port numbers	destination port numbers
a) $A \rightarrow S$	467	23
b) $B \rightarrow S$	513	23
c) $S \rightarrow A$	23	467
d) $S \rightarrow B$	23	513

e) Yes.

f) No.

Problem 2

Suppose the IP addresses of the hosts A, B, and C are a, b, c, respectively. (Note that a, b, c are distinct.)

To host A: Source port = 80, source IP address = b, dest port = 26145, dest IP address = a

To host C, left process: Source port = 80, source IP address = b, dest port = 7532, dest IP address = c

To host C, right process: Source port = 80, source IP address = b, dest port = 26145, dest IP address = c

Problem 3

Note, wrap around if overflow.

$$\begin{array}{r} 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ 1 \\ + \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \\ \hline 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \end{array}$$

$$\begin{array}{r}
 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1 \\
 +\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0 \\
 \hline
 0\ 0\ 1\ 0\ 1\ 1\ 1\ 0
 \end{array}$$

One's complement = 1 1 0 1 0 0 0 1.

To detect errors, the receiver adds the four words (the three original words and the checksum). If the sum contains a zero, the receiver knows there has been an error. All one-bit errors will be detected, but two-bit errors can be undetected (e.g., if the last digit of the first word is converted to a 0 and the last digit of the second word is converted to a 1).

Problem 4

- a) Adding the two bytes gives 11000001. Taking the one's complement gives 00111110.
- b) Adding the two bytes gives 01000000; the one's complement gives 10111111.
- c) First byte = 01010100; second byte = 01101101.

Problem 5

No, the receiver cannot be absolutely certain that no bit errors have occurred. This is because of the manner in which the checksum for the packet is calculated. If the corresponding bits (that would be added together) of two 16-bit words in the packet were 0 and 1 then even if these get flipped to 1 and 0 respectively, the sum still remains the same. Hence, the 1s complement the receiver calculates will also be the same. This means the checksum will verify even if there was transmission error.

Problem 6

Suppose the sender is in state "Wait for call 1 from above" and the receiver (the receiver shown in the homework problem) is in state "Wait for 1 from below." The sender sends a packet with sequence number 1, and transitions to "Wait for ACK or NAK 1," waiting for an ACK or NAK. Suppose now the receiver receives the packet with sequence number 1 correctly, sends an ACK, and transitions to state "Wait for 0 from below," waiting for a data packet with sequence number 0. However, the ACK is corrupted. When the rdt2.1 sender gets the corrupted ACK, it resends the packet with sequence number 1. However, the receiver is waiting for a packet with sequence number 0 and (as shown in the home work problem) always sends a NAK when it doesn't get a packet with sequence number 0. Hence the sender will always be sending a packet with sequence number 1, and the receiver will always be NAKing that packet. Neither will progress forward from that state.

Problem 7

To best answer this question, consider why we needed sequence numbers in the first place. We saw that the sender needs sequence numbers so that the receiver can tell if a data packet is a duplicate of an already received data packet. In the case of ACKs, the sender does not need this info (i.e., a sequence number on an ACK) to tell detect a duplicate ACK. A duplicate ACK is obvious to the rdt3.0 receiver, since when it has received the original ACK it transitioned to the next state. The duplicate ACK is not the ACK that the sender needs and hence is ignored by the rdt3.0 sender.

Problem 8

The sender side of protocol rdt3.0 differs from the sender side of protocol 2.2 in that timeouts have been added. We have seen that the introduction of timeouts adds the possibility of duplicate packets into the sender-to-receiver data stream. However, the receiver in protocol rdt.2.2 can already handle duplicate packets. (Receiver-side duplicates in rdt 2.2 would arise if the receiver sent an ACK that was lost, and the sender then retransmitted the old data). Hence the receiver in protocol rdt2.2 will also work as the receiver in protocol rdt 3.0.

Problem 9

Suppose the protocol has been in operation for some time. The sender is in state “Wait for call from above” (top left hand corner) and the receiver is in state “Wait for 0 from below”. The scenarios for corrupted data and corrupted ACK are shown in Figure 1.

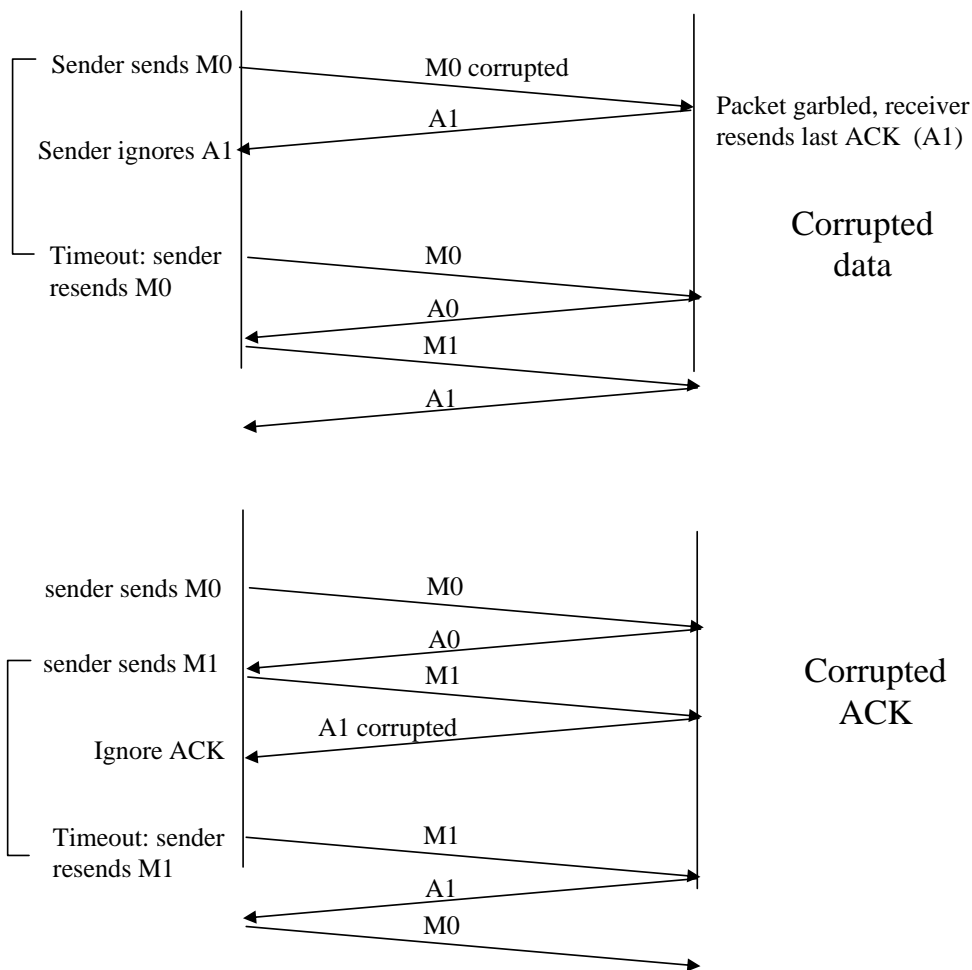


Figure 1: rdt 3.0 scenarios: corrupted data, corrupted ACK

Problem 10

Here, we add a timer, whose value is greater than the known round-trip propagation delay. We add a timeout event to the “Wait for ACK or NAK0” and “Wait for ACK or NAK1” states. If the timeout event occurs, the most recently transmitted packet is retransmitted. Let us see why this protocol will still work with the rdt2.1 receiver.

- Suppose the timeout is caused by a lost data packet, i.e., a packet on the sender-to-receiver channel. In this case, the receiver never received the previous transmission and, from the receiver's viewpoint, if the timeout retransmission is received, it looks *exactly* the same as if the original transmission is being received.
- Suppose now that an ACK is lost. The receiver will eventually retransmit the packet on a timeout. But a retransmission is exactly the same action that if an ACK is garbled. Thus the sender's reaction is the same with a loss, as with a garbled ACK. The rdt 2.1 receiver can already handle the case of a garbled ACK.

Problem 11

If the sending of this message were removed, the sending and receiving sides would deadlock, waiting for an event that would never occur. Here's a scenario:

- Sender sends pkt0, enter the "Wait for ACK0 state", and waits for a packet back from the receiver
- Receiver is in the "Wait for 0 from below" state, and receives a corrupted packet from the sender. Suppose it does not send anything back, and simply re-enters the 'wait for 0 from below' state.

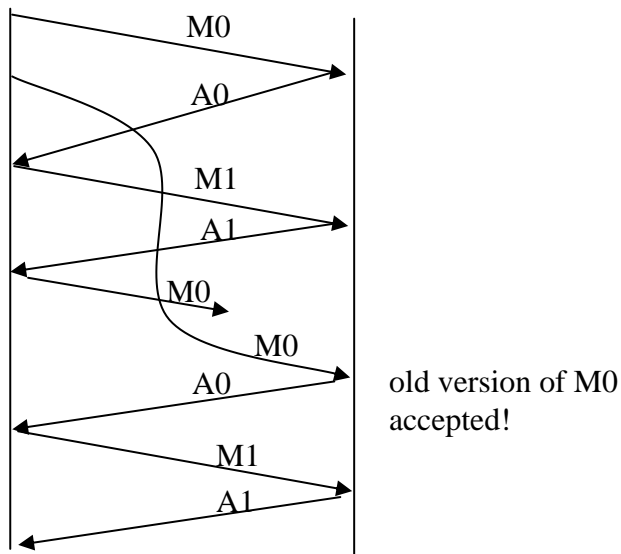
Now, the sender is awaiting an ACK of some sort from the receiver, and the receiver is waiting for a data packet from the sender – a deadlock!

Problem 12

The protocol would still work, since a retransmission would be what would happen if the packet received with errors has actually been lost (and from the receiver standpoint, it never knows which of these events, if either, will occur).

To get at the more subtle issue behind this question, one has to allow for premature timeouts to occur. In this case, if each extra copy of the packet is ACKed and each received extra ACK causes another extra copy of the current packet to be sent, the number of times packet n is sent will increase without bound as n approaches infinity.

Problem 13



Problem 14

In a NAK only protocol, the loss of packet x is only detected by the receiver when packet $x+1$ is received. That is, the receiver receives $x-1$ and then $x+1$, only when $x+1$ is received does the receiver realize that x was missed. If there is a long delay between the transmission of x and the transmission of $x+1$, then it will be a long time until x can be recovered, under a NAK only protocol.

On the other hand, if data is being sent often, then recovery under a NAK-only scheme could happen quickly. Moreover, if errors are infrequent, then NAKs are only occasionally sent (when needed), and ACKs are never sent – a significant reduction in feedback in the NAK-only case over the ACK-only case.

Problem 15

It takes 12 microseconds (or 0.012 milliseconds) to send a packet, as $1500 \times 8 / 10^9 = 12$ microseconds. In order for the sender to be busy 98 percent of the time, we must have

$$util = 0.98 = (0.012n) / 30.012$$

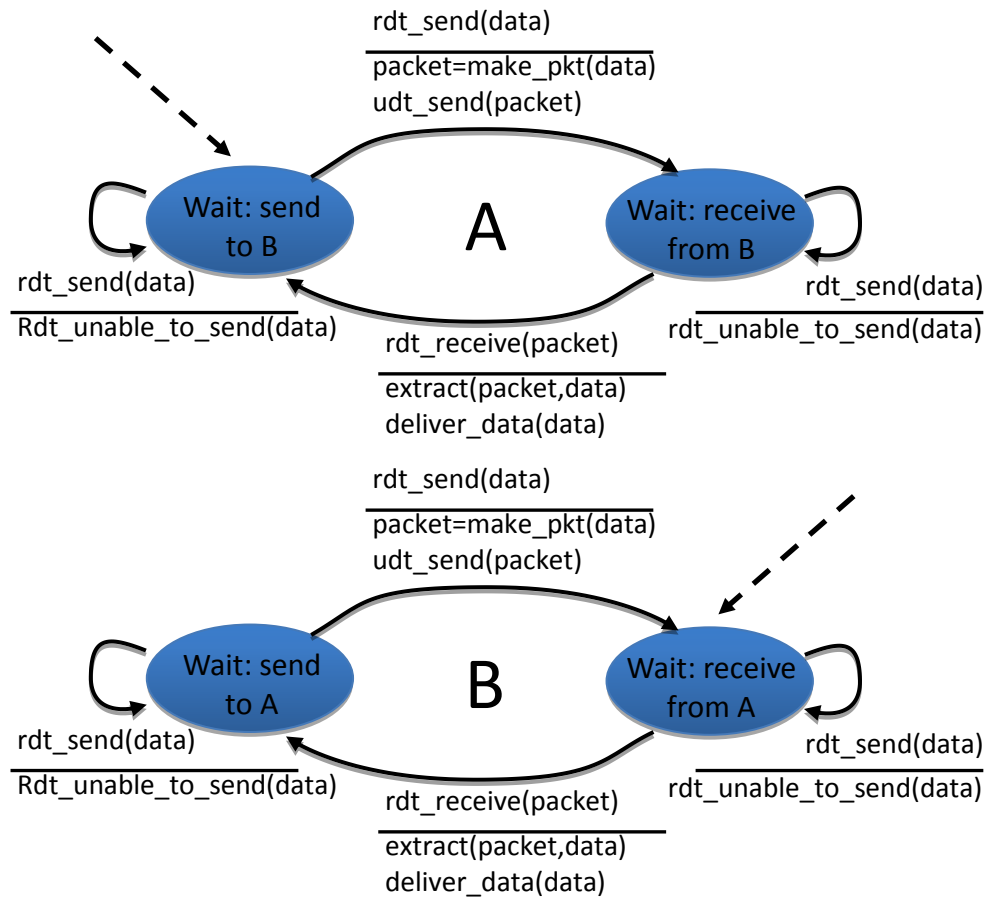
or n approximately 2451 packets.

Problem 16

Yes. This actually causes the sender to send a number of pipelined data into the channel.

Yes. Here is one potential problem. If data segments are lost in the channel, then the sender of rdt 3.0 won't re-send those segments, unless there are some additional mechanism in the application to recover from loss.

Problem 17



Problem 18

In our solution, the sender will wait until it receives an ACK for a pair of messages (seqnum and seqnum+1) before moving on to the next pair of messages. Data packets have a data field and carry a two-bit sequence number. That is, the valid sequence numbers are 0, 1, 2, and 3. (Note: you should think about why a 1-bit sequence number space of 0, 1 only would not work in the solution below.) ACK messages carry the sequence number of the data packet they are acknowledging.

The FSM for the sender and receiver are shown in Figure 2. Note that the sender state records whether (i) no ACKs have been received for the current pair, (ii) an ACK for seqnum (only) has been received, or an ACK for seqnum+1 (only) has been received. In this figure, we assume that the seqnum is initially 0, and that the sender has sent the first

two data messages (to get things going). A timeline trace for the sender and receiver recovering from a lost packet is shown below:

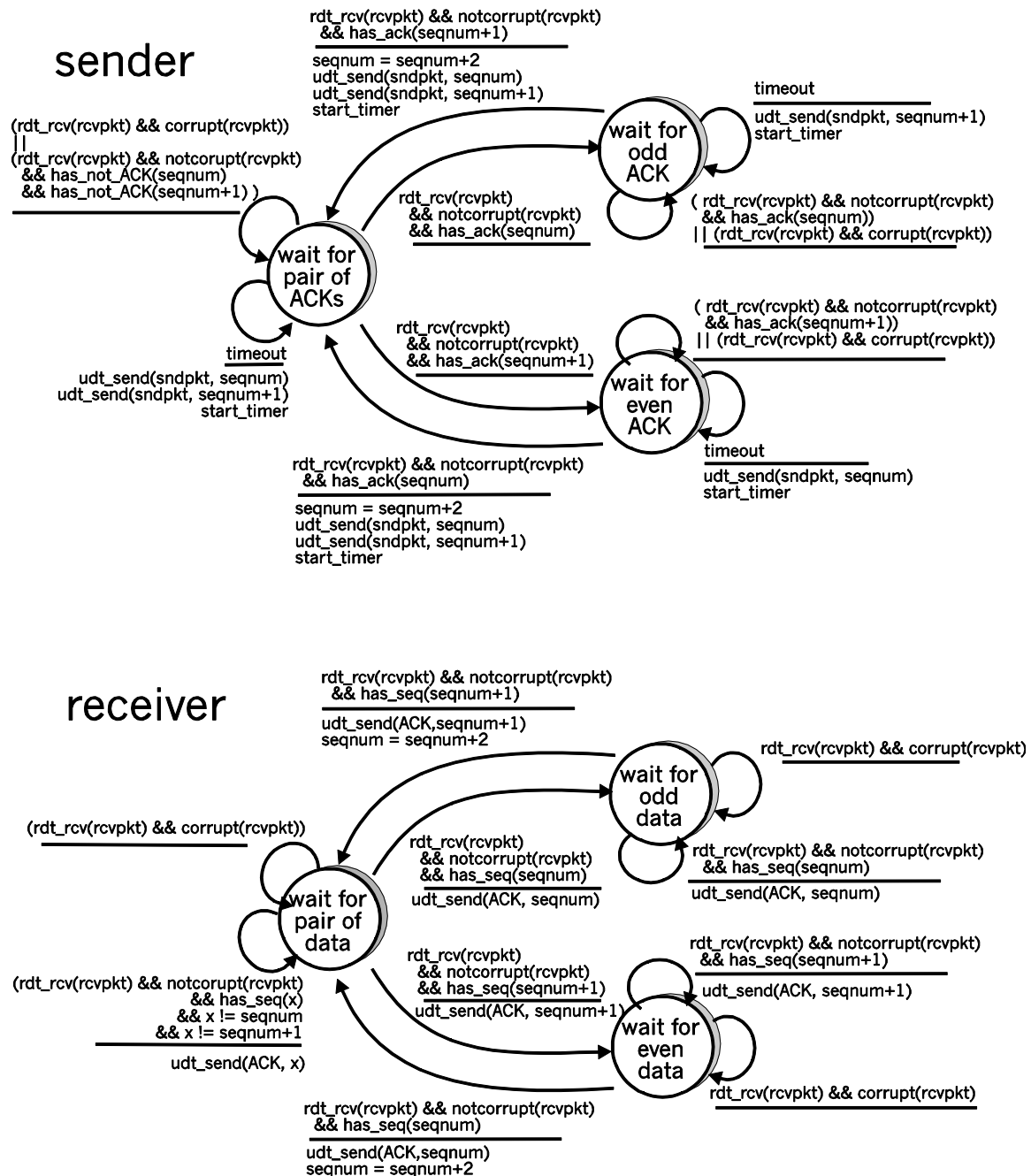


Figure 2: Sender and receiver for Problem (3.18)

Sender

Receiver

```
make pair (0,1)
send packet 0
```

Packet 0 drops
send packet 1

receive ACK 1
(timeout)
resend packet 0

receive ACK 0

receive packet 1
buffer packet 1
send ACK 1

receive packet 0
deliver pair (0,1)
send ACK 0

Problem 19

This problem is a variation on the simple stop and wait protocol (rdt3.0). Because the channel may lose messages and because the sender may resend a message that one of the receivers has already received (either because of a premature timeout or because the other receiver has yet to receive the data correctly), sequence numbers are needed. As in rdt3.0, a 0-bit sequence number will suffice here.

The sender and receiver FSM are shown in Figure 3. In this problem, the sender state indicates whether the sender has received an ACK from B (only), from C (only) or from neither C nor B. The receiver state indicates which sequence number the receiver is waiting for.

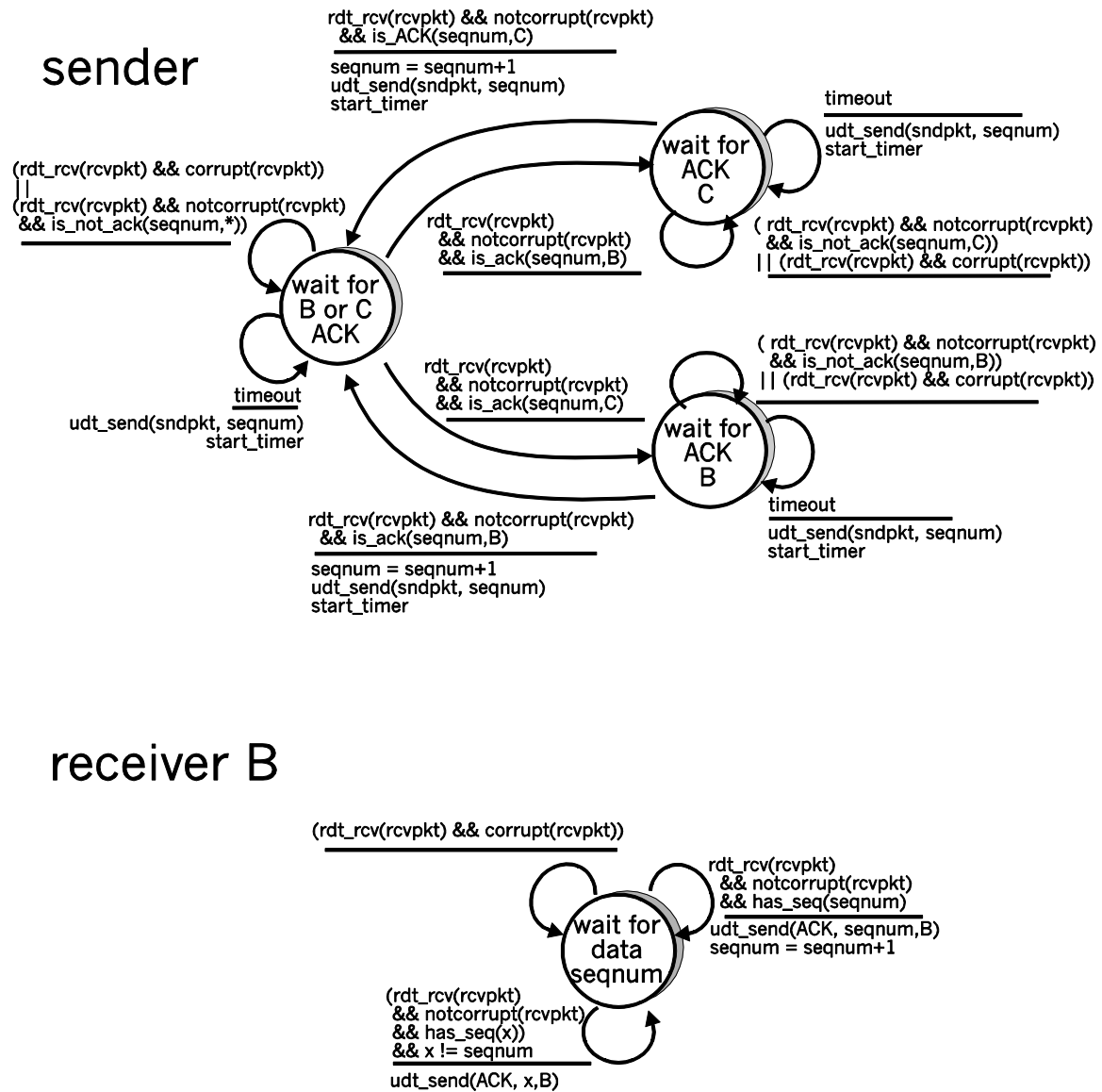


Figure 3. Sender and receiver for Problem 3.19(Problem 19)

Problem 20

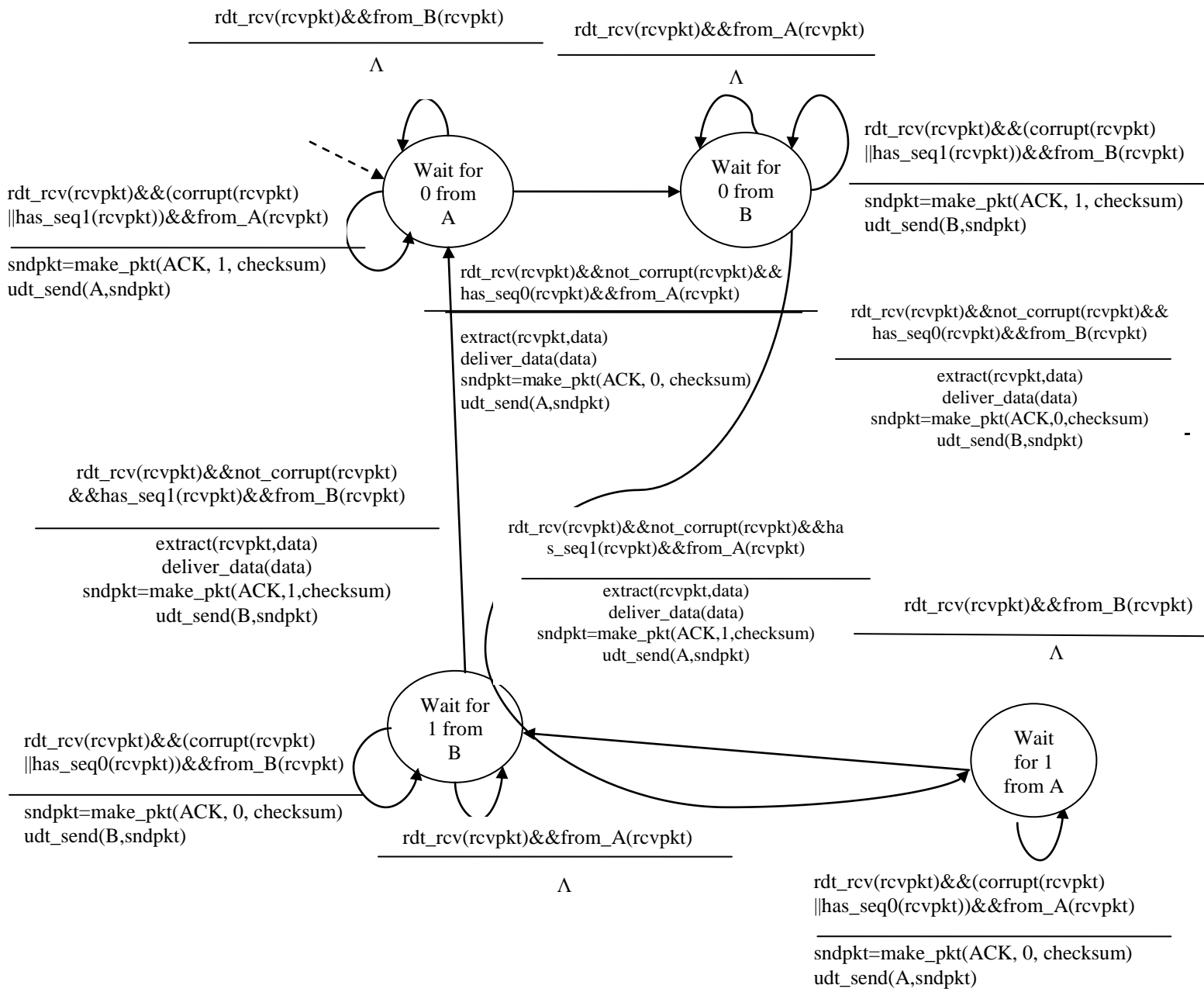


Figure 4: Receiver side FSM for 3.18

Sender

The sender side FSM is exactly same as given in Figure 3.15 in text

Problem 21

Because the A-to-B channel can lose request messages, A will need to timeout and retransmit its request messages (to be able to recover from loss). Because the channel delays are variable and unknown, it is possible that A will send duplicate requests (i.e., resend a request message that has already been received by B). To be able to detect duplicate request messages, the protocol will use sequence numbers. A 1-bit sequence number will suffice for a stop-and-wait type of request/response protocol.

