CHAPTER 26

PRACTICE SET

Questions

Q26-1. Probably Alice turned off her desktop, which stopped the FTP server, when she left the office. A server process should be running all the time, waiting for clients to access it.

Q26-2. In FTP, the control connection is actively opened by the client. The server is running all the time waiting for a client to make a control connection. The data-transfer connection, on the other hand, is actively opened by the server. The client issues a passive open and sends the ephemeral port (to be used for the data-transfer connection) to the server, which is done using a control-connection command. The server now issues an active open using the ephemeral port received from the client.

Q26-3. The answer is no. Data-transfer activity is controlled by the client. If a file needs to be transferred from the client site to the server site, the client needs to store it at the server site.

Q26-4. The file needs to be transferred as an *image file* (binary), which means the downloaded file should be the bit-by-bit copy of the audio file on the server.

Q26-5. One example can be the use of a *remote control* when we are watching TV. The remote control establishes a control connection with the TV set, which is separate from one-way multimedia connection.

Q26-6. The answer is no. Only the client can get the list of files from the server. In the client-server paradigm, the request is from the client; the server only can respond.

Q26-7. The file (or list of directories or files, which is consider as a file) is transmitted using stream mode (default), block mode, or compressed mode. The mode is determined by the client using the MODE command. In the stream mode, no header is added to the file bytes; in the other two modes, a header is added to the beginning of the file to show information about blocking or compressing.

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

Q26-8. We can have a control connection without a data-transfer connection in FTP. As a matter of fact, some tasks in FTP can be done without a data-transfer connection. For example, if a client needs to rename a file at the server site, there is no need for a data-transfer connection.

Q26-9. In an HTTP request message, a blank line signals the end of the headers and the beginning of the body of the message, if any.

Q26-10. Mail servers normally allocate a limited amount of storage for each mail-box. If the e-mails or attachments are not retrieved, the mail-box may become full and some old e-mails may be discarded.

Q26-11. The client changes the ASCII character to NVT characters. The server changes NVT characters to EBCDIC.

Q26-12. FTP has a specific format for exchanging commands and responses during the control connection. We can say that these formats can be defined as shown below:

Command:	**<command> [<space> <parameter> …]**
Response:	**<code> [<space> <text>] <CRLF>**

Q26-13. It depends on how the file is stored on the server. If the file is stored as a web page, embedded in an HTML document, we need to use HTTP to download the file. On the other hand, if the file is stored on the server without having been embedded in an HTML document, then we need to download it using FTP.

Q26-14. If Bob posts his clip on his website, Alice can get it by running an HTTP client (a browser) using a GET message. Since Alice is not running an HTTP server, she needs to use the PUT command and post her clip on Bob's site.

Q26-15. If the control connection is severed during a session, no more control information can be exchanged between the client control process and the server control process. Since the control connection and the data connection are two separate TCP connections, the severance of the control connection has no effect on the current data connection. However, when the current data connection is terminated, no new data connection can be made until the control connection is again established.

Q26-16. The task can be done using only one control connection, but two data-transfer connections are needed, one for retrieving and one for storing. Although the data-transfer connection is a two-way connection, one is used for data transfer, the other for acknowledging.

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

Q26-17. FTP definitely cannot use the services of UDP for control connection because the client and the server need to be connected during the whole session. Since UDP is not a connection-oriented protocol, it cannot do this task. FTP does not use the services of UDP during data transmission for another reason. A file to be transferred may be too large to fit in a single user datagram (UDP packet). In addition, UDP is not a reliable transport-layer protocol; for file transfer, reliability is an important issue. TCP is more appropriate for this purpose.

Q26-18. The HELO command is needed so that the client can identify itself using its domain name during connection establishment. The MAIL FROM message is needed to supply the server with a return mail address for returning errors and reporting messages.

Q26-19. The MAIL FROM in the envelope contains the source e-mail address while the FROM in the header contains the name of the sender.

Q26-20. The answer is yes. Two separate connections are needed for setup and teardown to use FTP.

Q26-21. The answer is no. We cannot have a data-transfer connection without a control connection. The data-transfer connection is established with one or more commands issued from a client; there should be a control connection to allow the client to issue these commands.

Q26-22.

a. PQDN (It does not end with a dot.)

b. PQDN (It does not end with a dot.)

c. FQDN (It does end with a dot.)

Q26-23. FTP allows the client to define the file format it can receive or send; the server needs to use the format acceptable to the client when sending a file and to accept the format used by the client when receiving a file.

Q26-24. TELNET allows a host to log into a remote computer that can offer application programs. After the user logs in, she can use any services provided by the remote computer. For example, the user can create a program in any computer language supported by the remote computer, compile it, run it, and see the result.

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

Q26-25. When a host logs into a remote computer, it can use all services available on that host. Since an FTP or an HTTP server provides services, the user can use these services. As a matter of fact, the TELNET interface allows the user to define the port number of the server she wants to use.

