Chapter 2 Review Questions

1. The Web: HTTP; file transfer: FTP; remote login: Telnet; e-mail: SMTP; BitTorrent file sharing: BitTorrent protocol

2. Network architecture refers to the organization of the communication process into layers (e.g., the five-layer Internet architecture). Application architecture, on the other hand, is designed by an application developer and dictates the broad structure of the application (e.g., client-server or P2P).

3. The process which initiates the communication is the client; the process that waits to be contacted is the server.

4. No. In a P2P file-sharing application, the peer that is receiving a file is typically the client and the peer that is sending the file is typically the server.

5. The IP address of the destination host and the port number of the socket in the destination process.

6. You would use UDP. With UDP, the transaction can be completed in one roundtrip time (RTT) - the client sends the transaction request into a UDP socket, and the server sends the reply back to the client's UDP socket. With TCP, a minimum of two RTTs are needed - one to set-up the TCP connection, and another for the client to send the request, and for the server to send back the reply.

7. One such example is remote word processing, for example, with Google docs. However, because Google docs runs over the Internet (using TCP), timing guarantees are not provided.

8. a) Reliable data transfer
    TCP provides a reliable byte-stream between client and server but UDP does not.
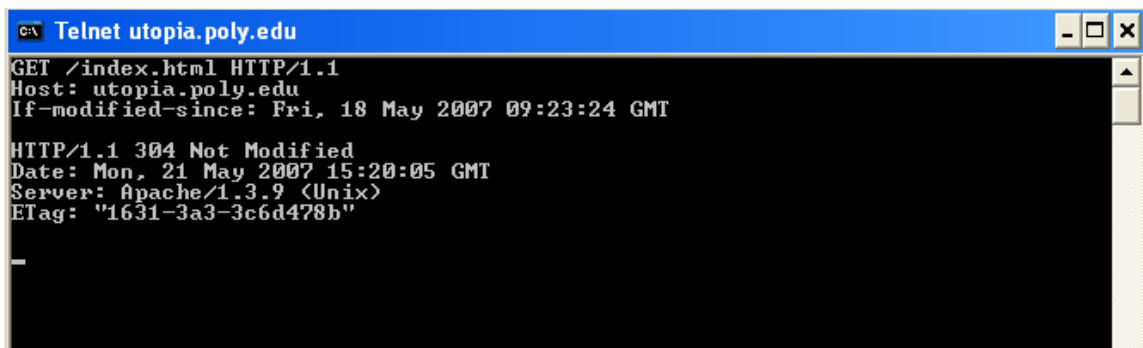
    b) A guarantee that a certain value for throughput will be maintained
    Neither

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

    d) Confidentiality (via encryption)
    Neither

9. SSL operates at the application layer. The SSL socket takes unencrypted data from the application layer, encrypts it and then passes it to the TCP socket. If the application developer wants TCP to be enhanced with SSL, she has to include the SSL code in the application.

10. A protocol uses handshaking if the two communicating entities first exchange control packets before sending data to each other. SMTP uses handshaking at the application layer whereas HTTP does not.

11. The applications associated with those protocols require that all application data be received in the correct order and without gaps. TCP provides this service whereas UDP does not.

12. When the user first visits the site, the server creates a unique identification number, creates an entry in its back-end database, and returns this identification number as a cookie number. This cookie number is stored on the user's host and is managed by the browser. During each subsequent visit (and purchase), the browser sends the cookie number back to the site. Thus the site knows when this user (more precisely, this browser) is visiting the site.

13. Web caching can bring the desired content "closer" to the user, possibly to the same LAN to which the user's host is connected. Web caching can reduce the delay for all objects, even objects that are not cached, since caching reduces the traffic on links.

14. Telnet is not available in Windows 7 by default. to make it available, go to Control Panel, Programs and Features, Turn Windows Features On or Off, Check Telnet client. To start Telnet, in Windows command prompt, issue the following command
    > telnet webserverver 80

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

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

15. FTP uses two parallel TCP connections, one connection for sending control information (such as a request to transfer a file) and another connection for actually transferring the file.

Because the control information is not sent over the same connection that the file is sent over, FTP sends control information out of band.

16. The message is first sent from Alice's host to her mail server over HTTP. Alice's mail server then sends the message to Bob's mail server over SMTP. Bob then transfers the message from his mail server to his host over POP3.