A (the requestor) has 4 states:

- **“Wait for Request 0 from above.”** Here the requestor is waiting for a call from above to request a unit of data. When it receives a request from above, it sends a request message, R0, to B, starts a timer and makes a transition to the “Wait for D0” state. When in the “Wait for Request 0 from above” state, A ignores anything it receives from B.
- **“Wait for D0”.** Here the requestor is waiting for a D0 data message from B. A timer is always running in this state. If the timer expires, A sends another R0 message, restarts the timer and remains in this state. If a D0 message is received from B, A stops the time and transits to the “Wait for Request 1 from above” state. If A receives a D1 data message while in this state, it is ignored.
- **“Wait for Request 1 from above.”** Here the requestor is again waiting for a call from above to request a unit of data. When it receives a request from above, it sends a request message, R1, to B, starts a timer and makes a transition to the “Wait for D1” state. When in the “Wait for Request 1 from above” state, A ignores anything it receives from B.
- **“Wait for D1”.** Here the requestor is waiting for a D1 data message from B. A timer is always running in this state. If the timer expires, A sends another R1 message, restarts the timer and remains in this state. If a D1 message is received from B, A stops the timer and transits to the “Wait for Request 0 from above” state. If A receives a D0 data message while in this state, it is ignored.

The data supplier (B) has only two states:

- **“Send D0.”** In this state, B continues to respond to received R0 messages by sending D0, and then remaining in this state. If B receives a R1 message, then it knows its D0 message has been received correctly. It thus discards this D0 data (since it has been received at the other side) and then transits to the “Send D1” state, where it will use D1 to send the next requested piece of data.

- **“Send D1.”** In this state, B continues to respond to received R1 messages by sending D1, and then remaining in this state. If B receives a R1 message, then it knows its D1 message has been received correctly and thus transits to the “Send D1” state.

Problem 22

- Here we have a window size of $N=3$. Suppose the receiver has received packet $k-1$, and has ACKed that and all other preceding packets. If all of these ACK's have been received by sender, then sender's window is $[k, k+N-1]$. Suppose next that none of the ACKs have been received at the sender. In this second case, the sender's window contains $k-1$ and the N packets up to and including $k-1$. The sender's window is thus $[k-N, k-1]$. By these arguments, the sender's window is of size 3 and begins somewhere in the range $[k-N, k]$.
- If the receiver is waiting for packet k , then it has received (and ACKed) packet $k-1$ and the $N-1$ packets before that. If none of those N ACKs have been yet received by the sender, then ACK messages with values of $[k-N, k-1]$ may still be propagating back. Because the sender has sent packets $[k-N, k-1]$, it must be the case that the sender has already received an ACK for $k-N-1$. Once the receiver has sent an ACK for $k-N-1$ it will never send an ACK that is less than $k-N-1$. Thus the range of in-flight ACK values can range from $k-N-1$ to $k-1$.

Problem 23

In order to avoid the scenario of Figure 3.27, we want to avoid having the leading edge of the receiver's window (i.e., the one with the “highest” sequence number) wrap around in the sequence number space and overlap with the trailing edge (the one with the “lowest” sequence number in the sender's window). That is, the sequence number space must be large enough to fit the entire receiver window and the entire sender window without this overlap condition. So - we need to determine how large a range of sequence numbers can be covered at any given time by the receiver and sender windows.

Suppose that the lowest-sequence number that the receiver is waiting for is packet m . In this case, its window is $[m, m+w-1]$ and it has received (and ACKed) packet $m-1$ and the $w-1$ packets before that, where w is the size of the window. If none of those w ACKs have been yet received by the sender, then ACK messages with values of $[m-w, m-1]$ may still be propagating back. If no ACKs with these ACK numbers have been received by the sender, then the sender's window would be $[m-w, m-1]$.

Thus, the lower edge of the sender's window is $m-w$, and the leading edge of the receiver's window is $m+w-1$. In order for the leading edge of the receiver's window to not overlap with the trailing edge of the sender's window, the sequence number space must

thus be big enough to accommodate $2w$ sequence numbers. That is, the sequence number space must be at least twice as large as the window size, $k \geq 2w$.

Problem 24

- a) True. Suppose the sender has a window size of 3 and sends packets 1, 2, 3 at t_0 . At t_1 ($t_1 > t_0$) the receiver ACKS 1, 2, 3. At t_2 ($t_2 > t_1$) the sender times out and resends 1, 2, 3. At t_3 the receiver receives the duplicates and re-acknowledges 1, 2, 3. At t_4 the sender receives the ACKs that the receiver sent at t_1 and advances its window to 4, 5, 6. At t_5 the sender receives the ACKs 1, 2, 3 the receiver sent at t_2 . These ACKs are outside its window.
- b) True. By essentially the same scenario as in (a).
- c) True.
- d) True. Note that with a window size of 1, SR, GBN, and the alternating bit protocol are functionally equivalent. The window size of 1 precludes the possibility of out-of-order packets (within the window). A cumulative ACK is just an ordinary ACK in this situation, since it can only refer to the single packet within the window.

Problem 25

- a) Consider sending an application message over a transport protocol. With TCP, the application writes data to the connection send buffer and TCP will grab bytes without necessarily putting a single message in the TCP segment; TCP may put more or less than a single message in a segment. UDP, on the other hand, encapsulates in a segment whatever the application gives it; so that, if the application gives UDP an application message, this message will be the payload of the UDP segment. Thus, with UDP, an application has more control of what data is sent in a segment.
- b) With TCP, due to flow control and congestion control, there may be significant delay from the time when an application writes data to its send buffer until when the data is given to the network layer. UDP does not have delays due to flow control and congestion control.

Problem 26

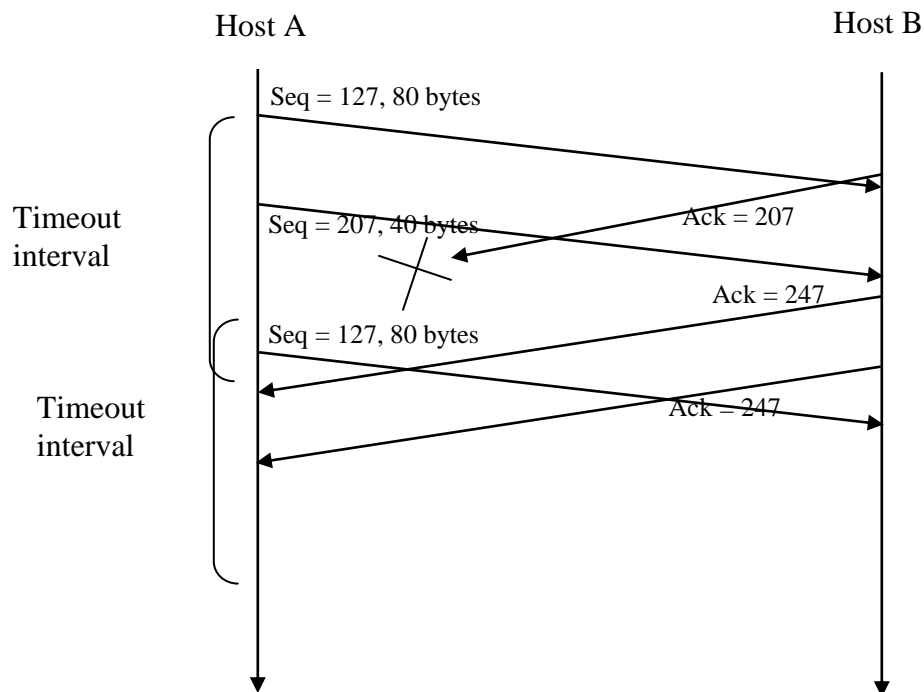
There are $2^{32} = 4,294,967,296$ possible sequence numbers.

- a) The sequence number does not increment by one with each segment. Rather, it increments by the number of bytes of data sent. So the size of the MSS is irrelevant -- the maximum size file that can be sent from A to B is simply the number of bytes representable by $2^{32} \approx 4.19$ Gbytes.

- b) The number of segments is $\left\lceil \frac{2^{32}}{536} \right\rceil = 8,012,999$. 66 bytes of header get added to each segment giving a total of 528,857,934 bytes of header. The total number of bytes transmitted is $2^{32} + 528,857,934 = 4.824 \times 10^9$ bytes.
- Thus it would take 249 seconds to transmit the file over a 155-Mbps link.

Problem 27

- In the second segment from Host A to B, the sequence number is 207, source port number is 302 and destination port number is 80.
- If the first segment arrives before the second, in the acknowledgement of the first arriving segment, the acknowledgement number is 207, the source port number is 80 and the destination port number is 302.
- If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, the acknowledgement number is 127, indicating that it is still waiting for bytes 127 and onwards.
-



Problem 28

Since the link capacity is only 100 Mbps, so host A's sending rate can be at most 100Mbps. Still, host A sends data into the receive buffer faster than Host B can remove data from the buffer. The receive buffer fills up at a rate of roughly 40Mbps. When the buffer is full, Host B signals to Host A to stop sending data by setting $RcvWindow = 0$. Host A then stops sending until it receives a TCP segment with $RcvWindow > 0$. Host A will thus repeatedly stop and start sending as a function of the $RcvWindow$ values it

receives from Host B. On average, the long-term rate at which Host A sends data to Host B as part of this connection is no more than 60Mbps.

Problem 29

- a) The server uses special initial sequence number (that is obtained from the hash of source and destination IPs and ports) in order to defend itself against SYN FLOOD attack.
- b) No, the attacker cannot create half-open or fully open connections by simply sending and ACK packet to the target. Half-open connections are not possible since a server using SYN cookies does not maintain connection variables and buffers for any connection before full connections are established. For establishing fully open connections, an attacker should know the special initial sequence number corresponding to the (spoofed) source IP address from the attacker. This sequence number requires the "secret" number that each server uses. Since the attacker does not know this secret number, she cannot guess the initial sequence number.
- c) No, the sever can simply add in a time stamp in computing those initial sequence numbers and choose a time to live value for those sequence numbers, and discard expired initial sequence numbers even if the attacker replay them.

Problem 30

- a) If timeout values are fixed, then the senders may timeout prematurely. Thus, some packets are re-transmitted even they are not lost.
- b) If timeout values are estimated (like what TCP does), then increasing the buffer size certainly helps to increase the throughput of that router. But there might be one potential problem. Queuing delay might be very large, similar to what is shown in Scenario 1.

Problem 31

$$EstimatedRTT = xSampleRTT + (1 - x)EstimatedRTT$$

$$DevRTT = y|SampleRTT - EstimatedRTT| + (1 - y)DevRTT$$

$$TimeoutInterval = EstimatedRTT + 4 * DevRTT$$

After obtaining first sampleRTT is

$$\begin{aligned} EstimatedRTT &= 0.125 * 106 + 0.875 * 100 \\ &= 100.75ms \end{aligned}$$

$$\begin{aligned} DevRTT &= 0.25 * |106 - 100.75| + 0.75 * 5 \\ &= 5.06ms \end{aligned}$$

$$TimeoutInterval = 100.75 + 4 * 5.06$$

$$= 120.99ms$$

After obtaining second sample $RTT = 120ms$:

$$\begin{aligned} EstimatedRTT &= 0.125 * 120 + 0.875 * 100.75 \\ &= 103.15ms \end{aligned}$$

$$\begin{aligned} DevRTT &= 0.25 * |120 - 103.15| + 0.75 * 5.06 \\ &= 8ms \end{aligned}$$

$$\begin{aligned} TimeoutInterval &= 103.15 + 4 * 8 \\ &= 135.15ms \end{aligned}$$

After obtaining Third sample $RTT = 140ms$:

$$\begin{aligned} EstimatedRTT &= 0.125 * 140 + 0.875 * 103.15 \\ &= 107.76ms \end{aligned}$$

$$\begin{aligned} DevRTT &= 0.25 * |140 - 107.76| + 0.75 * 8 \\ &= 14.06ms \end{aligned}$$

$$\begin{aligned} TimeoutInterval &= 107.76 + 4 * 14.06 \\ &= 164ms \end{aligned}$$

After obtaining fourth sample $RTT = 90ms$:

$$\begin{aligned} EstimatedRTT &= 0.125 * 90 + 0.875 * 107.76 \\ &= 105.54ms \end{aligned}$$

$$\begin{aligned} DevRTT &= 0.25 * |90 - 105.54| + 0.75 * 14.06 \\ &= 14.42ms \end{aligned}$$

$$\begin{aligned} TimeoutInterval &= 105.54 + 4 * 14.42 \\ &= 163.22ms \end{aligned}$$

After obtaining fifth sample $RTT = 115ms$:

$$\begin{aligned} EstimatedRTT &= 0.125 * 115 + 0.875 * 105.54 \\ &= 106.71ms \end{aligned}$$

$$\begin{aligned} DevRTT &= 0.25 * |115 - 106.71| + 0.75 * 14.42 \\ &= 12.88ms \end{aligned}$$

$$\begin{aligned} TimeoutInterval &= 106.71 + 4 * 12.88 \\ &= 158.23ms \end{aligned}$$

Problem 32

a)

Denote $EstimatedRTT^{(n)}$ for the estimate after the n th sample.

$$\begin{aligned} EstimatedRTT^{(4)} &= xSampleRTT_1 + \\ &\quad (1-x)[xSampleRTT_2 + \\ &\quad (1-x)[xSampleRTT_3 + (1-x)SampleRTT_4]] \end{aligned}$$

$$\begin{aligned}
&= x\text{SampleRTT}_1 + (1-x)x\text{SampleRTT}_2 \\
&+ (1-x)^2 x\text{SampleRTT}_3 + (1-x)^3 \text{SampleRTT}_4
\end{aligned}$$

b)

$$\begin{aligned}
\text{EstimatedRTT}^{(n)} &= x \sum_{j=1}^{n-1} (1-x)^{j-1} \text{SampleRTT}_j \\
&+ (1-x)^{n-1} \text{SampleRTT}_n
\end{aligned}$$

c)

$$\begin{aligned}
\text{EstimatedRTT}^{(\infty)} &= \frac{x}{1-x} \sum_{j=1}^{\infty} (1-x)^j \text{SampleRTT}_j \\
&= \frac{1}{9} \sum_{j=1}^{\infty} .9^j \text{SampleRTT}_j
\end{aligned}$$

The weight given to past samples decays exponentially.

Problem 33

Let's look at what could wrong if TCP measures `SampleRTT` for a retransmitted segment. Suppose the source sends packet P1, the timer for P1 expires, and the source then sends P2, a new copy of the same packet. Further suppose the source measures `SampleRTT` for P2 (the retransmitted packet). Finally suppose that shortly after transmitting P2 an acknowledgment for P1 arrives. The source will mistakenly take this acknowledgment as an acknowledgment for P2 and calculate an incorrect value of `SampleRTT`.

Let's look at what could be wrong if TCP measures `SampleRTT` for a retransmitted segment. Suppose the source sends packet P1, the timer for P1 expires, and the source then sends P2, a new copy of the same packet. Further suppose the source measures `SampleRTT` for P2 (the retransmitted packet). Finally suppose that shortly after transmitting P2 an acknowledgment for P1 arrives. The source will mistakenly take this acknowledgment as an acknowledgment for P2 and calculate an incorrect value of `SampleRTT`.

Problem 34

At any given time t , $\text{SendBase} - 1$ is the sequence number of the last byte that the sender knows has been received correctly, and in order, at the receiver. The actually last byte received (correctly and in order) at the receiver at time t may be greater if there are acknowledgements in the pipe. Thus

$$\text{SendBase} - 1 \leq \text{LastByteRcvd}$$

Problem 35

When, at time t , the sender receives an acknowledgement with value y , the sender knows for sure that the receiver has received everything up through $y-1$. The actual last byte received (correctly and in order) at the receiver at time t may be greater if $y \leq \text{SendBase}$ or if there are other acknowledgements in the pipe. Thus

$$y - 1 \leq \text{LastByteRcvd}$$

Problem 36

Suppose packets n , $n+1$, and $n+2$ are sent, and that packet n is received and ACKed. If packets $n+1$ and $n+2$ are reordered along the end-to-end-path (i.e., are received in the order $n+2$, $n+1$) then the receipt of packet $n+2$ will generate a duplicate ack for n and would trigger a retransmission under a policy of waiting only for second duplicate ACK for retransmission. By waiting for a triple duplicate ACK, it must be the case that *two* packets after packet n are correctly received, while $n+1$ was not received. The designers of the triple duplicate ACK scheme probably felt that waiting for two packets (rather than 1) was the right tradeoff between triggering a quick retransmission when needed, but not retransmitting prematurely in the face of packet reordering.

Problem 37

a) GoBackN:

A sends 9 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2, 3, 4, and 5.

B sends 8 ACKs. They are 4 ACKS with sequence number 1, and 4 ACKS with sequence numbers 2, 3, 4, and 5.

Selective Repeat:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2.

B sends 5 ACKs. They are 4 ACKS with sequence number 1, 3, 4, 5. And there is one ACK with sequence number 2.

TCP:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2.

B sends 5 ACKs. They are 4 ACKs with sequence number 2. There is one ACK with sequence number 6. Note that TCP always send an ACK with expected sequence number.

- b) TCP. This is because TCP uses fast retransmit without waiting until time out.

Problem 38

Yes, the sending rate is always roughly $cwnd/RTT$.

Problem 39

If the arrival rate increases beyond $R/2$ in Figure 3.46(b), then the total arrival rate to the queue exceeds the queue's capacity, resulting in increasing loss as the arrival rate increases. When the arrival rate equals $R/2$, 1 out of every three packets that leaves the queue is a retransmission. With increased loss, even a larger fraction of the packets leaving the queue will be retransmissions. Given that the maximum departure rate from the queue for one of the sessions is $R/2$, and given that a third or more will be transmissions as the arrival rate increases, the throughput of successfully deliver data can not increase beyond λ_{out} . Following similar reasoning, if half of the packets leaving the queue are retransmissions, and the maximum rate of output packets per session is $R/2$, then the maximum value of λ_{out} is $(R/2)/2$ or $R/4$.

Problem 40

- a) TCP slowstart is operating in the intervals [1,6] and [23,26]
- b) TCP congestion avoidance is operating in the intervals [6,16] and [17,22]
- c) After the 16th transmission round, packet loss is recognized by a triple duplicate ACK. If there was a timeout, the congestion window size would have dropped to 1.
- d) After the 22nd transmission round, segment loss is detected due to timeout, and hence the congestion window size is set to 1.
- e) The threshold is initially 32, since it is at this window size that slow start stops and congestion avoidance begins.
- f) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 16, the congestion windows size is 42. Hence the threshold is 21 during the 18th transmission round.
- g) The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 22, the congestion windows size is 29. Hence the threshold is 14 (taking lower floor of 14.5) during the 24th transmission round.
- h) During the 1st transmission round, packet 1 is sent; packet 2-3 are sent in the 2nd transmission round; packets 4-7 are sent in the 3rd transmission round; packets 8-15 are sent in the 4th transmission round; packets 16-31 are sent in the 5th transmission

round; packets 32-63 are sent in the 6th transmission round; packets 64 – 96 are sent in the 7th transmission round. Thus packet 70 is sent in the 7th transmission round.

- i) The threshold will be set to half the current value of the congestion window (8) when the loss occurred and congestion window will be set to the new threshold value + 3 MSS . Thus the new values of the threshold and window will be 4 and 7 respectively.
- j) threshold is 21, and congestion window size is 1.
- k) round 17, 1 packet; round 18, 2 packets; round 19, 4 packets; round 20, 8 packets; round 21, 16 packets; round 22, 21 packets. So, the total number is 52.

Problem 41

Refer to Figure 5. In Figure 5(a), the ratio of the linear decrease on loss between connection 1 and connection 2 is the same - as ratio of the linear increases: unity. In this case, the throughputs never move off of the AB line segment. In Figure 5(b), the ratio of the linear decrease on loss between connection 1 and connection 2 is 2:1. That is, whenever there is a loss, connection 1 decreases its window by twice the amount of connection 2. We see that eventually, after enough losses, and subsequent increases, that connection 1's throughput will go to 0, and the full link bandwidth will be allocated to connection 2.

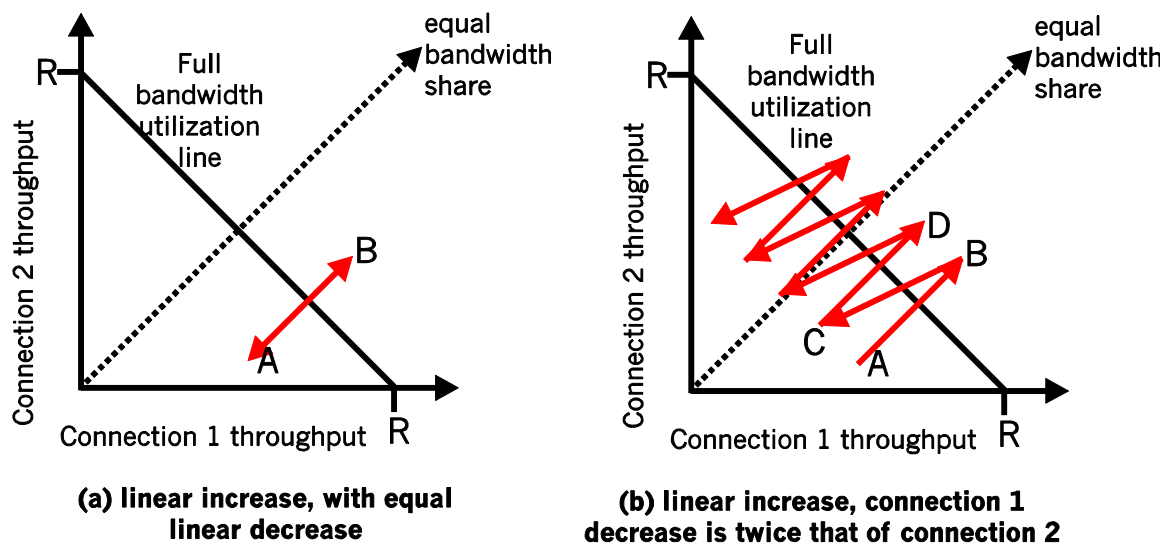


Figure 5: Lack of TCP convergence with linear increase, linear decrease

Problem 42

If TCP were a stop-and-wait protocol, then the doubling of the time out interval would suffice as a congestion control mechanism. However, TCP uses pipelining (and is therefore not a stop-and-wait protocol), which allows the sender to have multiple outstanding unacknowledged segments. The doubling of the timeout interval does not prevent a TCP sender from sending a large number of first-time-transmitted packets into the network, even when the end-to-end path is highly congested. Therefore a congestion-

control mechanism is needed to stem the flow of “data received from the application above” when there are signs of network congestion.

