

Chapter 26

Remote Logging, Electronic Mail, and File Transfer

26-1 REMOTE LOGGING

It would be impossible to write a specific client/server program for each demand. The better solution is a general-purpose client/server program that lets a user access any application program on a remote computer.

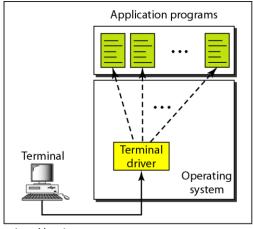
Topics discussed in this section:
TELNET



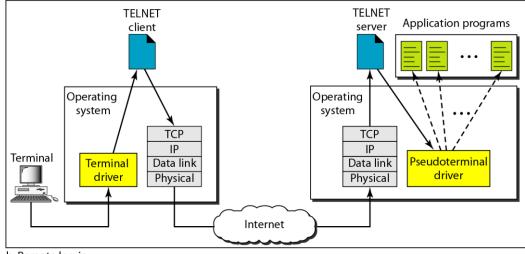
Note

TELNET is a general-purpose client/server application program.

Figure 26.1 Local and remote log-in



a. Local log-in



b. Remote log-in

Figure 26.2 Concept of NVT

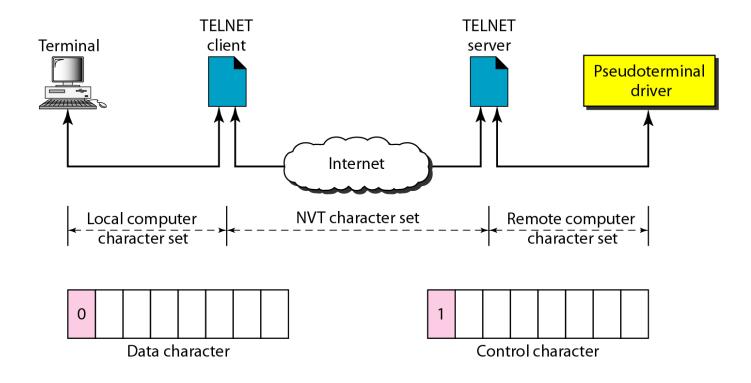


Table 26.1 Some NVT control characters

Character	Decimal	Binary	Meaning
EOF	236	11101100	End of file
EOR	239	11101111	End of record
SE	240	11110000	Suboption end
NOP	241	11110001	No operation
DM	242	11110010	Data mark
BRK	243	11110011	Break
IP	244	11110100	Interrupt process
AO	245	11110101	Abort output
AYT	246	11110110	Are you there?
EC	247	11110111	Erase character
EL	248	11111000	Erase line
GA	249	11111001	Go ahead
SB	250	11111010	Suboption begin
WILL	251	11111011	Agreement to enable option
WONT	252	111111100	Refusal to enable option
DO	253	11111101	Approval to option request
DONT	254	11111110	Denial of option request
IAC	255	11111111	Interpret (the next character) as control

Figure 26.3 An example of embedding



Typed at the remote terminal

Table 26.2 Options

Code	Option	Meaning
0	Binary	Interpret as 8-bit binary transmission.
1	Echo	Echo the data received on one side to the other.
3	Suppress go ahead	Suppress go-ahead signals after data.
5	Status	Request the status of TELNET.
6	Timing mark	Define the timing marks.
24	Terminal type	Set the terminal type.
32	Terminal speed	Set the terminal speed.
34	Line mode	Change to line mode.

Table 26.3 NVT character set for option negotiation

Character	Decimal	Binary	Meaning
WILL	251	11111011	1. Offering to enable
			2. Accepting a request to enable
WONT	252	11111100	1. Rejecting a request to enable
			2. Offering to disable
			3. Accepting a request to disable
DO	253	11111101	1. Approving an offer to enable
			2. Requesting to enable
DONT	254	11111110	1. Disapproving an offer to enable
			2. Approving an offer to disable
			3. Requesting to disable

Example 26.1

Figure 26.4 shows an example of option negotiation. In this example, the client wants the server to echo each character sent to the server. The echo option is enabled by the server because it is the server that sends the characters back to the user terminal. Therefore, the client should request from the server the enabling of the option using DO. The request consists of three characters: IAC, DO, and ECHO. The server accepts the request and enables the option. It informs the client by sending the three-character approval: IAC, WILL, and ECHO.