17.

Received:	from 65.54.246.203 (EHLO bay0-omc3-s3.bay0.hotmail.com) (65.54.246.203) by mta419.mail.mud.yahoo.com with SMTP; Sat, 19 May 2007 16:53:51 -0700
Received:	from hotmail.com ([65.55.135.106]) by bay0-omc3-s3.bay0.hotmail.com with Microsoft SMTPSVC(6.0.3790.2668); Sat, 19 May 2007 16:52:42 -0700
Received:	from mail pickup service by hotmail.com with Microsoft SMTPSVC; Sat, 19 May 2007 16:52:41 -0700
Message-ID:	<BAY130-F26D9E35BF59E0D18A819AFB9310@phx.gbl>
Received:	from 65.55.135.123 by by130fd.bay130.hotmail.msn.com with HTTP; Sat, 19 May 2007 23:52:36 GMT
From:	"prithula dhungel" <prithuladhungel@hotmail.com>
To:	prithula@yahoo.com
Bcc:	
Subject:	Test mail
Date:	Sat, 19 May 2007 23:52:36 +0000
Mime-Version:	1.0
Content-Type:	Text/html; format=flowed
Return-Path:	prithuladhungel@hotmail.com

Figure: A sample mail message header

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

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

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

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

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

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

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

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

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

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

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

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

18. With download and delete, after a user retrieves its messages from a POP server, the messages are deleted. This poses a problem for the nomadic user, who may want to access the messages from many different machines (office PC, home PC, etc.). In the download and keep configuration, messages are not deleted after the user retrieves the messages. This can also be inconvenient, as each time the user retrieves the stored messages from a new machine, all of non-deleted messages will be transferred to the new machine (including very old messages).

19. Yes an organization's mail server and Web server can have the same alias for a host name. The MX record is used to map the mail server's host name to its IP address.

20. You should be able to see the sender's IP address for a user with an .edu email address. But you will not be able to see the sender's IP address if the user uses a gmail account.

21. It is not necessary that Bob will also provide chunks to Alice. Alice has to be in the top 4 neighbors of Bob for Bob to send out chunks to her; this might not occur even if Alice provides chunks to Bob throughout a 30-second interval.

22. Recall that in BitTorrent, a peer picks a random peer and optimistically unchokes the peer for a short period of time. Therefore, Alice will eventually be optimistically unchoked by one of her neighbors, during which time she will receive chunks from that neighbor.

23. The overlay network in a P2P file sharing system consists of the nodes   participating in the file sharing system and the logical links between the nodes. There is a logical link (an "edge"

in graph theory terms) from node A to node B if there is a semi-permanent TCP connection between A and B. An overlay network does not include routers.

24. Mesh DHT: The advantage is in order to a route a message to the peer (with ID) that is closest to the key, only one hop is required; the disadvantage is that each peer must track all other peers in the DHT. Circular DHT: the advantage is that each peer needs to track only a few other peers; the disadvantage is that O(N) hops are needed to route a message to the peer that is closest to the key.

25.
   a) File Distribution
   b) Instant Messaging
   c) Video Streaming
   d) Distributed Computing

26. With the UDP server, there is no welcoming socket, and all data from different clients enters the server through this one socket. With the TCP server, there is a welcoming socket, and each time a client initiates a connection to the server, a new socket is created. Thus, to support n simultaneous connections, the server would need *n+1* sockets.

27. For the TCP application, as soon as the client is executed, it attempts to initiate a TCP connection with the server. If the TCP server is not running, then the client will fail to make a connection. For the UDP application, the client does not initiate connections (or attempt to communicate with the UDP server) immediately upon execution

Chapter 2 Problems

Problem 1

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

Problem 2

Access control commands:
USER, PASS, ACT, CWD, CDUP, SMNT, REIN, QUIT.

Transfer parameter commands:
PORT, PASV, TYPE STRU, MODE.

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

Problem 3

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

Problem 4

a) The document request was http://gaia.cs.umass.edu/cs453/index.html. The Host : field indicates the server's name and /cs453/index.html indicates the file name.

b) The browser is running HTTP version 1.1, as indicated just before the first <cr><lf> pair.

c) The browser is requesting a persistent connection, as indicated by the Connection: keep-alive.

d) This is a trick question. This information is not contained in an HTTP message anywhere. So there is no way to tell this from looking at the exchange of HTTP messages alone. One would need information from the IP datagrams (that carried the TCP segment that carried the HTTP GET request) to answer this question.

e) Mozilla/5.0. The browser type information is needed by the server to send different versions

of the same object to different types of browsers.

