Computer Networking: A Top-Down Approach Featuring the Internet, 3rd Edition

Solutions to Review Questions and Problems

Note: These solutions are incomplete. Complete solutions will be available by 1 September 2005

Version Date: July 1, 2004

This document contains the solutions to review questions and problems for the 3rd edition of *Computer Networking: A Top-Down Approach Featuring the Internet* by Jim Kurose and Keith Ross. These solutions are being made available to instructors ONLY. Please do NOT copy or distribute this document to others (even other instructors). Please do not post any solutions on a publicly-available Web site. We'll be happy to provide a copy (up-to-date) of this solution manual ourselves to anyone who asks.

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, Internet-connected PDAs, WebTVs, etc.

2. Suppose Alice, an ambassador of country A wants to invite Bob, an ambassador of country B, over for dinner. Alice doesn't simply just call Bob on the phone and say, "come to our dinner table now". Instead, she calls Bob and suggests a date and time. Bob may respond by saying he's not available that particular date, but he is available another date. Alice and Bob continue to send "messages" back and forth until they agree on a date and time. Bob then shows up at the embassy on the agreed date, hopefully not more than 15 minutes before or after the agreed time. Diplomatic protocols also allow for either Alice or Bob to politely cancel the engagement if they have reasonable excuses.

3. A networking program usually has two programs, each running on a different host, communicating with each other. The program that initiates the communication is the client. Typically, the client program requests and receives services from the server program.

4. The Internet provides its applications a connection-oriented service (TCP) and a connectionless service (UDP). Each Internet application makes use of one these two services. The two services will be discussed in detail in Chapter 3. Some of the principle characteristics of the connection-oriented service are:

   - Two end-systems first "handshake" before either starts to send application data to the other.
   - Provides reliable data transfer, i.e., all application data sent by one side of the connection arrives at the other side of the connection in order and without any gaps.
   - Provides flow control, i.e., it makes sure that neither end of a connection overwhelms the buffers in the other end of the connection by sending to many packets to fast.
   - Provides congestion control, i.e., regulates the amount of data that an application can send into the network, helping to prevent the Internet from entering a state of grid lock.

   The principle characteristics of connectionless service are:

   - No handshaking

- No guarantees of reliable data transfer
- No flow control or congestion control

5. Flow control and congestion control are two distinct control mechanisms with distinct objectives. Flow control makes sure that neither end of a connection overwhelms the buffers in the other end of the connection by sending to many packets to fast. Congestion control regulates the amount of data that an application can send into the network, helping to prevent congestion in the network core (i.e., in the buffers in the network routers).

6. The Internet's connection-oriented service provides reliable data transfer by using acknowledgements and retransmissions. When one side of the connection doesn't receive an acknowledgement (from the other side of the connection) for a packet it transmitted, it retransmits the packet.

7. 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.

8. In a packet switched network, the packets from different sources flowing on a link do not follow any fixed, pre-defined pattern. In TDM circuit switching, each host gets the same slot in a revolving TDM frame.

9. 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 t1, 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 $L/R_1 + L/R_2$.

10. In a VC network, each packet switch in the network core maintains connection state information for each VC passing through it. Some of this connection state information is maintained to a VC-number translation table. (See page 25)

11. The cons of VCs include (i) the need to have a signaling protocol to set-up and tear-down the VCs; (ii) the need to maintain connection state in the packet switches. For the pros, some researchers and engineers argue that it is easier to provide QoS services - such as services that guarantee a minimum transmission rate or services that guarantee maximum end-to-end packet delay – when VCs are used.

12. 1. Dial-up modem over telephone line: residential; 2. DSL over telephone line: residential or small office; 3. Cable to HFC: residential; 4. 100 Mbps switched Etherent: company; 5. Wireless LAN: mobile; 6. Cellular mobile access (for example, WAP): mobile

13. A tier-1 ISP connects to all other tier-1 ISPs; a tier-2 ISP connects to only a few of the tier-1 ISPs. Also, a tier-2 ISP is a customer of one or more tier-1

14. A POP is a group of one or more routers in an ISPs network at which routers in other ISPs can connect. NAPs are localized networks at which many ISPs (tier-1, tier-2 and lower-tier ISPs) can interconnect.

15. 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.

16. Ethernet LANs have transmission rates of 10 Mbps, 100 Mbps, 1 Gbps and 10 Gbps. For an X Mbps Ethernet (where X = 10, 100, 1,000 or 10,000), a user can continuously transmit at the rate X Mbps if that user is the only person sending data. If there are more than one active user, then each user cannot continuously transmit at X Mbps.

17. Ethernet most commonly runs over twisted-pair copper wire and "thin" coaxial cable. It also can run over fibers optic links and thick coaxial cable.

18. Dial up modems: up to 56 Kbps, bandwidth is dedicated; ISDN: up to 128 kbps, bandwidth is dedicated; ADSL: downstream channel is .5-8 Mbps, upstream channel is up to 1 Mbps, bandwidth is dedicated; HFC, downstream channel is 10-30 Mbps and upstream channel is usually less than a few Mbps, bandwidth is shared.

19. 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.

20. 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.

21. 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 1.7.1.

22. 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.

23. Routers process 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 layer four as well.) Link layer switches process layers 1 through 2. Hosts process all five layers.

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
HELO <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
WITHDRAWL <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.

a) A circuit-switched network would be well suited to the application described, 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 circuit with no significant waste. In addition, we need not worry greatly about the overhead costs of setting up and tearing down a circuit connection, which are amortized over the lengthy duration of a typical application session.

b) Given such generous link capacities, the network needs no congestion control mechanism. In the worst (most potentially congested) case, all the applications simultaneously transmit over one or more particular network links. However, since each link offers sufficient bandwidth to handle the sum of all of the applications' data rates, no congestion (very little queueing) will occur.

Problem 3.

a) We can *n* connections between each of the four pairs of adjacent switches. This gives a maximum of *4n* connections.
b) We can *n* connections passing through the switch in the upper-right-hand corner and another *n* connections passing through the switch in the lower-left-hand corner, giving a total of *2n* connections.

Problem 4.

Tollbooths are 100 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 two minutes, for the first tollbooth to service the 10 cars. Each of these cars have a propagation delay of 60 minutes before arriving at the second tollbooth. Thus, all the cars are lined up before the second tollbooth after 62 minutes. The whole process repeats itself for traveling between the second and third tollbooths. Thus the total delay is 124 minutes. (b) Delay between tollbooths is 7*12 seconds plus 60 minutes, i.e., 61 minutes and 24 seconds. The total delay is twice this amount, i.e., 122 minutes and 48 seconds.

Problem 5.

a) The time to transmit one packet onto a link is $(L+h)/R$. The time to deliver the packet over Q links is $Q(L+h)/R$. Thus the total latency is $t_s + Q(L+h)/R$.
b) $Q(L+2h)/R$
c) Because there is no store-and-forward delays at the links, the total delay is

$$t_s + (h+L)/R.$$

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

$$m = \frac{L}{R}S = \frac{100}{28\times10^3}\left(2.5\times10^8\right) = 893\,\text{km}.$$

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

$$\frac{48\cdot8}{64\times10^3}\,\text{sec}=6\text{msec}.$$

The time required to transmit the packet is

$$\frac{48\cdot8}{1\times10^6}\,\text{sec}=384\mu\,\text{sec}.$$

Propagation delay = 2 msec.
The delay until decoding is

$$6\text{msec} + 384\mu\,\text{sec} + 2\text{msec} = 8.384\text{msec}$$

A similar analysis shows that all bits experience a delay of 8.384 msec.

Problem 8.

a) 10 users can be supported because each user requires one tenth of the bandwidth.
b) $p = 0.1$.
c) $\binom{40}{n} p^n (1-p)^{40-n}$.
d) $1 - \sum_{n=0}^{9} \binom{40}{n} p^n (1-p)^{40-n}$.

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

$$P(\text{ “11 or more users” }) = 1 - P\left(\sum_{j=1}^{40} X_j \le 10 \right)$$

$$P\left(\sum_{j=1}^{40} X_j \le 10 \right) = P\left(\frac{\sum_{j=1}^{40} X_j - 4}{\sqrt{40 \cdot 0.1 \cdot 0.9}} \le \frac{6}{\sqrt{40 \cdot 0.1 \cdot 0.9}} \right)$$

$$\approx P\left(Z \le \frac{6}{\sqrt{3.6}} \right) = P(Z \le 3.16)$$

$$= 0.999$$

when Z is a standard normal r.v. Thus $P(\text{ “10 or more users” }) \approx 0.001$.

Problem 9.

a) 10,000
b) $\sum_{n=N+1}^{M} \binom{M}{n} p^n (1-p)^{M-n}$

Problem 10.

It takes LN/R seconds to transmit the N packets. Thus, the buffer is empty when a batch of N packets arrive.

The first of the N packets has no queueing delay. The 2nd packet has a queueing delay of L/R seconds. The n th packet has a delay of $(n-1)L/R$ seconds.

The average delay is

$$\frac{1}{N} \sum_{n=1}^{N} (n-1)L/R = \frac{L}{R} \frac{1}{N} \sum_{n=0}^{N-1} n = \frac{L}{R} \frac{1}{N} \frac{(N-1)N}{2} = \frac{L}{R} \frac{(N-1)}{2}.$$

Problem 11.

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 $x = L/R$.

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

Problem 12.

a) There are Q nodes (the source host and the $N-1$ routers). Let d^q_{proc} denote the processing delay at the q th node. Let R^q be the transmission rate of the q th link and let $d^q_{trans} = L/R^q$. Let d^q_{prop} be the propagation delay across the q th link. Then

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

b) Let d^q_{queue} denote the average queueing delay at node q. Then

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

Problem 13.

The command:

```
traceroute -q 20 www.eurecom.fr
```

will get 20 delay measurements from the issuing host to the host, www.eurecom.fr. The average and standard deviation of these 20 measurements can then be collected. Do you see any differences in your answers as a function of time of day?

Problem 14.

a) 40,000 bits
b) 40,000 bits
c) the bandwidth-delay product of a link is the maximum number of bits that can be in the link
d) 1 bit is 250 meters long, which is longer than a football field
e) *s/R*

Problem 15.

25 bps

Problem 16.

a) 40,000,000 bits
b) 400,000 bits
c) .25 meters

Problem 17.

a) $t_{trans} + t_{prop}$ = 400 msec + 40 msec = 440 msec
b) 10 * (t_{trans} + 2 t_{prop}) = 10*(40 msec + 80 msec) = 1.2 sec

Problem 18.

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

Problem 19.

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 if Figure 1.20 that allows the Baggage layer to implement the service or 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 the at 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 20.

a) time taken to send message from source host to first packet switch = $\frac{7.5\times10^6}{1.5\times10^6}\text{sec} = 5\,\text{sec}$. With store-and-forward switching, the total time to move message from source host to destination host = $5\,\text{sec}\times 3\ hops = 15\,\text{sec}$
b) time taken to send 1st packet from source host to first packet switch = . $\frac{1.5\times10^3}{1.5\times10^6}\text{sec} = 1\ m\,\text{sec}$. Time at which 2nd packet is received at the first switch = time at which 1st packet is received at the second switch = $2\times1\ m\,\text{sec} = 2\ m\,\text{sec}$
c) time at which 1st packet is received at the destination host = . $1\ m\,\text{sec}\times 3\ hops = 3\ m\,\text{sec}$. After this, every 1msec one packet will be received, thus time at which last (5000th) packet is received = $3\ m\,\text{sec} + 4999 * 1m\,\text{sec} = 5.002\,\text{sec}$. It can be seen that delay in using message segmentation is significantly less (almost 1/3rd).
d) drawbacks:
    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 packets regardless of its size, with message segmentation total amount of header bytes sent increases.

Problem 21.

Java Applet

Problem 22.

Time at which the 1st packet is received at the destination = $\frac{S+40}{R}\times 2$ sec. After this, one packet is received by the destination every $\frac{S+40}{R}$ sec. Thus delay in sending the whole file = $delay = \frac{S+40}{R}\times 2 + (\frac{F}{S}-1)\times(\frac{S+40}{R}) = \frac{S+40}{R}\times(\frac{F}{S}+1)$

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

$\frac{\partial delay}{\partial S} = 0 \Rightarrow \frac{F}{R}(\frac{1}{S} - \frac{40+S}{S^2}) + \frac{1}{R} = 0 \Rightarrow S = \sqrt{40F}$

Chapter 2 Review Questions

1. The Web: HTTP; file transfer: FTP; remote login: Telnet; Network News: NNTP; e-mail: SMTP.

2. Network architecture refers to organization of communication into layers (e.g., the five-layer Internet architecture). Application architecture, on the other hand, is designed by an application developer and dictates how the application is (e.g., client-server or P2P)
3. Instant Messaging involves the initiator to contact a centralized server to locate the address (IP address.) of the receiver: client server model. After this, the instant messaging can be peer to peer – message between the two communicating parties are sent directly between them.

4. The process which initiates a service request is the client; the process that waits to be contacted is the server.

5. No. As stated in the text, all communication sessions have a client side and a server side. 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.

6. The IP address of the destination hosts and the port numbers of the destination socket.

7. You probably use a browser and a mail reader on a daily basis. You may also use an FTP user agent, a Telnet user agent, an audio/video player user agent (such as a Real Networks player), an instant messaging agent, a P2P file-sharing agent, etc.

8. There are no good examples of an application that requires no data loss and timing. If you know of one, send an e-mail to the authors.

9. 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.

10. The applications that use those protocols require that all application data is received in the correct order and without gaps. TCP provides this service whereas UDP does not.