Figure 26.4 Example 26.1: Echo option

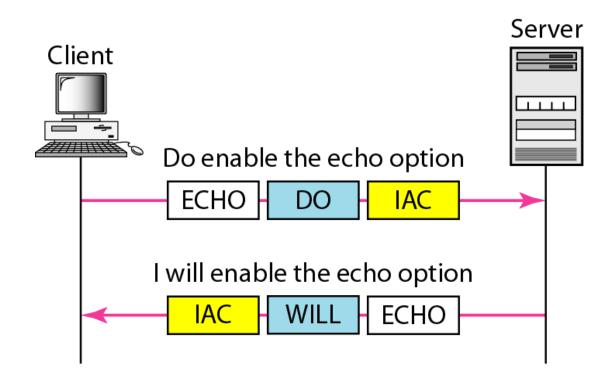


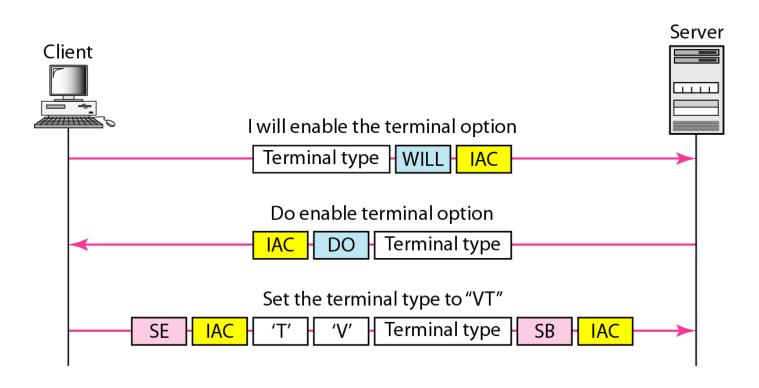
Table 26.4 Character set for suboptions

Character	Decimal	Binary	Meaning
SE	240	11110000	Suboption end
SB	250	11111010	Suboption begin

Example 26.2

Figure 26.5 shows an example of suboption negotiation. In this example, the client wants to negotiate the type of the terminal.

Figure 26.5 Example of suboption negotiation



26-2 ELECTRONIC MAIL

One of the most popular Internet services is electronic mail (e-mail). The designers of the Internet probably never imagined the popularity of this application program. Its architecture consists of several components that we discuss in this chapter.

Topics discussed in this section:

Architecture

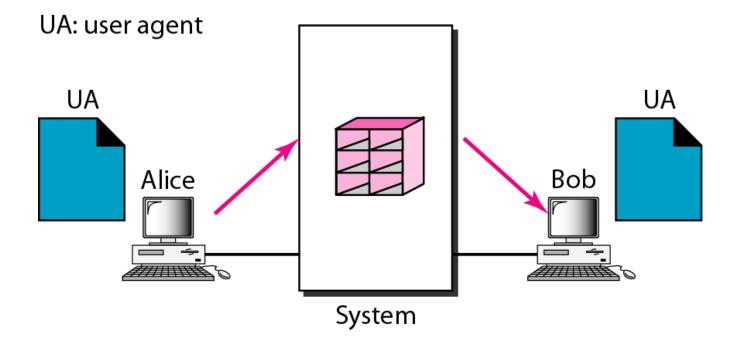
User Agent

Message Transfer Agent: SMTP

Message Access Agent: POP and IMAP

Web-Based Mail

Figure 26.6 First scenario in electronic mail



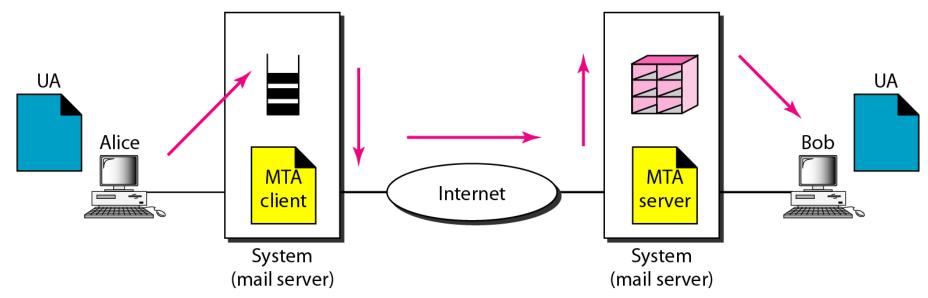
Note

When the sender and the receiver of an e-mail are on the same system, we need only two user agents.