Problem 43

In this problem, there is no danger in overflowing the receiver since the receiver’s receive buffer can hold the entire file. Also, because there is no loss and acknowledgements are returned before timers expire, TCP congestion control does not throttle the sender. However, the process in host A will not continuously pass data to the socket because the send buffer will quickly fill up. Once the send buffer becomes full, the process will pass data at an average rate or $R \ll S$.

Problem 44

- a) It takes 1 RTT to increase CongWin to 6 MSS; 2 RTTs to increase to 7 MSS; 3 RTTs to increase to 8 MSS; 4 RTTs to increase to 9 MSS; 5 RTTs to increase to 10 MSS; 6 RTTs to increase to 11 MSS; and 7 RTTs to increase to 12MSS.
- b) In the first RTT 5 MSS was sent; in the second RTT 6 MSS was sent; in the third RTT 7 MSS was sent; in the fourth RTT 8 MSS was sent; in the fifth RTT, 9 MSS was sent; and in the sixth RTT, 10 MSS was sent. Thus, up to time 6 RTT, $5+6+7+8+9+10 = 45$ MSS were sent (and acknowledged). Thus, we can say that the average throughput up to time 6 RTT was $(45 \text{ MSS})/(6 \text{ RTT}) = 7.5 \text{ MSS/RTT}$.

Problem 45

- a) The loss rate, L , is the ratio of the number of packets lost over the number of packets sent. In a cycle, 1 packet is lost. The number of packets sent in a cycle is

$$\begin{aligned}
 \frac{W}{2} + \left(\frac{W}{2} + 1\right) + \cdots + W &= \sum_{n=0}^{W/2} \left(\frac{W}{2} + n\right) \\
 &= \left(\frac{W}{2} + 1\right) \frac{W}{2} + \sum_{n=0}^{W/2} n \\
 &= \left(\frac{W}{2} + 1\right) \frac{W}{2} + \frac{W/2(W/2 + 1)}{2} \\
 &= \frac{W^2}{4} + \frac{W}{2} + \frac{W^2}{8} + \frac{W}{4} \\
 &= \frac{3}{8}W^2 + \frac{3}{4}W
 \end{aligned}$$

Thus the loss rate is

$$L = \frac{1}{\frac{3}{8}W^2 + \frac{3}{4}W}$$

b) For W large, $\frac{3}{8}W^2 \gg \frac{3}{4}W$. Thus $L \approx 8/3W^2$ or $W \approx \sqrt{\frac{8}{3L}}$. From the text, we therefore have

$$\begin{aligned} \text{average throughput} &= \frac{3}{4} \sqrt{\frac{8}{3L}} \cdot \frac{MSS}{RTT} \\ &= \frac{1.22 \cdot MSS}{RTT \cdot \sqrt{L}} \end{aligned}$$

Problem 46

- Let W denote the max window size measured in segments. Then, $W \cdot MSS/RTT = 10\text{Mbps}$, as packets will be dropped if the maximum sending rate exceeds link capacity. Thus, we have $W \cdot 1500 \cdot 8/0.15 = 10 \cdot 10^6$, then W is about 125 segments.
- As congestion window size varies from $W/2$ to W , then the average window size is $0.75W = 94$ (ceiling of 93.75) segments. Average throughput is $94 \cdot 1500 \cdot 8/0.15 = 7.52\text{Mbps}$.
- $94/2 \cdot 0.15 = 7.05$ seconds, as the number of RTTs (that this TCP connections needs in order to increase its window size from $W/2$ to W) is given by $W/2$. Recall the window size increases by one in each RTT.

Problem 47

Let W denote max window size. Let S denote the buffer size. For simplicity, suppose TCP sender sends data packets in a round by round fashion, with each round corresponding to a RTT. If the window size reaches W , then a loss occurs. Then the sender will cut its congestion window size by half, and waits for the ACKs for $W/2$ outstanding packets before it starts sending data segments again. In order to make sure the link always busying sending data, we need to let the link busy sending data in the period $W/(2 \cdot C)$ (this is the time interval where the sender is waiting for the ACKs for the $W/2$ outstanding packets). Thus, S/C must be no less than $W/(2 \cdot C)$, that is, $S \geq W/2$.

Let T_p denote the one-way propagation delay between the sender and the receiver. When the window size reaches the minimum $W/2$ and the buffer is empty, we need to make sure the link is also busy sending data. Thus, we must have $W/2/(2T_p) \geq C$, thus, $W/2 \geq C \cdot 2T_p$.

Thus, $S \geq C \cdot 2T_p$.

Problem 48

- Let W denote the max window size. Then, $W \cdot \text{MSS} / \text{RTT} = 10\text{Gbps}$, as packets will be dropped if maximum sending rate reaches link capacity. Thus, we have $W \cdot 1500 \cdot 8 / 0.15 = 10 \cdot 10^9$, then $W = 125000$ segments.
- As congestion window size varies from $W/2$ to W , then the average window size is $0.75W = 93750$ segments. Average throughput is $93750 \cdot 1500 \cdot 8 / 0.1 = 7.5\text{Gbps}$.
- $93750/2 \cdot 0.15 / 60 = 117$ minutes. In order to speed up the window increase process, we can increase the window size by a much larger value, instead of increasing window size only by one in each RTT. Some protocols are proposed to solve this problem, such as ScalableTCP or HighSpeed TCP.

Problem 49

As TCP's average throughput B is given by $B = \frac{1.22 \cdot \text{MSS}}{\text{RTT} \cdot \sqrt{L}}$, so we know that,

$$L = (1.22 \cdot \text{MSS} / (B \cdot \text{RTT}))^2$$

Since between two consecutive packet losses, there are $1/L$ packets sent by the TCP sender, thus, $T = (1/L) \cdot \text{MSS} / B$. Thus, we find that $T = B \cdot \text{RTT}^2 / (1.22^2 \cdot \text{MSS})$, that is, T is a function of B .

Problem 50

- The key difference between C1 and C2 is that C1's RTT is only half of that of C2. Thus C1 adjusts its window size after 50 msec, but C2 adjusts its window size after 100 msec. Assume that whenever a loss event happens, C1 receives it after 50msec and C2 receives it after 100msec. We further have the following simplified model of TCP. After each RTT, a connection determines if it should increase window size or not. For C1, we compute the average total sending rate in the link in the previous 50 msec. If that rate exceeds the link capacity, then we assume that C1 detects loss and reduces its window size. But for C2, we compute the average total sending rate in the link in the previous 100msec. If that rate exceeds the link capacity, then we assume that C2 detects loss and reduces its window size. Note that it is possible that the average sending rate in last 50msec is higher than the link capacity, but the average sending rate in last 100msec is smaller than or equal to the link capacity, then in this case, we assume that C1 will experience loss event but C2 will not.

The following table describes the evolution of window sizes and sending rates based on the above assumptions.

	C1		C2	
Time (msec)	Window Size (num. of segments sent in next 50msec)	Average data sending rate (segments per second, =Window/0.05)	Window Size(num. of segments sent in next 100msec)	Average data sending rate (segments per second, =Window/0.1)
0	10	200 (in [0-50]msec)	10	100 (in [0-50]msec)

50	5 (decreases window size as the avg. total sending rate to the link in last 50msec is $300 = 200 + 100$)	100 (in [50-100]msec)		100 (in [50-100]msec)
100	2 (decreases window size as the avg. total sending rate to the link in last 50msec is $200 = 100 + 100$)	40	5 (decreases window size as the avg. total sending rate to the link in last 100msec is $250 = (200 + 100)/2 + (100 + 100)/2$)	50
150	1 (decreases window size as the avg. total sending rate to the link in last 50msec is $90 = (40 + 50)$)	20		50
200	1 (no further decrease, as window size is already 1)	20	2 (decreases window size as the avg. total sending rate to the link in last 100msec is $80 = (40 + 20)/2 + (50 + 50)/2$)	20
250	1 (no further decrease, as window size	20		20

	is already 1)			
300	1 (no further decrease, as window size is already 1)	20	1 (decreases window size as the avg. total sending rate to the link in last 100msec is $40 = (20+20)/2 + (20+20)/2$)	10
350	2	40		10
400	1	20	1	10
450	2	40		10
500	1 (decreases window size as the avg. total sending rate to the link in last 50msec is $50 = (40+10)$)	20	1	10
550	2	40		10
600	1	20	1	10
650	2	40		10
700	1	20	1	10
750	2	40		10
800	1	20	1	10
850	2	40		10
900	1	20	1	10
950	2	40		10
1000	1	20	1	10

Based on the above table, we find that after 1000 msec, C1's and C2's window sizes are 1 segment each.

- b) No. In the long run, C1's bandwidth share is roughly twice as that of C2's, because C1 has shorter RTT, only half of that of C2, so C1 can adjust its window size twice as fast as C2. If we look at the above table, we can see a cycle every 200msec, e.g. from 850msec to 1000msec, inclusive. Within a cycle, the sending rate of C1 is $(40+20+40+20) = 120$, which is thrice as large as the sending of C2 given by $(10+10+10+10) = 40$.

Problem 51

- a) Similarly as in last problem, we can compute their window sizes over time in the following table. Both C1 and C2 have the same window size 2 after 2200msec.

Time (msec)	C1		C2	
	Window Size (num. of segments sent in next 100msec)	Data sending speed (segments per second, $=\text{Window}/0.1$)	Window Size (num. of segments sent in next 100msec)	Data sending speed (segments per second, $=\text{Window}/0.1$)
0	15	150 (in [0-100]msec)	10	100 (in [0-100]msec)
100	7	70	5	50
200	3	30	2	20
300	1	10	1	10
400	2	20	2	20
500	1	10	1	10
600	2	20	2	20
700	1	10	1	10
800	2	20	2	20
900	1	10	1	10
1000	2	20	2	20
1100	1	10	1	10
1200	2	20	2	20
1300	1	10	1	10
1400	2	20	2	20
1500	1	10	1	10
1600	2	20	2	20
1700	1	10	1	10
1800	2	20	2	20
1900	1	10	1	10
2000	2	20	2	20
2100	1	10	1	10
2200	2	20	2	20

- b) Yes, this is due to the AIMD algorithm of TCP and that both connections have the same RTT.
- c) Yes, this can be seen clearly from the above table. Their max window size is 2.
- d) No, this synchronization won't help to improve link utilization, as these two connections act as a single connection oscillating between min and max window size. Thus, the link is not fully utilized (recall we assume this link has no buffer). One possible way to break the synchronization is to add a finite buffer to the link and randomly drop packets in the buffer before buffer overflow. This will cause different connections cut their window sizes at different times. There are many AQM (Active Queue Management) techniques to do that, such as RED (Random Early Detect), PI (Proportional and Integral AQM), AVQ (Adaptive Virtual Queue), and REM (Random Exponential Marking), etc.

Problem 52

Note that W represents the maximum window size.

First we can find the total number of segments sent out during the interval when TCP changes its window size from $W/2$ up to and include W . This is given by:

$$S = W/2 + (W/2)*(1+\alpha) + (W/2)*(1+\alpha)^2 + (W/2)*(1+\alpha)^3 + \dots + (W/2)*(1+\alpha)^k$$

We find $k = \log_{(1+\alpha)} 2$, then $S = W*(2\alpha+1)/(2\alpha)$.

Loss rate L is given by:

$$L = 1/S = (2\alpha) / (W*(2\alpha+1)).$$

The time that TCP takes to increase its window size from $W/2$ to W is given by:

$$k*RTT = (\log_{(1+\alpha)} 2) * RTT,$$

which is clearly independent of TCP's average throughput.

Note, TCP's average throughput is given by:

$$B = MSS * S / ((k+1)*RTT) = MSS / (L*(k+1)*RTT).$$

Note that this is different from TCP which has average throughput: $B = \frac{1.22 \cdot MSS}{RTT \cdot \sqrt{L}}$,

where the square root of L appears in the denominator.

Problem 53

Let's assume 1500-byte packets and a 100 ms round-trip time. From the TCP throughput

equation $B = \frac{1.22 \cdot MSS}{RTT \cdot \sqrt{L}}$, we have

$$10 \text{ Gbps} = 1.22 * (1500*8 \text{ bits}) / (.1 \text{ sec} * \text{sqrt}(L)), \text{ or}$$

$$\text{sqrt}(L) = 14640 \text{ bits} / (10^9 \text{ bits}) = 0.00001464, \text{ or}$$

$$L = 2.14 * 10^{-10}$$

Problem 54

An advantage of using the earlier values of $cwnd$ and $ssthresh$ at t_2 is that TCP would not have to go through slow start and congestion avoidance to ramp up to the throughput value obtained at t_1 . A disadvantage of using these values is that they may be no longer accurate. In particular, if the path has become more congested between t_1 and t_2 , the sender will send a large window's worth of segments into an already (more) congested path.

Problem 55

a) The server will send its response to Y.

- b) The server can be certain that the client is indeed at Y. If it were at some other address spoofing Y, the SYNACK would have been sent to the address Y, and the TCP in that host would not send the TCP ACK segment back. Even if the attacker were to send an appropriately timed TCP ACK segment, it would not know the correct server sequence number (since the server uses random initial sequence numbers.)

Problem 56

- a) Referring to the figure below, we see that the total delay is

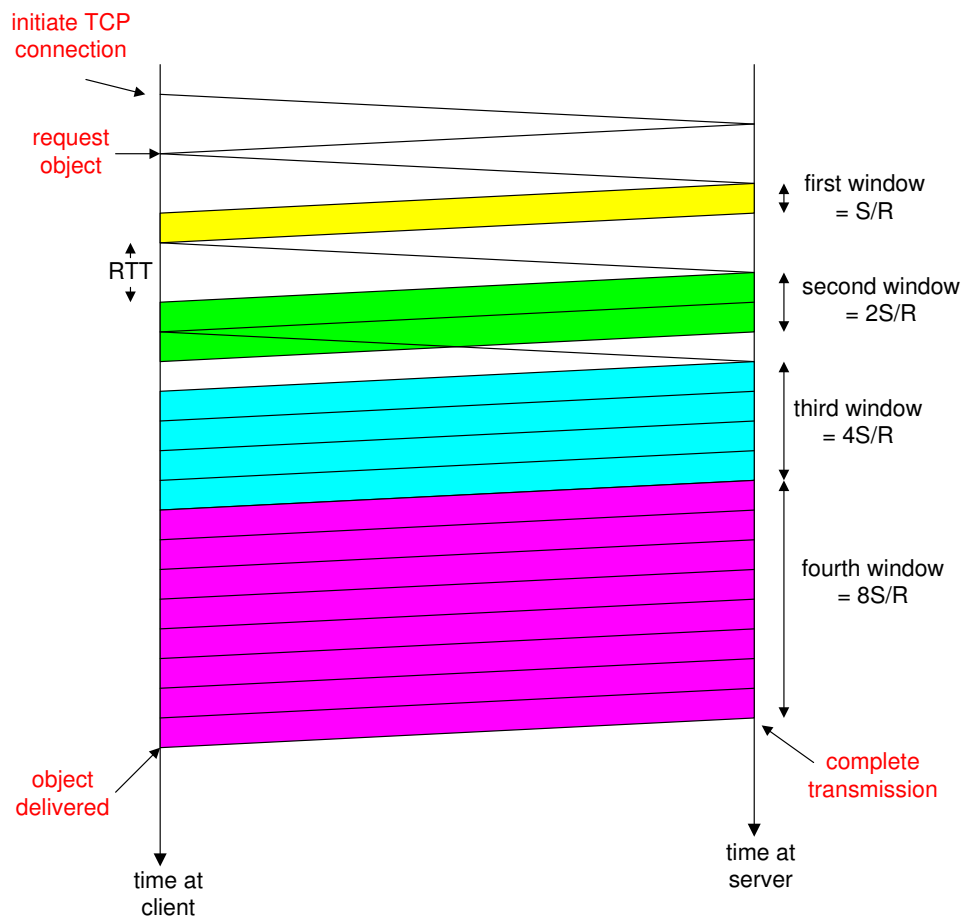
$$RTT + RTT + S/R + RTT + S/R + RTT + 12S/R = 4RTT + 14 S/R$$

- b) Similarly, the delay in this case is:

$$RTT + RTT + S/R + RTT + S/R + RTT + S/R + RTT + 8S/R = 5RTT + 11 S/R$$

- c) Similarly, the delay in this case is:

$$RTT + RTT + S/R + RTT + 14 S/R = 3 RTT + 15 S/R$$



to make the router's job easier, including using a datagram network layer rather than a virtual-circuit network layer, using a streamlined and fixed-sized header (as in IPv6), eliminating fragmentation (also done in IPv6), and providing the one and only best-effort service. Perhaps the most important trick here is *not* to keep track of individual flows, but instead base routing decisions solely on hierarchically structured destination addresses in the datagrams. It is interesting to note that the postal service has been using this approach for many years.

In this chapter, we also looked at the underlying principles of routing algorithms. We learned how routing algorithms abstract the computer network to a graph with nodes and links. With this abstraction, we can exploit the rich theory of shortest-path routing in graphs, which has been developed over the past 40 years in the operations research and algorithms communities. We saw that there are two broad approaches: a centralized (global) approach, in which each node obtains a complete map of the network and independently applies a shortest-path routing algorithm; and a decentralized approach, in which individual nodes have only a partial picture of the entire network, yet the nodes work together to deliver packets along the shortest routes. We also studied how hierarchy is used to deal with the problem of scale by partitioning large networks into independent administrative domains called autonomous systems (ASs). Each AS independently routes its datagrams through the AS, just as each country independently routes its postal mail through the country. We learned how centralized, decentralized, and hierarchical approaches are embodied in the principal routing protocols in the Internet: RIP, OSPF, and BGP. We concluded our study of routing algorithms by considering broadcast and multicast routing.

Having completed our study of the network layer, our journey now takes us one step further down the protocol stack, namely, to the link layer. Like the network layer, the link layer is also part of the network core. But we will see in the next chapter that the link layer has the much more localized task of moving packets between nodes on the same link or LAN. Although this task may appear on the surface to be trivial compared with that of the network layer's tasks, we will see that the link layer involves a number of important and fascinating issues that can keep us busy for a long time.



Homework Problems and Questions

Chapter 4 Review Questions

SECTIONS 4.1–4.2

- R1. Let's review some of the terminology used in this textbook. Recall that the name of a transport-layer packet is *segment* and that the name of a link-layer packet is *frame*. What is the name of a network-layer packet? Recall that both routers and link-layer switches are called *packet switches*. What is the fundamental difference between a router and link-layer switch? Recall that we use the term *routers* for both datagram networks and VC networks.

- R2. What are the two most important network-layer functions in a datagram network? What are the three most important network-layer functions in a virtual-circuit network?
- R3. What is the difference between routing and forwarding?
- R4. Do the routers in both datagram networks and virtual-circuit networks use forwarding tables? If so, describe the forwarding tables for both classes of networks.
- R5. Describe some hypothetical services that the network layer can provide to a single packet. Do the same for a flow of packets. Are any of your hypothetical services provided by the Internet's network layer? Are any provided by ATM's CBR service model? Are any provided by ATM's ABR service model?
- R6. List some applications that would benefit from ATM's CBR service model.

SECTION 4.3

- R7. Discuss why each input port in a high-speed router stores a shadow copy of the forwarding table.
- R8. Three types of switching fabrics are discussed in Section 4.3. List and briefly describe each type. Which, if any, can send multiple packets across the fabric in parallel?
- R9. Describe how packet loss can occur at input ports. Describe how packet loss at input ports can be eliminated (without using infinite buffers).
- R10. Describe how packet loss can occur at output ports. Can this loss be prevented by increasing the switch fabric speed?
- R11. What is HOL blocking? Does it occur in input ports or output ports?

SECTION 4.4

- R12. Do routers have IP addresses? If so, how many?
- R13. What is the 32-bit binary equivalent of the IP address 223.1.3.27?
- R14. Visit a host that uses DHCP to obtain its IP address, network mask, default router, and IP address of its local DNS server. List these values.
- R15. Suppose there are three routers between a source host and a destination host. Ignoring fragmentation, an IP datagram sent from the source host to the destination host will travel over how many interfaces? How many forwarding tables will be indexed to move the datagram from the source to the destination?
- R16. Suppose an application generates chunks of 40 bytes of data every 20 msec, and each chunk gets encapsulated in a TCP segment and then an IP datagram. What percentage of each datagram will be overhead, and what percentage will be application data?
- R17. Suppose Host A sends Host B a TCP segment encapsulated in an IP datagram. When Host B receives the datagram, how does the network layer in Host B

know it should pass the segment (that is, the payload of the datagram) to TCP rather than to UDP or to something else?

- R18. Suppose you purchase a wireless router and connect it to your cable modem. Also suppose that your ISP dynamically assigns your connected device (that is, your wireless router) one IP address. Also suppose that you have five PCs at home that use 802.11 to wirelessly connect to your wireless router. How are IP addresses assigned to the five PCs? Does the wireless router use NAT? Why or why not?
- R19. Compare and contrast the IPv4 and the IPv6 header fields. Do they have any fields in common?
- R20. It has been said that when IPv6 tunnels through IPv4 routers, IPv6 treats the IPv4 tunnels as link-layer protocols. Do you agree with this statement? Why or why not?

SECTION 4.5

- R21. Compare and contrast link-state and distance-vector routing algorithms.
- R22. Discuss how a hierarchical organization of the Internet has made it possible to scale to millions of users.
- R23. Is it necessary that every autonomous system use the same intra-AS routing algorithm? Why or why not?

SECTION 4.6

- R24. Consider Figure 4.37. Starting with the original table in *D*, suppose that *D* receives from *A* the following advertisement:

Destination Subnet	Next Router	Number of Hops to Destination
z	C	10
w	—	1
x	—	1
....

Will the table in *D* change? If so how?

- R25. Compare and contrast the advertisements used by RIP and OSPF.
- R26. Fill in the blank: RIP advertisements typically announce the number of hops to various destinations. BGP updates, on the other hand, announce the _____ to the various destinations.
- R27. Why are different inter-AS and intra-AS protocols used in the Internet?
- R28. Why are policy considerations as important for intra-AS protocols, such as OSPF and RIP, as they are for an inter-AS routing protocol like BGP?

- R29. Define and contrast the following terms: *subnet*, *prefix*, and *BGP route*.
- R30. How does BGP use the NEXT-HOP attribute? How does it use the AS-PATH attribute?
- R31. Describe how a network administrator of an upper-tier ISP can implement policy when configuring BGP.

SECTION 4.7

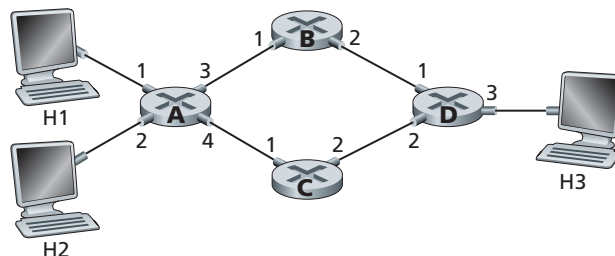
- R32. What is an important difference between implementing the broadcast abstraction via multiple unicasts, and a single network- (router-) supported broadcast?
- R33. For each of the three general approaches we studied for broadcast communication (uncontrolled flooding, controlled flooding, and spanning-tree broadcast), are the following statements true or false? You may assume that no packets are lost due to buffer overflow and all packets are delivered on a link in the order in which they were sent.
 - a. A node may receive multiple copies of the same packet.
 - b. A node may forward multiple copies of a packet over the same outgoing link.
- R34. When a host joins a multicast group, must it change its IP address to that of the multicast group it is joining?
- R35. What are the roles played by the IGMP protocol and a wide-area multicast routing protocol?
- R36. What is the difference between a group-shared tree and a source-based tree in the context of multicast routing?