11. In both cases, the site must keep a database record for the user. With cookies, the user does not explicitly provide a username and password each time it visits the site. However, browser identifies the user by sending the user's cookie number each time the user accesses the site.

12. In persistent HTTP without pipelining, the browser first waits to receive a HTTP response from the server before issuing a new HTTP request. In persistent HTTP with

pipelining, the browser issues requests as soon as it has a need to do so, without waiting for response messages from the server.

13. Web caching can bring the desired content "closer" to the user, perhaps 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.

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. Message is 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.

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. 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 directed link (a "directed edge" in graph theory terms) from node A to node B if node A sends datagrams that are explicitly addressed to node B's IP address. An overlay network does not include routers. With Gnutella, when a node wants to join the Gnutella network, it first discovers ("out of band") the IP address of one or more nodes already in the network. It then sends join messages to these nodes. When the node receives confirmations, it becomes a member of the of Gnutella network. Nodes maintain their logical links with periodic refresh messages.

21. Three companies as of this writing (August 2002) are KaZaA, eDonkey, Bit Torrent.

22. 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.

23. 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

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, RNTO, ABOR, DELE, RMD, MRD, PWD, LIST, NLST, SITE, SYST, STAT, HELP, NOOP`.

Problem 3.

SFTP: 115, NNTP: 119.

Problem 4.

Application layer protocols: DNS and HTTP
Transport layer protocols: UDP for DNS; TCP for HTTP

Problem 5.

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 including the connection-token "close" in the Connection-header field of the http request/reply.

Problem 6.

The total amount of time to get the IP address is

$$RTT_1 + RTT_2 + \Lambda + RTT_n .$$

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

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

Problem 7.

a)

$$RTT_1 + \Lambda + RTT_n + 2RTT_o + 3 \cdot 2RTT_o$$
$$= 8RTT_o + RTT_1 + \Lambda + RTT_n .$$

b)

$$RTT_1 + \Lambda + RTT_n + 2RTT_o + 2RTT_o$$
$$= 4RTT_o + RTT_1 + \Lambda + RTT_n .$$

c)

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

Problem 8.

HTTP/1.0: GET, POST, HEAD.
HTTP/1.1: GET, POST, HEAD, OPTIONS, PUT, DELETE, TRACE, CONNECT.

See RFCs for explanations.

Problem 9.

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

Δ= (900,000 bits)/(1,500,000 bits/sec) = .6 sec

The traffic intensity on the link is (1.5 requests/sec)(.6 msec/request) = .9. Thus, the average access delay is (.6 sec)/(1 - .9) = 6 seconds. The total average response time is therefore 6 sec + 2 sec = 8 sec.

b) The traffic intensity on the access link is reduced by 40% since the 40% of the requests are satisfied within the institutional network. Thus the average access delay

is (.6 sec)/[1 – (.6)(.9)] = 1.2 seconds. The response time is approximately zero if the request is satisfied by the cache (which happens with probability .4); the average response time is 1.2 sec + 2 sec = 3.2 sec for cache misses (which happens 60% of the time). So the average response time is (.4)(0 sec) + (.6)(3.2 sec) = 1.92 seconds. Thus the average response time is reduced from 8 sec to 1.92 sec.

Problem 11.

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 users mailbox. This command is useful for "download and keep". By keeping a file that lists the messages retrieved in earlier sessions, the client can use the UIDL command to determine which messages on the server have already been seen.

Problem 13.

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

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) If under an attack a victim can analyze the source address of packets, the victim can then use whois to obtain information about domain from which attack is coming and possibly inform the administrators of the origin domain.

Problem 14.

No. Because the link is full duplex, you have 128 kbps in each direction, and the uploading does not interfere with the downloading.

Problem 15.

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

Problem 16.

a) In this case, each of the five Gnutella clients immediately learns that it has one less neighbor. Consider one of these five clients, called, Bob. Suppose Bob has only three neighbors after X drops out. Then Bob needs to establish a TCP connection with another peer. Bob should have a fresh list of active peers; he sequentially contacts peers on this list until one accepts his TCP connection attempt.

b) In this case, Bob does not immediately know that X has departed. Bob will only learn about X's departure when it attempts to send a message (query or ping) to X. When Bob attempts to send a message, Bob's TCP will make several unsuccessful attempts to send the message to B. Bob's TCP will then inform the Gnutella client that X is down. Bob

will then try to establish a TCP connection with a new peer (see part (a)) to rebuild a fifth connection.

Problem 17.

a) The advantage of sending the QueryHit message directly over a TCP connection from Bob to Alice is that the QueryHit message is routed by the underlying Internet without passing through intermediate peers; thus, the delay in sending the message from Bob to Alice should be substantially less. The disadvantage is that each peer that has a match would ask Alice to open a TCP connection; Alice may therefore have to opens tens or hundreds of TCP connections for a given query. Furthermore, there will be additional complications if Alice is behind a NAT (see Chapter 4).

b) When a QueryHit message enters a peer, the peer records in a table the MessageID along with an identifier of the TCP socket from which the message arrived. When the same peer receives a QueryHit message with the same MessageID, it indexes the table and determines the socket to which it should forward the message.

c) KR

Problem 18.

Answer is unchanged for Ping/Pong messages. Replace Query message with Ping message and QueryHit message with Pong message

Problem 19.

a) Each super-group leader is responsible for roughly $200^2 = 40{,}000$ nodes. Therefore, we would need about 100 super-group leaders to support 4 million nodes.
b) Each group leader might store the meta-data for all of the files its children are sharing. A super-group leader might store all of the meta-data that its group-leader children store. An ordinary node would first send a query to its group leader. The group leader would respond with matches and then possibly forward the message to its super-group leader. The super-group leader would respond (through the overlay network) with its matches. The super-group leader may further forward the query to other super-group leaders.

Problem 20.

Alice sends her query to at most N neighbors. Each of these neighbors forwards the query to at most M = N-1 neighbors. Each of those neighbors forwards the query to at most M neighbors. Thus the maximum number of query messages is

$$N + NM + NM^2 + \ldots + NM^{(K-1)}$$
$$= N(1 + M + M^2 + \ldots + M^{(K-1)})$$
$$= N(1-M^K)/(1-M)$$

$= N[(N-1)^K - 1]/(N-2)$.

Problem 21.

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 22.

See Web-server programming assignment for this chapter for guidance.

Problem 23.

With the original line, 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 replacement 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 (using the .getPort() method). 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
Server socket = 9876

Chapter 3 Review Questions

1. Source port number y and destination port number x.

2. 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.

3. 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.

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

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

6. 3 segments. First segment: seq = 43, ack =80; Second segment: seq = 80, ack = 44 ; Third segment; seq = 44, ack = 81

7. R/2

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

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.

$$\begin{array}{rcccccccc} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ + & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}$$

$$\begin{array}{rcccccccc} & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ + & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ \hline & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

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

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.

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 5.

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 6.

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 7.

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.

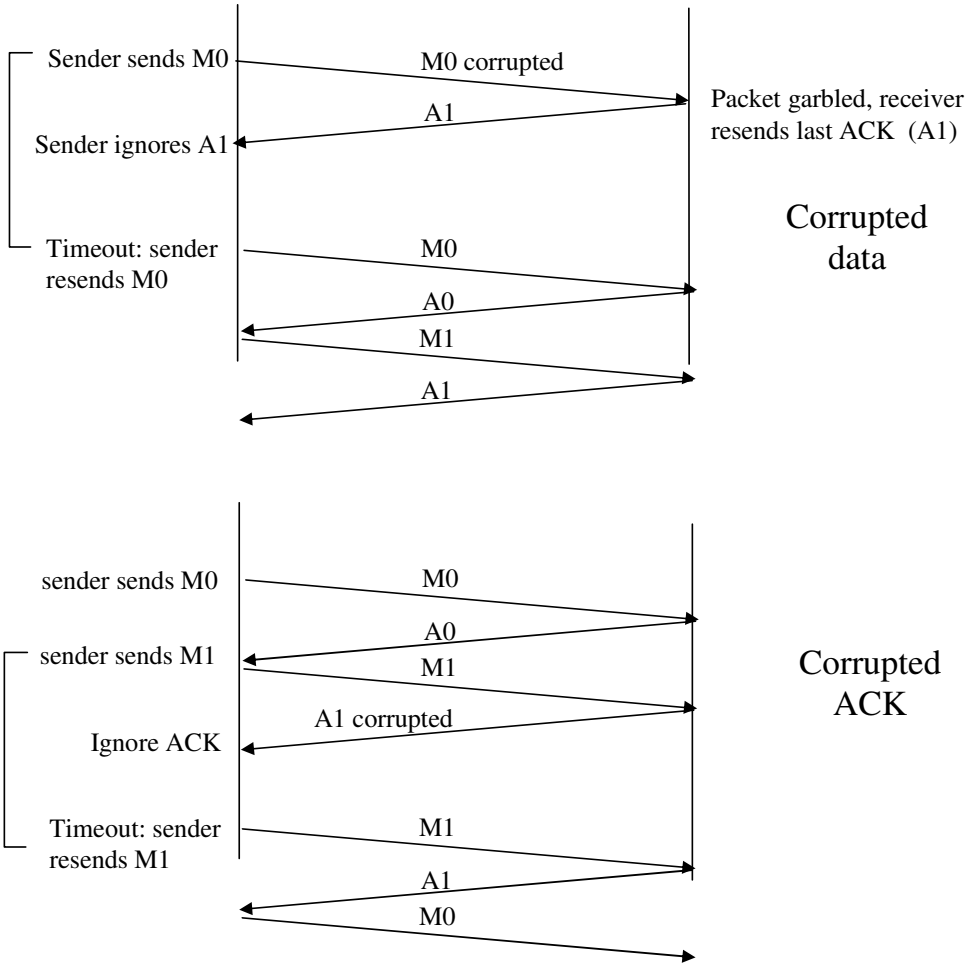


Figure 1: rdt 3.0 scenarios: corrupted data, corrupted ACK

Problem 8.

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.

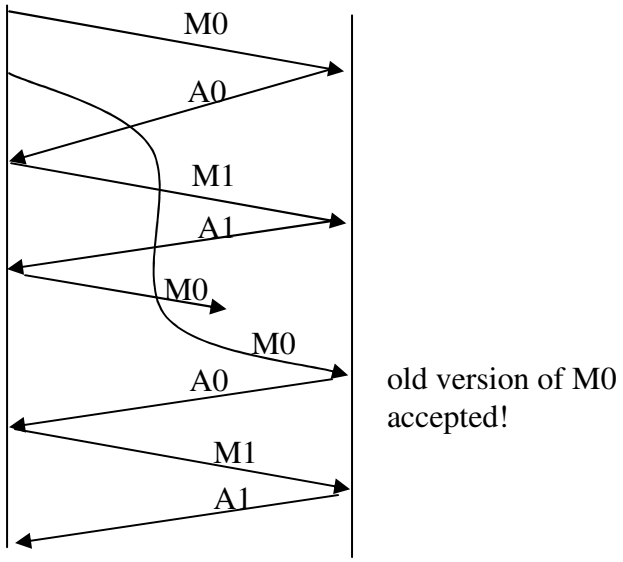
- Suppose the timeout is caused by a lost data packet, i.e., a packet on the sender-to-receiver channel. In this case, the receiver never received the previous transmission and, from the receiver's viewpoint, if the timeout retransmission is received, it look *exactly* the same as if the original transmission 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 is take 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 9.

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 (which is not what the question asked, sorry). 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 10.



Problem 11.

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 receivers 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 ACK are never sent – a significant reduction in feedback in the NAK-only case over the ACK-only case.

Problem 12.

It takes 8 microseconds (or 0.008 milliseconds) to send a packet. in order for the sender to be busy 90 percent of the time, we must have

$$util = 0.9 = (0.008n) / 30.016$$

or n approximately 3377 packets.

Problem 13.

In our GBN reliable data transfer protocol, the sender keep sending packets until it receives a NAK. A NAK is generated for packet n only if all packets up to $n-1$ have been correctly received. That is, n is always the smallest sequence number of a packet that is yet to be received. When the receives a NAK for packet n, it simply begins ending again from packet n onwards. This is like the GBN protocol in the text, except that there is no maximum number of unacknowledged packets in the pipeline. Note the sender can't actually know how many packets are unacknowledged. If the current sequence number if k and the last NAK was for packet n, then there may be as many as $k-(n-1)$ unacknowledged packets in the pipeline.

Note also that a receiver does not know that packet n is missing until a packet with a *higher* sequence number is received! (The "gap" in the sequence numbers of received packets tells the receiver that a packet in the gap is lost). Thus, when the data rate is low (i.e., a large amount of time between packet transmissions), it will take longer for a receiver to notice a missing packet than when the data rate is high.

Problem 14.