Q26-26. If the message has no body section, the lack of any character after the blank line is an indication that the message is over. If the message has a body, either the body needs to have an end-of-file marker or the message should have the Content-Length header to define the size of the body.

Problems

P26-1. HTTP provides some presentation features, using the request and response headers. For example, HTTP messages can define the format and the language of the messages exchanged, which is a kind of presentation.

P26-2. To store a video clip, we choose the organization to be *page* and the file type to be *image* (binary). The following shows the transaction:

Server:	220 (Service ready)
Client:	**USER John**
Server:	331 (User name OK. Password?)
Client:	**PASS xxxx**
Server:	220 (User login OK)
Client:	**PORT 56002**
Server:	150 (Data connection will open shortly)
Client:	**STRU P**
Server:	200 (OK)
Client:	**TYPE I**
Server:	200 (OK)
Client:	**STOR /top/videos/general/video2**
Server:	125 (Data connection OK)
Transferring video files from the client to the server	
Server:	226 (Closing data connection)
Client:	**Quit**
Server:	221 (Server closing)

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-3. The following shows an example. We have shown only part of the transaction (when the client starts retrieving the message):

```
Client: RETR 1
Server: +OK 230 octets
Server: …                                   // Message contents
Server: .                                   // Dot means end of mail
Client: DELE 1
Server: +OK message 1 deleted
Client: RETR 2
Server: OK 400 octets
Server: …                                   // Message contents
Server: .                                   // Dot means end of mail
Client: DELE 2
Server: +OK message 2 deleted
```

P26-4. Since there are several programs to do the job, we post on the book web site. See extra materials for Chapter 26.

P26-5. The following gives the meaning and usage of each command:

a. The UIDL (unique ID listing) is used by the client to provide a unique identifier for the message (if used with an argument) or for all messages (if no argument is used).

b. The TOP command, always with two arguments, is used by the client to ask the server to return the top lines of a particular message. In this case, it means to return the top 15 lines of message 1.

c. The USER command is used by the client to define its user identification.

d. The PASS command is used by the client to define its password.

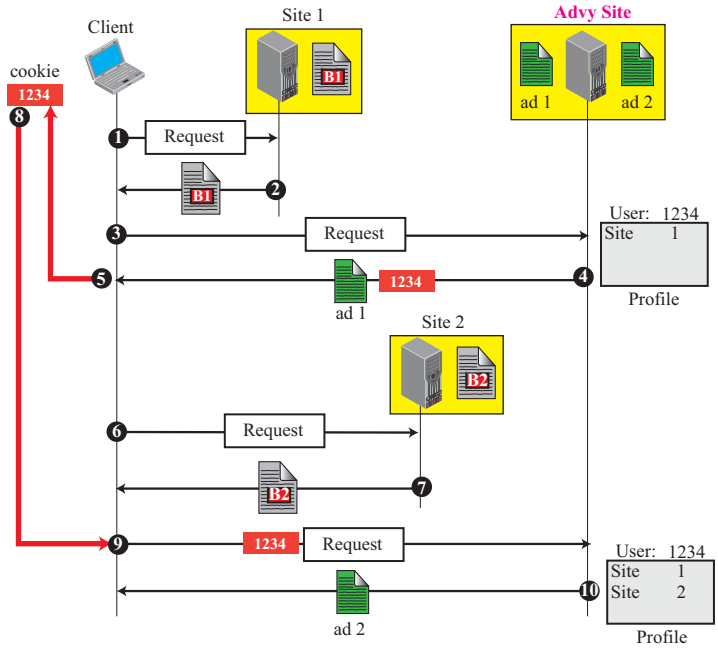
P26-6. When HTTP operates in persistent-connection mode, it is actually implementing a session that can last for a period of time. This is simulating the session layer defined in the OSI model.

P26-7.

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

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-8. The following shows a simple example with only three sites.



The first two sites, site 1 and site 2, are popular sites that people visit. The third site, Advy site, is the site of an advertising company. The advertising company pays some fees to the other two sites to include advertisement banners, B1 and B2, related to the corporations advertised by Advy. The banners, however, are not actual advertisement images; they include URLs that refer to the Advy site. When the user (client in the figure) visits one of these sites, the banner is sent to the user. The user's browser accesses the Advy site to retrieve the actual advertisement. At this time the Advy server sends a cookie with an identification number (1234 in the figure) to the user's browser and, at the same time, creates a user profile showing which sites the user has visited. The advertisement company can later sell this profile to other organizations for profit.

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-9. The following shows the commands and responses. The file mode in this case needs to be C (compressed).