Problems

- P1. In this question, we consider some of the pros and cons of virtual-circuit and datagram networks.
 - a. Suppose that routers were subjected to conditions that might cause them to fail fairly often. Would this argue in favor of a VC or datagram architecture? Why?
 - b. Suppose that a source node and a destination require that a fixed amount of capacity always be available at all routers on the path between the source and destination node, for the exclusive use of traffic flowing between this source and destination node. Would this argue in favor of a VC or datagram architecture? Why?
 - c. Suppose that the links and routers in the network never fail and that routing paths used between all source/destination pairs remains constant. In this scenario, does a VC or datagram architecture have more control traffic overhead? Why?

- P2. Consider a virtual-circuit network. Suppose the VC number is an 8-bit field.
- What is the maximum number of virtual circuits that can be carried over a link?
 - Suppose a central node determines paths and VC numbers at connection setup. Suppose the same VC number is used on each link along the VC's path. Describe how the central node might determine the VC number at connection setup. Is it possible that there are fewer VCs in progress than the maximum as determined in part (a) yet there is no common free VC number?
 - Suppose that different VC numbers are permitted in each link along a VC's path. During connection setup, after an end-to-end path is determined, describe how the links can choose their VC numbers and configure their forwarding tables in a decentralized manner, without reliance on a central node.
- P3. A bare-bones forwarding table in a VC network has four columns. What is the meaning of the values in each of these columns? A bare-bones forwarding table in a datagram network has two columns. What is the meaning of the values in each of these columns?
- P4. Consider the network below.
- Suppose that this network is a datagram network. Show the forwarding table in router A, such that all traffic destined to host H3 is forwarded through interface 3.
 - Suppose that this network is a datagram network. Can you write down a forwarding table in router A, such that all traffic from H1 destined to host H3 is forwarded through interface 3, while all traffic from H2 destined to host H3 is forwarded through interface 4? (Hint: this is a trick question.)
 - Now suppose that this network is a virtual circuit network and that there is one ongoing call between H1 and H3, and another ongoing call between H2 and H3. Write down a forwarding table in router A, such that all traffic from H1 destined to host H3 is forwarded through interface 3, while all traffic from H2 destined to host H3 is forwarded through interface 4.
 - Assuming the same scenario as (c), write down the forwarding tables in nodes B, C, and D.



- P5. Consider a VC network with a 2-bit field for the VC number. Suppose that the network wants to set up a virtual circuit over four links: link A, link B,

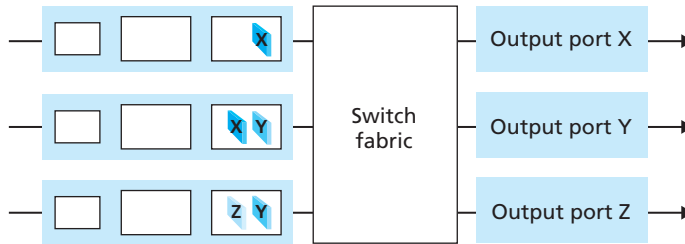
link C, and link D. Suppose that each of these links is currently carrying two other virtual circuits, and the VC numbers of these other VCs are as follows:

Link A	Link B	Link C	Link D
00	01	10	11
01	10	11	00

In answering the following questions, keep in mind that each of the existing VCs may only be traversing one of the four links.

- a. If each VC is required to use the same VC number on all links along its path, what VC number could be assigned to the new VC?
 - b. If each VC is permitted to have different VC numbers in the different links along its path (so that forwarding tables must perform VC number translation), how many different combinations of four VC numbers (one for each of the four links) could be used?
- P6. In the text we have used the term *connection-oriented service* to describe a transport-layer service and *connection service* for a network-layer service. Why the subtle shades in terminology?
- P7. Suppose two packets arrive to two different input ports of a router at exactly the same time. Also suppose there are no other packets anywhere in the router.
- a. Suppose the two packets are to be forwarded to two *different* output ports. Is it possible to forward the two packets through the switch fabric at the same time when the fabric uses a *shared bus*?
 - b. Suppose the two packets are to be forwarded to two *different* output ports. Is it possible to forward the two packets through the switch fabric at the same time when the fabric uses a *crossbar*?
 - c. Suppose the two packets are to be forwarded to the *same* output port. Is it possible to forward the two packets through the switch fabric at the same time when the fabric uses a *crossbar*?
- P8. In Section 4.3, we noted that the maximum queuing delay is $(n-1)D$ if the switching fabric is n times faster than the input line rates. Suppose that all packets are of the same length, n packets arrive at the same time to the n input ports, and all n packets want to be forwarded to *different* output ports. What is the maximum delay for a packet for the (a) memory, (b) bus, and (c) crossbar switching fabrics?
- P9. Consider the switch shown below. Suppose that all datagrams have the same fixed length, that the switch operates in a slotted, synchronous manner, and that in one time slot a datagram can be transferred from an input port to an output port. The switch fabric is a crossbar so that at most one datagram can

be transferred to a given output port in a time slot, but different output ports can receive datagrams from different input ports in a single time slot. What is the minimal number of time slots needed to transfer the packets shown from input ports to their output ports, assuming any input queue scheduling order you want (i.e., it need not have HOL blocking)? What is the largest number of slots needed, assuming the worst-case scheduling order you can devise, assuming that a non-empty input queue is never idle?



- P10. Consider a datagram network using 32-bit host addresses. Suppose a router has four links, numbered 0 through 3, and packets are to be forwarded to the link interfaces as follows:

Destination Address Range	Link Interface
11100000 00000000 00000000 00000000 through 11100000 00111111 11111111 11111111	0
11100000 01000000 00000000 00000000 through 11100000 01000000 11111111 11111111	1
11100000 01000001 00000000 00000000 through 11100001 01111111 11111111 11111111	2
otherwise	3

- Provide a forwarding table that has five entries, uses longest prefix matching, and forwards packets to the correct link interfaces.
- Describe how your forwarding table determines the appropriate link interface for datagrams with destination addresses:

```

11001000 10010001 01010001 01010101
11100001 01000000 11000011 00111100
11100001 10000000 00010001 01110111
    
```

- P11. Consider a datagram network using 8-bit host addresses. Suppose a router uses longest prefix matching and has the following forwarding table:

Prefix Match	Interface
00	0
010	1
011	2
10	2
11	3

For each of the four interfaces, give the associated range of destination host addresses and the number of addresses in the range.

- P12. Consider a datagram network using 8-bit host addresses. Suppose a router uses longest prefix matching and has the following forwarding table:

Prefix Match	Interface
1	0
10	1
111	2
otherwise	3

For each of the four interfaces, give the associated range of destination host addresses and the number of addresses in the range.

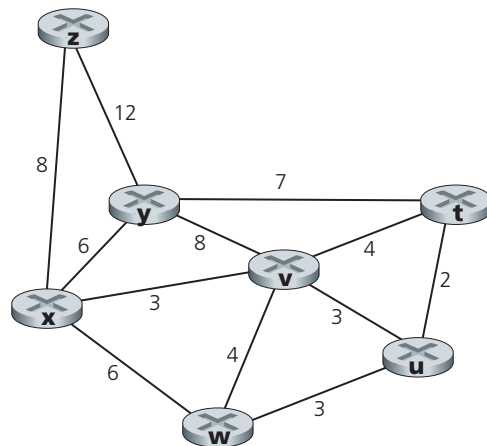
- P13. Consider a router that interconnects three subnets: Subnet 1, Subnet 2, and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 223.1.17/24. Also suppose that Subnet 1 is required to support at least 60 interfaces, Subnet 2 is to support at least 90 interfaces, and Subnet 3 is to support at least 12 interfaces. Provide three network addresses (of the form a.b.c.d/x) that satisfy these constraints.
- P14. In Section 4.2.2 an example forwarding table (using longest prefix matching) is given. Rewrite this forwarding table using the a.b.c.d/x notation instead of the binary string notation.
- P15. In Problem P10 you are asked to provide a forwarding table (using longest prefix matching). Rewrite this forwarding table using the a.b.c.d/x notation instead of the binary string notation.

- P16. Consider a subnet with prefix 128.119.40.128/26. Give an example of one IP address (of form xxx.xxx.xxx.xxx) that can be assigned to this network. Suppose an ISP owns the block of addresses of the form 128.119.40.64/26. Suppose it wants to create four subnets from this block, with each block having the same number of IP addresses. What are the prefixes (of form a.b.c.d/x) for the four subnets?
- P17. Consider the topology shown in Figure 4.17. Denote the three subnets with hosts (starting clockwise at 12:00) as Networks A, B, and C. Denote the subnets without hosts as Networks D, E, and F.
- Assign network addresses to each of these six subnets, with the following constraints: All addresses must be allocated from 214.97.254/23; Subnet A should have enough addresses to support 250 interfaces; Subnet B should have enough addresses to support 120 interfaces; and Subnet C should have enough addresses to support 120 interfaces. Of course, subnets D, E and F should each be able to support two interfaces. For each subnet, the assignment should take the form a.b.c.d/x or a.b.c.d/x – e.f.g.h/y.
 - Using your answer to part (a), provide the forwarding tables (using longest prefix matching) for each of the three routers.
- P18. Use the whois service at the American Registry for Internet Numbers (<http://www.arin.net/whois>) to determine the IP address blocks for three universities. Can the whois services be used to determine with certainty the geographical location of a specific IP address? Use www.maxmind.com to determine the locations of the Web servers at each of these universities.
- P19. Consider sending a 2400-byte datagram into a link that has an MTU of 700 bytes. Suppose the original datagram is stamped with the identification number 422. How many fragments are generated? What are the values in the various fields in the IP datagram(s) generated related to fragmentation?
- P20. Suppose datagrams are limited to 1,500 bytes (including header) between source Host A and destination Host B. Assuming a 20-byte IP header, how many datagrams would be required to send an MP3 consisting of 5 million bytes? Explain how you computed your answer.
- P21. Consider the network setup in Figure 4.22. Suppose that the ISP instead assigns the router the address 24.34.112.235 and that the network address of the home network is 192.168.1/24.
- Assign addresses to all interfaces in the home network.
 - Suppose each host has two ongoing TCP connections, all to port 80 at host 128.119.40.86. Provide the six corresponding entries in the NAT translation table.

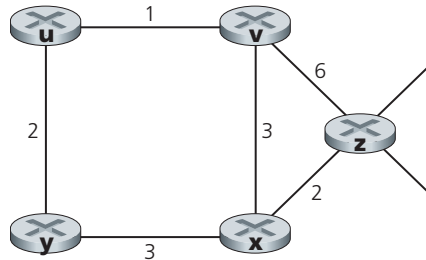
- P22. Suppose you are interested in detecting the number of hosts behind a NAT. You observe that the IP layer stamps an identification number sequentially on each IP packet. The identification number of the first IP packet generated by a host is a random number, and the identification numbers of the subsequent IP packets are sequentially assigned. Assume all IP packets generated by hosts behind the NAT are sent to the outside world.
- Based on this observation, and assuming you can sniff all packets sent by the NAT to the outside, can you outline a simple technique that detects the number of unique hosts behind a NAT? Justify your answer.
 - If the identification numbers are not sequentially assigned but randomly assigned, would your technique work? Justify your answer.
- P23. In this problem we'll explore the impact of NATs on P2P applications. Suppose a peer with username Arnold discovers through querying that a peer with username Bernard has a file it wants to download. Also suppose that Bernard and Arnold are both behind a NAT. Try to devise a technique that will allow Arnold to establish a TCP connection with Bernard without application-specific NAT configuration. If you have difficulty devising such a technique, discuss why.
- P24. Looking at Figure 4.27, enumerate the paths from y to u that do not contain any loops.
- P25. Repeat Problem P24 for paths from x to z , z to u , and z to w .
- P26. Consider the following network. With the indicated link costs, use Dijkstra's shortest-path algorithm to compute the shortest path from x to all network nodes. Show how the algorithm works by computing a table similar to Table 4.3.



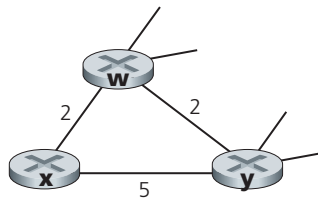
VideoNote
Dijkstra's algorithm:
discussion and example



- P27. Consider the network shown in Problem P26. Using Dijkstra's algorithm, and showing your work using a table similar to Table 4.3, do the following:
- Compute the shortest path from t to all network nodes.
 - Compute the shortest path from u to all network nodes.
 - Compute the shortest path from v to all network nodes.
 - Compute the shortest path from w to all network nodes.
 - Compute the shortest path from y to all network nodes.
 - Compute the shortest path from z to all network nodes.
- P28. Consider the network shown below, and assume that each node initially knows the costs to each of its neighbors. Consider the distance-vector algorithm and show the distance table entries at node z .

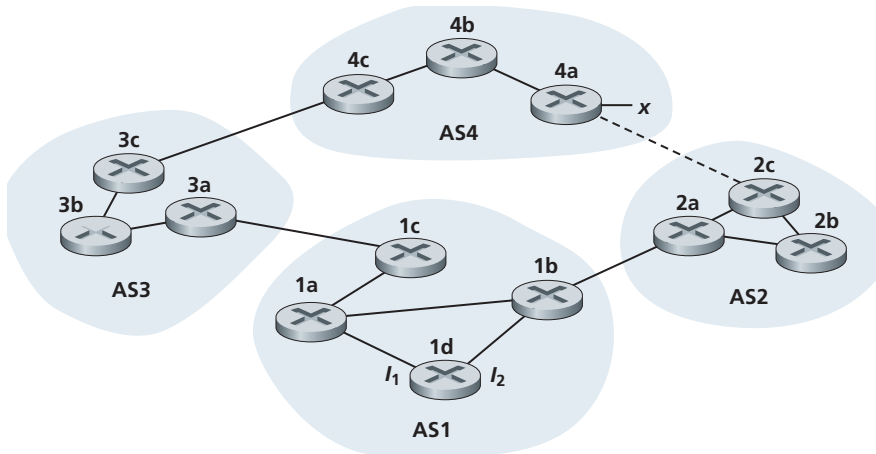


- P29. Consider a general topology (that is, not the specific network shown above) and a synchronous version of the distance-vector algorithm. Suppose that at each iteration, a node exchanges its distance vectors with its neighbors and receives their distance vectors. Assuming that the algorithm begins with each node knowing only the costs to its immediate neighbors, what is the maximum number of iterations required before the distributed algorithm converges? Justify your answer.
- P30. Consider the network fragment shown below. x has only two attached neighbors, w and y . w has a minimum-cost path to destination u (not shown) of 5, and y has a minimum-cost path to u of 6. The complete paths from w and y to u (and between w and y) are not shown. All link costs in the network have strictly positive integer values.



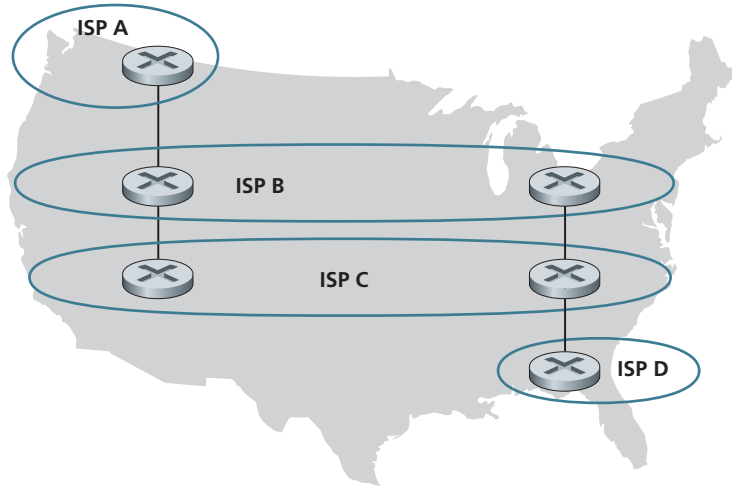
- a. Give x 's distance vector for destinations w , y , and u .
 - b. Give a link-cost change for either $c(x,w)$ or $c(x,y)$ such that x will inform its neighbors of a new minimum-cost path to u as a result of executing the distance-vector algorithm.
 - c. Give a link-cost change for either $c(x,w)$ or $c(x,y)$ such that x will *not* inform its neighbors of a new minimum-cost path to u as a result of executing the distance-vector algorithm.
- P31. Consider the three-node topology shown in Figure 4.30. Rather than having the link costs shown in Figure 4.30, the link costs are $c(x,y) = 3$, $c(y,z) = 6$, $c(z,x) = 4$. Compute the distance tables after the initialization step and after each iteration of a synchronous version of the distance-vector algorithm (as we did in our earlier discussion of Figure 4.30).
- P32. Consider the count-to-infinity problem in the distance vector routing. Will the count-to-infinity problem occur if we decrease the cost of a link? Why? How about if we connect two nodes which do not have a link?
- P33. Argue that for the distance-vector algorithm in Figure 4.30, each value in the distance vector $D(x)$ is non-increasing and will eventually stabilize in a finite number of steps.
- P34. Consider Figure 4.31. Suppose there is another router w , connected to router y and z . The costs of all links are given as follows: $c(x,y) = 4$, $c(x,z) = 50$, $c(y,w) = 1$, $c(z,w) = 1$, $c(y,z) = 3$. Suppose that poisoned reverse is used in the distance-vector routing algorithm.
- a. When the distance vector routing is stabilized, router w , y , and z inform their distances to x to each other. What distance values do they tell each other?
 - b. Now suppose that the link cost between x and y increases to 60. Will there be a count-to-infinity problem even if poisoned reverse is used? Why or why not? If there is a count-to-infinity problem, then how many iterations are needed for the distance-vector routing to reach a stable state again? Justify your answer.
 - c. How do you modify $c(y,z)$ such that there is no count-to-infinity problem at all if $c(y,x)$ changes from 4 to 60?
- P35. Describe how loops in paths can be detected in BGP.
- P36. Will a BGP router always choose the loop-free route with the shortest AS-path length? Justify your answer.
- P37. Consider the network shown below. Suppose AS3 and AS2 are running OSPF for their intra-AS routing protocol. Suppose AS1 and AS4 are running RIP for their intra-AS routing protocol. Suppose eBGP and iBGP are used for the inter-AS routing protocol. Initially suppose there is *no* physical link between AS2 and AS4.

- Router 3c learns about prefix x from which routing protocol: OSPF, RIP, eBGP, or iBGP?
- Router 3a learns about x from which routing protocol?
- Router 1c learns about x from which routing protocol?
- Router 1d learns about x from which routing protocol?

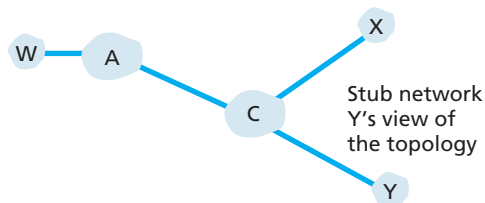


- P38. Referring to the previous problem, once router 1d learns about x it will put an entry (x, I) in its forwarding table.
- Will I be equal to I_1 or I_2 for this entry? Explain why in one sentence.
 - Now suppose that there is a physical link between AS2 and AS4, shown by the dotted line. Suppose router 1d learns that x is accessible via AS2 as well as via AS3. Will I be set to I_1 or I_2 ? Explain why in one sentence.
 - Now suppose there is another AS, called AS5, which lies on the path between AS2 and AS4 (not shown in diagram). Suppose router 1d learns that x is accessible via AS2 AS5 AS4 as well as via AS3 AS4. Will I be set to I_1 or I_2 ? Explain why in one sentence.
- P39. Consider the following network. ISP B provides national backbone service to regional ISP A. ISP C provides national backbone service to regional ISP D. Each ISP consists of one AS. B and C peer with each other in two places using BGP. Consider traffic going from A to D. B would prefer to hand that traffic over to C on the West Coast (so that C would have to absorb the cost of carrying the traffic cross-country), while C would prefer to get the traffic via its East Coast peering point with B (so that B would have carried the traffic across the country). What BGP mechanism might C use, so that B would hand over A-to-D traffic at its East Coast

peering point? To answer this question, you will need to dig into the BGP specification.



- P40. In Figure 4.42, consider the path information that reaches stub networks W, X, and Y. Based on the information available at W and X, what are their respective views of the network topology? Justify your answer. The topology view at Y is shown below.



- P41. Consider Figure 4.42. B would never forward traffic destined to Y via X based on BGP routing. But there are some very popular applications for which data packets go to X first and then flow to Y. Identify one such application, and describe how data packets follow a path not given by BGP routing.
- P42. In Figure 4.42, suppose that there is another stub network V that is a customer of ISP A. Suppose that B and C have a peering relationship, and A is a customer of both B and C. Suppose that A would like to have the traffic destined to W to come from B only, and the traffic destined to V from either B or C. How should A advertise its routes to B and C? What AS routes does C receive?
- P43. Suppose ASs X and Z are not directly connected but instead are connected by AS Y. Further suppose that X has a peering agreement with Y, and that Y has

a peering agreement with Z. Finally, suppose that Z wants to transit all of Y's traffic but does not want to transit X's traffic. Does BGP allow Z to implement this policy?

- P44. Consider the seven-node network (with nodes labeled t to z) in Problem P26. Show the minimal-cost tree rooted at z that includes (as end hosts) nodes u , v , w , and y . Informally argue why your tree is a minimal-cost tree.
- P45. Consider the two basic approaches identified for achieving broadcast, unicast emulation and network-layer (i.e., router-assisted) broadcast, and suppose spanning-tree broadcast is used to achieve network-layer broadcast. Consider a single sender and 32 receivers. Suppose the sender is connected to the receivers by a binary tree of routers. What is the cost of sending a broadcast packet, in the cases of unicast emulation and network-layer broadcast, for this topology? Here, each time a packet (or copy of a packet) is sent over a single link, it incurs a unit of cost. What topology for interconnecting the sender, receivers, and routers will bring the cost of unicast emulation and true network-layer broadcast as far apart as possible? You can choose as many routers as you'd like.
- P46. Consider the operation of the reverse path forwarding (RPF) algorithm in Figure 4.44. Using the same topology, find a set of paths from all nodes to the source node A (and indicate these paths in a graph using thicker-shaded lines as in Figure 4.44) such that if these paths were the least-cost paths, then node B would receive a copy of A 's broadcast message from nodes A , C , and D under RPF.
- P47. Consider the topology shown in Figure 4.44. Suppose that all links have unit cost and that node E is the broadcast source. Using arrows like those shown in Figure 4.44 indicate links over which packets will be forwarded using RPF, and links over which packets will not be forwarded, given that node E is the source.
- P48. Repeat Problem P47 using the graph from Problem P26. Assume that z is the broadcast source, and that the link costs are as shown in Problem P26.
- P49. Consider the topology shown in Figure 4.46, and suppose that each link has unit cost. Suppose node C is chosen as the center in a center-based multicast routing algorithm. Assuming that each attached router uses its least-cost path to node C to send join messages to C , draw the resulting center-based routing tree. Is the resulting tree a minimum-cost tree? Justify your answer.
- P50. Repeat Problem P49, using the graph from Problem P26. Assume that the center node is v .
- P51. In Section 4.5.1 we studied Dijkstra's link-state routing algorithm for computing the unicast paths that are individually the least-cost paths from the source to all destinations. The union of these paths might be thought of as forming a **least-unicast-cost path tree** (or a shortest unicast path tree, if all link costs are identical). By constructing a counterexample, show that the least-cost path tree is *not* always the same as a minimum spanning tree.

- P52. Consider a network in which all nodes are connected to three other nodes. In a single time step, a node can receive all transmitted broadcast packets from its neighbors, duplicate the packets, and send them to all of its neighbors (except to the node that sent a given packet). At the next time step, neighboring nodes can receive, duplicate, and forward these packets, and so on. Suppose that uncontrolled flooding is used to provide broadcast in such a network. At time step t , how many copies of the broadcast packet will be transmitted, assuming that during time step 1, a single broadcast packet is transmitted by the source node to its three neighbors.
- P53. We saw in Section 4.7 that there is no network-layer protocol that can be used to identify the hosts participating in a multicast group. Given this, how can multicast applications learn the identities of the hosts that are participating in a multicast group?
- P54. Design (give a pseudocode description of) an application-level protocol that maintains the host addresses of all hosts participating in a multicast group. Specifically identify the network service (unicast or multicast) that is used by your protocol, and indicate whether your protocol is sending messages in-band or out-of-band (with respect to the application data flow among the multicast group participants) and why.
- P55. What is the size of the multicast address space? Suppose now that two multicast groups randomly choose a multicast address. What is the probability that they choose the same address? Suppose now that 1,000 multicast groups are ongoing at the same time and choose their multicast group addresses at random. What is the probability that they interfere with each other?



Socket Programming Assignment

At the end of Chapter 2, there are four socket programming assignments. Below, you will find a fifth assignment which employs ICMP, a protocol discussed in this chapter.