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:

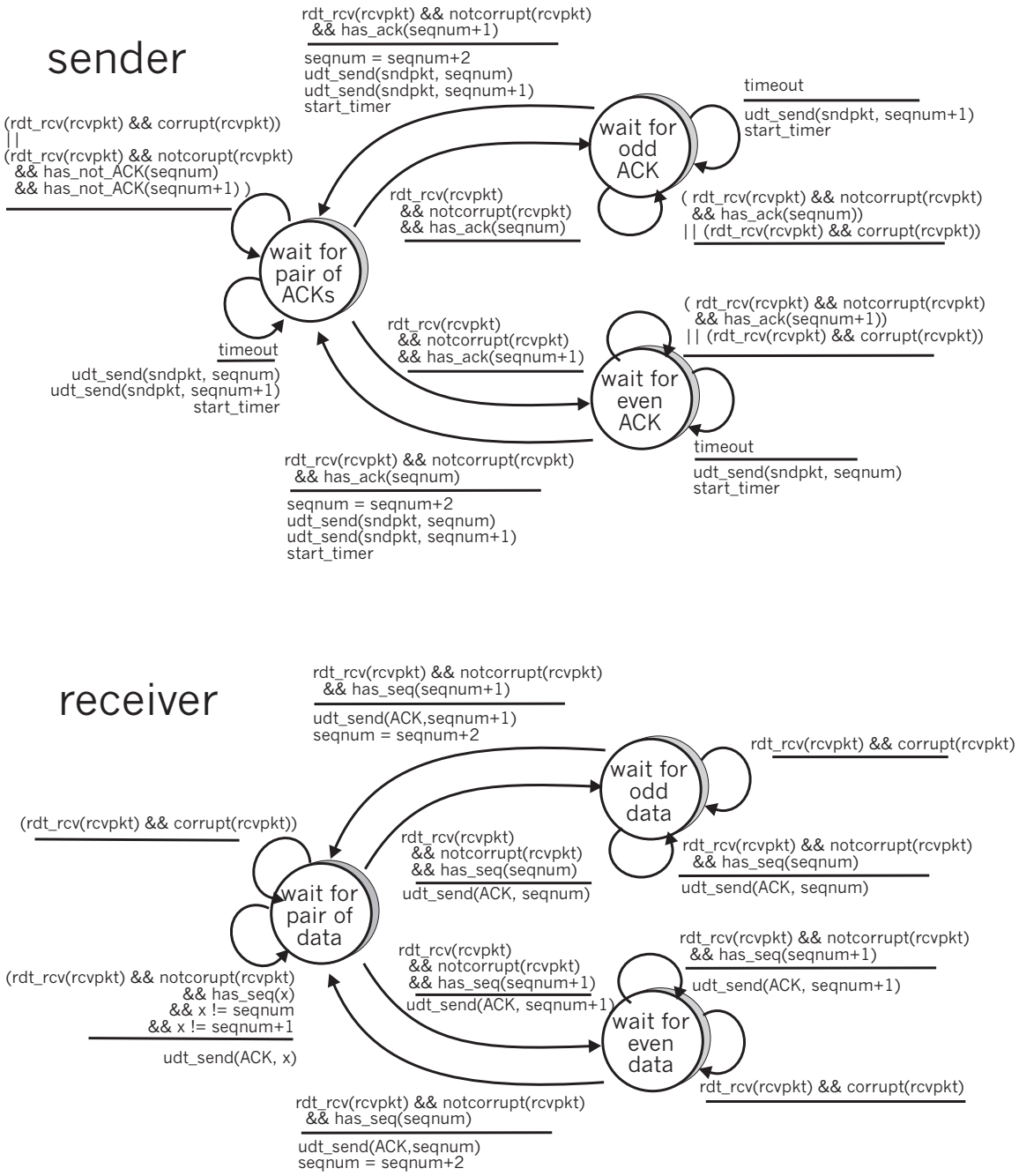


Figure 2: Sender and receiver for Problem 3.11

```
Sender                                  Receiver

make pair (0,1)
send packet 0
                    Packet 0 drops
send packet 1
                                        receive packet 1
                                        buffer packet 1
```

```
                                                send ACK 1
receive ACK 1
(timeout)
resend packet 0
                                                receive packet 0
                                                deliver pair (0,1)
                                                send ACK 0
receive ACK 0
```

Problem 15.

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.

Problem 16.

a) Here we have a window size of N=3. Suppose the receiver has received packet k-1, and has ACKed that and all other preceeding 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 senders window is of size 3 and begins somewhere in the range [k-N,k].

b) 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 that k-N-1. Thus the range of in-flight ACK values can range from k-N-1 to k-1.

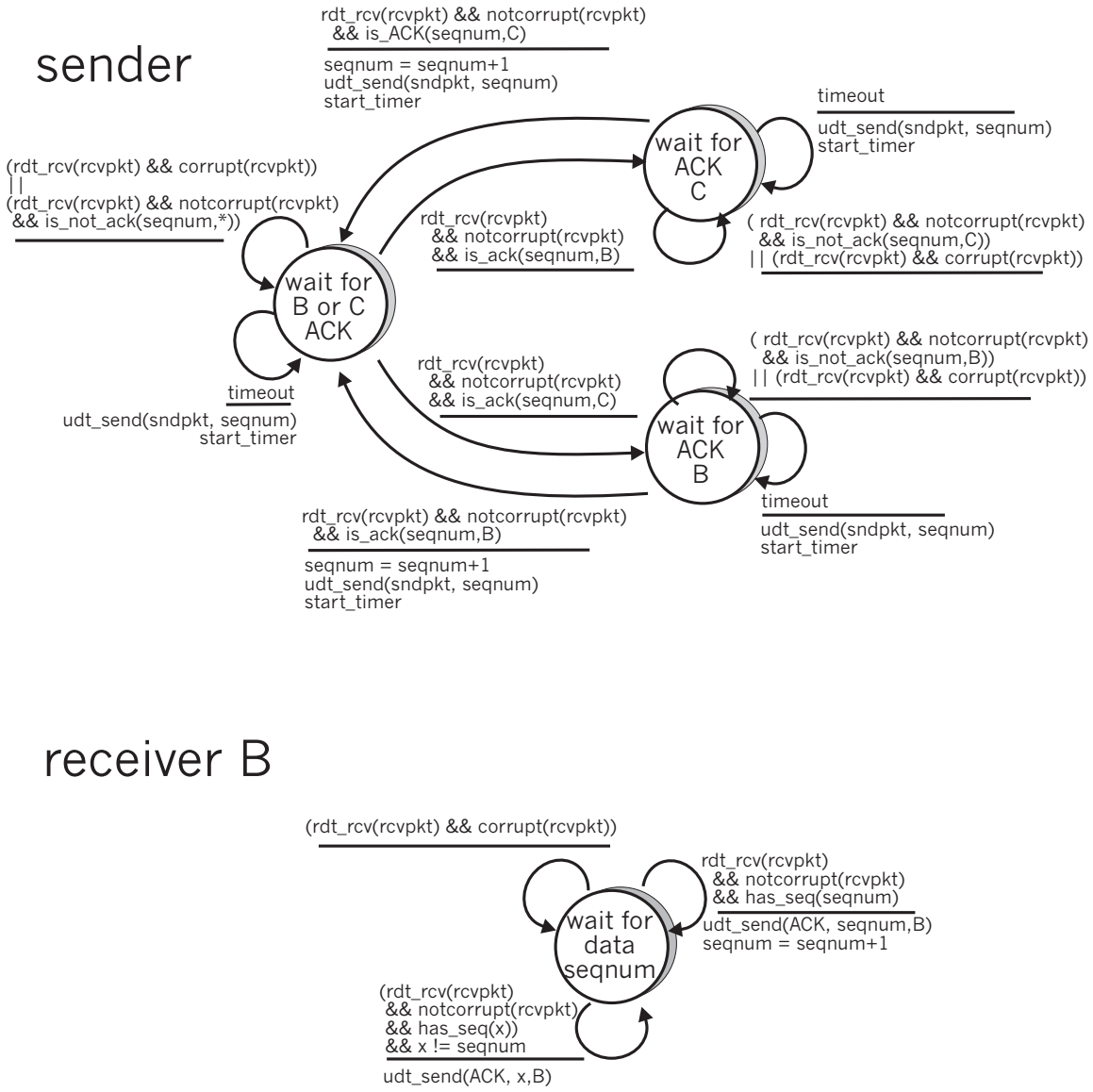


Figure 3. Sender and receiver for Problem 3.12

Problem 17.

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 18.

In order to avoid the scenario of Figure 3.26, 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 (see Problem 13).

Suppose that the lowest-sequence number that the receiver is waiting for is packet m. In this case, it's 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 receivers 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 19.

a) True. Suppose the sender has a window size of 3 and sends packets 1, 2, 3 at $t0$. At $t1$ ($t1 > t0$) the receiver ACKS 1, 2, 3. At $t2$ ($t2 > t1$) the sender times out and resends 1, 2, 3. At $t3$ the receiver receives the duplicates and re-acknowledges 1, 2, 3. At $t4$ the sender receives the ACKs that the receiver sent at $t1$ and advances its window to 4, 5, 6. At $t5$ the sender receives the ACKs 1, 2, 3 the receiver sent at $t2$. 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 20.

There are $2^{32} = 4{,}294{,}967{,}296$ possible sequence numbers.
a) The sequence number does not increment by one with each segment. Rather, is 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 $\left\lceil \frac{2^{32}}{1460} \right\rceil = 2{,}941{,}758.66$ bytes of header get added to each segment giving a total of 194,156,028 bytes of header. The total number of bytes transmitted is $2^{32} + 194{,}156{,}028 = 3{,}591 \times 10^7$ bits.
Thus it would take 3,591 seconds = 59 minutes to transmit the file over a 10~Mbps link.

Problem 21.

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

$$EstimatedRTT^{(1)} = SampleRTT_1$$

$$EstimatedRTT^{(2)} = xSampleRTT_1 + (1-x)SampleRTT_2$$

$$\begin{aligned} EstimatedRTT^{(3)} &= xSampleRTT_1 \\ &\quad + (1-x)[xSampleRTT_2 + (1-x)SampleRTT_3] \\ &= xSampleRTT_1 + (1-x)xSampleRTT_2 \\ &\quad + (1-x)^2 SampleRTT_3 \end{aligned}$$

$$\begin{aligned} EstimatedRTT^{(4)} &= xSampleRTT_1 + (1-x)EstimatedRTT^{(3)} \\ &= xSampleRTT_1 + (1-x)xSampleRTT_2 \\ &\quad + (1-x)^2 xSampleRTT_3 + (1-x)^3 SampleRTT_4 \end{aligned}$$

b)

$$\begin{aligned} EstimatedRTT^{(n)} &= x\sum_{j=1}^{n-1}(1-x)^j SampleRTT_j \\ &\quad + (1-x)^n SampleRTT_n \end{aligned}$$

c)

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

The weight given to past samples decays exponentially.

Figure 4: Lack of TCP convergence with linear increase, linear decrease

Problem 22.

JK

Problem 23.

At any given time *t*, `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

`SendBase-1 ≤ LastByteRcvd`

Problem 24.

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 ≤ SendBase` or if there are other acknowledgements in the pipe. Thus

`y-1 ≤ LastByteRvcd`

Problem 25.

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 26.

JK

Problem 27.

a) TCP slowstart is operating in the intervals [1,6] and [23,26]
b) TCP congestion advoidance 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 slowtart 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 16, the congestion windows size is 42. Hence the threshold is 21 during the 18th 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; packets15-31 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 congestion window and threshold will be set to half the current value of the congestion window (8) when the loss occurred. Thus the new values of the threshold and window will be 4.

Problem 28.

Refer to Figure 4. In Figure 4(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 4(a), 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.

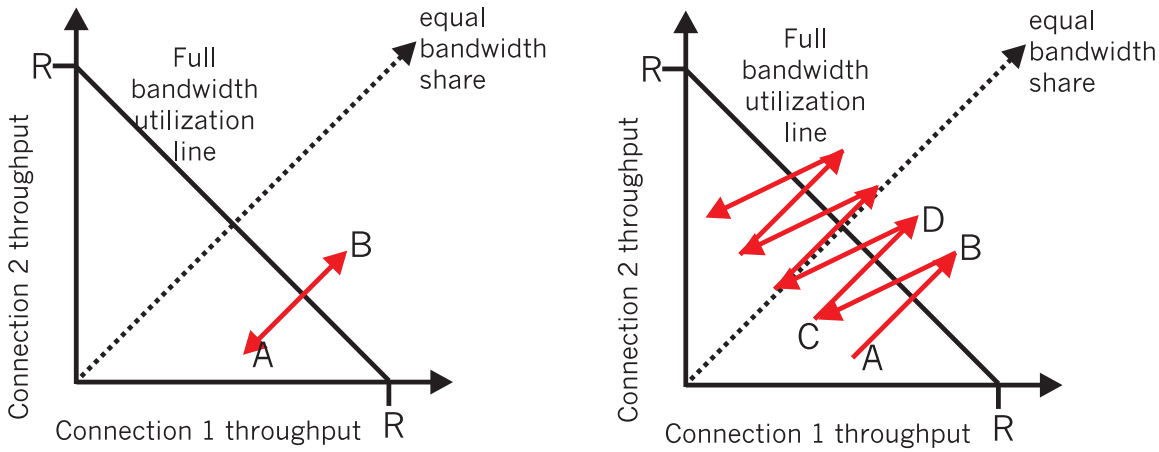


Figure 4: Lack of TCP convergence with linear increase, linear decrease

Problem 29.

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 30.

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 << S.

Problem 31

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

$$\frac{W}{2}+\left(\frac{W}{2}+1\right)+\Lambda\ +W=\sum_{n=0}^{W/2}(\frac{W}{2}+n)$$

$$=\left(\frac{W}{2}+1\right)\frac{W}{2}+\sum_{n=0}^{W/2}n$$

$$=\left(\frac{W}{2}+1\right)\frac{W}{2}+\frac{W/2(W/2+1)}{2}$$

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

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

Thus the loss rate is

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

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

$$\text{average throughput} = \frac{3}{4}\sqrt{\frac{8}{3L}} \cdot \frac{MSS}{RTT}$$

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

Problem 32.

JK

Problem 33

JK

Problem 34.