Figure 26.7 Second scenario in electronic mail

UA: user agent

MTA: message transfer agent

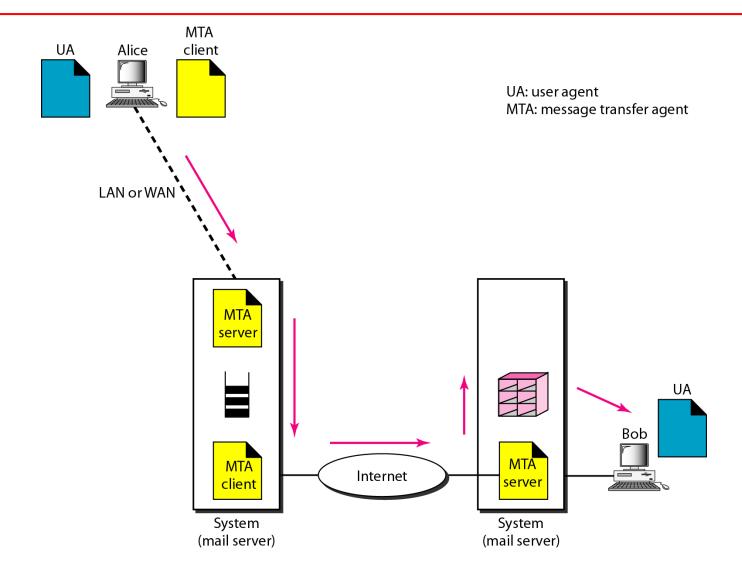


-

Note

When the sender and the receiver of an e-mail are on different systems, we need two UAs and a pair of MTAs (client and server).

Figure 26.8 Third scenario in electronic mail



-

Note

When the sender is connected to the mail server via a LAN or a WAN, we need two UAs and two pairs of MTAs (client and server).

Figure 26.9 Fourth scenario in electronic mail

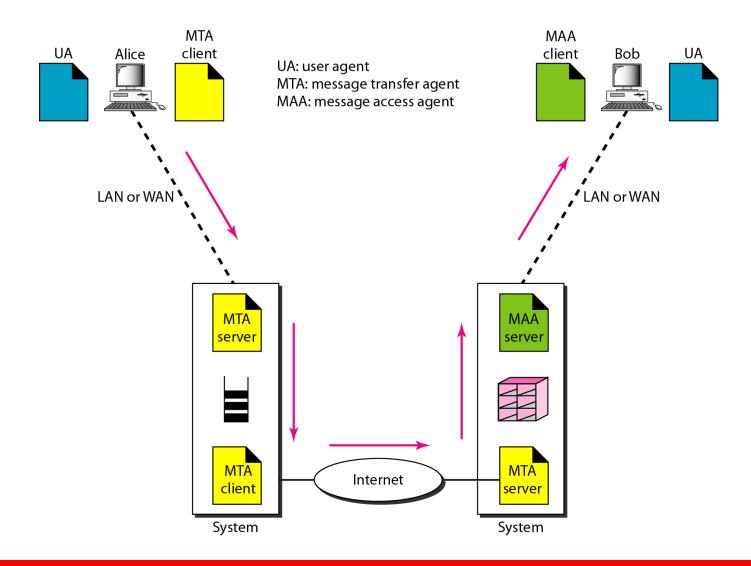
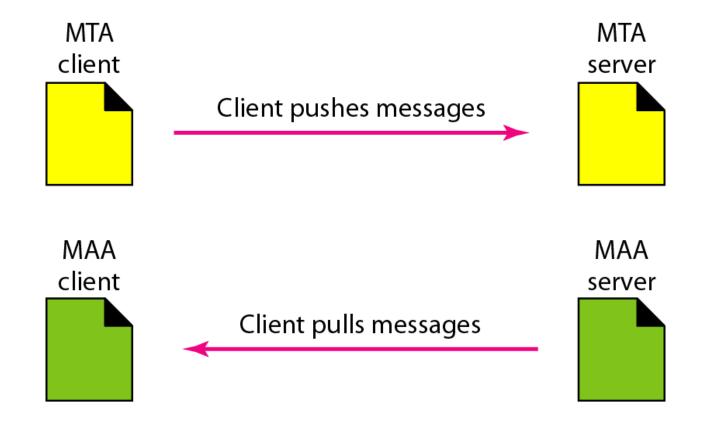