Assignment 5: ICMP Ping

Ping is a popular networking application used to test from a remote location whether a particular host is up and reachable. It is also often used to measure latency between the client host and the target host. It works by sending ICMP “echo request” packets (i.e., ping packets) to the target host and listening for ICMP “echo response” replies (i.e., pong packets). Ping measures the RRT, records packet loss, and calculates a statistical summary of multiple ping-pong exchanges (the minimum, mean, max, and standard deviation of the round-trip times).

Chapter 4 Review Questions

1. A network-layer packet is a datagram. A router forwards a packet based on the packet's IP (layer 3) address. A link-layer switch forwards a packet based on the packet's MAC (layer 2) address.
2. Datagram-based network layer: forwarding; routing. Additional function of VC-based network layer: call setup.
3. Forwarding is about moving a packet from a router's input port to the appropriate output port. Routing is about determining the end-to-routes between sources and destinations.
4. Yes, both use forwarding tables. For descriptions of the tables, see Section 4.2.
5. Single packet: guaranteed delivery; guaranteed delivery with guaranteed maximum delay. Flow of packets: in-order packet delivery; guaranteed minimal bandwidth; guaranteed maximum jitter. None of these services is provided by the Internet's network layer. ATM's CBR service provides both guaranteed delivery and timing. ABR does not provide any of these services.
6. Interactive live multimedia applications, such as IP telephony and video conference, could benefit from ATM CBR's service, which maintains timing.
7. With the shadow copy, the forwarding lookup is made locally, at each input port, without invoking the centralized routing processor. Such a decentralized approach avoids creating a lookup processing bottleneck at a single point within the router.
8. Switching via memory; switching via a bus; switching via an interconnection network. An interconnection network can forward packets in parallel as long as all the packets are being forwarded to different output ports.
9. If the rate at which packets arrive to the fabric exceeds switching fabric rate, then packets will need to queue at the input ports. If this rate mismatch persists, the queues will get larger and larger and eventually overflow the input port buffers, causing packet loss. Packet loss can be eliminated if the switching fabric speed is at least n times as fast as the input line speed, where n is the number of input ports.
10. Assuming input and output line speeds are the same, packet loss can still occur if the rate at which packets arrive to a single output port exceeds the line speed. If this rate mismatch persists, the queues will get larger and larger and eventually overflow the output port buffers, causing packet loss. Note that increasing switch fabric speed cannot prevent this problem from occurring.

11. HOL blocking: Sometimes the a packet that is first in line at an input port queue must wait because there is no available buffer space at the output port to which it wants to be forwarded. When this occurs, all the packets behind the first packet are blocked, even if their output queues have room to accommodate them. HOL blocking occurs at the input port.
12. Yes. They have one address for each interface.
13. 11011111 00000001 00000011 00011100.
14. Students will get different correct answers for this question.
15. 8 interfaces; 3 forwarding tables.
16. 50% overhead.
17. The 8-bit protocol field in the IP datagram contains information about which transport layer protocol the destination host should pass the segment to.
18. Typically the wireless router includes a DHCP server. DHCP is used to assign IP addresses to the 5 PCs and to the router interface. Yes, the wireless router also uses NAT as it obtains only one IP address from the ISP.
19. IPv6 has a fixed length header, which does not include most of the options an IPv4 header can include. Even though the IPv6 header contains two 128 bit addresses (source and destination IP address) the whole header has a fixed length of 40 bytes only. Several of the fields are similar in spirit. Traffic class, payload length, next header and hop limit in IPv6 are respectively similar to type of service, datagram length, upper-layer protocol and time to live in IPv4.
20. Yes, because the entire IPv6 datagram (including header fields) is encapsulated in an IPv4 datagram.
21. Link state algorithms: Computes the least-cost path between source and destination using complete, global knowledge about the network. Distance-vector routing: The calculation of the least-cost path is carried out in an iterative, distributed manner. A node only knows the neighbor to which it should forward a packet in order to reach given destination along the least-cost path, and the cost of that path from itself to the destination.
22. Routers are organized into autonomous systems (ASs). Within an AS, all routers run the same intra-AS routing protocol. The problem of scale is solved since an router in an AS need only know about routers within its AS and the subnets that attach to the AS. To route across ASes, the inter-AS protocol is based on the AS graph and does not take individual routers into account.

23. No. Each AS has administrative autonomy for routing within an AS.
24. No. The advertisement tells D that it can get to z in 11 hops by way of A. However, D can already get to z by way of B in 7 hops. Therefore, there is no need to modify the entry for z in the table. If, on the other hand, the advertisement said that A were only 4 hops away from z by way of C, then D would indeed modify its forwarding table.
25. With OSPF, a router periodically broadcasts routing information to all other routers in the AS, not just to its neighboring routers. This routing information sent by a router has one entry for each of the router's neighbors; the entry gives the distance from the router to the neighbor. A RIP advertisement sent by a router contains information about all the networks in the AS, although this information is only sent to its neighboring routers.
26. "sequence of ASs on the routes"
27. Policy: Among ASs, policy issues dominate. It may well be important that traffic originating in a given AS not be able to pass through another specific AS. Similarly, a given AS may want to control what transit traffic it carries between other ASs. Within an AS, everything is nominally under the same administrative control and thus policy issues a much less important role in choosing routes within an AS.

Scale: The ability of a routing algorithm and its data structures to scale to handle routing to/among large numbers of networks is a critical issue in inter-AS routing. Within an AS, scalability is less of a concern. For one thing, if a single administrative domain becomes too large, it is always possible to divide it into two ASs and perform inter-AS routing between the two new ASs.

Performance: Because inter-AS routing is so policy oriented, the quality (for example, performance) of the routes used is often of secondary concern (that is, a longer or more costly route that satisfies certain policy criteria may well be taken over a route that is shorter but does not meet that criteria). Indeed, we saw that among ASs, there is not even the notion of cost (other than AS hop count) associated with routes. Within a single AS, however, such policy concerns are of less importance, allowing routing to focus more on the level of performance realized on a route.

28. ISP C can use the BGP Multi-Exit Descriptor to suggest to ISP B that the preferred route to ISP D is through the east coast peering point. For example, the east coast BGP router in ISP C can advertise a route to D with an MED value of 5. The west coast router in ISP C can advertise a route to D with an MED value of 10. Since a lower value is preferred, ISP B knows that ISP C wants to receive traffic on the east coast. In practice, a router can ignore the MED value, and so ISP B can still use hot potato routing to pass traffic to ISP C destined to ISP D via the west coast peering point.

29. A **subnet** is a portion of a larger network; a subnet does not contain a router; its boundaries are defined by the router and host interfaces. A **prefix** is the network portion of a CIDRized address; it is written in the form a.b.c.d/x ; A prefix covers one or more subnets. When a router advertises a prefix across a BGP session, it includes with the prefix a number of BGP attributes. In BGP jargon, a prefix along with its attributes is a **BGP route** (or simply a **route**).
30. Routers use the AS-PATH attribute to detect and prevent looping advertisements; they also use it in choosing among multiple paths to the same prefix. The NEXT-HOP attribute indicates the IP address of the first router along an advertised path (outside of the AS receiving the advertisement) to a given prefix. When configuring its forwarding table, a router uses the NEXT-HOP attribute.
31. A tier-1 ISP B may not to carry transit traffic between two other tier-1 ISPs, say A and C, with which B has peering agreements. To implement this policy, ISP B would not advertise to A routes that pass through C; and would not advertise to C routes that pass through A.
32. N-way unicast has a number of drawbacks, including:
- Efficiency: multiple copies of the same packet are sent over the same link for potentially many links; source must generate multiple copies of same packet
 - Addressing: the source must discover the address of all the recipients
33. a) uncontrolled flooding: T; controlled flooding: T; spanning-tree: F
- b) uncontrolled flooding: T; controlled flooding: F; spanning-tree: F
34. False
35. IGMP is a protocol run only between the host and its first-hop multicast router. IGMP allows a host to specify (to the first-hop multicast router) the multicast group it wants to join. It is then up to the multicast router to work with other multicast routers (i.e., run a multicast routing protocol) to ensure that the data for the host-joined multicast group is routed to the appropriate last-hop router and from there to the host.
36. In a group-shared tree, all senders send their multicast traffic using the same routing tree. With source-based tree, the multicast datagrams from a given source are routed over s specific routing tree constructed for that source; thus each source may have a different source-based tree and a router may have to keep track of several source-based trees for a given multicast group.

Chapter 4 Problems

Problem 1

- a) With a connection-oriented network, every router failure will involve the routing of that connection. At a minimum, this will require the router that is “upstream” from the failed router to establish a new downstream part of the path to the destination node, with all of the requisite signaling involved in setting up a path. Moreover, all of the routers on the initial path that are downstream from the failed node must take down the failed connection, with all of the requisite signaling involved to do this.

With a connectionless datagram network, no signaling is required to either set up a new downstream path or take down the old downstream path. We have seen, however, that routing tables will need to be updated (e.g., either via a distance vector algorithm or a link state algorithm) to take the failed router into account. We have seen that with distance vector algorithms, this routing table change can sometimes be localized to the area near the failed router. Thus, a datagram network would be preferable. Interestingly, the design criteria that the initial ARPAnet be able to function under stressful conditions was one of the reasons that datagram architecture was chosen for this Internet ancestor.

- b) In order for a router to maintain an available fixed amount of capacity on the path between the source and destination node for that source-destination pair, it would need to know the characteristics of the traffic from all sessions passing through that link. That is, the router must have per-session state in the router. This is possible in a connection-oriented network, but not with a connectionless network. Thus, a connection-oriented VC network would be preferable.
- c) In this scenario, datagram architecture has more control traffic overhead. This is due to the various packet headers needed to route the datagrams through the network. But in VC architecture, once all circuits are set up, they will never change. Thus, the signaling overhead is negligible over the long run.

Problem 2

- a) Maximum number of VCs over a link = $28 = 256$.
- b) The centralized node could pick any VC number which is free from the set $\{0, 1, \dots, 28-1\}$. In this manner, it is not possible that there are fewer VCs in progress than 256 without there being any common free VC number.
- c) Each of the links can independently allocate VC numbers from the set $\{0, 1, \dots, 28-1\}$. Thus, a VC will likely have a different VC number for each link along its path. Each router in the VC's path must replace the VC number of each arriving packet with the VC number associated with the outbound link.

Problem 3

For a VC forwarding table, the columns are : Incoming Interface, Incoming VC Number, Outgoing Interface, Outgoing VC Number. For a datagram forwarding table, the columns are: Destination Address, Outgoing Interface.

Problem 4

- a) Data destined to host H3 is forwarded through interface 3

Destination Address	Link Interface
H3	3

- b) No, because forwarding rule is only based on destination address.
- c) One possible configuration is:

Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	12	3	22
2	63	4	18

Note, that the two flows could actually have the same VC numbers.

- d) One possible configuration is:

Router B.			
Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	22	2	24

Router C.			
Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	18	2	50

Router D.			
Incoming interface	Incoming VC#	Outgoing Interface	Outgoing VC#
1	24	3	70
2	50	3	76

Problem 5

- a) No VC number can be assigned to the new VC; thus the new VC cannot be established in the network.

- b) Each link has two available VC numbers. There are four links. So the number of combinations is $2^4 = 16$. One example combination is (10,00,00,10).

Problem 6

In a virtual circuit network, there *is* an end-to-end connection in the sense that each router along the path must maintain state for the connection; hence the terminology *connection service*. In a connection-oriented transport service over a connectionless network layer, such as TCP over IP, the end systems maintain connection state; however the routers have no notion of any connections; hence the terminology *connection-oriented service*.

Problem 7

- a) No, you can only transmit one packet at a time over a shared bus.
- b) Yes, as discussed in the text, as long as the two packets use different input busses and different output busses, they can be forwarded in parallel.
- c) No, in this case the two packets would have to be sent over the same output bus at the same time, which is not possible.

Problem 8

- a) $(n-1)D$
- b) $(n-1)D$
- c) 0

Problem 9

The minimal number of time slots needed is 3. The scheduling is as follows.

Slot 1: send X in top input queue, send Y in middle input queue.

Slot 2: send X in middle input queue, send Y in bottom input queue

Slot 3: send Z in bottom input queue.

Largest number of slots is still 3. Actually, based on the assumption that a non-empty input queue is never idle, we see that the first time slot always consists of sending X in the top input queue and Y in either middle or bottom input queue, and in the second time slot, we can always send two more datagram, and the last datagram can be sent in third time slot.

NOTE: Actually, if the first datagram in the bottom input queue is X, then the worst case would require 4 time slots.

Problem 10

a)

Prefix Match	Link Interface
11100000 00	0
11100000 01000000	1
1110000	2
11100001 1	3
otherwise	3

- b) Prefix match for first address is 5th entry: link interface 3
Prefix match for second address is 3rd entry: link interface 2
Prefix match for third address is 4th entry: link interface 3

Problem 11

Destination Address Range	Link Interface
00000000 through 00111111	0
01000000 through 01011111	1
01100000 through 01111111	2
10000000 through 10111111	2
11000000 through 11111111	3

number of addresses for interface 0 = $2^6 = 64$

number of addresses for interface 1 = $2^5 = 32$

number of addresses for interface 2 = $2^6 + 2^5 = 64 + 32 = 96$

number of addresses for interface 3 = $2^6 = 64$

Problem 12

Destination Address Range	Link Interface
11000000 through (32 addresses) 11011111	0
10000000 through(64 addresses) 10111111	1
11100000 through (32 addresses) 11111111	2
00000000 through (128 addresses) 01111111	3

Problem 13

223.1.17.0/26
223.1.17.128/25
223.1.17.192/28

Problem 14

Destination Address	Link Interface
200.23.16/21	0
200.23.24/24	1
200.23.24/21	2
otherwise	3

Problem 15

Destination Address	Link Interface
11100000 00 (224.0/10)	0
11100000 01000000 (224.64/16)	1
1110000 (224/8)	2
11100001 1 (225.128/9)	3
otherwise	3

Problem 16

Any IP address in range 128.119.40.128 to 128.119.40.191

Four equal size subnets: 128.119.40.64/28, 128.119.40.80/28, 128.119.40.96/28, 128.119.40.112/28

Problem 17

From 214.97.254/23, possible assignments are

- a) Subnet A: 214.97.255/24 (256 addresses)
Subnet B: 214.97.254.0/25 - 214.97.254.0/29 (128-8 = 120 addresses)
Subnet C: 214.97.254.128/25 (128 addresses)

Subnet D: 214.97.254.0/31 (2 addresses)
Subnet E: 214.97.254.2/31 (2 addresses)
Subnet F: 214.97.254.4/30 (4 addresses)
- b) To simplify the solution, assume that no datagrams have router interfaces as ultimate destinations. Also, label D, E, F for the upper-right, bottom, and upper-left interior subnets, respectively.

Router 1

Longest Prefix Match

11010110 01100001 11111111
11010110 01100001 11111110 00000000
11010110 01100001 11111110 0000001

Outgoing Interface

Subnet A
Subnet D
Subnet F

Router 2

Longest Prefix Match

11010110 01100001 11111111 00000000
11010110 01100001 11111110 0
11010110 01100001 11111110 0000001

Outgoing Interface

Subnet D
Subnet B
Subnet E

Router 3

Longest Prefix Match

Outgoing Interface

11010110 01100001 11111111 000001	Subnet F
11010110 01100001 11111110 0000001	Subnet E
11010110 01100001 11111110 1	Subnet C

Problem 18

The IP address blocks of Polytechnic Institute of New York University are:

NetRange: 128.238.0.0 - 128.238.255.255

CIDR: 128.238.0.0/16

The IP address blocks Stanford University are:

NetRange: 171.64.0.0 - 171.67.255.255

CIDR: 171.64.0.0/14

The IP address blocks University of Washington are:

NetRange: 140.142.0.0 - 140.142.255.255

CIDR: 140.142.0.0/16

No, the whois services cannot be used to determine with certainty the geographical location of a specific IP address.

www.maxmind.com is used to determine the locations of the Web servers at Polytechnic Institute of New York University, Stanford University and University of Washington.

Locations of the Web server at Polytechnic Institute of New York University is

Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
128.238.24.30	US	United States	NY	New York	Brooklyn	11201	40.6944	-73.9906	Polytechnic University	Polytechnic University	501	718

Locations of the Web server Stanford University is

Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
171.64.13.26	US	United States	CA	California	Stanford	94305	37.4178	-122.1720	Stanford University	Stanford University	807	650

Locations of the Web server at University of Massachusetts is

Hostname	Country Code	Country Name	Region	Region Name	City	Postal Code	Latitude	Longitude	ISP	Organization	Metro Code	Area Code
128.119.103.148	US	United States	MA	Massachusetts	Amherst	01003	42.3896	-72.4534	University of Massachusetts	University of Massachusetts	543	413

Problem 19

The maximum size of data field in each fragment = 680 (because there are 20 bytes IP header). Thus the number of required fragments = $\left\lceil \frac{2400 - 20}{680} \right\rceil = 4$

Each fragment will have Identification number 422. Each fragment except the last one will be of size 700 bytes (including IP header). The last datagram will be of size 360 bytes (including IP header). The offsets of the 4 fragments will be 0, 85, 170, 255. Each of the first 3 fragments will have flag=1; the last fragment will have flag=0.

Problem 20

MP3 file size = 5 million bytes. Assume the data is carried in TCP segments, with each TCP segment also having 20 bytes of header. Then each datagram can carry 1500-40=1460 bytes of the MP3 file

Number of datagrams required = $\left\lceil \frac{5 \times 10^6}{1460} \right\rceil = 3425$. All but the last datagram will be 1,500

bytes; the last datagram will be 960+40 = 1000 bytes. Note that here there is no fragmentation – the source host does not create datagrams larger than 1500 bytes, and these datagrams are smaller than the MTUs of the links.

Problem 21

- a) Home addresses: 192.168.1.1, 192.168.1.2, 192.168.1.3 with the router interface being 192.168.1.4
b)

NAT Translation Table

WAN Side	LAN Side
24.34.112.235, 4000	192.168.1.1, 3345
24.34.112.235, 4001	192.168.1.1, 3346
24.34.112.235, 4002	192.168.1.2, 3445
24.34.112.235, 4003	192.168.1.2, 3446
24.34.112.235, 4004	192.168.1.3, 3545
24.34.112.235, 4005	192.168.1.3, 3546

Problem 22

- a) Since all IP packets are sent outside, so we can use a packet sniffer to record all IP packets generated by the hosts behind a NAT. As each host generates a sequence of IP packets with sequential numbers and a distinct (very likely, as they are randomly chosen from a large space) initial identification number (ID), we can group IP packets with consecutive IDs into a cluster. The number of clusters is the number of hosts behind the NAT.

For more practical algorithms, see the following papers.

“A Technique for Counting NATted Hosts”, by Steven M. Bellovin, appeared in IMW’02, Nov. 6-8, 2002, Marseille, France.

“Exploiting the IPID field to infer network path and end-system characteristics.”

Weifeng Chen, Yong Huang, Bruno F. Ribeiro, Kyoungwon Suh, Honggang Zhang, Edmundo de Souza e Silva, Jim Kurose, and Don Towsley.

PAM’05 Workshop, March 31 - April 01, 2005. Boston, MA, USA.

- b) However, if those identification numbers are not sequentially assigned but randomly assigned, the technique suggested in part (a) won’t work, as there won’t be clusters in sniffed data.

Problem 23

It is not possible to devise such a technique. In order to establish a direct TCP connection between Arnold and Bernard, either Arnold or Bob must initiate a connection to the other. But the NATs covering Arnold and Bob drop SYN packets arriving from the WAN side. Thus neither Arnold nor Bob can initiate a TCP connection to the other if they are both behind NATs.

Problem 24

y-x-u, y-x-v-u, y-x-w-u, y-x-w-v-u,
y-w-u, y-w-v-u, y-w-x-u, y-w-x-v-u, y-w-v-x-u,
y-z-w-u, y-z-w-v-u, y-z-w-x-u, y-z-w-x-v-u, y-z-w-v-x-u,

Problem 25

x to z:

x-y-z, x-y-w-z,
x-w-z, x-w-y-z,
x-v-w-z, x-v-w-y-z,
x-u-w-z, x-u-w-y-z,
x-u-v-w-z, x-u-v-w-y-z

z to u:

z-w-u,
z-w-v-u, z-w-x-u, z-w-v-x-u, z-w-x-v-u, z-w-y-x-u, z-w-y-x-v-u,
z-y-x-u, z-y-x-v-u, z-y-x-w-u, z-y-x-w-y-u, z-y-x-v-w-u,
z-y-w-v-u, z-y-w-x-u, z-y-w-v-x-u, z-y-w-x-v-u, z-y-w-y-x-u, z-y-w-y-x-v-u

z to w:

z-w, z-y-w, z-y-x-w, z-y-x-v-w, z-y-x-u-w, z-y-x-u-v-w, z-y-x-v-u-w

Problem 26

Step	N'	$D(t), p(t)$	$D(u), p(u)$	$D(v), p(v)$	$D(w), p(w)$	$D(y), p(y)$	$D(z), p(z)$
0	x	∞	∞	3,x	6,x	6,x	8,x
1	xv	7,v	6,v	3,x	6,x	6,x	8,x
2	xvu	7,v	6,v	3,x	6,x	6,x	8,x
3	xvuuv	7,v	6,v	3,x	6,x	6,x	8,x
4	xvuuvy	7,v	6,v	3,x	6,x	6,x	8,x
5	xvuuvyt	7,v	6,v	3,x	6,x	6,x	8,x
6	xvuuvytz	7,v	6,v	3,x	6,x	6,x	8,x

Problem 27

a)

Step	N'	$D(x), p(x)$	$D(u), p(u)$	$D(v), p(v)$	$D(w), p(w)$	$D(y), p(y)$	$D(z), p(z)$
0	t	∞	2,t	4,t	∞	7,t	∞
1	tu	∞	2,t	4,t	5,u	7,t	∞
2	tuv	7,v	2,t	4,t	5,u	7,t	∞
3	tuvw	7,v	2,t	4,t	5,u	7,t	∞
4	tuvwxy	7,v	2,t	4,t	5,u	7,t	15,x
5	tuvwxyt	7,v	2,t	4,t	5,u	7,t	15,x
6	tuvwxytz	7,v	2,t	4,t	5,u	7,t	15,x

b)

Step	N'	$D(x), p(x)$	$D(t), p(t)$	$D(v), p(v)$	$D(w), p(w)$	$D(y), p(y)$	$D(z), p(z)$
	u	∞	2,u	3,u	3,u	∞	∞
	ut	∞	2,u	3,u	3,u	9,t	∞
	utv	6,v	2,u	3,u	3,u	9,t	∞
	utvw	6,v	2,u	3,u	3,u	9,t	∞
	utvwxy	6,v	2,u	3,u	3,u	9,t	14,x
	utvwxyt	6,v	2,u	3,u	3,u	9,t	14,x
	utvwxytz	6,v	2,u	3,u	3,u	9,t	14,x

c)

Step	N'	$D(x), p(x)$	$D(u), p(u)$	$D(t), p(t)$	$D(w), p(w)$	$D(y), p(y)$	$D(z), p(z)$
	v	3,v	3,v	4,v	4,v	8,v	∞
	vx	3,v	3,v	4,v	4,v	8,v	11,x
	vxu	3,v	3,v	4,v	4,v	8,v	11,x
	vxut	3,v	3,v	4,v	4,v	8,v	11,x
	vxutw	3,v	3,v	4,v	4,v	8,v	11,x
	vxutwy	3,v	3,v	4,v	4,v	8,v	11,x
	vxutwyz	3,v	3,v	4,v	4,v	8,v	11,x

d)

Step	N'	$D(x), p(x)$	$D(u), p(u)$	$D(v), p(v)$	$D(t), p(t)$	$D(y), p(y)$	$D(z), p(z)$
	w	6,w	3,w	4,w	∞	∞	∞
	wu	6,w	3,w	4,w	5,u	∞	∞
	wuv	6,w	3,w	4,w	5,u	12,v	∞
	wuvt	6,w	3,w	4,w	5,u	12,v	∞
	wuvtx	6,w	3,w	4,w	5,u	12,v	14,x
	wuvtxy	6,w	3,w	4,w	5,u	12,v	14,x
	wuvtxyz	6,w	3,w	4,w	5,u	12,v	14,x

e)

Step	N'	$D(x), p(x)$	$D(u), p(u)$	$D(v), p(v)$	$D(w), p(w)$	$D(t), p(t)$	$D(z), p(z)$
	y	6,y	∞	8,y	∞	7,y	12,y
	yx	6,y	∞	8,y	12,x	7,y	12,y
	yxt	6,y	9,t	8,y	12,x	7,y	12,y
	yxtv	6,y	9,t	8,y	12,x	7,y	12,y
	yxtvu	6,y	9,t	8,y	12,x	7,y	12,y
	yxtvuw	6,y	9,t	8,y	12,x	7,y	12,y
	yxtvuwx	6,y	9,t	8,y	12,x	7,y	12,y

f)

Step	N'	$D(x), p(x)$	$D(u), p(u)$	$D(v), p(v)$	$D(w), p(w)$	$D(y), p(y)$	$D(t), p(t)$
	z	8,z	∞	∞	∞	12,z	∞
	zx	8,z	∞	11,x	14,x	12,z	∞
	zxv	8,z	14,v	11,x	14,x	12,z	15,v
	zxvy	8,z	14,v	11,x	14,x	12,z	15,v
	zxvyu	8,z	14,v	11,x	14,x	12,z	15,v

zxvyuw	8,z	14,v	11,x	14,x	12,z	15,v
zxvyuwt	8,z	14,v	11,x	14,x	12,z	15,v

Problem 28

		Cost to				
		u	v	x	y	z
From	v	∞	∞	∞	∞	∞
	x	∞	∞	∞	∞	∞
	z	∞	6	2	∞	0

		Cost to				
		u	v	x	y	z
From	v	1	0	3	∞	6
	x	∞	3	0	3	2
	z	7	5	2	5	0

		Cost to				
		u	v	x	y	z
From	v	1	0	3	3	5
	x	4	3	0	3	2
	z	6	5	2	5	0

		Cost to				
		u	v	x	y	z
From	v	1	0	3	3	5
	x	4	3	0	3	2
	z	6	5	2	5	0

Problem 29

The wording of this question was a bit ambiguous. We meant this to mean, “the number of iterations from when the algorithm is run for the first time” (that is, assuming the only information the nodes initially have is the cost to their nearest neighbors). We assume

that the algorithm runs synchronously (that is, in one step, all nodes compute their distance tables at the same time and then exchange tables).