The minimum latency is 2RTT+O/R. The minimum W that achieves this latency is

$$W = \min\left\{w : w \geq \frac{RTT + S/R}{S/R}\right\}$$
$$= \left\lceil \frac{RTT}{S/R} \right\rceil + 1.$$

R	min latency	W
28 Kbps	28.77 sec	2
100 Kbps	8.2 sec	4
1 Mbps	1 sec	25
10 Mbps	0.28 sec	235

Problem 35.

a)

$$K = \text{ number of windows that cover the object}$$

$$= \min\{k : 3^0 + 3^1 + \Lambda + 3^{k-1} \geq O/S\}$$

$$= \min\left\{k : \frac{1-3^k}{1-3} \geq O/S\right\}$$

$$= \min\{k : 3^k \geq 1 + 2O/S\}$$
$$= \lceil \log_3(1 + 2O/S) \rceil$$

b) Q is the number of times the server would idle for an object of infinite size.

$$Q = \max\left\{k : RTT + \frac{S}{R} - \frac{S}{R}3^{k-1} \geq 0\right\}$$

$$= \left\lfloor 1 + \log_3(1 + \frac{RTT}{S/R}) \right\rfloor$$

c)

$$\text{latency} = \frac{O}{R} + 2RTT + \sum_{k=1}^{P} \text{stall}_k$$

$$= \frac{O}{R} + 2RTT + \sum_{k=1}^{P}\left(RTT + \frac{S}{R} - \frac{S}{R}3^{k-1}\right)$$

$$= \frac{O}{R} + 2RTT + P(RTT + S/R) - \frac{(3^P - 1)}{2}\frac{S}{R}$$

Problem 36.

R	O/R	P	Min latency	Latency with slow start
28 Kbps	29.25 s	3	31.25 sec	33.18 sec
100 Kbps	8.19 s	5	10.19 sec	13.86 sec
1 Mbps	819 msec	7	2.81 sec	9.26 sec
10 Mbps	82 msec	7	2 sec	9 sec

Problem 37.
a) T
b) T
c) F

d) F

Problem 38.

a)

$$Q = \max\left\{k : RTT + \frac{S}{R} - \frac{S}{R}2^{k-1} \geq 0\right\}$$

$$= \max\left\{k : 2^{k-1} \leq 1 + \frac{RTT}{S/R}\right\}$$

$$= \max\left\{k : k \leq \log_2\left(1 + \frac{RTT}{S/R}\right) + 1\right\}$$

$$= \left\lfloor \log_2\left(1 + \frac{RTT}{S/R}\right)\right\rfloor + 1$$

b) Beginning with the equation for latency on page 277, we note that the term within the brackets is non-zero only for k ≤ Q. So we can replace the K-1 in the summand with P and remove the $[\ldots]^+$ restriction. Applying the identity to the resulting equation gives the desired result (after a little algebraic manipulation).

Problem 39.

When the sever sends a segment, it has to wait a time of $TS/R + RTT$ for the acknowledgement to arrive. The transmission time of the *k*th window is $(S/R)2^{k-1}$. The idle time for the *k*th window is

$$\left[\frac{TS}{R} + RTT - 2^{k-1}\frac{S}{R}\right]^+$$

The number of idle periods, *Q,* is

$$Q = \max\left\{k : RTT + \frac{TS}{R} - \frac{S}{R}2^{k-1} \geq 0\right\}$$

$$= \max\left\{k : 2^{k-1} \leq T + \frac{RTT}{S/R}\right\}$$

$$= \max\left\{k : k \le \log_2\left(T + \frac{RTT}{S/R}\right) + 1\right\}$$

$$= \left\lfloor \log_2\left(T + \frac{RTT}{S/R}\right)\right\rfloor + 1$$

The number of times the server idles is $P = \min(Q, K-1)$. The latency is

$$\text{latency} = 2RTT + \frac{O}{R} + \sum_{k=1}^{P}\left(RTT + \frac{TS}{R} - \frac{S}{R}2^{k-1}\right)$$

which simplifies to

$$\text{latency} = 2RTT + \frac{O}{R} + P\left(RTT + \frac{TS}{R}\right) - (2^P - 1)\frac{S}{R} + (T-1)\frac{S}{R}$$

Problem 40.

The fraction of the response time that is due to slow start is y/(x+y) where x = 2 RTT + O/R and y = P (RTT + S/R) – (2^P – 1) S/R

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 link to the appropriate output link. 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. KR

6. KR

7. KR

8. switching via memory; switching via a bus; switching via an interconnection network

9. Packet loss occurs if queue size at the input port grows large because of slow switching fabric speed and thus exhausting router's buffer space. It 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. Packet loss can occur if the queue size at the output port grows large because of slow outgoing line-speed.

11. HOL blocking – a queued packet in an input queue must wait for transfer through the fabric because it is blocked by another packet at the head of the line. It occurs at the input port.

12. Yes. They have one address for each interface.

13. 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 1 1 0 0

14.

15. 8 interfaces; 4 routing tables

16.

17. 50% overhead

18. The 8-bit protocol field in the IP datagram contains information about which transport layer protocol the destination host should pass the segment to.

19. KR

20. See Section 4.4.4

21. Yes, because the entire IPv6 datagram (including header fields) is encapsulated in an an IPv4 datagram

22. 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.

23. Routers are aggregated into autonomous systems (ASs). Within an AS, all routers run the same intra-AS routing protocol. Special gateway routers in the various ASs run the inter-autonomous system routing protocol that determines the routing paths among the ASs. The problem of scale is solved since an intra-AS router need only know about routers within its AS and the gateway router(s) in its AS.

24. No. Each AS has administrative autonomy for routing within an AS.

25. 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.

26. 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.

27. "sequence of ASs on the routes"

28. See "Principles in Practice" on page 384

29. JK

30. KR

31. KR

32. KR

33. JK

34. JK

35. False

36. 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.

37. 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 tress 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 router 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. Interesting, the design criteria that the initial ARPAnet be able to function under stressful conditions was one of the reasons that a datagram architecture was chosen for this Internet ancestor.

b) In order for a router to determine the delay (or a bound on delay) along an outgoing link, 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 network would be preferable.

Problem 2.

a) maximum number over a link = $2^{16} = 65536$
b) centralized node could pick any VC number which is 'free' from the $[0,2^{16}-1]$ set. It is possible that there are fewer VCs in progress than the maximum number without there being any common free VC number - ?
c) ?

(need help)

Problem 3.

For VC forwarding table the columns mean the following : Incoming Interface, Incoming VC number, Outgoing Interface and Outgoing VC number.
For datagram routing table the columns may mean the following : Destination address, Outgoing Interface.

Problem 4.

KR

Problem 5.

KR

Problem 6.

JK

Problem 7.

a)

Prefix Match	**Link Interface**
11100000	0
11100001 00000000	1
11100001	2

otherwise 3

b) Prefix match for first address is 4th entry -> link interface 3
Prefix match for second address is 2nd entry -> link interface 1
Prefix match for first address is 3rd entry -> link interface 2

Problem 8.

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

number of addresses in each range = $2^6 = 64$

Problem 9.

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

otherwise(128 addresses) 3

please confirm the correctness

Problem 10.

223.1.17.0/25
223.1.17.128/26
223.1.17.192/26

Problem 11.

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

Problem 12.

Destination Address	**Link Interface**
224/8	0
225.0/16	1
225/8	2
otherwise	3

Problem 13.

Example of IP address - 101.101.101.65
Four equal size subnets – 101.101.128/19, 101.101.160/19, 101.101.192/19, 101.101.224/19

Problem 14.

Possible assignments may be

a) Subnet A – 214.97.255/24
Subnet B – 214.97.254.0/25
Subnet C – 214.97.254.128/25

Subnet D – 214.97.253.0/31

Subnet E – 214.97.253.2/31
Subnet F – 214.97.253.4/31

b) **Router 1**

Longest Prefix Match	Outgoing Interface
11010110 01100001 11111111	Subnet A
11010110 01100001 11111101 0000000	Subnet D

Router 2

Longest Prefix Match	Outgoing Interface
11010110 01100001 11111110 0	Subnet B
11010110 01100001 11111101 0000010	Subnet E

Router 3

Longest Prefix Match	Outgoing Interface
11010110 01100001 11111110 1	Subnet C
11010110 01100001 11111101 0000100	Subnet F

Problem 15.

Maximum size of each fragmented datagram in user bytes = 480 (20 bytes IP header).

Number of fragmented datagrams required $= \left\lceil \frac{3000-20}{480} \right\rceil = 7$

Each fragmented datagram will have Identification number 422
Each fragmented datagram except the last one will be of size 500 bytes and including 20 bytes of IP header.
Last datagram will be of size 40 bytes (including 20 bytes header).

Problem 16.

Mp3 size = 4 million bytes.
Each datagram can carry 1500-20=1480 bytes of user data.

Number of datagrams required = $= \left\lceil \frac{4\times10^6}{1480} \right\rceil = 2703$

Problem 17.

a) Home addresses – 192.168.0.1, 192.168.0.2, 192.168.0.3 with the router interface being 192.168.0.4

b)

NAT Translation Table

WAN Side	LAN Side
128.119.40.86, 80	192.168.0.1, 3345
128.119.40.86, 80	192.168.0.1, 3355
128.119.40.86, 80	192.168.0.2, 3365
128.119.40.86, 80	192.168.0.2, 3375
128.119.40.86, 80	192.168.0.3, 3245
128.119.40.86, 80	192.168.0.3, 3945

Problem 18.

a) NAT will not have an entry for a connection initiated from the WAN side, hence will drop incoming packets from Arnold.
b) Bernard can know the IP address of Arnold through Cindy. Then, the p2p application can initiate a connection through NAT to Arnold and upload the file.

Problem 19.

There is a super-peer (say Cindy) which is not behind any NAT. Arnold and Bernard connect to the super-peer first, and send it the IP address they think they have; the server notes both that address and the address it sees in the UDP header. The server then sends both addresses to the other peers. At this point, everyone knows everyone else's address(es).

To open up peer-to-peer connections, Arnold and Bernard send a UDP packet to the new peer, and the new peer sends a UDP packet to each of Arnold and Bernard. Since nobody knows at first whether they are behind the same NAT, the first packet is always sent to both the public and the private address.

This causes everyone's NAT to open up a bidirectional hole for the UDP traffic to go through. Once the first reply comes back from each peer, the sender knows which return address to use, and can stop sending to both addresses.

This technique is also called NAT P2P hole punching.

Problem 20.

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

Problem 21.

N'	D(s), p(s)	D(t), p(t)	D(u), p(u)	D(v), p(v)	D(w), p(w)	D(y), p(y)	D(z), p(z)
	inf	inf	inf	3,x	1,x	6,x	inf
w	Inf	inf	4,w	2,w		6,x	inf
wv	Inf	11,v	3,v			3,v	inf
wvu	7,u	5,u				3,v	inf
wvuy	7,u	5,u					17,y
wvuyt	6,t						7,t
Wvuyts							7,t

Problem 22.

a) from node S to all nodes (note: ties broken in favor of leftmost column)

N'	D(t), p(t)	D(u), p(u)	D(v), p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
	1,s	4,s	inf	inf	Inf	inf	inf
t		3,t	t,10	inf	Inf	5,t	3,t
tu			4,u	6,u	Inf	5,t	3,t
tuz			4,u	6,u	Inf	5,t	
tuzv				5,v	7,v	5,t	
tuzvw					6,w	5,t	
tuzvwy					6,w		

b) from node B to all nodes (note: ties broken in favor of leftmost column)

N'	D(s), p(s)	D(u), p(u)	D(v), p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
	1,t	2,t	9,t	inf	Inf	4,t	2,t
S		2,t	9,t	inf	Inf	4,t	2,t
Su			3,u	5,u	Inf	4,t	2,t
suz			3,u	5,u	Inf	4,t	
suzv				4,v	6,v	4,t	
suzvw					5,w	4,t	
suzvwy					5,w		

c) from node C to all nodes (note: ties broken in favor of leftmost column)

N	D(s), p(s)	D(t), p(t)	D(v), p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
	4,u	2,u	1,u	3,u	Inf	inf	inf
v	4,u	2,u		2,v	4,v	2,v	inf
vt	3,t			2,v	4,v	2,v	4,t
vtw	3,t				3,w	2,v	4,t
vtwy	3,t				3,w		4,t
vtwys					3,w		4,t
vtwysx							4,t

d) from node D to all nodes (note: ties broken in favor of leftmost column)

N'	D(s), p(s)	D(t), p(t)	D(u),p (u)	D(w),p (w)	D(x),p (x)	D(y),p (y)	D(z),p (z)
	inf	9,v	1,v	1,w	3,v	1,v	inf
U	5,u	3,u		1,w	3,v	1,v	inf
Uw	5,u	3,u			2,w	1,v	inf
uwy	5,u	3,u			2,w		15,y
uwyx	5,u	3,u					15,y
uwyxt	4,t						6,t
uwyxts							6,t

e) from node E to all nodes (note: ties broken in favor of leftmost column)

N'	D(s) ,p(s)	D(t),p (t)	D(u),p (u)	D(v),p (v)	D(x),p (x)	D(y),p (y)	D(z),p (z) z
	inf	inf	3,w	1,w	1,x	inf	inf
v	inf	10,v	2,v		1,x	2,v	inf
vx	inf	10,v	2,v			2,v	inf
vxu	6,u	4,u				2,v	inf
vxuy	6,u	4,u					16,y
vxuyt	5,t						6,t
vxuyts							6,t

f) from node Y to all nodes (note: ties broken in favor of leftmost column)

N'	D(s) ,p(s)	D(t),p (t)	D(u),p (u)	D(v),p (v)	D(w),p (w)	D(x),p (x)	D(z),p (z)
	inf	4,y	inf	1,y	inf	6,y	14,y
v	inf	4,y	2,v		2,v	4,v	14,y
vu	6,u	4,u			2,v	4,v	14,y
vuw	6,u	4,u				3,w	14,y
vuwx	6,u	4,u					14,y
vuwxt	5,t						6,t
vuwxts							6,t

g) from node Z to all nodes (note: ties broken in favor of leftmost column)

N	D(s),p(s)	D(t),p(t)	D(u),p(u)	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)
	inf	2,z	inf	inf	inf	inf	14,z
T	3,t		4,t	11,t	inf	inf	6,t
Ts			4,t	11,t	inf	inf	6,t
Tsu				4,u	7,u	inf	6,t
tsuv					6,v	8,v	6,t
tsuvw						7,w	6,t
tsuvwy						7,w	

Problem 23.