Figure 26.10 Push versus pull in electronic email



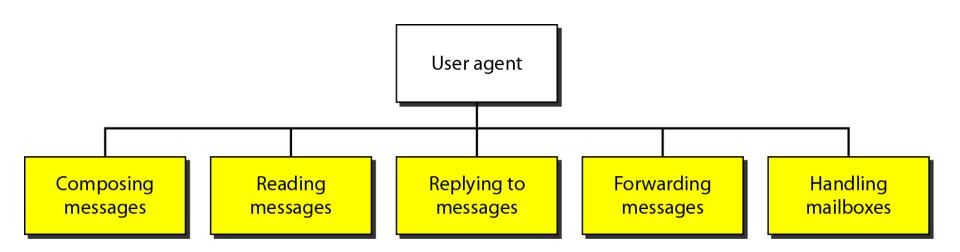
-

Note

When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs and a pair of MAAs.

This is the most common situation today.

Figure 26.11 Services of user agent



-

Note

Some examples of command-driven user agents are *mail*, *pine*, and *elm*.

-

Note

Some examples of GUI-based user agents are *Eudora*, *Outlook*, and *Netscape*.

Figure 26.12 Format of an e-mail

Behrouz Forouzan De Anza College Cupertino, CA 96014

> Sophia Fegan Com-Net Cupertino, CA 95014

Sophia Fegan Com-Net Cupertino, CA 95014 Jan. 5, 2005

Subject: Network

Dear Ms. Fegan: We want to inform you that our network is working properly after the last repair.