Server:	220 (Service ready)
Client:	**USER Jane**
Server:	331 (User name OK. Password?)
Client:	**PASS xxxx**
Server:	220 (User login OK)
Client:	**PORT 61017**
Server:	150 (Data connection will open shortly)
Client:	**STRU F**
Server:	200 (OK)
Client:	**TYPE E**
Server:	200 (OK)
Client:	**MODE C**
Server:	200 (OK)
Client:	**RETR /usr/users/report/huge**
Server:	125 (Data connection OK)
Transferring the file from the server to the client	
Server:	226 (Closing data connection)
Client:	**Quit**
Server:	221 (Server closing)

P26-10. SMTP provides some presentation features by allowing us to define the contents of the e-mail (using MIME).

P26-11. The result is "Vw/w" in ASCII as shown below:

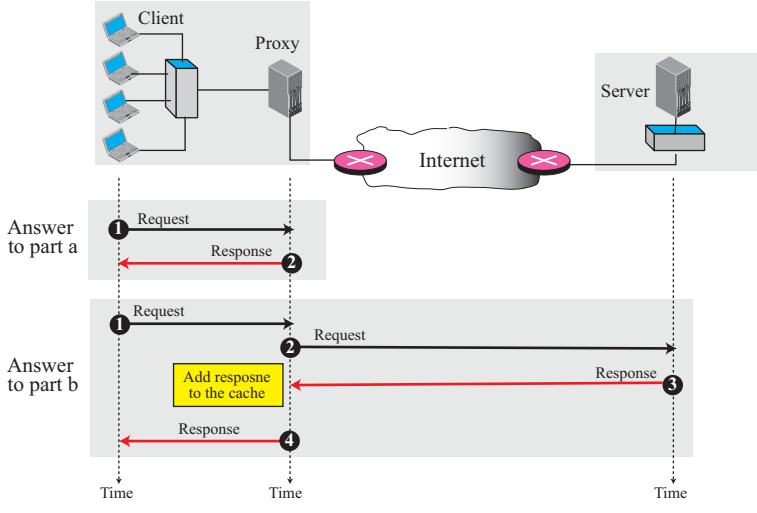
Original:	01010111 00001111 11110000			
Grouped by six:	010101	110000	111111	110000
Base64:	21	48	63	48
ASCII:	V	w	/	w

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-12. The following shows the set of commands and responses. Note that there is no data transfer connection in this; we have only a control connection.

Server:	220 (Service ready)
Client:	**USER Jan**
Server:	331 (User name OK. Password?)
Client:	**PASS xxxx**
Server:	220 (User login OK)
Client:	**MKD /user/users/letters/Jan**
Server:	200 (Command OK)
Client:	**Quit**
Server:	221 (Server closing)

P26-13. The following shows a simple example. In part a, the request can be responded to by the proxy server. In part b, the proxy needs to send the request to the true server. When the response is received, the proxy server saves it in the cache for future use, and then sends it to the client.



© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-14. The following shows an example. We have shown only part of the transaction (when the client starts retrieving the message):

```
Client: RETR 1
Server: +OK 192 octets
Server: …                                   // Message contents
Server: .                                   // Dot means end of mail
Client: RETR 2
Server: +OK 300 octets
Server: …                                   // Message contents
Server: .                                   // Dot means end of mail
```

P26-15. The following shows a possible request and response:

a. A possible request

```
GET /usr/users/doc HTTP /1.1
Date: Fri, 26-Nov-04 16:46:23 GMT
MIME-version: 1.0
Accept: image/gif
Accept: image/jpeg
Last modified: Mon, 22-Nov-04
```

b. A possible response

```
HTTP/1.1 200 OK
Date: Fri, 26-Nov-04 16:46:26 GMT
Server: Challenger
MIME-version: 1.0
Content-length: 4623
(Body of document)
```

P26-16. The following gives the meaning and usage of each command:

a. The STAT command, which takes no argument, asks for the status of the mailbox. The server normally responds with the number of messages and the total size in bytes.

b. The LIST command, which can have no argument or one argument, asks information about all messages or one particular message. If it is used with no argument, it asks information about all messages. The server gives one response for each message (the message number and size). If it is used with one argument, like LIST 3, it means that the server needs to give information about a particular message

c. The DELE command, which always needs an argument, is used to delete a message at the server site. In this case, the client is asking to delete message 4 at the server.

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-17. SMTP does not create a session between the client and sender in which the client can send some e-mails in a single session. The e-mails need to be sent one by one. However, POP3 allows the user to retrieve all e-mails received in the mailbox in one session.