The distance table in z is:

		Cost to			
		u	**v**	**x**	**y**
From	**v**	6	5	8	9
	x	4	5	2	3
	y	13	14	11	10

Problem 24.

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 25.

The distance table in X is:

		Cost to W	Y	A
From	**W**	1	?	6
	Y	?	4	10

Note that there is not enough information given in the problem (purposefully!) to determine the distance table entries D(W,Y) and D(Y,W). To know these values, we would need to know Y's minimum cost to W, and vice versa.

Since X's least cost path to A goes through W, a change in the link cost c(X,W) will cause X to inform its neighbors of a new minimum cost path to A, Since X's least cost path to A does not go through Y, a change in the link cost c(X,Y) will not cause X to inform its neighbors of a new minimum cost path to A.

Problem 26.

Node x table

		Cost to X	Y	Z
	X	0	5	2
from	**Y**	inf	inf	Inf
	Z	inf	inf	Inf

		Cost to X	Y	Z
	X	0	5	2
from	**Y**	3	0	6
	Z	2	6	0

Node y table

		Cost to X	Y	Z
	X	inf	inf	Inf
from	**Y**	5	0	6
	Z	inf	inf	Inf

		Cost to X	Y	Z
	X	0	5	2
from	**Y**	5	0	6
	Z	2	6	0

Node z table

		Cost to X	Y	Z
	X	inf	inf	Inf
from	**Y**	inf	inf	Inf
	Z	2	6	0

		Cost to X	Y	Z
	X	0	5	2
from	**Y**	5	0	6
	Z	2	6	0

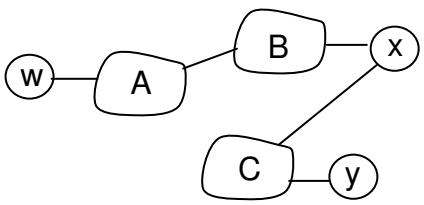
Problem 27.

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.

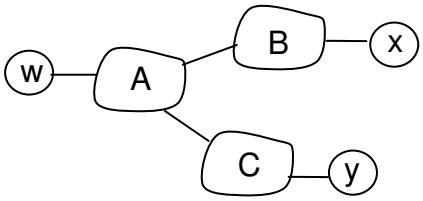
Problem 28.

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 29.



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 30.

The minimal spanning tree has G connected to D at a cost of 1, E connected to D at a cost of 1; C connected to D at a cost of 1; C connected to B at a a cost of 2; B connected to A at a cost of 1.
We can argue informally that the tree cost is minimal as follows. C, D, E, and G can be connected to each other at a cost of 3. One can't do any better than that. In order to connect A in to any of C, D, E, G, the minimum cost is 3 (via B).

Problem 31.

JK

Problem 32.

JK

Problem 33.

JK

Problem 34.

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

Problem 35.

JK

Problem 36.

Jk

Problem 37. Rakesh

Since there is no network layer protocol to identify hosts participating in a group, this must be done at a higher layer. In practice, this is done using an application-level protocol over IP multicast.

Problem 38.

Pseudo code

Problem 39.

$32-4=28$ 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

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

Ignoring cross-product terms, this is approximately equal to

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

Chapter 5 Review Questions

1. 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 emerging from 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. 2^{48} MAC addresses; 2^{32} IPv4 addresses; 2^{128} IPv6 addresses.

8. 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.

9. 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).

10. 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.

11. The three Ethernet technologies have identical frame structures.

12. 20 million transitions per second.

13. After the 5^{th} collision, the adapter chooses from {0, 1, 2, …, 31}. The probability that it chooses 4 is 1/32. It waits 204.8 microseconds.

14. When a receiving station correctly and completely receives a frame, it needs to send an acknowledgement to the transmitting station. The acknowledgement should be sent and correctly received before some other station attempts to transmit a frame. By making SIFS much smaller than DIFS, the receiving station can transmit its acknowledgement before other stations think that the channel is clear.

Chapter 5 Problems

Problem 1.

The rightmost column and bottom row are for parity bits.

1 0 1 0 0
1 0 1 0 0
1 0 1 0 0
1 0 1 1 1
0 0 0 1 1

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.

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

00000000 00000001
00000010 00000011
00000100 00000101
00000110 00000111
00001000 00001001

00010100 00011001

The one's complement of the sum is 11101011 11100110.

Problem 4.

If we divide 1001 into 10101010000 we get 10010111, with a remainder of R = 001.

Problem 5.

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

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

b)

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

$$\lim_{N\to\infty}(1-\frac{1}{N}) = 1 \qquad \lim_{N\to\infty}(1-\frac{1}{N})^N = \frac{1}{e}$$

Thus

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

Problem 6.

$$E(p) = Np(1-p)^{2(N-1)}$$
$$E'(p) = N(1-p)^{2(N-2)} - Np2(N-1)(1-p)^{2(N-3)}$$
$$= N(1-p)^{2(N-3)}((1-p) - p2(N-1))$$

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

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

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

Problem 7.

Pure Aloha has an efficiency of zero for almost all values of p. See Figure 6.

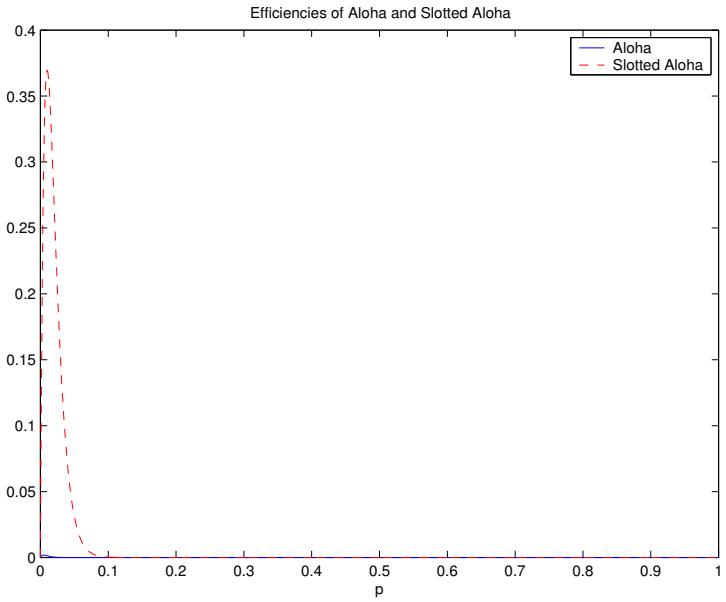


Figure 6: Efficiency of Aloha and Slotted Aloha

Problem 8.

The length of a polling round is

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

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

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

Problem 9.

a), b), c) See Figure 7.

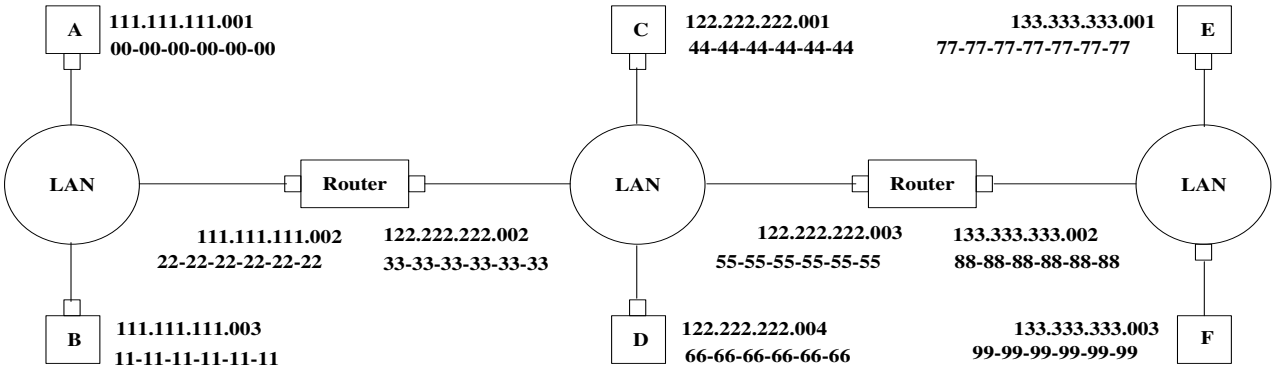


Figure 7: Solution to problem 12

d)

1. Forwarding table in A determines that the datagram should be routed to interface 111.111.111.002.
2. Host A uses ARP to determine the LAN address for 111.111.111.002, namely 22-22-22-22-22-22.
3. The adapter in A creates and Ethernet packet with Ethernet destination address 22-22-22-22-22-22.
4. The first router receives the packet and extracts the datagram. The forwarding table in this router indicates that the datagram is to be routed to 122.222.222.003.
5. The first router then uses ARP to obtain the associated Ethernet address, namely 55-55-55-55-55-55.
6. The process continues until the packet has reached Host F.

e)

ARP in A must now determine the LAN address of 111.111.111.002. Host A sends out an ARP query packet within a broadcast Ethernet frame. The first router receives the

query packet and sends to Host A an ARP response packet. This ARP response packet is carried by an Ethernet frame with Ethernet destination address 00-00-00-00-00-00.

.

Problem 10.

Wait for 51,200 bit times. For 10 Mbps, this wait is

$$\frac{51.2\times10^{3}\,bits}{10\times10^{6}\,bps}=5.12\,\text{msec}$$

For 100 Mbps, the wait is 512 μ sec.

Problem 11.

At $t=0$ A transmits. At $t=576$, A would finish transmitting. In the worst case, B begins transmitting at time $t=224$. At time $t=224+225=449$ B's first bit arrives at A. Because $449<576$, A aborts before completing the transmission of the packet, as it is supposed to do.

Thus A cannot finish transmitting before it detects that B transmitted. This implies that if A does not detect the presence of a host, then no other host begins transmitting while A is transmitting.

Problem 12.

Time, t	Event
0	A and B begin transmission
225	A and B detect collision
273	A and B finish transmitting jam signal
273+225 = 498	B's last bit arrives at A; A detects an idle channel
498+96=594	A starts transmitting
273+512 = 785	B returns to Step2 B must sense idle channel for 96 bit times before it transmits
594+225=819	A's transmission reaches B

Because A's retransmission reaches B before B's scheduled retransmission time, 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 13.

We want $1/(1+5a) = .5$ or, equivalently, $a = .2 = t_{prop} / t_{trans}$. $t_{prop} = d/(1.8 \times 10^8)$ m/sec and $t_{trans} = (576 \text{ bits})/(10^8 \text{ bits/sec}) = 5.76\mu$ sec. Solving for d we obtain $d = 265$ meters. For the 100 Mbps Ethernet standard, the maximum distance between two hosts is 200 m.

For transmitting station A to detect whether any other station transmitted during A's interval, t_{trans} must be greater than $2t_{prop} = 2 \cdot 265\text{m}/1.8 \times 10^8 \text{ m/sec} = 2.94\mu$ sec. Because $2.94 < 5.76$, A will detect B's signal before the end of its transmission.

Problem 14.

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

$$x = E[X] = E[Y] - 1 = \frac{1-\beta}{\beta}$$

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

$$x = \frac{1 - Np(1-p)^{N-1}}{Np(1-p)^{N-1}}$$

$$\text{efficiency} = \frac{k}{k+x} = \frac{k}{k + \dfrac{1 - Np(1-p)^{N-1}}{Np(1-p)^{N-1}}}$$

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)

$$\text{efficiency} = \frac{k}{k + \frac{1-(1-\frac{1}{N})^{N-1}}{(1-\frac{1}{N})^{N-1}}}$$

$$\lim_{N\to\infty} \text{efficiency} = \frac{k}{k + \frac{1-1/e}{1/e}} = \frac{k}{k+e-1}$$

d) Clearly, $\frac{k}{k+e-1}$ approaches 1 as $k \to \infty$.

Problem 15.

a)

$$\frac{900m}{2\cdot 10^8 m/\sec} + 4\cdot\frac{20bits}{10\times 10^6 bps}$$

$$= (4.5\times 10^{-6} + 8\times 10^{-6})\sec$$
$$= 12.5\mu\sec$$

b)

- At time $t = 0$, both A and B transmit.
- At time $t = 12.5\mu\sec$, A detects a collision.
- At time $t = 25\mu\sec$ last bit of B's aborted transmission arrives at A.
- At time $t = 37.5\mu\sec$ first bit of A's retransmission arrives at B.
- At time $t = 37.5\mu\sec + \frac{1000bits}{10\times 10^6 bps} = 137.5\mu\sec$ A's packet is completely delivered at B.

c) $12.5\mu\sec + 5\cdot 100\mu\sec = 512.5\mu\sec$

Problem 16.