Yours truly, Behrouz Forouzan

a. Postal mail

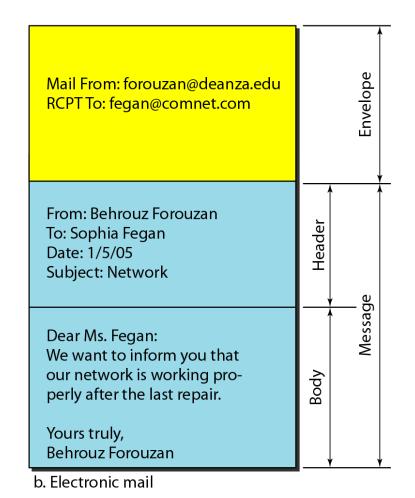


Figure 26.13 E-mail address

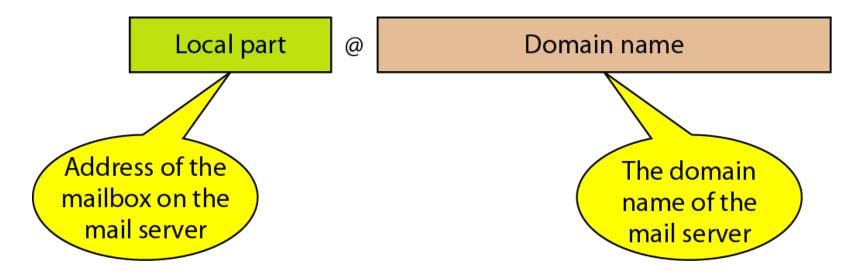


Figure 26.14 MIME

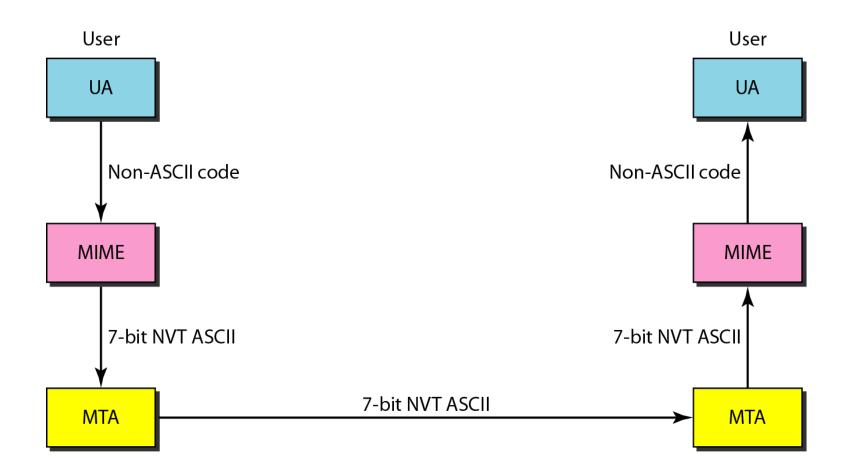


Figure 26.15 MIME header

E-mail header

MIME-Version: 1.1

Content-Type: type/subtype

Content-Transfer-Encoding: encoding type

Content-Id: message id

Content-Description: textual explanation of nontextual contents

E-mail body

MIME headers

Table 26.5 Data types and subtypes in MIME

Туре	Subtype	Description	
Text	Plain	Unformatted	
TOAC	HTML	HTML format (see Chapter 27)	
	Mixed	Body contains ordered parts of different data types	
Multipart	Parallel	Same as above, but no order	
	Digest	Similar to mixed subtypes, but the default is message/ RFC822	
	Alternative	Parts are different versions of the same message	
	RFC822	Body is an encapsulated message	
Message	Partial	Body is a fragment of a bigger message	
	External-Body	Body is a reference to another message	
Image	JPEG	Image is in JPEG format	
	GIF	Image is in GIF format	
Video	MPEG	Video is in MPEG format	
Audio	Basic	Single-channel encoding of voice at 8 kHz	
Application	PostScript	Adobe PostScript	
	Octet-stream	General binary data (8-bit bytes)	

Table 26.6 Content-transfer-encoding

Туре	Description
7-bit	NVT ASCII characters and short lines
8-bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base-64	6-bit blocks of data encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters encoded as an equals sign followed by an ASCII code

Figure 26.16 SMTP range

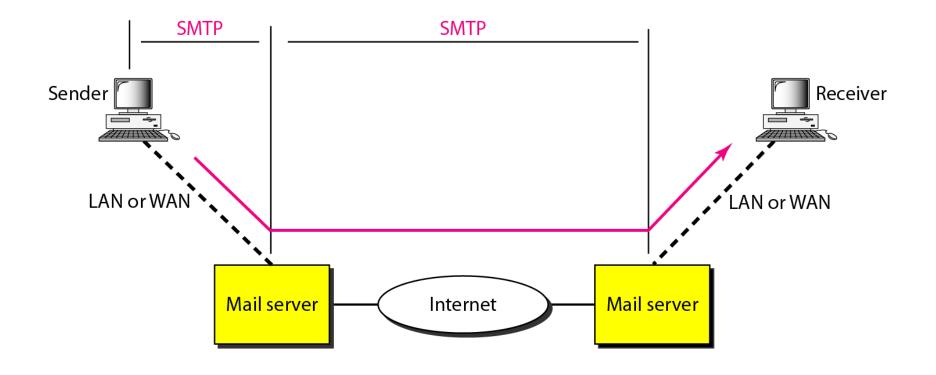


Figure 26.17 Commands and responses

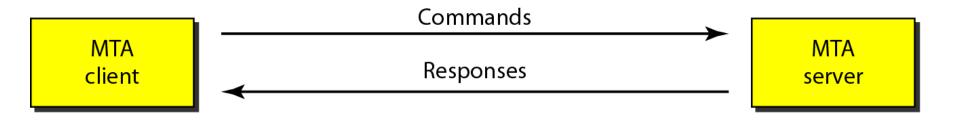


Figure 26.18 Command format

Keyword: argument(s)