Problem 5

a) The status code of 200 and the phrase OK indicate that the server was able to locate the document successfully. The reply was provided on Tuesday, 07 Mar 2008 12:39:45 Greenwich Mean Time.

b) The document index.html was last modified on Saturday 10 Dec 2005 18:27:46 GMT.

c) There are 3874 bytes in the document being returned.

d) The first five bytes of the returned document are : <!doc. The server agreed to a persistent connection, as indicated by the Connection: Keep-Alive field

Problem 6

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

Problem 7

The total amount of time to get the IP address is
$RTT_1 + RTT_2 + \cdots + RTT_n$.
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
$2RTT_o + RTT_1 + RTT_2 + \cdots + RTT_n$

Problem 8

a)

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

b)

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

c)

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

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:

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

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

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

Problem 10

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

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

$(200/150 + Tp + 200/150 + Tp + 200/150 + Tp + 100{,}000/150 + Tp)$
$+ (200/(150/10) + Tp + 200/(150/10) + Tp + 200/(150/10) + Tp + 100{,}000/(150/10) + Tp)$
$= 7377 + 8*Tp \text{ (seconds)}$

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

$(200/150+Tp + 200/150 +Tp + 200/150+Tp + 100,000/150+ Tp)$
$+ 10*(200/150+Tp + 100,000/150+ Tp)$
$=7351 + 24*Tp$ (seconds)

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

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

Problem 11
a) Yes, because Bob has more connections, he can get a larger share of the link bandwidth.
b) Yes, Bob still needs to perform parallel downloads; otherwise he will get less bandwidth than the other four users.

Problem 12

Server.py

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

Problem 13

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

Problem 14

SMTP uses a line containing only a period to mark the end of a message body.
HTTP uses "Content-Length header field" to indicate the length of a message body.

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

Problem 15

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

Problem 16

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

Problem 17

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

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

c) **C:** list
   S: 1 498
   S: 2 912
   S: .
   C: retr 1

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

Problem 18

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

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

c) *Local Domain: www.mindspring.com*
Web servers : www.mindspring.com
        207.69.189.21, 207.69.189.22,
        207.69.189.23, 207.69.189.24,
        207.69.189.25, 207.69.189.26, 207.69.189.27,
        207.69.189.28
Mail Servers : mx1.mindspring.com (207.69.189.217)
        mx2.mindspring.com (207.69.189.218)
        mx3.mindspring.com (207.69.189.219)
        mx4.mindspring.com (207.69.189.220)
Name Servers: itchy.earthlink.net (207.69.188.196)
        scratchy.earthlink.net (207.69.188.197)

www.yahoo.com
Web Servers: www.yahoo.com (216.109.112.135, 66.94.234.13)
Mail Servers: a.mx.mail.yahoo.com (209.191.118.103)
        b.mx.mail.yahoo.com (66.196.97.250)
        c.mx.mail.yahoo.com (68.142.237.182, 216.39.53.3)
        d.mx.mail.yahoo.com (216.39.53.2)
        e.mx.mail.yahoo.com (216.39.53.1)
        f.mx.mail.yahoo.com (209.191.88.247, 68.142.202.247)
        g.mx.mail.yahoo.com (209.191.88.239, 206.190.53.191)
Name Servers: ns1.yahoo.com (66.218.71.63)
        ns2.yahoo.com (68.142.255.16)
        ns3.yahoo.com (217.12.4.104)
        ns4.yahoo.com (68.142.196.63)
        ns5.yahoo.com (216.109.116.17)

ns8.yahoo.com (202.165.104.22)
ns9.yahoo.com (202.160.176.146)

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

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

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

d) The yahoo web server has multiple IP addresses
   www.yahoo.com (216.109.112.135, 66.94.234.13)

e) The address range for Polytechnic University: 128.238.0.0 – 128.238.255.255

f) An attacker can use the *whois* database and nslookup tool to determine the IP address ranges,
   DNS server addresses, etc., for the target institution.

g) By analyzing the source address of attack packets, the victim can use whois to obtain
   information about domain from which the attack is coming and possibly inform the
   administrators of the origin domain.