The time required to fill $L\cdot 8$ bits is

$$\frac{L\cdot 8}{64\times 10^3}\sec = \frac{L}{8}m\sec.$$

b) For $L = 1{,}500$, the packetization delay is

$$\frac{1500}{8} m\sec = 187.5 m\sec .$$

For $L = 48,$ the packetization delay is

$$\frac{48}{8} m\sec = 6 m\sec .$$

c)

$$\text{Store-and-forward delay} = \frac{L \cdot 8 + 40}{R}$$

For $L = 1{,}500$, the delay is

$$\frac{1500 \cdot 8 + 40}{155 \times 10^6} \sec \approx \frac{12}{155} m\sec \approx 77 \mu\sec$$

For $L = 48,$ store-and-forward delay $< 1\mu\sec$.

d) Store-and-forward delay is small for both cases for typical ATM link speeds. However, packetization delay for $L = 1500$ is too large for real-time voice applications.

Problem 17.

JK

Chapter 6 Review Questions

1. Beacon Frames transmitted in each of the 11 channels help a wireless station to identify nearby APs.
2. Possible approaches: (1) based on the MAC address of wireless host (2) user name /password combination. In both cases AP relays information to an authentication server.
3. False
4. In wireless channels bit error rates are high and collision detection can not be effectively done.
5. False
6. Each wireless station can set an RTS threshold such that the RTS/CTS sequence is used only when the frame is longer than the threshold. This ensures that RTS/CTS mechanism is used only for large enough frames.
7. Advantages would be : (1) hidden terminal problem mitigated, (2) overall reduction in collision rates for DATA and ACK frames.
   Disadvantage : introduces delay and consumes channel resources.
8. Switch has an entry in its forwarding table which associates the wireless station with the earlier AP. The new AP spoofs wireless station' s MAC address and broadcasts the frame to the switch. This forces the switch to update its forwarding table making the association between wireless station and the new AP.
9. UMTS to GSM and CDMA-2000 to IS-95.

Chapter 6 Problems

Problem 1.

Output corresponding to bit d_1 = [-1,1,-1,1,-1,1,-1,1]
Output corresponding to bit d0 = [1,-1,1,-1,1,-1,1,-1]

Problem 2.

Sender 2 output = [1,-1,1,1,1,-1,1,1]; [1,-1,1,1,1,-1,1,1]

Problem 3.

$$d_2^1 = \frac{1\times1+(-1)\times(-1)+1\times1+1\times1+1\times1+(-1)\times(-1)+1\times1+1\times1}{8} = 1$$

$$d_2^2 = \frac{1\times1+(-1)\times(-1)+1\times1+1\times1+1\times1+(-1)\times(-1)+1\times1+1\times1}{8} = 1$$

Problem 4.

a) Each AP will typically have different SSID, hence 802.11 protocol will not break down at all. However, there will be a significant interference from the presence of other AP transmitting on the same channel, lower effective data rates.
b) No interference at all between channel 1 and 11.

Problem 5.

It is done to ensure a fair chances to other wireless stations competing for channel access. Otherwise, only the wireless channel which just transmitted will have the best chances of accessing the channel again, leading to increased collisions also.

Problem 6.

DIFS + <RTS> + SIFS + <CTS> + DIFS + <FRAME> + SIFS + <ACK>

Problem 7.

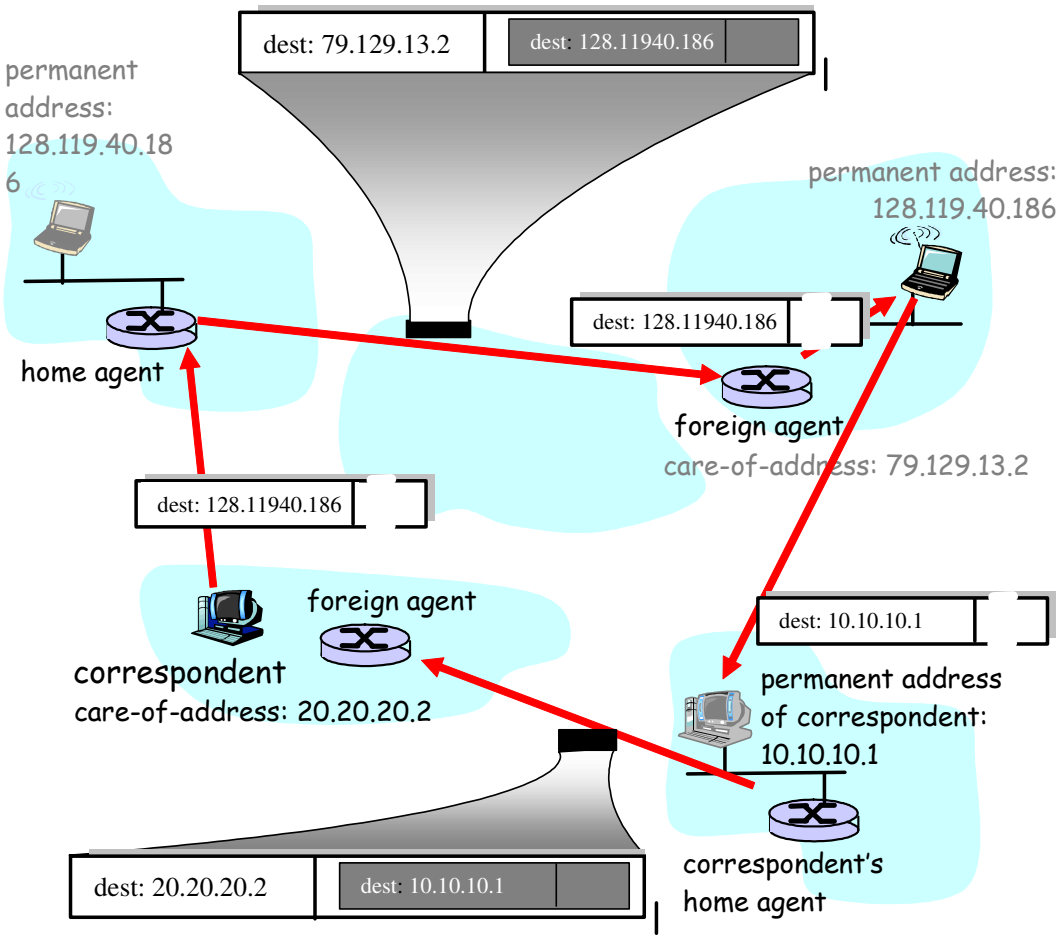
a) Recall that in distance vector routing, information about a change in destination passes only between neighboring nodes, whcn the neighboring nodes exhange routing updates/information. (The is in contrast to link state routing, where all changes in routing are broadcast to all routers, and thus all routers learn about changes in the network after just one link state broadcast). Thus, all routers will not be able to route to the mobile node immediately, under the assumption of distance vector routing.

b) Under distance vector routing, different routers may indeed have a different view of the visited network for the mobile node. A router will not know about the changed visited network until that information propagates to it via the pair-wise exchanges of routing information between routers on the path to the mobile node.

c) The timescale is roughly on the order of the diameter of the network (i.e., the length of the longest source-destination path). This is because routing information propagates only via pair-wise exchange between neighboring routers on the path. Thus the time it would take to propagate information from any point in the network to any other point is, in worst case, on the order of the diameter of the network

Problem 8.

If the correspondent is mobile, then any datagrams destined to the correspondent would have to pass through the **correspondent's home agent**. The **foreign agent** in the



network being visited would also need to be involved, since it is this foreign agent that notifies the correspondent's home agent of the location of the correspondent. Datagrams received by the correspondent's home agent would need to be encapsulated/tunneled between the correspondent's home agent and foreign agent, (as in the case of the encapsulated diagram at the top of Figure 6.19.

Problem 9.

Because datagrams must be first forward to the home agent, and from there to the mobile, the delays will generally be longer than via direct routing. Note that it *is* possible,

however, that the direct delay from the correspondent to the mobile (i.e., if the datagram is not routed through the home agent) could actually be smaller than the sum of the delay from the correspondent to the home agent and from there to the mobile. It would depend on the delays on these various path segments. Note that indirect routing also adds a home agent processing (e.g., encapsulation) delay.

Problem 10.

First, we note that chaining was discussed at the end of section 6.5. In the case of chaining using indirect routing through a home agent, the following events would happen:

- The mobile node arrives at A, A notifies the home agent that the mobile is now visiting A and that datagrams to the mobile should now be forwarded to the specified care-of-address (COA) in A.
- The mobile node moves to B. The foreign agent at B must notify the foreign agent at A that the mobile is no longer resident in A but in fact is resident in B and has the specified COA in B. From then on, the foreign agent in A will forward datagrams it receives that are addressed to the mobile's COA in A to the mobile's COA in B.
- The mobile node moves to C. The foreign agent at C must notify the foreign agent at B that the mobile is no longer resident in B but in fact is resident in C and has the specified COA in C. From then on, the foreign agent in B will forward datagrams it receives (from the foreign agent in A) that are addressed to the mobile's COA in B to the mobile's COA in C.

Note that when the mobile goes offline (i.e., has no address) or returns to its home network, the datagram-forwarding state maintained by the foreign agents in A, B and C must be removed. This teardown must also be done through signaling messages. Note that the home agent is not aware of the mobile's mobility beyond A, and that the correspondent is not at all aware of the mobile's mobility.

In the case that chaining is not used, the following events would happen:

- The mobile node arrives at A, A notifies the home agent that the mobile is now visiting A and that datagrams to the mobile should now be forwarded to the specified care-of-address (COA) in A.
- The mobile node moves to B. The foreign agent at B must notify the foreign agent at A and the home agent that the mobile is no longer resident in A but in fact is resident in B and has the specified COA in B. The foreign agent in A can remove its state about the mobile, since it is no longer in A. From then on, the home agent will forward datagrams it receives that are addressed to the mobile's COA in B.
- The mobile node moves to C. The foreign agent at C must notify the foreign agent at B and the home agent that the mobile is no longer resident in B but in fact is resident in C and has the specified COA in C. The foreign agent in B can remove its state about the mobile, since it is no longer in B. From then on, the home agent will forward datagrams it receives that are addressed to the mobile's COA in C.

-

When the mobile goes offline or returns to its home network, the datagram-forwarding state maintained by the foreign agent in C must be removed. This teardown must also be done through signaling messages. Note that the home agent is always aware of the mobile's current foreign network. However, the correspondent is still blissfully unaware of the mobile's mobility.

Problem 11.

Two mobiles could certainly have the same care-of-address in the same visited network. Indeed, if the care-of-address is the address of the foreign agent, then this address would be the same. Once the foreign agent decapsulates the tunneled datagram and determines the address of the mobile, then separate addresses would need to be used to send the datagrams separately to their different destinations (mobiles) within the visited network.

Problem 12.

JK

Chapter 7 Review Questions

1. Streaming stored audio/video: pause/resume, re-positioning, fast-forward; real-time interactive audio and video: people communicating and responding in real time.

2. Camp 1: No fundamental changes in TCP/IP protocols; add bandwidth where needed; also use caching, content distribution networks, and multicast overlay networks. Camp 2: Provide a network service that allows applications to reserve bandwidth in the network. Camp 3, differentiated service: introduce simple classifying and policing schemes at the edge of the network, and give different datagrams different levels of service according to their class in the router queues.

3. Figure 6.1: simple, doesn't require meta file or streaming server; Figure 6.2: allows media player to interact directly with the web server, doesn't require a streaming server; Figure 6.3: media player interacts directly with a streaming server, which has been designed for the specific streaming application.

4. End-to-end delay is the time it takes a packet to travel across the network from source to destination. Delay jitter is the fluctuation of end-to-end delay from packet to the next packet.

5. A packet that arrives after its scheduled playout time can not be be played out. Therefore, from the perspective of the application, the packet has been lost.

6. First scheme: send a redundant encoded chunk after every n chunks; the redundant chunk is obtained by exclusive OR-ing the n original chunks. Second scheme: send a lower-resolution low-bit rate scheme along with the original stream. Interleaving does not increase the bandwidth requirements of a stream.

7. RTP streams in different sessions: different multicast addresses; RTP streams in the same session: SSRC field; RTP packets are distinguished from RTCP packets by using distinct port numbers.

8. Reception report packets: includes info about fraction of packets lost, last sequence number, interarrival jitter; sender report packets: timestamp and wall clock time of most recently generated RTP packet, number of packets sent, number of bytes sent ; source description packets: e-mail address of the sender, the sender's name, the application that generates the RTP stream.

9. In non-preemptive priority queuing, the transmission of a packet is not interrupted once it has begun. In preemptive priority queuing, the transmission of a packet will be interrupted if a higher priority packet arrives before transmission completes. This would mean that portions of the packet would be sent into the network as separate chunks; these chunks would no longer all have the appropriate header fields. For this reason, preemptive priority queuing is not used.

10. A scheduling discipline that is not work conserving is time division multiplexing, whereby a rotating frame is partitioned into slots, with each slot exclusively available to a particular class.

11. Scalability: Per-flow resource reservation implies the need for a router to process resource reservations and maintain per-flow state for each flow passing through the router. Flexibly service: The Intserv framework provides for a small number of pre-specified service classes.

Chapter 7 Problems

Problem 3.
$x(t)$ will continue to grow until the client buffer becomes full. Once the client buffer becomes full, the client application will drain the receive TCP buffer at rate d. TCP flow control will then throttle the sender's transmission rate so that the average of $x(t)$ after the client buffer becomes full is approximately d.