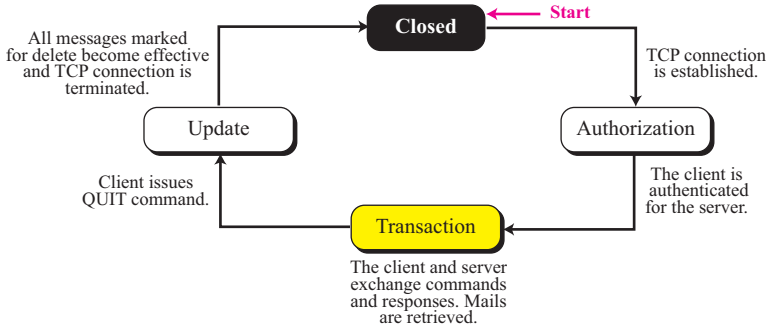
P26-18.

a. The text is $1000 \times 8 = 8000$ bits. This results in $8000/6 = 1333.33$ or 1334 characters. We have $1334 - 1000 = 334$ redundant bytes. The ratio of redundant bytes to the entire message length is $334/1334 \approx 25\%$.

b. In this case, $900 + (100 \times 3) = 1200$ bytes are in the encoded message. There are 200 redundant bytes. The ratio of redundant bytes to the entire message length is $200/1200 \approx 16.7\%$.

c. The efficiency in part *a* is $(1000)/(1334) = 75\%$. The efficiency in part *b* is $(1000/1200) = 83\%$. The efficiency is improved 8%.

P26-19. The following shows the four states and the transitions from one state to another.



P26-20. The client or server can use one of the general headers called *Connection,* and set the *connection-token* = *close* to define a nonpersistent connection.

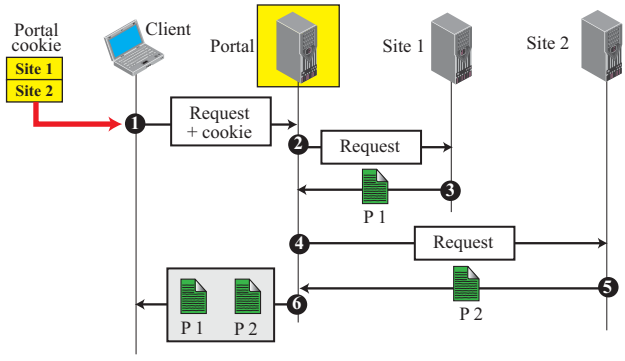
P26-21. Since there are several programs to do the job, we post on the book web site. See extra materials for Chapter 26.

P26-22. The result is "O=AFQ" in ASCII as shown below

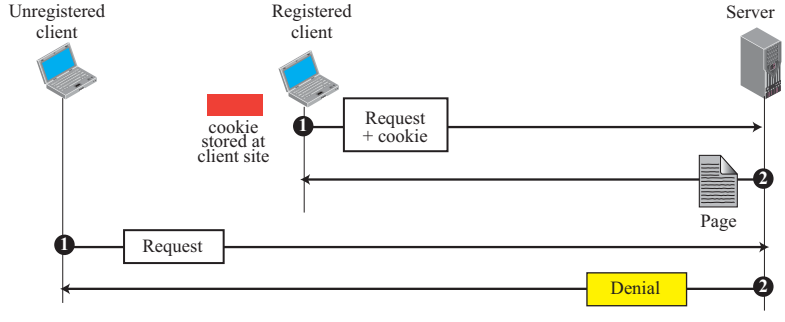
Original:	010011111010111101110001		
Grouped by eight:	01001111	10101111	01110001
ASCII:	O	=AF	Q

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-23. The following shows a simple situation. A *portal* is a special site that holds the often-visited URLs for each client. The cookie stored in the browser under the name of the portal holds the list of the sites the user normally needs to check periodically. When the user clicks on the portal web page, a request is sent with the cookie to the portal site with the list of desired web pages. The portal then fetches the current pages from the corresponding site, compiles a page, and sends it to the browser.



P26-24. The following shows a simple situation. When a client registers, a cookie is sent to the browser and stored there. Next time, when the client wants to visit the same site, the browser checks the cookie list and adds the corresponding cookie to the request. The server recognizes the cookie, checks the registration, and sends the page. An unregistered client does not have the cookie related to the site; the access to the web page is denied.



© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.

P26-25. The following shows the set of commands and responses. Note that there is no data transfer connection in this; we have only a control connection. We need to give the name of the file to be renamed (/usr/users/report/file1) and then the new name (/usr/top/letters/file1).

Server:	220 (Service ready)
Client:	**USER Maria**
Server:	331 (User name OK. Password?)
Client:	**PASS xxxx**
Server:	220 (User login OK)
Client:	**RNFR** */usr/users/report/file1*
Server:	200 (Command OK)
Client:	**RNTO** */usr/top/letters/file1*
Server:	200 (Command OK)
Client:	**Quit**
Server:	221 (Server closing)

© 2012 by McGraw-Hill Education. This is proprietary material solely for authorized instructor use. Not authorized for sale or distribution in any manner. This document may not be copied, scanned, duplicated, forwarded, distributed, or posted on a website, in whole or part.