Problem 19

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

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

   ;; AUTHORITY SECTION:
   edu.   172800 IN   NS   E.GTLD-SERVERS.NET.
   edu.   172800 IN   NS   A.GTLD-SERVERS.NET.
   edu.   172800 IN   NS   G3.NSTLD.COM.
   edu.   172800 IN   NS   D.GTLD-SERVERS.NET.
   edu.   172800 IN   NS   H3.NSTLD.COM.

edu.	172800 IN	NS	L3.NSTLD.COM.
edu.	172800 IN	NS	M3.NSTLD.COM.
edu.	172800 IN	NS	C.GTLD-SERVERS.NET.

Among all returned edu DNS servers, we send a query to the first one.
dig +norecurse @E.GTLD-SERVERS.NET any gaia.cs.umass.edu

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

Among all three returned authoritative DNS servers, we send a query to the first one.
dig +norecurse @ns1.umass.edu any gaia.cs.umass.edu

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

b) The answer for google.com could be:
    a.root-servers.net
    E.GTLD-SERVERS.NET
    ns1.google.com(authoritative)

Problem 20

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

For a complete measurement study, see:
Craig E. Wills, Mikhail Mikhailov, Hao Shang
"Inferring Relative Popularity of Internet Applications by Actively Querying DNS Caches", in IMC'03, October 27-29, 2003, Miami Beach, Florida, USA

Problem 21

Yes, we can use dig to query that Web site in the local DNS server.
For example, "dig cnn.com" will return the query time for finding cnn.com. If cnn.com was just accessed a couple of seconds ago, an entry for cnn.com is cached in the local DNS cache, so the query time is 0 msec. Otherwise, the query time is large.

Problem 22

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

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

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

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

Where, $F = 15$ Gbits $= 15 * 1024$ Mbits
  $u_s = 30$ Mbps
  $d_{min} = d_i = 2$ Mbps

Note, 300Kbps = 300/1024 Mbps.

Client Server

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

Peer to Peer

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

Problem 23

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

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

c) From Section 2.6 we know that

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

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

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

We can similarly show that:

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

Combining Equation 2 and Equation 3 gives the desired result.

Problem 24

a) Define u = u1 + u2 + ... + uN. By assumption

$u_s <= (u_s + u)/N$ Equation 1

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

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

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

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

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

Thus each peer receives the file in F/u_s.

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

$u_s >= (u_s + u)/N$ Equation 2

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

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

Problem 27

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

Problem 28

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

Problem 29

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

Problem 30

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

Such mismatches may degrade the search performance. For example, consider a logical path p1 (consisting of only two logical links): A→B→C, where A and B are neighboring peers, and B and C are neighboring peers. Suppose that there is another logical path p2 from A to C (consisting of 3 logical links): A→D→E→C.

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

Problem 31

a) If you run TCPClient first, then the client will attempt to make a TCP connection with a non-existent server process. A TCP connection will not be made.

b) UDPClient doesn't establish a TCP connection with the server. Thus, everything should work fine if you first run UDPClient, then run UDPServer, and then type some input into the keyboard.

c) If you use different port numbers, then the client will attempt to establish a TCP connection with the wrong process or a non-existent process. Errors will occur.

Problem 32

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

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

Before:

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

After:

Client socket = 5432

Problem 33

Yes, you can configure many browsers to open multiple simultaneous connections to a Web site. The advantage is that you will you potentially download the file faster. The disadvantage is that you may be hogging the bandwidth, thereby significantly slowing down the downloads of other users who are sharing the same physical links.

Problem 34

For an application such as remote login (telnet and ssh), a byte-stream oriented protocol is very natural since there is no notion of message boundaries in the application. When a user types a character, we simply drop the character into the TCP connection.
In other applications, we may be sending a series of messages that have inherent boundaries between them. For example, when one SMTP mail server sends another SMTP mail server several email messages back to back. Since TCP does not have a mechanism to indicate the boundaries, the application must add the indications itself, so that receiving side of the application can distinguish one message from the next. If each message were instead put into a

distinct UDP segment, the receiving end would be able to distinguish the various messages without any indications added by the sending side of the application.

Problem 35

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

Problem 36

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