Problem 4.
No, they are not the same thing. The client application reads data from the TCP receive buffer and puts it in the client buffer. If the client buffer becomes full, then application will stop reading from the TCP receive buffer until some room opens up in the client buffer.

Problem 5.
a) $160 + h$ bytes are sent every 20 msec. Thus the transmission rate is

$$\frac{(160+h)\cdot 8}{20} Kbps = (64 + .4h) Kbps$$

b)

IP header:	20bytes
UDP header:	8bytes
RTP header:	12bytes

h=40 bytes (a 25% increase in the transmission rate!)

Problem 6.
a) Denote $d^{(n)}$ for the estimate after the *n*th sample.

$$d^{(1)} = r_4 - t_4$$

$$d^{(2)} = u(r_3 - t_3) + (1-u)(r_4 - t_4)$$

$$d^{(3)} = u(r_2 - t_2) + (1-u)\left[u(r_3 - t_3) + (1-u)(r_4 - t_4)\right]$$

$$= u(r_2 - t_2) + (1-u)u(r_3 - t_3) + (1-u)^2(r_4 - t_4)$$

$$d^{(4)} = u(r_1 - t_1) + (1-u)d^{(3)}$$

$$= u(r_1 - t_1) + (1-u)u(r_2 - t_2) + (1-u)^2 u(r_3 - t_3) + (1-u)^3(r_4 - t_4)$$

b)

$$d^{(n)} = u\sum_{j=1}^{n-1}(1-u)^j(r_j - t_j) + (1-u)^n(r_n - t_n)$$

c)

$$d^{(\infty)} = \frac{u}{1-u}\sum_{j=1}^{\infty}(1-u)^j(r_j - t_j)$$

$$= \frac{1}{9}\sum_{j=1}^{\infty}.9^j(r_j - t_j)$$

The weight given to past samples decays exponentially.

Problem 7.

a) Denote $v^{(n)}$ for the estimate after the *n*th sample. Let $\Delta_j = r_j - t_j$.

$$v^{(1)} = \left| \ \Delta_4 - d^{(1)} \ \right| \quad (=0)$$

$$v^{(2)} = u\left| \ \Delta_3 - d^{(2)} \ \right| + (1-u)\left| \ \Delta_4 - d^{(1)} \ \right|$$

$$v^{(3)} = u\left| \ \Delta_2 - d^{(3)} \ \right| + (1-u)v^{(2)}$$

$$= u\left| \ \Delta_2 - d^{(3)} \ \right| + u(1-u)\left| \ \Delta_3 - d^{(2)} \ \right| + (1-u)^2\left| \ \Delta_4 - d^{(1)} \ \right|$$

$$v^{(4)} = u\left| \ \Delta_1 - d^{(4)} \ \right| + (1-u)v^{(3)}$$

$$= u\left| \ \Delta_1 - d^{(4)} \ \right| + (1-u)u\left| \ \Delta_2 - d^{(3)} \ \right| + u(1-u)^2\left| \ \Delta_3 - d^{(2)} \ \right|$$

$$+ (1-u)^3\left| \ \Delta_4 - d^{(1)} \ \right|$$

$$= u\left[\ \left| \ \Delta_1 - d^{(4)} \ \right| + (1-u)\left| \ \Delta_2 - d^{(3)} \ \right| + (1-u)^2\left| \ \Delta_3 - d^{(2)} \ \right| \ \right]$$

$$+(1-u)^3\left|\ \Delta_4 - d^{(1)}\ \right|$$

b)

$$v^{(n)} = u\sum_{j=1}^{n-1}(1-u)^{j-1}\left|\ \Delta_j - d^{(n-j+1)}\ \right| + (1-u)^n\left|\ \Delta_n - d^{(1)}\ \right|$$

Problem 8.

a) $r_1 - t_1 + r_2 - t_2 + \ldots + r_{n-1} - t_{n-1} = (n-1)d_{n-1}$

Substituting this into the expression for d_n gives

$$d_n = \frac{n-1}{n}d_{n-1} + \frac{r_n - t_n}{n}$$

b) The delay estimate in part (a) is an average of the delays. It gives equal weight to recent delays and to "old" delays. The delay estimate in Section 6.3 gives more weight to recent delays; delays in the distant past have relatively little impact on the estimate.

Problem 9.

The two procedures are very similar. They both use the same formula, thereby resulting in exponentially decreasing weights for past samples.

One difference is that for estimating average RTT, the time when the data is sent and when the acknowledgement is received is recorded on the same machine. For the delay estimate, the two values are recorded on different machines. Thus the sample delay can actually be negative.

Problem 10.
a) If there is packet lost, then other packets may have been transmitted in between the two received packets.
b) Let S_i denote the sequence number of the *i*th received packet. If

$$t_i - t_{i-1} > 20m\,\text{sec}$$

and

$$S_i = S_{i-1} + 1$$

then packet *I* begins a new talkspurt.

Problem 11.

a) Both schemes require 25% more bandwidth. The first scheme has a playback delay of 5 packets. The second scheme has a delay of 2 packets.

b) The first scheme will be able to reconstruct the original high-quality audio encoding. The second scheme will use the low quality audio encoding for the lost packets and will therefore have lower overall quality.

c) For the first scheme, many of the original packets will be lost and audio quality will be very poor. For the second scheme, every audio chunk will be available at the receiver, although only the low quality version will be available for every other chunk. Audio quality will be acceptable.

Problem 12.

The interarrival jitter *J* is defined to be the mean deviation (smoothed absolute value) of the difference *D* in packet spacing at the receiver compared to the sender for a pair of packets. As shown in the equation below, this is equivalent to the "relative transit time" for the two packets; the relative transit time is the difference between a packet's RTP timestamp and the receiver's clock at the time of arrival. If T_i is the RTP timestamp for packet i and r_i is the time of arrival in RTP timestamp units for packet *i*, then for two packets *i* and *j*, *D* is defined as

$$D(i, j) = (r_i - r_j) - (s_i - s_j) = (r_j - s_j) - (r_i - s_i)$$

The interarrival jitter is calculated continuously as each data packet *i* is received, using this difference *D* for that packet and the previous packet *i*-1 in order of arrival (not necessarily in sequence), according to the formula

$$J = J + \frac{|D(i, j) - J|}{16}$$

Whenever a reception report is issued, the current value of *J* is sampled.

Problem 13.

a) As discussed in Chapter 2, UDP sockets are identified by the two-tuple consisting of destination IP address and destination port number. So the two packets will indeed pass through the same socket.

b) Yes, Alice only needs one socket. Bob and Claire will choose different SSRC's, so Alice will be able distinguish between the two streams. Another question we could have asked is: How does Alice's software know which stream (ie SSRC) belongs to Bob and which stream belongs to Alice? Indeed, Alice's software may want to display the sender's name when the sender is talking. Alice's software gets the SSRC to name mapping from the RTCP source description reports.

Problem 14

a) The session bandwidth is 4* 100 kbps = 400 kbps. Five percent of the session bandwidth is 20 kbps.
b) Sends each user is both a sender and receiver, each user gets 5 kbps for RTCP packets (receiver reports, sender reports, and source description packets).

Problem 15.

a) Like HTTP, all request and response methods are in ASCII text. RTSP also has methods (SETUP, PLAY, PAUSE), and the server responds with standardized reply codes. Yes, using the GET method, HTTP can be used to request a stream.
b) RTSP messages use different port numbers than the media streams. Thus RTSP is out-of-band. HTTP objects are sent within the HTTP response message. Thus HTTP is in-band. HTTP does not keep session state: each request is handled independently. RTSP uses the Session ID to maintain session state. For example, in the lab (programming assignment) for this chapter, the RTSP server is in one several states for each session ID. When in the pause state, the server stores the number of the frame at which the pause occurred.

Problem 17.

a) true
b) true
c) No, RTP streams can be sent to/from any port number. See the SIP example in Section 6.4.3
d) No, typically they are assigned different SSRC values.
e) True
f) False, she is indicating that she wishes to *receive* GSM audio
g) False, she is indicating that she wishes to *receive* audio on port 48753
h) True, 5060 for both source and destination port numbers
i) True
j) False, this is a requirement of H.323 and not SIP.

Problem 18.

a) One possible sequence is 1 2 1 3 1 2 1 3 1 2 1 3 …
Another possible sequence is 1 1 2 1 1 3 1 1 2 1 1 3 1 1 2 1 1 3 …
b) 1 1 3 1 1 3 1 1 3 1 1 3 …

Problem 19.

See figure below. For the second leaky bucket, $r = p, b = 1$.

Problem 20.

No.

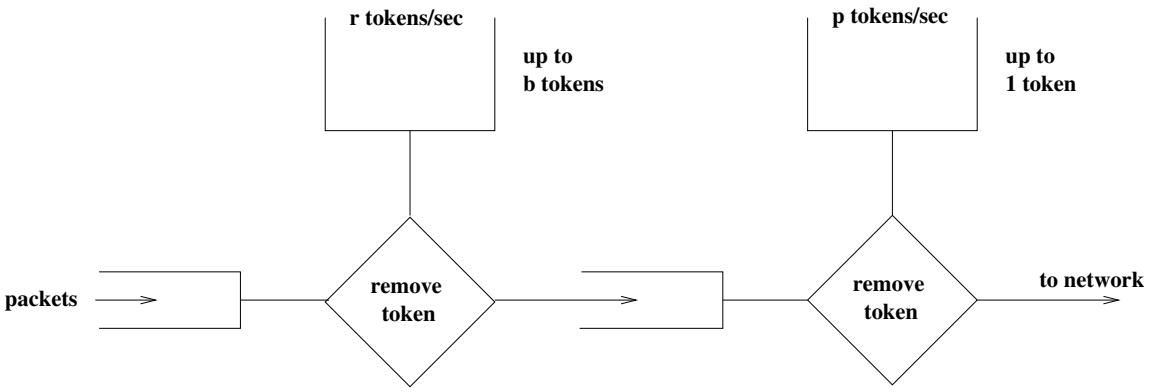


Figure: Solution to problem 19

Problem 21.

Let τ be a time at which flow 1 traffic starts to accumulate in the queue. We refer to τ as the beginning of a flow-1 busy period. Let $t > \tau$ be another time in the same flow-1 busy period. Let $T_1(\tau, t)$ be the amount of flow-1 traffic transmitted in the interval $[\tau, t]$. Clearly,

$$T_1(\tau, t) \geq \frac{W_1}{\sum W_j} R(t - \tau)$$

Let $Q_1(t)$ be the amount of flow-1 traffic in the queue at time *t*. Clearly,

$$Q_1(t) = b_1 + r_1(t - \tau) - T_1(\tau, t)$$

$$\leq b_1 + r_1(t - \tau) + \frac{W_1}{\sum W_j} R(t - \tau)$$

$$= b_1 + (t - \tau)\left[r_1 - \frac{W_1}{\sum W_j} R \right]$$

Since $r_1 < \frac{W_1}{\sum W_j} R$, $Q_1(t) \leq b_1$. Thus the maximum amount of flow-1 traffic in the queue is b_1. The minimal rate at which this traffic is served is $\frac{W_1 R}{\sum W_j}$.

Thus, the maximum delay for a flow-1 bit is

$$\frac{b_1}{W_1 R / \sum W_j} = d_{\max}.$$

Chapter 8 Review Questions

Question 1.
Confidentiality is the property that the original plaintext message can not be determined by an attacker who intercepts the ciphertext-encryption of the original plaintext message. Message integrity is the property that the receiver can detect whether the message sent (whether encrypted or not) was altered in transit. The two are thus different concepts, and one can have one without the other. An encrypted message that is altered in transmit may still be confidential (the attacker can not determine the original plaintext) but will not have message integrity if the error is undetected. Similarly, a message that is altered in transit (and detected) could have been sent in plaintext and thus would not be confidential.

Question 2.
A passive intruder only monitors ("sniffs", intercepts) messages. An active intruder can also monitor traffic, but will also actively send messages into the network

Question 3.
One important difference between symmetric and public key systems is that in symmetric key systems both the sender and receiver must know the same (secret) key. In public key systems, the encryption and decryption keys are distinct. The encryption key is known by the entire world (including the sender), but the decryption key is known only by the receiver.

Question 4.
In this case, a known plaintext attack is performed. If, somehow, the message encrypted by the sender was chosen by the attacker, then this would be a chosen-plaintext attack.

Question 5.
If each user wants to communicate with N other users, then each pair of users must have a shared symmetric key. There are N*(N-1)/2 such pairs and thus there are N*(N-1)/2 keys. With a public key system, each user has a public key which is known to all, and a private key (which is secret and only known by the user). There are thus 2N keys in the public key system.

Question 6.
A nonce is used to ensure that the person being authenticated is "live." Nonces thus are used to combat playback attacks.

Question 7.
Once in a lifetimes means that the entity sending the noce will never again use that value to check whether another entity is "live".

Question 8.
In a man-in-the-middle attacker, the attacker interposes him/herself between the sender and receiver, often performing some transformation (e.g., re-encoding or altering) of data

between the sender and receiver. Man-in-the-middle attacks can be particularly pernicious since (as shown in Figure 7.13) the sender and receiver will each receive what the other has sent and since they are using encryption would think that they have achieved confidentiality.

Question 9.
Suppose Bob sends an encrypted document to Alice. To be verifiable, Alice must be able to convince herself that Bob sent the encrypted document. To be non-forgeable, Alice must be able to convince herself that only Bob could have sent the encrypted document (e.g.,, non one else could have guess a key and encrypted/sent the document) To be non-reputiable, Alice must be able to convince someone else that only Bob could have sent the document. To illustrate the latter distinction, suppose Bob and Alice share a secret key, and they are the only ones in the world who know the key. If Alice receives a document that was encrypted with the key, and knows that she did not encrypt the document herself, then the document is known to be verifiable and non-forgeable (assuming a suitably strong encryption system was used). However, Alice can *not* convince someone else that Bob must have sent the document, since in fact Alice knew the key herself and could have encrypted/sent the document.