At each iteration, a node exchanges distance tables with its neighbors. Thus, if you are node A, and your neighbor is B, all of B's neighbors (which will all be one or two hops from you) will know the shortest cost path of one or two hops to you after one iteration (i.e., after B tells them its cost to you).

Let d be the “diameter” of the network - the length of the longest path without loops between any two nodes in the network. Using the reasoning above, after $d - 1$ iterations, all nodes will know the shortest path cost of d or fewer hops to all other nodes. Since any path with greater than d hops will have loops (and thus have a greater cost than that path with the loops removed), the algorithm will converge in at most $d - 1$ iterations.

ASIDE: if the DV algorithm is run as a result of a change in link costs, there is no a priori bound on the number of iterations required until convergence unless one also specifies a bound on link costs.

Problem 30

a) $D_x(w) = 2$, $D_x(y) = 4$, $D_x(u) = 7$

b) First consider what happens if $c(x,y)$ changes. If $c(x,y)$ becomes larger or smaller (as long as $c(x,y) \geq 1$), the least cost path from x to u will still have cost at least 7. Thus a change in $c(x,y)$ (if $c(x,y) \geq 1$) will not cause x to inform its neighbors of any changes.

If $c(x,y) = \delta < 1$, then the least cost path now passes through y and has cost $\delta + 6$.

Now consider if $c(x,w)$ changes. If $c(x,w) = \epsilon \leq 1$, then the least-cost path to u continues to pass through w and its cost changes to $5 + \epsilon$; x will inform its neighbors of this new cost. If $c(x,w) = \delta > 6$, then the least cost path now passes through y and has cost 11; again x will inform its neighbors of this new cost.

c) Any change in link cost $c(x,y)$ (and as long as $c(x,y) \geq 1$) will not cause x to inform its neighbors of a new minimum-cost path to u .

Problem 31

Node x table

Cost to

	x	y	z
x	0	3	4

From	y	∞	∞	∞
	z	∞	∞	∞

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node y table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	3	0	6
	z	∞	∞	∞

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node z table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	∞	∞	∞
	z	4	6	0

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Problem 32

NO, this is because that decreasing link cost won't cause a loop (caused by the next-hop relation of between two nodes of that link). Connecting two nodes with a link is equivalent to decreasing the link weight from infinite to the finite weight.

Problem 33

At each step, each updating of a node's distance vectors is based on the Bellman-Ford equation, i.e., only decreasing those values in its distance vector. There is no increasing in values. If no updating, then no message will be sent out. Thus, $D(x)$ is non-increasing. Since those costs are finite, then eventually distance vectors will be stabilized in finite steps.

Problem 34

a)

Router z	Informs w, $D_z(x)=\infty$
	Informs y, $D_z(x)=6$
Router w	Informs y, $D_w(x)=\infty$
	Informs z, $D_w(x)=5$
Router y	Informs w, $D_y(x)=4$
	Informs z, $D_y(x)=4$

b) Yes, there will be a count-to-infinity problem. The following table shows the routing converging process. Assume that at time t_0 , link cost change happens. At time t_1 , y updates its distance vector and informs neighbors w and z. In the following table, “ \rightarrow ” stands for “informs”.

time	t_0	t_1	t_2	t_3	t_4
Z	\rightarrow w, $D_z(x)=\infty$ \rightarrow y, $D_z(x)=6$		No change	\rightarrow w, $D_z(x)=\infty$ \rightarrow y, $D_z(x)=11$	
W	\rightarrow y, $D_w(x)=\infty$ \rightarrow z, $D_w(x)=5$		\rightarrow y, $D_w(x)=\infty$ \rightarrow z, $D_w(x)=10$		No change
Y	\rightarrow w, $D_y(x)=4$ \rightarrow z, $D_y(x)=4$	\rightarrow w, $D_y(x)=9$ \rightarrow z, $D_y(x)=\infty$		No change	\rightarrow w, $D_y(x)=14$ \rightarrow z, $D_y(x)=\infty$

We see that w, y, z form a loop in their computation of the costs to router x. If we continue the iterations shown in the above table, then we will see that, at t_{27} , z detects that its least cost to x is 50, via its direct link with x. At t_{29} , w learns its least cost to x is 51 via z. At t_{30} , y updates its least cost to x to be 52 (via w). Finally, at time t_{31} , no updating, and the routing is stabilized.

time	t_{27}	t_{28}	t_{29}	t_{30}	t_{31}
Z	\rightarrow w, $D_z(x)=50$ \rightarrow y, $D_z(x)=50$				via w, ∞ via y, 55 via z, 50
W		\rightarrow y, $D_w(x)=\infty$ \rightarrow z, $D_w(x)=50$	\rightarrow y, $D_w(x)=51$ \rightarrow z, $D_w(x)=\infty$		via w, ∞ via y, ∞ via z, 51
Y		\rightarrow w, $D_y(x)=53$ \rightarrow z, $D_y(x)=\infty$		\rightarrow w, $D_y(x)=\infty$ \rightarrow z, $D_y(x)=52$	via w, 52 via y, 60 via z, 53

- c) cut the link between y and z.

Problem 35

Since full AS path information is available from an AS to a destination in BGP, loop detection is simple – if a BGP peer receives a route that contains its own AS number in the AS path, then using that route would result in a loop.

Problem 36

The chosen path is not necessarily the shortest AS-path. Recall that there are many issues to be considered in the route selection process. It is very likely that a longer loop-free path is preferred over a shorter loop-free path due to economic reason. For example, an AS might prefer to send traffic to one neighbor instead of another neighbor with shorter AS distance.

Problem 37

- a) eBGP
- b) iBGP
- c) eBGP
- d) iBGP

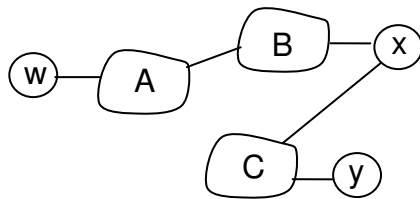
Problem 38

- a) I1 because this interface begins the least cost path from 1d towards the gateway router 1c.
- b) I2. Both routes have equal AS-PATH length but I2 begins the path that has the closest NEXT-HOP router.
- c) I1. I1 begins the path that has the shortest AS-PATH.

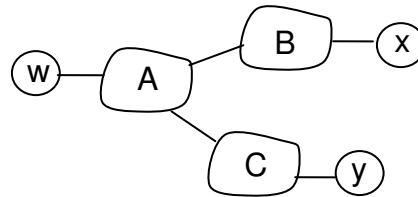
Problem 39

One way for C to force B to hand over all of B's traffic to D on the east coast is for C to only advertise its route to D via its east coast peering point with C.

Problem 40



X's view of the topology



W's view of the topology

In the above solution, X does not know about the AC link since X does not receive an advertised route to w or to y that contain the AC link (i.e., X receives no advertisement containing both AS A and AS C on the path to a destination).

Problem 41

BitTorrent file sharing and Skype P2P applications.

Consider a BitTorrent file sharing network in which peer 1, 2, and 3 are in stub networks W, X, and Y respectively. Due the mechanism of BitTorrent's file sharing, it is quire possible that peer 2 gets data chunks from peer 1 and then forwards those data chunks to 3. This is equivalent to B forwarding data that is finally destined to stub network Y.

Problem 42

A should advise to B two routes, AS-paths A-W and A-V.

A should advise to C only one route, A-V.

C receives AS paths: B-A-W, B-A-V, A-V.

Problem 43

Since Z wants to transit Y's traffic, Z will send route advertisements to Y. In this manner, when Y has a datagram that is destined to an IP that can be reached through Z, Y will have the option of sending the datagram through Z. However, if Z advertizes routes to Y, Y can re-advertize those routes to X. Therefore, in this case, there is nothing Z can do from preventing traffic from X to transit through Z.

Problem 44

The minimal spanning tree has z connected to y via x at a cost of 14(=8+6).

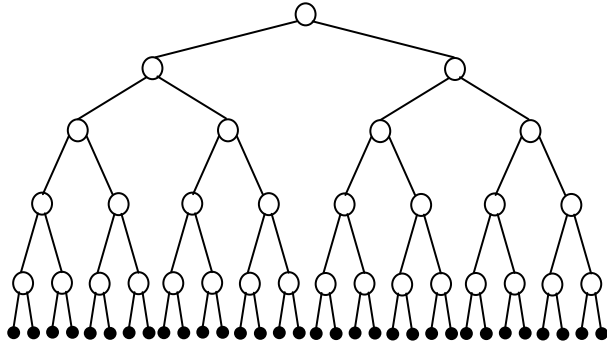
z connected to v via x at a cost of 11(=8+3);

z connected to u via x and v, at a cost of 14(=8+3+3);

z connected to w via x, v, and u, at a cost of 17(=8+3+3+3).

This can be obtained by Prim's algorithm to grow a minimum spanning tree.

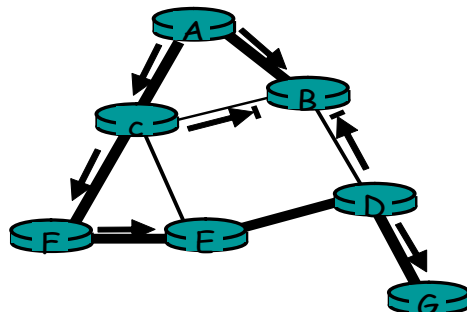
Problem 45



The 32 receivers are shown connected to the sender in the binary tree configuration shown above. With network-layer broadcast, a copy of the message is forwarded over each link exactly once. There are thus 62 link crossings ($2+4+8+16+32$). With unicast emulation, the sender unicasts a copy to each receiver over a path with 5 hops. There are thus 160 link crossings (5×32).

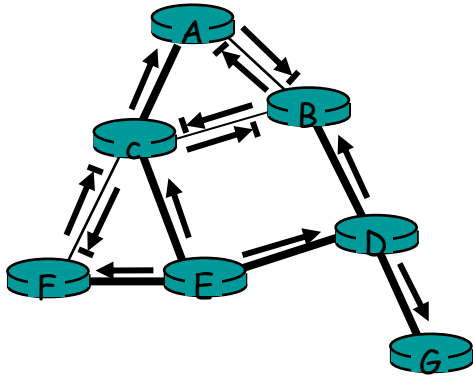
A topology in which all receivers are in a line, with the sender at one end of the line, will have the largest disparity between the cost of network-layer broadcast and unicast emulation.

Problem 46

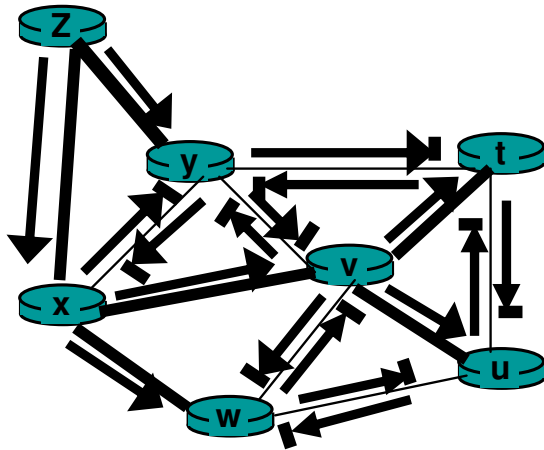


The thicker shaded lines represent the shortest path tree from A to all destinations. Other solutions are possible, but in these solutions, B can not route to either C or D from A.

Problem 47



Problem 48



Problem 49

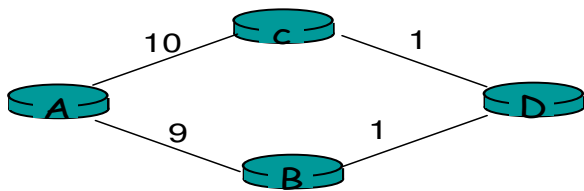
The center-based tree for the topology shown in the original figure connects A to C; B to C; E to C; and F to C (all directly). D connects to C via E, and G connects to C via D, E. This center-based tree is different from the minimal spanning tree shown in the figure.

Problem 50

The center-based tree for the topology shown in the original figure connects t to v; u to v; w to v; x to v; and y to v (all directly). And z connected to v via x. This center-based tree is different from the minimal spanning tree.

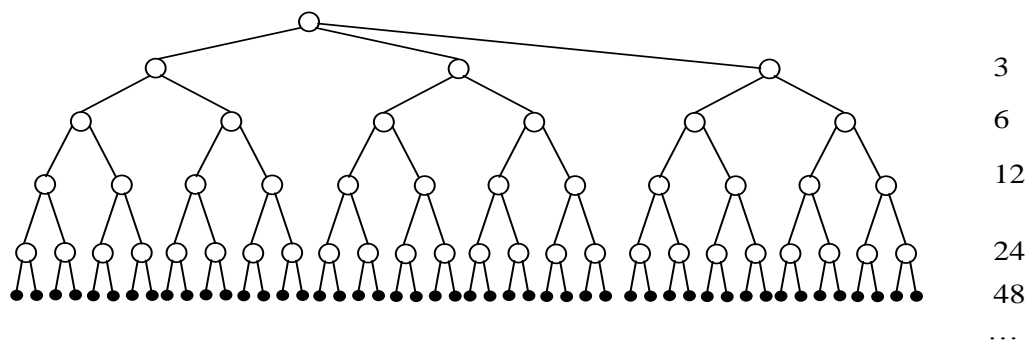
Problem 51

Dijkstra's algorithm for the network below, with node A as the source, results in a least-unicast-cost path tree of links AC, AB, and BD, with an overall free cost of 20. The minimum spanning tree contains links AB, BD, and DC, at a cost of 11.



Problem 52

After 1 step 3 copies are transmitted, after 2 steps 6 copies are transmitted. After 3 steps, 12 copies are transmitted, and so on. After k steps, $3 \cdot 2^{k-1}$ copies will be transmitted in that step.



Problem 53

The protocol must be built at the application layer. For example, an application may periodically multicast its identity to all other group members in an application-layer message.

Problem 54

A simple application-layer protocol that will allow all members to know the identity of all other members in the group is for each instance of the application to send a multicast message containing its identity to all other members. This protocol sends message in-band, since the multicast channel is used to distribute the identification messages as well as multicast data from the application itself. The use of the in-band signaling makes use of the existing multicast distribution mechanism, leading to a very simple design.

Problem 55

$32 - 4 = 28$ bits are available for multicast addresses. Thus, the size of the multicast address space is $N = 2^{28}$.

The probability that two groups choose the same address is

$$\frac{1}{N} = 2^{-28} = 3.73 \cdot 10^{-9}$$

The probability that 1000 groups all have different addresses is

$$\frac{N \cdot (N-1) \cdot (N-2) \cdots (N-999)}{N^{1000}} = \left(1 - \frac{1}{N}\right) \left(1 - \frac{2}{N}\right) \cdots \left(1 - \frac{999}{N}\right)$$

Ignoring cross-product terms, this is approximately equal to

$$1 - \left(\frac{1 + 2 + \cdots + 999}{N}\right) = 1 - \frac{999 \cdot 1000}{2N} = 0.998$$

some networking texts) thus requires a firm foundation in all layers of the protocol stack—a foundation that our study of the link layer has now completed!



Homework Problems and Questions

Chapter 5 Review Questions

SECTIONS 5.1–5.2

- R1. Consider the transportation analogy in Section 5.1.1. If the passenger is analogous to a datagram, what is analogous to the link layer frame?
- R2. If all the links in the Internet were to provide reliable delivery service, would the TCP reliable delivery service be redundant? Why or why not?
- R3. What are some of the possible services that a link-layer protocol can offer to the network layer? Which of these link-layer services have corresponding services in IP? In TCP?

SECTION 5.3

- R4. Suppose two nodes start to transmit at the same time a packet of length L over a broadcast channel of rate R . Denote the propagation delay between the two nodes as d_{prop} . Will there be a collision if $d_{\text{prop}} < L/R$? Why or why not?
- R5. In Section 5.3, we listed four desirable characteristics of a broadcast channel. Which of these characteristics does slotted ALOHA have? Which of these characteristics does token passing have?
- R6. In CSMA/CD, after the fifth collision, what is the probability that a node chooses $K = 4$? The result $K = 4$ corresponds to a delay of how many seconds on a 10 Mbps Ethernet?
- R7. Describe polling and token-passing protocols using the analogy of cocktail party interactions.
- R8. Why would the token-ring protocol be inefficient if a LAN had a very large perimeter?

SECTION 5.4

- R9. How big is the MAC address space? The IPv4 address space? The IPv6 address space?
- R10. Suppose nodes A, B, and C each attach to the same broadcast LAN (through their adapters). If A sends thousands of IP datagrams to B with each encapsulating frame addressed to the MAC address of B, will C's adapter process these frames? If so, will C's adapter pass the IP datagrams in these frames to the network layer C? How would your answers change if A sends frames with the MAC broadcast address?

- R11. Why is an ARP query sent within a broadcast frame? Why is an ARP response sent within a frame with a specific destination MAC address?
- R12. For the network in Figure 5.19, the router has two ARP modules, each with its own ARP table. Is it possible that the same MAC address appears in both tables?
- R13. Compare the frame structures for 10BASE-T, 100BASE-T, and Gigabit Ethernet. How do they differ?
- R14. Consider Figure 5.15. How many subnetworks are there, in the addressing sense of Section 4.4?
- R15. What is the maximum number of VLANs that can be configured on a switch supporting the 802.1Q protocol? Why?
- R16. Suppose that N switches supporting K VLAN groups are to be connected via a trunking protocol. How many ports are needed to connect the switches? Justify your answer.



Problems

- P1. Suppose the information content of a packet is the bit pattern 1110 0110 1001 1101 and an even parity scheme is being used. What would the value of the field containing the parity bits be for the case of a two-dimensional parity scheme? Your answer should be such that a minimum-length checksum field is used.
- P2. Show (give an example other than the one in Figure 5.5) that two-dimensional parity checks can correct and detect a single bit error. Show (give an example of) a double-bit error that can be detected but not corrected.
- P3. Suppose the information portion of a packet (D in Figure 5.3) contains 10 bytes consisting of the 8-bit unsigned binary ASCII representation of string “Networking.” Compute the Internet checksum for this data.
- P4. Consider the previous problem, but instead suppose these 10 bytes contain
 - a. the binary representation of the numbers 1 through 10.
 - b. the ASCII representation of the letters B through K (uppercase).
 - c. the ASCII representation of the letters b through k (lowercase).Compute the Internet checksum for this data.
- P5. Consider the 7-bit generator, $G=10011$, and suppose that D has the value 1010101010. What is the value of R ?
- P6. Consider the previous problem, but suppose that D has the value
 - a. 1001010101.
 - b. 0101101010.
 - c. 1010100000.

- P7. In this problem, we explore some of the properties of the CRC. For the generator $G (=1001)$ given in Section 5.2.3, answer the following questions.
- Why can it detect any single bit error in data D?
 - Can the above G detect any odd number of bit errors? Why?
- P8. In Section 5.3, we provided an outline of the derivation of the efficiency of slotted ALOHA. In this problem we'll complete the derivation.
- Recall that when there are N active nodes, the efficiency of slotted ALOHA is $Np(1-p)^{N-1}$. Find the value of p that maximizes this expression.
 - Using the value of p found in (a), find the efficiency of slotted ALOHA by letting N approach infinity. *Hint:* $(1 - 1/N)^N$ approaches $1/e$ as N approaches infinity.
- P9. Show that the maximum efficiency of pure ALOHA is $1/(2e)$. *Note:* This problem is easy if you have completed the problem above!
- P10. Consider two nodes, A and B, that use the slotted ALOHA protocol to contend for a channel. Suppose node A has more data to transmit than node B, and node A's retransmission probability p_A is greater than node B's retransmission probability, p_B .
- Provide a formula for node A's average throughput. What is the total efficiency of the protocol with these two nodes?
 - If $p_A = 2p_B$, is node A's average throughput twice as large as that of node B? Why or why not? If not, how can you choose p_A and p_B to make that happen?
 - In general, suppose there are N nodes, among which node A has retransmission probability $2p$ and all other nodes have retransmission probability p . Provide expressions to compute the average throughputs of node A and of any other node.
- P11. Suppose four active nodes—nodes A, B, C and D—are competing for access to a channel using slotted ALOHA. Assume each node has an infinite number of packets to send. Each node attempts to transmit in each slot with probability p . The first slot is numbered slot 1, the second slot is numbered slot 2, and so on.
- What is the probability that node A succeeds for the first time in slot 5?
 - What is the probability that some node (either A, B, C or D) succeeds in slot 4?
 - What is the probability that the first success occurs in slot 3?
 - What is the efficiency of this four-node system?
- P12. Graph the efficiency of slotted ALOHA and pure ALOHA as a function of p for the following values of N :
- $N=15$.
 - $N=25$.
 - $N=35$.

- P13. Consider a broadcast channel with N nodes and a transmission rate of R bps. Suppose the broadcast channel uses polling (with an additional polling node) for multiple access. Suppose the amount of time from when a node completes transmission until the subsequent node is permitted to transmit (that is, the polling delay) is d_{poll} . Suppose that within a polling round, a given node is allowed to transmit at most Q bits. What is the maximum throughput of the broadcast channel?
- P14. Consider three LANs interconnected by two routers, as shown in Figure 5.33.
- Assign IP addresses to all of the interfaces. For Subnet 1 use addresses of the form 192.168.1.xxx; for Subnet 2 use addresses of the form 192.168.2.xxx; and for Subnet 3 use addresses of the form 192.168.3.xxx.
 - Assign MAC addresses to all of the adapters.
 - Consider sending an IP datagram from Host E to Host B. Suppose all of the ARP tables are up to date. Enumerate all the steps, as done for the single-router example in Section 5.4.1.
 - Repeat (c), now assuming that the ARP table in the sending host is empty (and the other tables are up to date).
- P15. Consider Figure 5.33. Now we replace the router between subnets 1 and 2 with a switch S1, and label the router between subnets 2 and 3 as R1.

