
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.
- Q26-3.** 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.
- Q26-5.** 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.
- Q26-7.** 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.
- Q26-9.** 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.
- Q26-11.** 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.
- Q26-13.** 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.

- Q26-15.** 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.
- Q26-17.** 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.
- Q26-19.** 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.
- Q26-21.** The MAIL FROM in the envelope contains the source e-mail address while the FROM in the header contains the name of the sender.
- Q26-23.** The client changes the ASCII character to NVT characters. The server changes NVT characters to EBCDIC.
- 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.

Problems

P26-1. The following shows a possible request and response:

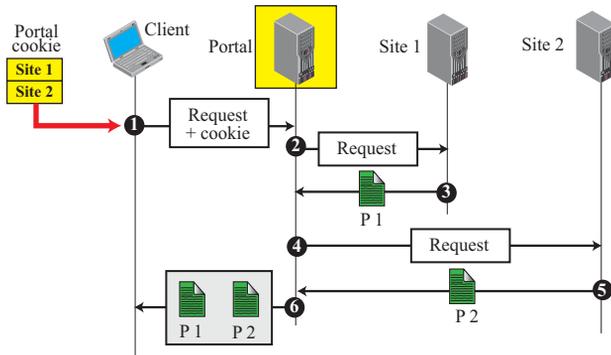
a. A possible request

```
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
```

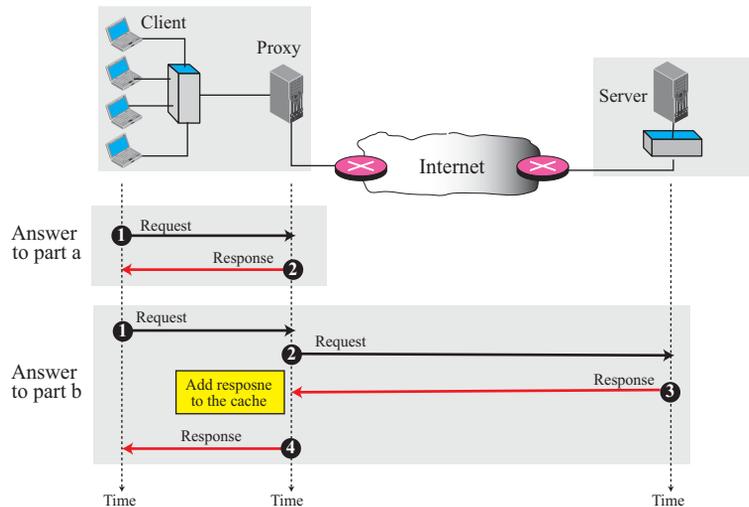
b. A possible response

```
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)
```

- P26-3.** 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.



- P26-5.** 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.



P26-7. 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.

P26-9.

MIME-version: 1.1

Content-Type: Text/Plain

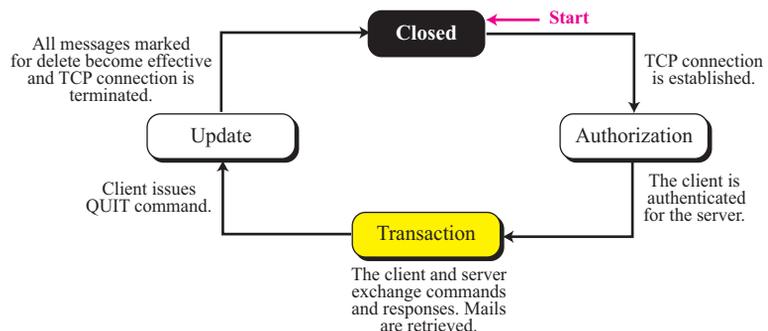
Content-Transfer-Encoding: 7bit

P26-11. Since there are several programs to do the job, we post on the book web site. See extra materials for Chapter 26.

P26-13. 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

P26-15. The following shows the four states and the transitions from one state to another.



P26-17. 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.

P26-19. 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
```

P26-21. 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.

P26-23. 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)

P26-25. 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)