Question 10.
One requirement of a message digest is that given a message M, it is very difficult to find another message M' that has the same message digest and, as a corollary, that given a message digest value it is difficult to find a message M" that has that given message digest value. We have "message integrity" in the sense that we have reasonable confidence that given a message M and its signed message digest that the message was not altered since the message digest was computed and signed. This is not true of the Internet checksum, where we saw in Figure 7.18 that it easy to find two messages with the same Internet checksum.

Question 11.
A public-key signed message digest is "better" in that one need only encrypt (using the private key) a short message digest, rather than the entire message. Since public key encryption with a technique like RSA is expensive, it's desirable to have to sign (encrypt) a smaller amount of data than a larger amount of data.

Question 12.
The message associated with a message digest value need not be encrypted. Encrypting the message provides for confidentiality, which the message digest provides for integrity – two different goals.

Question 13.
A key distribution center is used to create a distribute a symmetric session key for two communicating parties, requiring only that the two parties each have their own symmetric key that allows them to encrypt/decrypt communication to/from the key distribution center. A certification authority binds an individual's identity with a public key. The CA signs that key with its (the CAs) private key. Thus, given the public key of a CA, one can

retrieve the CA-signed public key for an entity, verify the CA's signature, and then have the CA-certified public key for an entity.

Question 14.
The AH provides for authentication and message integrity, while ESP provides for authentication, integrity, and confidentiality.

Chapter 8 Problems

Problem 1.
The encoding of "This is an easy problem" is "uasi si my cmiw lokngch".
The decoding of "rmij'u uamu xyj" is "wasn't that fun".

Problem 2.
If Trudy knew that the words "bob" and "alice" appeared in the text, then she would know the ciphertext for b,o,a,l,i,c,e (since "bob" is the only palindrome in the message, and "alice" is the only 5-letter word. If Trudy knows the ciphertext for 7 of the letters, then she only needs to try 19!, rather than 26!, plaintext-ciphertext pairs. The difference between 19! and 26! is 26*25*24...*20, which is 3315312000, or approximately 10^9.

Problem 3.
Every letter in the alphabet appears in the phrase "The quick fox jumps over the lazy brown dog." Given this phrase in a chosen plaintext attack (where the attacker has both the plain text, and the ciphertext), the Caesar cipher would be broken - the intruder would know the ciphertext character for every plaintext character. However, the Vigenere cipher does not alway translate a given plaintext character to the same ciphertext character each time, and hence a Vigenere cipher would not be immediately broken by this chosen plaintext attack.

Problem 4.
We are given $p = 3$ and $q = 11$. We thus have $n = 33$ and $q = 11$. Choose $e = 9$ (it might be a good idea to give students a hint that 9 is a good value to choose, since the resulting calculations are less likely to run into numerical stability problems than other choices for e.) since 3 and $(p-1)*(q-1) = 20$ have no common factors. Choose $d = 9$ also so that $e*d = 81$ and thus $e*d-1 = 80$ is exactly divisible by 20. We can now perform the RSA encryption and decryption using $n = 33$, $e = 9$ and $d = 9$.

letter	m	m**e	ciphertext = m**e mod 33
h	8	134217728	29
e	5	1953125	20
l	12	5159780352	12
l	12	5159780352	12
o	15	38443359375	3

ciphertext	c**d	m = c**d mod n	letter
29	14507145975869	8	h

```
20          512000000000    5           e
12          5159780352      12          l
12          5159780352      12          l
3           19683           15          o
```

Problem 5.

Bob does not know if he is talking to Trudy or Alice initially. Bob and Alice share a secret key K_{A-B} that is unknown to Trudy. Trudy wants Bob to authenticate her (Trudy) as Alice. Trudy is going to have Bob authenticate himself, and waits for Bob to start:

1. **Bob-to-Trudy: "I am Bob"** Commentary: Bob starts to authenticate himself. Bob's authentication of himself to the other side then stops for a few steps.
2. **Trudy-to-Bob: "I am Alice"** Commentary: Trudy starts to authenticate herself as Alice
3. **Bob-to-Trudy: "R"** Commentary: Bob responds to step 2 by sending a nonce in reply. Trudy does not yet know $K_{A-B}(R)$ so she can not yet reply.
4. **Trudy-to-Bob: "R"** Commentary: Trudy responds to step 1 now continuing Bob's authentication, picking as the nonce for Bob to encrypt, *the exact same value that Bob sent her to encrypt in Step 3.*
5. **Bob-to-Trudy: "**$K_{A-B}(R)$" Bob completes his own authentication of himself to the other side by encrypting the nonce he was sent in step 4. Trudy now has $K_{A-B}(R)$. (Note: she does not have, nor need, K_{A-B}
6. *Trudy-to-Bob:* **"**$K_{A-B}(R)$" Trudy completes her authentication, responding to the R that Bob sent in step 3 above with $K_{A-B}(R)$. Since Trudy has returned the properly encrypted nonce that Bob send in step 3, Bob thinks Trudy is Alice!

Problem 6.

This wouldn't really solve the problem. Just as Bob thinks (incorrectly) that he is authenticating Alice in the first half of Figure 7.14, so too can Trudy fool Alice into thinking (incorrectly) that she is authenticating Bob. The root of the problem that neither Bob nor Alice can tell is the public key they are getting is indeed the public key of Alice of Bob.

Problem 7.

As discussed in section 7.4.2, full blown encryption is more computationally complex than a message digest such as MD5.

Problem 8.

The message

```
I  O  U  2
0  0  .  8
9  B  O  B
```

has the same checksum.

Problem 9.
If Alice wants to ensure that the KDC is live (that is, the message she will be receiving ack from the KDC are not part of a playback attack), she can include a nonce, R0 in the initial message ($K_{A-KDC}(A, B, R0)$) to the KDC. The KDC would then include R) in the reply back to Alice, thus proving the KDC is indeed live. Note that it is already assumed that only the KDC and Alice know the key to decrypt $K_{A-KDC}(A, B, R0)$.

Problem 10.
The message from Alice is encoded using a key that is only known to Alice and the KDC. Therefore the KDC knows (by definition) that anyone using the key must be Alice. It is interesting to think about what damage Trudy could do if she obtains Alice's key. In this case, she can impersonate Alice to anyone; see also the answer to question 11.

Problem 11.
If the KDC goes down, no one can communicate securely, as a first step in communication (see Figure 7.19) is to get the one-time session key from the KDC. If the CA goes down, then as along as the CA's public key is known, one can still communicate securely using previously-issued certificates (recall that once a certificate is issued, the CA is not explicitly involved in any later communication among parties using the CA's certificate. Of course, if the CA goes down, no new certificates can be issued.

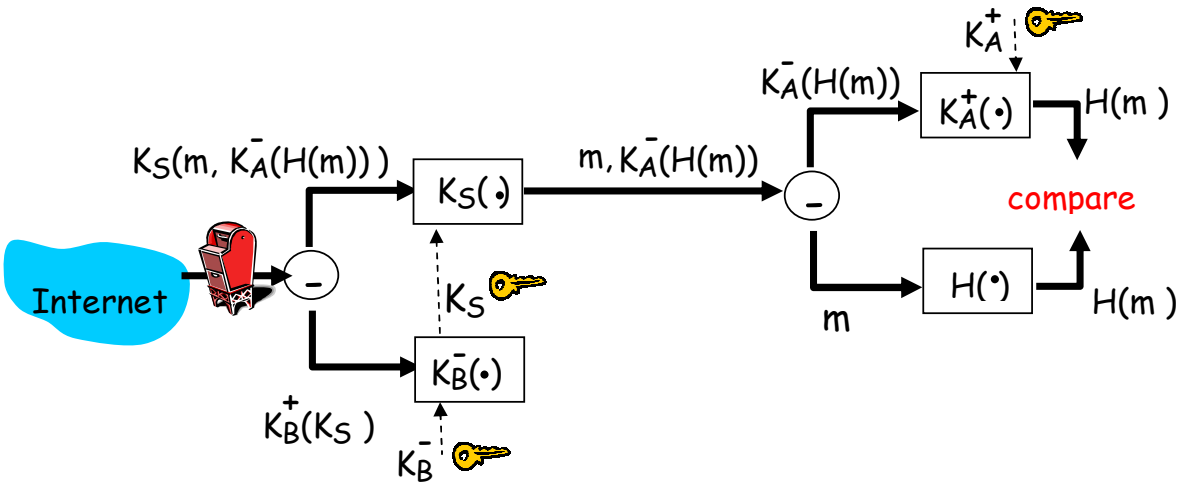
Problem 12.

datagram #	Source IP addr	Dest IP add	desired action	action under R1, R2, R3	action under R2 R1 R3
P1	111.11.11.1 (hacker subnet)	222.22.6.6 (corp.net)	deny	deny(R2)	deny(R2)
P2	111.11.11.1 (hacket subnet)	222.22.22.2 (special subnet)	permit	permit(R1)	deny(R2)
P3	111.11.6.6 (univ. net, not hacker subnet)	222.22.22.2 (special subnet)	permit	permit(R1)	permit(R1)
P4	111.11.6.6 (univ. net, not hacker subnet)	222.22.6.6 (corp. net)	deny	deny(R3)	deny(R3)

Under the ordering R1, R2, R3, removing R2 would have no effect. Note that under this ordering, R2 is only involved in the denial of P1. Since P1 would also be denied under R3, the removal of P2 would have no effect.

Under the ordering R2, R1, R3, if R2 is removed then P2 would be admitted.

Problem 13.



Chapter 9 Review Questions

Question 1.
A network manager would like to have network management capabilities when (a) a component of the network fails, (b) a component of the network is about to fail, and is acting "flaky" (c) a component of the network has been compromised from a security standpoint and is attacking the network, e.g., by launching a DOS attack by flooding the network with packets, (d) traffic levels exceed a certain threshold on a link, causing packets to be dropped, (e) everything is running smoothly (in order to *know* that everything is running smoothly and there are no problems). There are many additional reasons as well.

Question 2.
Performance management, fault management, configuration management, accounting management, security management.

Question 3.
Network management is more narrowly defined, as it focuses on the resources in the network – monitoring their functions and controlling their operation. These resources are combined (used) in various ways to implement services. Note that while the network resources may all be functioning as they should, they may not be sufficient to implement a service with a given level of performance; this latter concern is an aspect of service management.

Question 4.

- ***Managing entity:*** control the collection, processing, analysis, display of network management information, and is used by the network manager to control the devices in the network.
- ***Managed device:*** a piece of network equipment that is under the control of the managing entity.
- ***Management agent:*** a software process running on a managed device that communicated with the managing entity and takes action on the managed device under the control of the managing entity.
- ***MIB:*** pieces of information associated with all of the managed objects in a device.
- ***Network management protocol:*** runs between the managing entity of the management agents on the managed devices, allowing the agents to alert the managing entity to potential problems, and allowing the managing entity to send commands to the management agents.

Question 5.
The SMI is a data-definition language used to defined the pieces of information in an SNMP MIB.

Question 6.

The trap message is sent by the management agent to the managing entity (and requires no response from the managing entity). A request-response message is sent by the managing entity, with the response coming back from the management agent.

Question 7.
GetRequest, GetNextRequest, GetBulkRequest, SetRequest, InformRequest, Response, Trap

Question 8.
The SNMP engine is the part of an SNMP implementation that handles the dispatching, processing, authentication, access control, and timeliness of he SNMP messages. See Figure 8.5.

Question 9.
The ASN.1 object identifier tree provides a standard way to name objects.

Question 10.
The role of the presentation layer is to allow the sending and receiving of data in a machine-independent format (i.e., without regard to the particular storage and architectural conventions of the sender and receiver).

Question 11.
The Internet does not have presentation layer. It is up to the implementers of protocols to make sure that data is sent and received in the desired format.

Question 12.
In TLV encoding, each piece of data is tagged with its type, length, and value.

Chapter 9 Problems

Problem 1.
Request response mode will generally have more overhead (measured in terms of the number of messages exchanged) for several reasons. First, each piece of information received by the manager requires two messages: the poll and the response. Trapping generates only a single message to the sender. If the manager really only wants to be notified when a condition occurs, polling has more overhead, since many of the polling messages may indicate that the waited-for condition has not yet occurred. Trapping generates a message only when the condition occurs.

Trapping will also immediately notify the manager when an event occurs. With polling, the manager needs will need to wait for half a polling cycle (on average) between when the event occurs and the manager discovers (via its poll message) that the event has occurred.

If a trap message is lost, the managed device will not send another copy. If a poll message, or its response, is lost the manager would know there has been a lost message (since the reply never arrives). Hence the manager could repoll, if needed.

Problem 2.
Often, the time when network management is most needed is in times of stress, when the network may be severely congested and packets are being lost. With SNMP running over TCP, TCP's congestion control would cause SNMP to back-off and stop sending messages at precisely the time when the network manager needs to send SNMP messages.

Problem 3.
1.3.6.1.2.1.5

Problem 4.
Microsoft file formats are under 1.2.840.113556.4 (see section 8.3.2)

Problem 5.
Your probably be under 1.2.840

Problem 6.

```
3  7  'J' 'a' 'c' 'k' 's' 'o' 'n' 2  2  1  15
```