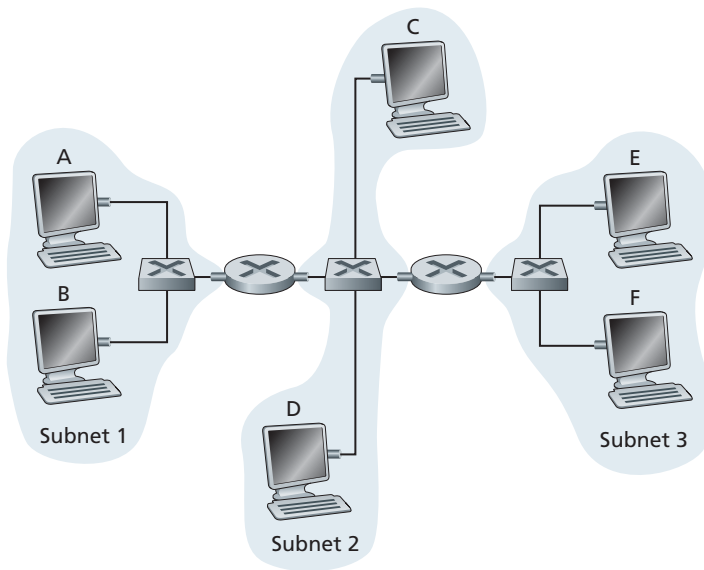


Figure 5.33 ♦ Three subnets, interconnected by routers



VideoNote
Sending a datagram
between subnets: link-
layer and network-layer
addressing

- a. Consider sending an IP datagram from Host E to Host F. Will Host E ask router R1 to help forward the datagram? Why? In the Ethernet frame containing the IP datagram, what are the source and destination IP and MAC addresses?
 - b. Suppose E would like to send an IP datagram to B, and assume that E's ARP cache does not contain B's MAC address. Will E perform an ARP query to find B's MAC address? Why? In the Ethernet frame (containing the IP datagram destined to B) that is delivered to router R1, what are the source and destination IP and MAC addresses?
 - c. Suppose Host A would like to send an IP datagram to Host B, and neither A's ARP cache contains B's MAC address nor does B's ARP cache contain A's MAC address. Further suppose that the switch S1's forwarding table contains entries for Host B and router R1 only. Thus, A will broadcast an ARP request message. What actions will switch S1 perform once it receives the ARP request message? Will router R1 also receive this ARP request message? If so, will R1 forward the message to Subnet 3? Once Host B receives this ARP request message, it will send back to Host A an ARP response message. But will it send an ARP query message to ask for A's MAC address? Why? What will switch S1 do once it receives an ARP response message from Host B?
- P16. Consider the previous problem, but suppose now that the router between subnets 2 and 3 is replaced by a switch. Answer questions (a)–(c) in the previous problem in this new context.
- P17. Recall that with the CSMA/CD protocol, the adapter waits $K \cdot 512$ bit times after a collision, where K is drawn randomly. For $K = 100$, how long does the adapter wait until returning to Step 2 for a 10 Mbps broadcast channel? For a 100 Mbps broadcast channel?
- P18. Suppose nodes A and B are on the same 10 Mbps broadcast channel, and the propagation delay between the two nodes is 325 bit times. Suppose CSMA/CD and Ethernet packets are used for this broadcast channel. Suppose node A begins transmitting a frame and, before it finishes, node B begins transmitting a frame. Can A finish transmitting before it detects that B has transmitted? Why or why not? If the answer is yes, then A incorrectly believes that its frame was successfully transmitted without a collision. *Hint:* Suppose at time $t = 0$ bits, A begins transmitting a frame. In the worst case, A transmits a minimum-sized frame of $512 + 64$ bit times. So A would finish transmitting the frame at $t = 512 + 64$ bit times. Thus, the answer is no, if B's signal reaches A before bit time $t = 512 + 64$ bits. In the worst case, when does B's signal reach A?
- P19. Suppose nodes A and B are on the same 10 Mbps broadcast channel, and the propagation delay between the two nodes is 245 bit times. Suppose A and B send Ethernet frames at the same time, the frames collide, and then A and B choose different values of K in the CSMA/CD algorithm. Assuming

no other nodes are active, can the retransmissions from A and B collide? For our purposes, it suffices to work out the following example. Suppose A and B begin transmission at $t = 0$ bit times. They both detect collisions at $t = 245$ bit times. Suppose $K_A = 0$ and $K_B = 1$. At what time does B schedule its retransmission? At what time does A begin transmission? (*Note:* The nodes must wait for an idle channel after returning to Step 2—see protocol.) At what time does A's signal reach B? Does B refrain from transmitting at its scheduled time?

- P20. In this problem, you will derive the efficiency of a CSMA/CD-like multiple access protocol. In this protocol, time is slotted and all adapters are synchronized to the slots. Unlike slotted ALOHA, however, the length of a slot (in seconds) is much less than a frame time (the time to transmit a frame). Let S be the length of a slot. Suppose all frames are of constant length $L = kRS$, where R is the transmission rate of the channel and k is a large integer. Suppose there are N nodes, each with an infinite number of frames to send. We also assume that $d_{\text{prop}} < S$, so that all nodes can detect a collision before the end of a slot time. The protocol is as follows:

- If, for a given slot, no node has possession of the channel, all nodes contend for the channel; in particular, each node transmits in the slot with probability p . If exactly one node transmits in the slot, that node takes possession of the channel for the subsequent $k - 1$ slots and transmits its entire frame.
- If some node has possession of the channel, all other nodes refrain from transmitting until the node that possesses the channel has finished transmitting its frame. Once this node has transmitted its frame, all nodes contend for the channel.

Note that the channel alternates between two states: the productive state, which lasts exactly k slots, and the nonproductive state, which lasts for a random number of slots. Clearly, the channel efficiency is the ratio of $k/(k + x)$, where x is the expected number of consecutive unproductive slots.

- a. For fixed N and p , determine the efficiency of this protocol.
 - b. For fixed N , determine the p that maximizes the efficiency.
 - c. Using the p (which is a function of N) found in (b), determine the efficiency as N approaches infinity.
 - d. Show that this efficiency approaches 1 as the frame length becomes large.
- P21. Consider Figure 5.33 in problem P14. Provide MAC addresses and IP addresses for the interfaces at Host A, both routers, and Host F. Suppose Host A sends a datagram to Host F. Give the source and destination MAC addresses in the frame encapsulating this IP datagram as the frame is transmitted (i) from A to the left router, (ii) from the left router to the right router, (iii) from the right router to F. Also give the source and destination IP addresses in the IP datagram encapsulated within the frame at each of these points in time.

- P22. Suppose now that the leftmost router in Figure 5.33 is replaced by a switch. Hosts A, B, C, and D and the right router are all star-connected into this switch. Give the source and destination MAC addresses in the frame encapsulating this IP datagram as the frame is transmitted (i) from A to the switch, (ii) from the switch to the right router, (iii) from the right router to F. Also give the source and destination IP addresses in the IP datagram encapsulated within the frame at each of these points in time.
- P23. Consider Figure 5.15. Suppose that all links are 100 Mbps. What is the maximum total aggregate throughput that can be achieved among the 9 hosts and 2 servers in this network? You can assume that any host or server can send to any other host or server. Why?
- P24. Suppose the three departmental switches in Figure 5.15 are replaced by hubs. All links are 100 Mbps. Now answer the questions posed in problem P23.
- P25. Suppose that *all* the switches in Figure 5.15 are replaced by hubs. All links are 100 Mbps. Now answer the questions posed in problem P23.
- P26. Let's consider the operation of a learning switch in the context of a network in which 6 nodes labeled A through F are star connected into an Ethernet switch. Suppose that (i) B sends a frame to E, (ii) E replies with a frame to B, (iii) A sends a frame to B, (iv) B replies with a frame to A. The switch table is initially empty. Show the state of the switch table before and after each of these events. For each of these events, identify the link(s) on which the transmitted frame will be forwarded, and briefly justify your answers.
- P27. In this problem, we explore the use of small packets for Voice-over-IP applications. One of the drawbacks of a small packet size is that a large fraction of link bandwidth is consumed by overhead bytes. To this end, suppose that the packet consists of P bytes and 5 bytes of header.
- Consider sending a digitally encoded voice source directly. Suppose the source is encoded at a constant rate of 128 kbps. Assume each packet is entirely filled before the source sends the packet into the network. The time required to fill a packet is the **packetization delay**. In terms of L , determine the packetization delay in milliseconds.
 - Packetization delays greater than 20 msec can cause a noticeable and unpleasant echo. Determine the packetization delay for $L = 1,500$ bytes (roughly corresponding to a maximum-sized Ethernet packet) and for $L = 50$ (corresponding to an ATM packet).
 - Calculate the store-and-forward delay at a single switch for a link rate of $R = 622$ Mbps for $L = 1,500$ bytes, and for $L = 50$ bytes.
 - Comment on the advantages of using a small packet size.

- P28. Consider the single switch VLAN in Figure 5.25, and assume an external router is connected to switch port 1. Assign IP addresses to the EE and CS hosts and router interface. Trace the steps taken at both the network layer and the link layer to transfer an IP datagram from an EE host to a CS host (*Hint*: reread the discussion of Figure 5.19 in the text).
- P29. Consider the MPLS network shown in Figure 5.29, and suppose that routers R5 and R6 are now MPLS enabled. Suppose that we want to perform traffic engineering so that packets from R6 destined for A are switched to A via R6-R4-R3-R1, and packets from R5 destined for A are switched via R5-R4-R2-R1. Show the MPLS tables in R5 and R6, as well as the modified table in R4, that would make this possible.
- P30. Consider again the same scenario as in the previous problem, but suppose that packets from R6 destined for D are switched via R6-R4-R3, while packets from R5 destined to D are switched via R4-R2-R1-R3. Show the MPLS tables in all routers that would make this possible.
- P31. In this problem, you will put together much of what you have learned about Internet protocols. Suppose you walk into a room, connect to Ethernet, and want to download a Web page. What are all the protocol steps that take place, starting from powering on your PC to getting the Web page? Assume there is nothing in our DNS or browser caches when you power on your PC. (*Hint*: the steps include the use of Ethernet, DHCP, ARP, DNS, TCP, and HTTP protocols.) Explicitly indicate in your steps how you obtain the IP and MAC addresses of a gateway router.
- P32. Consider the data center network with hierarchical topology in Figure 5.30. Suppose now there are 80 pairs of flows, with ten flows between the first and ninth rack, ten flows between the second and tenth rack, and so on. Further suppose that all links in the network are 10 Gbps, except for the links between hosts and TOR switches, which are 1 Gbps.
- Each flow has the same data rate; determine the maximum rate of a flow.
 - For the same traffic pattern, determine the maximum rate of a flow for the highly interconnected topology in Figure 5.31.
 - Now suppose there is a similar traffic pattern, but involving 20 hosts on each hosts and 160 pairs of flows. Determine the maximum flow rates for the two topologies.
- P33. Consider the hierarchical network in Figure 5.30 and suppose that the data center needs to support email and video distribution among other applications. Suppose four racks of servers are reserved for email and four racks are reserved for video. For each of the applications, all four racks must lie below a single tier-2 switch since the tier-2 to tier-1 links do not have sufficient bandwidth to support the intra-application traffic. For the email application,

suppose that for 99.9 percent of the time only three racks are used, and that the video application has identical usage patterns.

- a. For what fraction of time does the email application need to use a fourth rack? How about for the video application?
- b. Assuming email usage and video usage are independent, for what fraction of time do (equivalently, what is the probability that) both applications need their fourth rack?
- c. Suppose that it is acceptable for an application to have a shortage of servers for 0.001 percent of time or less (causing rare periods of performance degradation for users).

Discuss how the topology in Figure 5.31 can be used so that only seven racks are collectively assigned to the two applications (assuming that the topology can support all the traffic).



Wireshark Labs

At the companion Web site for this textbook, <http://www.awl.com/kurose-ross>, you'll find a Wireshark lab that examines the operation of the IEEE 802.3 protocol and the Wireshark frame format. A second Wireshark lab examines packet traces taken in a home network scenario.

Chapter 5 Review Questions

1. The transportation mode, e.g., car, bus, train, car.
2. Although each link guarantees that an IP datagram sent over the link will be received at the other end of the link without errors, it is not guaranteed that IP datagrams will arrive at the ultimate destination in the proper order. With IP, datagrams in the same TCP connection can take different routes in the network, and therefore arrive out of order. TCP is still needed to provide the receiving end of the application the byte stream in the correct order. Also, IP can lose packets due to routing loops or equipment failures.
3. Framing: there is also framing in IP and TCP; link access; reliable delivery: there is also reliable delivery in TCP; flow control: there is also flow control in TCP; error detection: there is also error detection in IP and TCP; error correction; full duplex: TCP is also full duplex.
4. There will be a collision in the sense that while a node is transmitting it will start to receive a packet from the other node.
5. Slotted Aloha: 1, 2 and 4 (slotted ALOHA is only partially decentralized, since it requires the clocks in all nodes to be synchronized). Token ring: 1, 2, 3, 4.
6. After the 5th collision, the adapter chooses from $\{0, 1, 2, \dots, 31\}$. The probability that it chooses 4 is $1/32$. It waits 204.8 microseconds.
7. In polling, a discussion leader allows only one participant to talk at a time, with each participant getting a chance to talk in a round-robin fashion. For token ring, there isn't a discussion leader, but there is wine glass that the participants take turns holding. A participant is only allowed to talk if the participant is holding the wine glass.
8. When a node transmits a frame, the node has to wait for the frame to propagate around the entire ring before the node can release the token. Thus, if L/R is small as compared to t_{prop} , then the protocol will be inefficient.
9. 2^{48} MAC addresses; 2^{32} IPv4 addresses; 2^{128} IPv6 addresses.
10. C's adapter will process the frames, but the adapter will not pass the datagrams up the protocol stack. If the LAN broadcast address is used, then C's adapter will both process the frames and pass the datagrams up the protocol stack.
11. An ARP query is sent in a broadcast frame because the querying host does not which adapter address corresponds to the IP address in question. For the response, the sending node knows the adapter address to which the response should be sent, so

there is no need to send a broadcast frame (which would have to be processed by all the other nodes on the LAN).

12. No it is not possible. Each LAN has its own distinct set of adapters attached to it, with each adapter having a unique LAN address.
13. The three Ethernet technologies have identical frame structures.
14. 2 (the internal subnet and the external internet)
15. In 802.1Q there is a 12- bit VLAN identifier. Thus $2^{12} = 4,096$ VLANs can be supported.
16. We can string the N switches together. The first and last switch would use one port for trunking; the middle N-2 switches would use two ports. So the total number of ports is $2 + 2(N-2) = 2N-2$ ports.

Chapter 5 Problems

Problem 1

```
1 1 1 0 1
0 1 1 0 0
1 0 0 1 0
1 1 0 1 1
1 1 0 0 0
```

Problem 2

Suppose we begin with the initial two-dimensional parity matrix:

```
0 0 0 0
1 1 1 1
0 1 0 1
1 0 1 0
```

With a bit error in row 2, column 3, the parity of row 2 and column 3 is now wrong in the matrix below:

```
0 0 0 0
1 1 0 1
0 1 0 1
1 0 1 0
```


Now suppose there is a bit error in row 2, column 2 and column 3. The parity of row 2 is now correct! The parity of columns 2 and 3 is wrong, but we can't detect in which rows the error occurred!

```
0 0 0 0
1 0 0 1
0 1 0 1
1 0 1 0
```

The above example shows that a double bit error can be detected (if not corrected).

Problem 3

```
01001100 01101001
+ 01101110 01101011
```

```
-----
10111010 11010100
+ 00100000 01001100
```

```
-----
11011011 00100000
+ 01100001 01111001
```

```
-----
00111100 10011010 (overflow, then wrap around)
+ 01100101 01110010
```

```
-----
10100010 00001100
```

The one's complement of the sum is 01011101 11110011

Problem 4

a) To compute the Internet checksum, we add up the values at 16-bit quantities:

```
00000001 00000010
00000011 00000100
00000101 00000110
00000111 00001000
00001001 00001010
-----
00011001 00011110
```

The one's complement of the sum is 11100110 11100001.

b) To compute the Internet checksum, we add up the values at 16-bit quantities:

```

01000010 01000011
01000100 01000101
01000110 01000111
01001000 01001001
01001010 01001011
-----
10011111 10100100

```

The one's complement of the sum is 01100000 01011011

- c) To compute the Internet checksum, we add up the values at 16-bit quantities:

```

01100010 01100011
01100100 01100101
01100110 01100111
01101000 01101001
01101010 01101011
-----
00000000 00000101

```

The one's complement of the sum is 11111111 11111010.

Problem 5

If we divide 10011 into 1010101010 0000, we get 1011011100, with a remainder of R=0100. Note that, G=10011 is CRC-4-ITU standard.

Problem 6

- a) we get 1000100011, with a remainder of R=0101.
- b) we get 1011111111, with a remainder of R=0001.
- c) we get 0101101110, with a remainder of R=0010.

Problem 7

- a) Without loss of generality, suppose i th bit is flipped, where $0 \leq i \leq d+r-1$ and assume that the least significant bit is 0th bit.

A single bit error means that the received data is $K = D * 2^r \text{ XOR } R + 2^i$. It is clear that if we divide K by G, then the remainder is not zero. In general, if G contains at least two 1's, then a single bit error can always be detected.

- b) The key insight here is that G can be divided by 11 (binary number), but any number of odd-number of 1's cannot be divided by 11. Thus, a sequence (not necessarily contiguous) of odd-number bit errors cannot be divided by 11, thus it cannot be divided by G.

Problem 8

a)

$$E(p) = Np(1-p)^{N-1}$$

$$\begin{aligned} E'(p) &= N(1-p)^{N-1} - Np(N-1)(1-p)^{N-2} \\ &= N(1-p)^{N-2}((1-p) - p(N-1)) \end{aligned}$$

$$E'(p) = 0 \Rightarrow p^* = \frac{1}{N}$$

b)

$$E(p^*) = N \frac{1}{N} \left(1 - \frac{1}{N}\right)^{N-1} = \left(1 - \frac{1}{N}\right)^{N-1} = \frac{\left(1 - \frac{1}{N}\right)^N}{1 - \frac{1}{N}}$$

$$\lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right) = 1 \quad \lim_{N \rightarrow \infty} \left(1 - \frac{1}{N}\right)^N = \frac{1}{e}$$

Thus

$$\lim_{N \rightarrow \infty} E(p^*) = \frac{1}{e}$$

Problem 9

$$E(p) = Np(1-p)^{2(N-1)}$$

$$\begin{aligned} E'(p) &= N(1-p)^{2(N-2)} - Np2(N-1)(1-p)^{2(N-3)} \\ &= N(1-p)^{2(N-3)}((1-p) - p2(N-1)) \end{aligned}$$

$$E'(p) = 0 \Rightarrow p^* = \frac{1}{2N-1}$$

$$E(p^*) = \frac{N}{2N-1} \left(1 - \frac{1}{2N-1}\right)^{2(N-1)}$$

$$\lim_{N \rightarrow \infty} E(p^*) = \frac{1}{2} \cdot \frac{1}{e} = \frac{1}{2e}$$

Problem 10

- a) A's average throughput is given by $p_A(1-p_B)$.
Total efficiency is $p_A(1-p_B) + p_B(1-p_A)$.
- b) A's throughput is $p_A(1-p_B) = 2p_B(1-p_B) = 2p_B - 2(p_B)^2$.
B's throughput is $p_B(1-p_A) = p_B(1-2p_B) = p_B - 2(p_B)^2$.
Clearly, A's throughput is not twice as large as B's.
In order to make $p_A(1-p_B) = 2p_B(1-p_A)$, we need that $p_A = 2 - (p_A / p_B)$.
- c) A's throughput is $2p(1-p)^{N-1}$, and any other node has throughput $p(1-p)^{N-2}(1-2p)$.

Problem 11

- a) $(1 - p(A))^4 p(A)$
where, $p(A)$ = probability that A succeeds in a slot
 $p(A) = p(\text{A transmits and B does not and C does not and D does not})$
 $= p(\text{A transmits}) p(\text{B does not transmit}) p(\text{C does not transmit}) p(\text{D does not transmit})$
 $= p(1 - p)(1 - p)(1 - p) = p(1 - p)^3$

Hence, $p(\text{A succeeds for first time in slot 5})$
 $= (1 - p(A))^4 p(A) = (1 - p(1 - p)^3)^4 p(1 - p)^3$

- b) $p(\text{A succeeds in slot 4}) = p(1-p)^3$
 $p(\text{B succeeds in slot 4}) = p(1-p)^3$
 $p(\text{C succeeds in slot 4}) = p(1-p)^3$
 $p(\text{D succeeds in slot 4}) = p(1-p)^3$

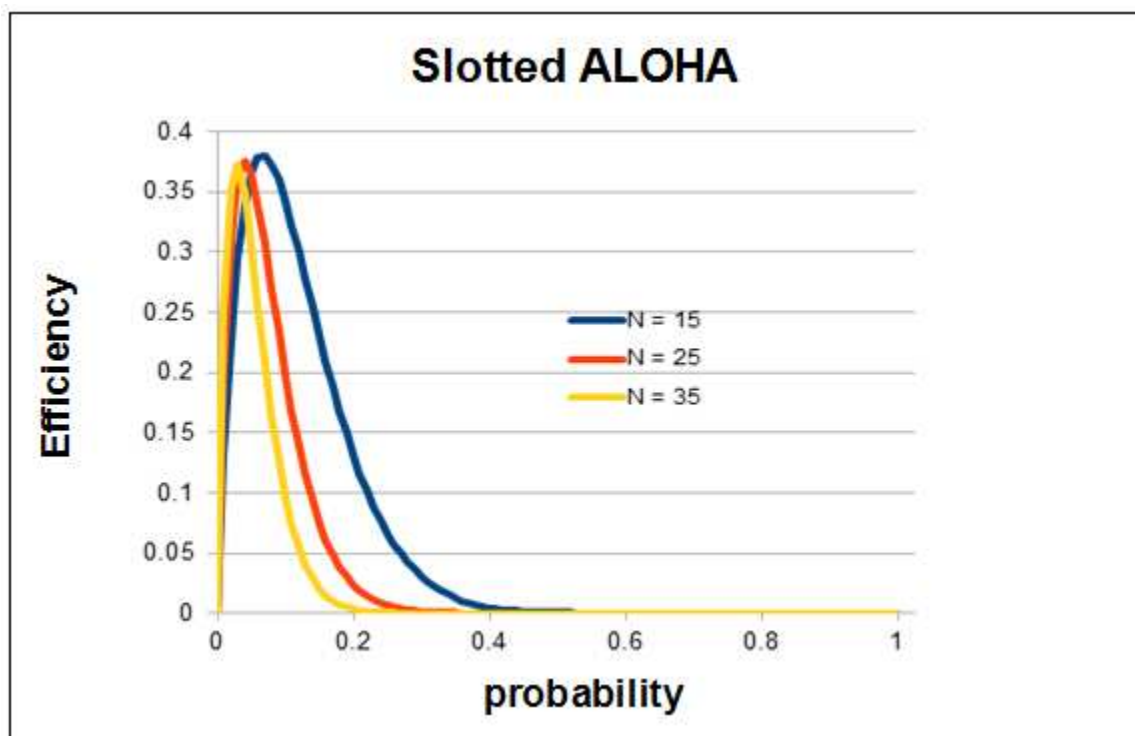
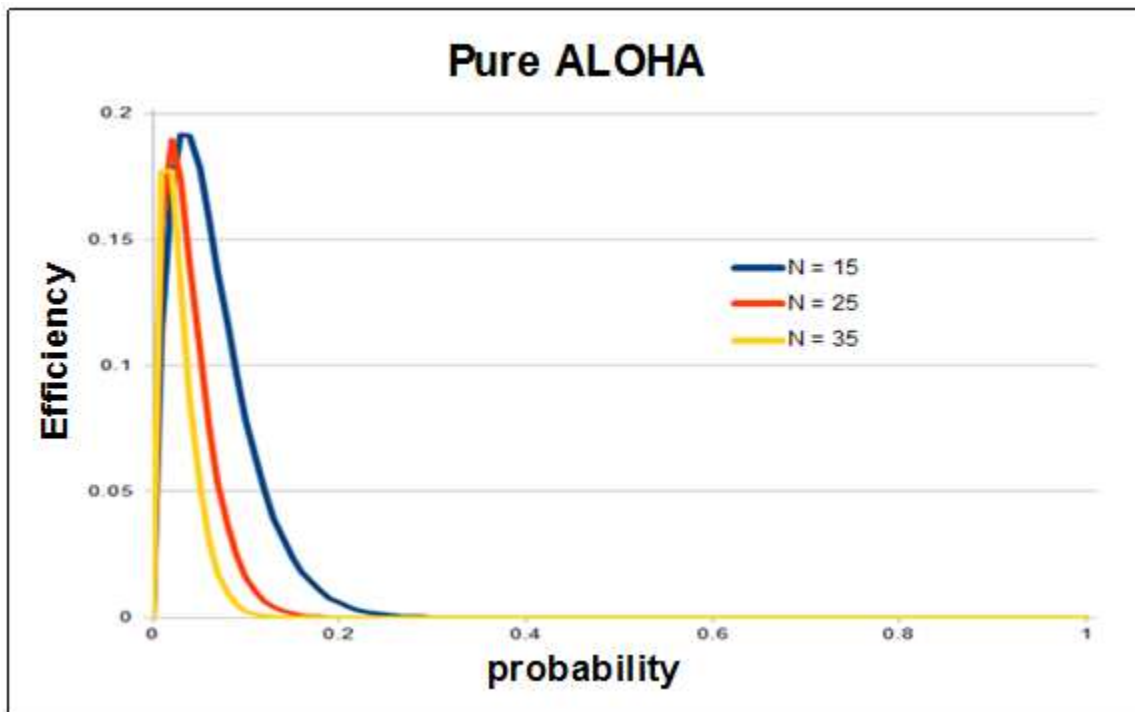
$p(\text{either A or B or C or D succeeds in slot 4}) = 4 p(1-p)^3$
(because these events are mutually exclusive)

- c) $p(\text{some node succeeds in a slot}) = 4 p(1-p)^3$
 $p(\text{no node succeeds in a slot}) = 1 - 4 p(1-p)^3$

Hence, $p(\text{first success occurs in slot 3}) = p(\text{no node succeeds in first 2 slots}) p(\text{some node succeeds in 3rd slot}) = (1 - 4 p(1-p)^3)^2 4 p(1-p)^3$

- d) efficiency = $p(\text{success in a slot}) = 4 p(1-p)^3$

Problem 12



Problem 13

The length of a polling round is

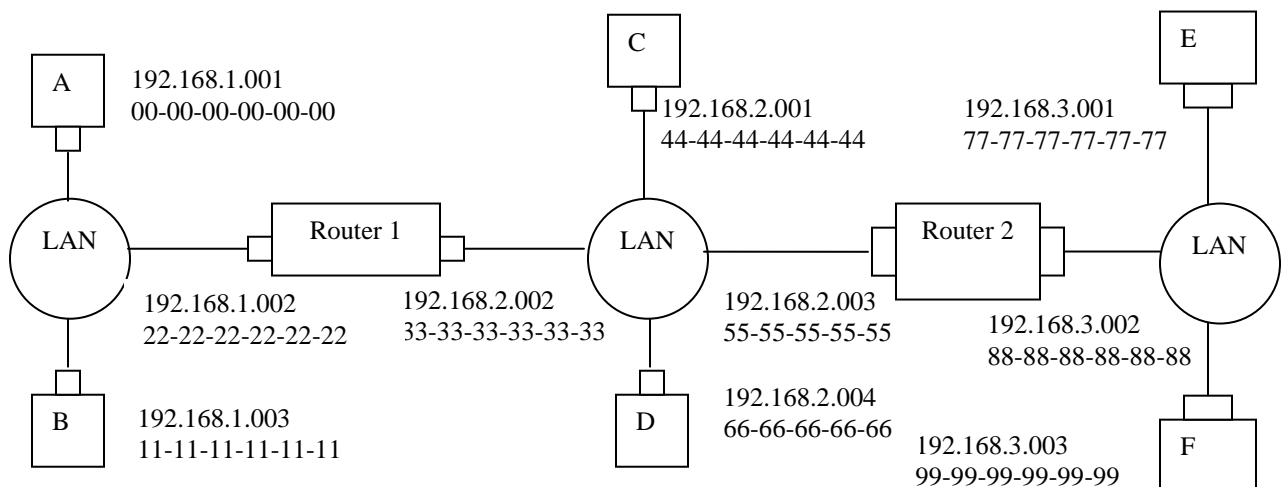
$$N(Q/R + d_{poll}).$$

The number of bits transmitted in a polling round is NQ . The maximum throughput therefore is

$$\frac{NQ}{N(Q/R + d_{poll})} = \frac{R}{1 + \frac{d_{poll}R}{Q}}$$

Problem 14

a), b) See figure below.



c)

1. Forwarding table in E determines that the datagram should be routed to interface 192.168.3.002.
2. The adapter in E creates an Ethernet packet with Ethernet destination address 88-88-88-88-88-88.
3. Router 2 receives the packet and extracts the datagram. The forwarding table in this router indicates that the datagram is to be routed to 198.162.2.002.
4. Router 2 then sends the Ethernet packet with the destination address of 33-33-33-33-33-33 and source address of 55-55-55-55-55-55 via its interface with IP address of 198.162.2.003.
5. The process continues until the packet has reached Host B.

- d) ARP in E must now determine the MAC address of 198.162.3.002. Host E sends out an ARP query packet within a broadcast Ethernet frame. Router 2 receives the query packet and sends to Host E an ARP response packet. This ARP response packet is carried by an Ethernet frame with Ethernet destination address 77-77-77-77-77-77.

Problem 15

- a) No. E can check the subnet prefix of Host F's IP address, and then learn that F is on the same LAN. Thus, E will not send the packet to the default router R1.

Ethernet frame from E to F:

Source IP = E's IP address

Destination IP = F's IP address

Source MAC = E's MAC address

Destination MAC = F's MAC address

- b) No, because they are not on the same LAN. E can find this out by checking B's IP address.

Ethernet frame from E to R1:

Source IP = E's IP address

Destination IP = B's IP address

Source MAC = E's MAC address

Destination MAC = The MAC address of R1's interface connecting to Subnet 3.

- c) Switch S1 will broadcast the Ethernet frame via both its interfaces as the received ARP frame's destination address is a broadcast address. And it learns that A resides on Subnet 1 which is connected to S1 at the interface connecting to Subnet 1. And, S1 will update its forwarding table to include an entry for Host A.

Yes, router R1 also receives this ARP request message, but R1 won't forward the message to Subnet 3.

B won't send ARP query message asking for A's MAC address, as this address can be obtained from A's query message.

Once switch S1 receives B's response message, it will add an entry for host B in its forwarding table, and then drop the received frame as destination host A is on the same interface as host B (i.e., A and B are on the same LAN segment).

Problem 16

Lets call the switch between subnets 2 and 3 S2. That is, *router R1 between subnets 2 and 3 is now replaced with switch S2.*

- a) No. E can check the subnet prefix of Host F's IP address, and then learn that F is on the same LAN segment. Thus, E will not send the packet to S2.

Ethernet frame from E to F:

Source IP = E's IP address

Destination IP = F's IP address

Source MAC = E's MAC address

Destination MAC = F's MAC address

- b) Yes, because E would like to find B's MAC address. In this case, E will send an ARP query packet with destination MAC address being the broadcast address.

This query packet will be re-broadcast by switch 1, and eventually received by Host B.

Ethernet frame from E to S2:

Source IP = E's IP address

Destination IP = B's IP address

Source MAC = E's MAC address

Destination MAC = broadcast MAC address: FF-FF-FF-FF-FF-FF.

- c) Switch S1 will broadcast the Ethernet frame via both its interfaces as the received ARP frame's destination address is a broadcast address. And it learns that A resides on Subnet 1 which is connected to S1 at the interface connecting to Subnet 1. And, S1 will update its forwarding table to include an entry for Host A.

Yes, router S2 also receives this ARP request message, and S2 will broadcast this query packet to all its interfaces.

B won't send ARP query message asking for A's MAC address, as this address can be obtained from A's query message.

Once switch S1 receives B's response message, it will add an entry for host B in its forwarding table, and then drop the received frame as destination host A is on the same interface as host B (i.e., A and B are on the same LAN segment).

Problem 17

Wait for 51,200 bit times. For 10 Mbps, this wait is

$$\frac{51.2 \times 10^3 \text{ bits}}{10 \times 10^6 \text{ bps}} = 5.12 \text{ msec}$$

For 100 Mbps, the wait is 512 μ sec.

Problem 18

At $t = 0$ A transmits. At $t = 576$, A would finish transmitting. In the worst case, B begins transmitting at time $t=324$, which is the time right before the first bit of A 's frame arrives at B . At time $t=324+325=649$ B 's first bit arrives at A . Because $649 > 576$, A finishes transmitting before it detects that B has transmitted. So A incorrectly thinks that its frame was successfully transmitted without a collision.

Problem 19

Time, t	Event
0	A and B begin transmission
245	A and B detect collision
293	A and B finish transmitting jam signal
$293+245 = 538$	B 's last bit arrives at A ; A detects an idle channel
$538+96=634$	A starts transmitting
$293+512 = 805$	B returns to Step2 B must sense idle channel for 96 bit times before it transmits
$634+245=879$	A 's transmission reaches B

Because A 's retransmission reaches B before B 's scheduled retransmission time ($805+96$), B refrains from transmitting while A retransmits. Thus A and B do not collide. Thus the factor 512 appearing in the exponential backoff algorithm is sufficiently large.

Problem 20

a) Let Y be a random variable denoting the number of slots until a success:

$$P(Y = m) = \beta(1 - \beta)^{m-1},$$

where β is the probability of a success.

This is a geometric distribution, which has mean $1/\beta$. The number of consecutive wasted slots is $X = Y - 1$ that

$$x = E[X] = E[Y] - 1 = \frac{1 - \beta}{\beta}$$

$$\beta = Np(1 - p)^{N-1}$$

$$x = \frac{1 - Np(1-p)^{N-1}}{Np(1-p)^{N-1}}$$

$$= \frac{k}{k+x} = \frac{k}{k + \frac{1 - Np(1-p)^{N-1}}{Np(1-p)^{N-1}}}$$

efficiency

b)

Maximizing efficiency is equivalent to minimizing x , which is equivalent to maximizing β . We know from the text that β is maximized at $p = \frac{1}{N}$.

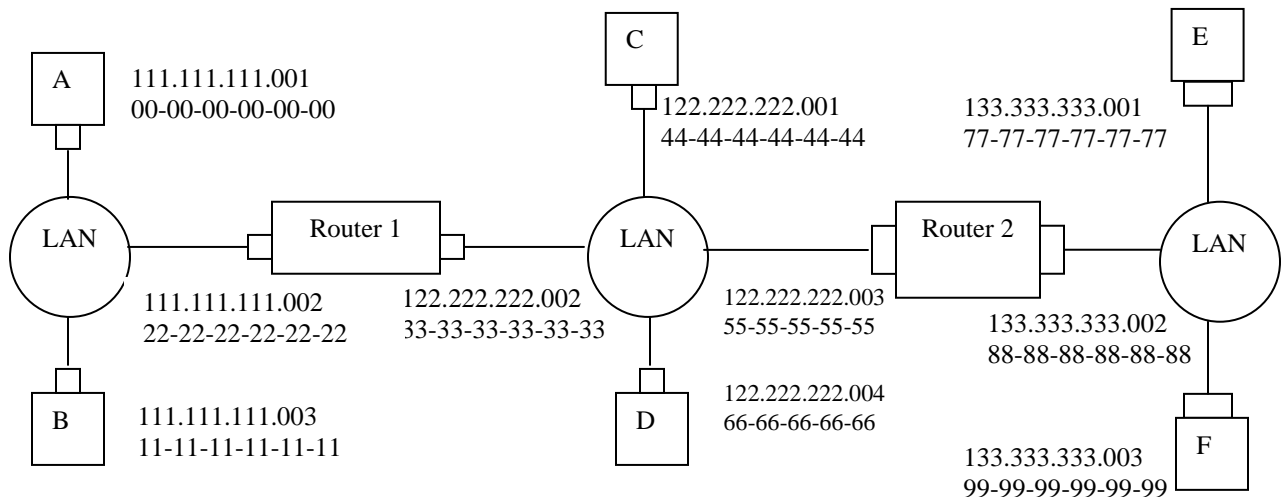
c)

$$\text{efficiency} = \frac{k}{k + \frac{1 - (1 - \frac{1}{N})^{N-1}}{(1 - \frac{1}{N})^{N-1}}}$$

$$\lim_{N \rightarrow \infty} \text{efficiency} = \frac{k}{k + \frac{1 - 1/e}{1/e}} = \frac{k}{k + e - 1}$$

d) Clearly, $\frac{k}{k + e - 1}$ approaches 1 as $k \rightarrow \infty$.

Problem 21



- i) from A to left router: Source MAC address: 00-00-00-00-00-00
Destination MAC address: 22-22-22-22-22-22
Source IP: 111.111.111.001
Destination IP: 133.333.333.003
- ii) from the left router to the right router: Source MAC address: 33-33-33-33-33-33
Destination MAC address: 55-55-55-55-55-55
Source IP: 111.111.111.001
Destination IP: 133.333.333.003
- iii) from the right router to F: Source MAC address: 88-88-88-88-88-88
Destination MAC address: 99-99-99-99-99-99
Source IP: 111.111.111.001
Destination IP: 133.333.333.003

Problem 22

- i) from A to switch: Source MAC address: 00-00-00-00-00-00
Destination MAC address: 55-55-55-55-55-55
Source IP: 111.111.111.001
Destination IP: 133.333.333.003
- ii) from switch to right router: Source MAC address: 00-00-00-00-00-00
Destination MAC address: 55-55-55-55-55-55
Source IP: 111.111.111.001
Destination IP: 133.333.333.003
- iii) from right router to F: Source MAC address: 88-88-88-88-88-88
Destination MAC address: 99-99-99-99-99-99
Source IP: 111.111.111.001
Destination IP: 133.333.333.003

Problem 23

If all the $11=9+2$ nodes send out data at the maximum possible rate of 100 Mbps, a total aggregate throughput of $11 \times 100 = 1100$ Mbps is possible.

Problem 24

Each departmental hub is a single collision domain that can have a maximum throughput of 100 Mbps. The links connecting the web server and the mail server has a maximum

throughput of 100 Mbps. Hence, if the three collision domains and the web server and mail server send out data at their maximum possible rates of 100 Mbps each, a maximum total aggregate throughput of 500 Mbps can be achieved among the 11 end systems.

Problem 25

All of the 11 end systems will lie in the same collision domain. In this case, the maximum total aggregate throughput of 100 Mbps is possible among the 11 end systems.

Problem 26

Action	Switch Table State	Link(s) packet is forwarded to	Explanation
B sends a frame to E	Switch learns interface corresponding to MAC address of B	A, C, D, E, and F	Since switch table is empty, so switch does not know the interface corresponding to MAC address of E
E replies with a frame to B	Switch learns interface corresponding to MAC address of E	B	Since switch already knows interface corresponding to MAC address of B
A sends a frame to B	Switch learns the interface corresponding to MAC address of A	B	Since switch already knows the interface corresponding to MAC address of B
B replies with a frame to A	Switch table state remains the same as before	A	Since switch already knows the interface corresponding to MAC address of A

Problem 27

a) The time required to fill $L \cdot 8$ bits is

$$\frac{L \cdot 8}{128 \times 10^3} \text{ sec} = \frac{L}{16} \text{ msec.}$$

b) For $L = 1,500$, the packetization delay is

$$\frac{1500}{16} \text{ msec} = 93.75 \text{ msec.}$$

For $L = 50$, the packetization delay is

$$\frac{50}{16} \text{ msec} = 3.125 \text{ msec}.$$

c) Store-and-forward delay = $\frac{L \cdot 8 + 40}{R}$

For $L = 1,500$, the delay is

$$\frac{1500 \cdot 8 + 40}{622 \times 10^6} \text{ sec} \approx 19.4 \mu \text{ sec}$$

For $L = 50$, store-and-forward delay $< 1 \mu \text{ sec}$.

- d) Store-and-forward delay is small for both cases for typical link speeds. However, packetization delay for $L = 1500$ is too large for real-time voice applications.

Problem 28

The IP addresses for those three computers (from left to right) in EE department are: 111.111.1.1, 111.111.1.2, 111.111.1.3. The subnet mask is 111.111.1/24.

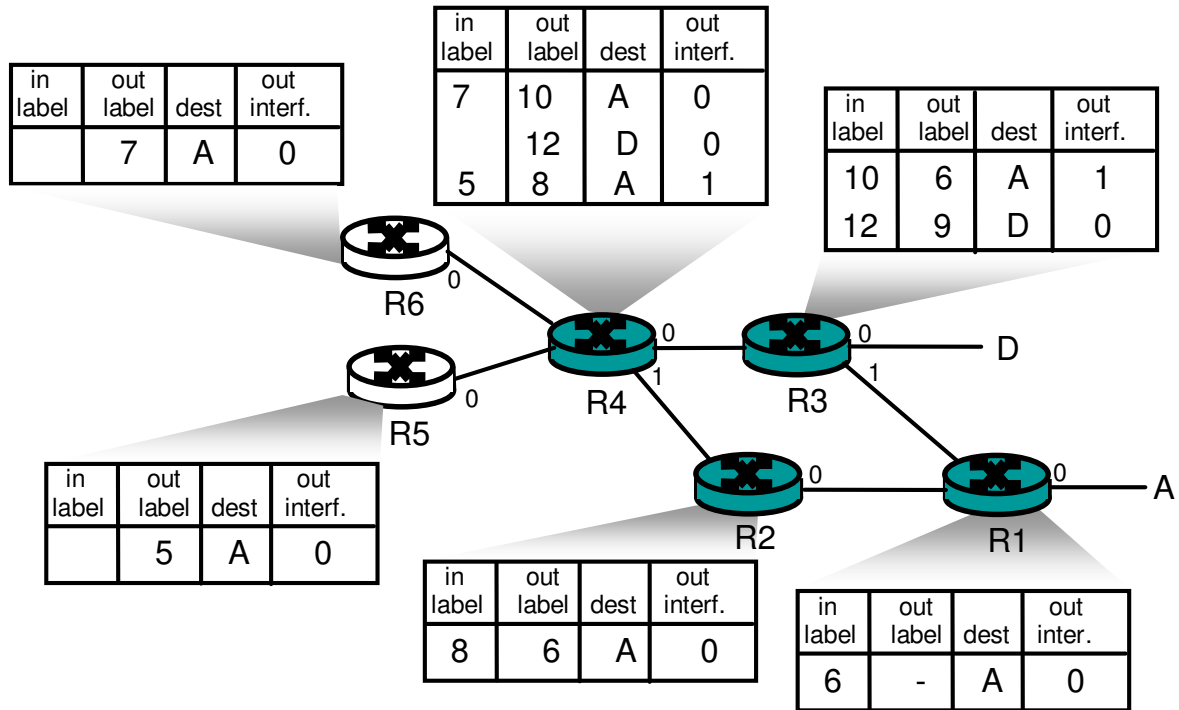
The IP addresses for those three computers (from left to right) in CS department are: 111.111.2.1, 111.111.2.2, 111.111.2.3. The subnet mask is 111.111.2/24.

The router's interface card that connects to port 1 can be configured to contain two sub-interface IP addresses: 111.111.1.0 and 111.111.2.0. The first one is for the subnet of EE department, and the second one is for the subnet of CS department. Each IP address is associated with a VLAN ID. Suppose 111.111.1.0 is associated with VLAN 11, and 111.111.2.0 is associated with VLAN 12. This means that each frame that comes from subnet 111.111.1/24 will be added an 802.1q tag with VLAN ID 11, and each frame that comes from 111.111.2/24 will be added an 802.1q tag with VLAN ID 12.

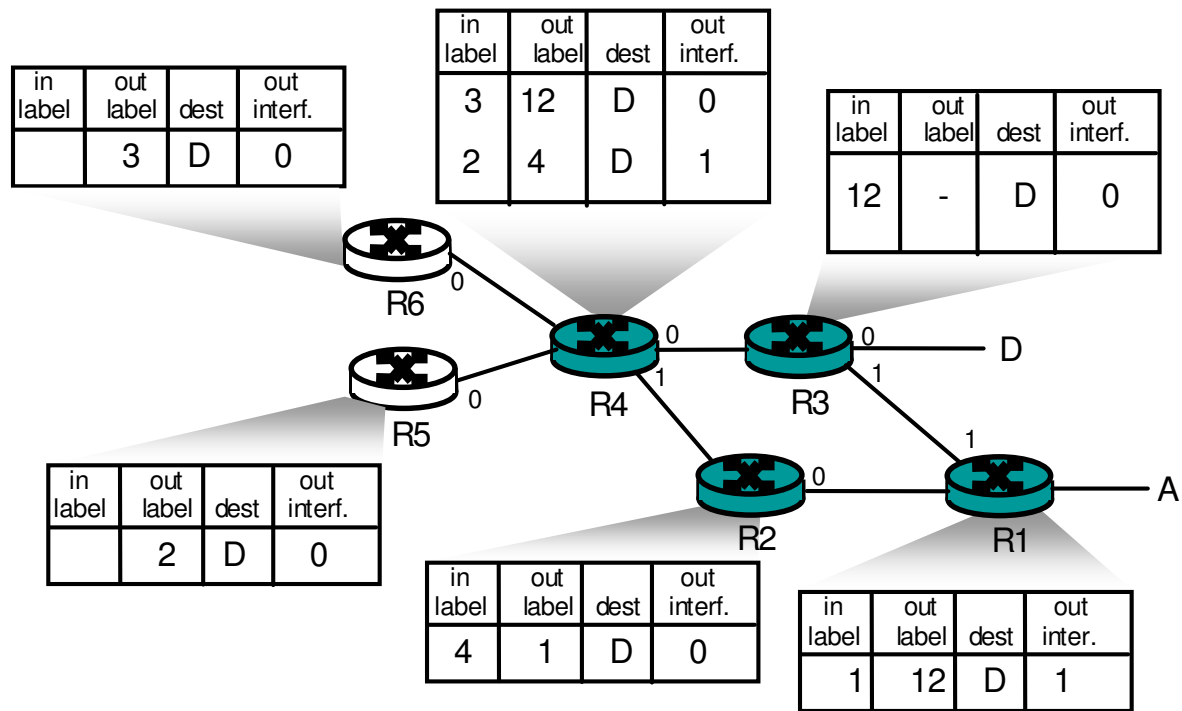
Suppose that host A in EE department with IP address 111.111.1.1 would like to send an IP datagram to host B (111.111.2.1) in CS department. Host A first encapsulates the IP datagram (destined to 111.111.2.1) into a frame with a destination MAC address equal to the MAC address of the router's interface card that connects to port 1 of the switch. Once the router receives the frame, then it passes it up to IP layer, which decides that the IP datagram should be forwarded to subnet 111.111.2/24 via sub-interface 111.111.2.0. Then the router encapsulates the IP datagram into a frame and sends it to port 1. Note that this frame has an 802.1q tag VLAN ID 12. Once the switch receives the frame port 1, it knows that this frame is destined to VLAN with ID 12, so the switch will send the frame

to Host B which is in CS department. Once Host B receives this frame, it will remove the 802.1q tag.

Problem 29



Problem 30



Problem 31

(The following description is short, but contains all major key steps and key protocols involved.)

Your computer first uses DHCP to obtain an IP address. You computer first creates a special IP datagram destined to 255.255.255.255 in the DHCP server discovery step, and puts it in a Ethernet frame and broadcast it in the Ethernet. Then following the steps in the DHCP protocol, you computer is able to get an IP address with a given lease time.

A DHCP server on the Ethernet also gives your computer a list of IP addresses of first-hop routers, the subnet mask of the subnet where your computer resides, and the addresses of local DNS servers (if they exist).

Since your computer's ARP cache is initially empty, your computer will use ARP protocol to get the MAC addresses of the first-hop router and the local DNS server.

Your computer first will get the IP address of the Web page you would like to download. If the local DNS server does not have the IP address, then your computer will use DNS protocol to find the IP address of the Web page.

Once your computer has the IP address of the Web page, then it will send out the HTTP request via the first-hop router if the Web page does not reside in a local Web server. The

HTTP request message will be segmented and encapsulated into TCP packets, and then further encapsulated into IP packets, and finally encapsulated into Ethernet frames. Your computer sends the Ethernet frames destined to the first-hop router. Once the router receives the frames, it passes them up into IP layer, checks its routing table, and then sends the packets to the right interface out of all of its interfaces.

Then your IP packets will be routed through the Internet until they reach the Web server.

The server hosting the Web page will send back the Web page to your computer via HTTP response messages. Those messages will be encapsulated into TCP packets and then further into IP packets. Those IP packets follow IP routes and finally reach your first-hop router, and then the router will forward those IP packets to your computer by encapsulating them into Ethernet frames.

Problem 32

- a) Each flow evenly shares a link's capacity with other flows traversing that link, then the 80 flows crossing the B to access-router 10 Gbps links (as well as the access router to border router links) will each only receive $10 \text{ Gbps} / 80 = 125 \text{ Mbps}$
- b) In Topology of Figure 5.31, there are four distinct paths between the first and third tier-2 switches, together providing 40 Gbps for the traffic from racks 1-4 to racks 9-12. Similarly, there are four links between second and fourth tier-2 switches, together providing 40 Gbps for the traffic from racks 5-8 to 13-16. Thus the total aggregate bandwidth is 80 Gbps, and the value per flow rate is 1 Gbps.
- c) Now 20 flows will need to share each 1 Gbps bandwidth between pairs of TOR switches. So the host-to-host bit rate will be 0.5 Gbps.

Problem 33

- a) Both email and video application uses the fourth rack for 0.1 percent of the time.
- b) Probability that both applications need fourth rack is $0.001 * 0.001 = 10^{-6}$.
- c) Suppose the first three racks are for video, the next rack is a shared rack for both video and email, and the next three racks are for email. Let's assume that the fourth rack has all the data and software needed for both the email and video applications. With the topology of Figure 5.31, both applications will have enough intra-bandwidth as long as both are not simultaneously using the fourth rack. From part b, both are using the fourth rack for no more than .00001 % of time, which is within the .0001% requirement.