Table 26.7 Commands

Keyword	Argument(s)	
HELO	Sender's host name	
MAIL FROM	Sender of the message	
RCPT TO	Intended recipient of the message	
DATA	Body of the mail	
QUIT		
RSET		
VRFY	Name of recipient to be verified	
NOOP		
TURN		
EXPN	Mailing list to be expanded	
HELP	Command name	
SEND FROM	Intended recipient of the message	
SMOL FROM	Intended recipient of the message	
SMAL FROM	Intended recipient of the message	

Table 26.8 Responses

Code	Description			
Positive Completion Reply				
211	System status or help reply			
214	Help message			
220	Service ready			
221	Service closing transmission channel			
250	Request command completed			
251	User not local; the message will be forwarded			
Positive Intermediate Reply				
354	Start mail input			
Transient Negative Completion Reply				
421	Service not available			
450	Mailbox not available			
451	Command aborted: local error			
452	Command aborted: insufficient storage			

Table 26.8 Responses (continued)

Code	Description			
Permanent Negative Completion Reply				
500	Syntax error; unrecognized command			
501	Syntax error in parameters or arguments			
502	Command not implemented			
503	Bad sequence of commands			
504	Command temporarily not implemented			
550	Command is not executed; mailbox unavailable			
551	User not local			
552	Requested action aborted; exceeded storage location			
553	Requested action not taken; mailbox name not allowed			
554	Transaction failed			

Example 26.3

Let us see how we can directly use SMTP to send an e-mail and simulate the commands and responses we described in this section. We use TELNET to log into port 25 (the well-known port for SMTP). We then use the commands directly to send an e-mail. In this example, forouzanb@adelphia.net is sending an e-mail to himself. The first few lines show TELNET trying to connect to the Adelphia mail server. After connection, we can type the SMTP commands and then receive the responses, as shown on the next slide. Note that we have added, for clarification, some comment lines, designated by the "=" signs. These lines are not part of the e-mail procedure.

Example 26.3 (continued)

```
$ telnet mail.adelphia.net 25
Trying 68.168.78.100 . . .
Connected to mail.adelphia.net (68.168.78.100).
```

```
220 mta13.adelphia.net SMTP server ready Fri, 6 Aug 2004 . . .

HELO mail.adelphia.net
250 mta13.adelphia.net
```

Example 26.3 (continued)

```
Mail Transfer
MAIL FROM: forouzanb@adelphia.net
 250 Sender <forouzanb@adelphia.net> Ok
RCPT TO: forouzanb@adelphia.net
 250 Recipient <forouzanb@adelphia.net> Ok
DATA
 354 Ok Send data ending with <CRLF>.<CRLF>
From: Forouzan
TO: Forouzan
This is a test message
to show SMTP in action.
```

Example 26.3 (continued)

250 Message received: adelphia.net@mail.adelphia.net
QUIT
221 mta13.adelphia.net SMTP server closing connection
Connection closed by foreign host.

Figure 26.19 POP3 and IMAP4

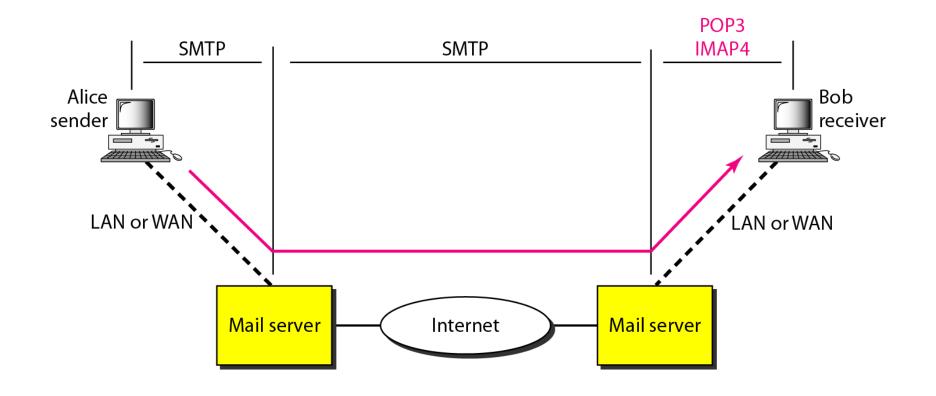
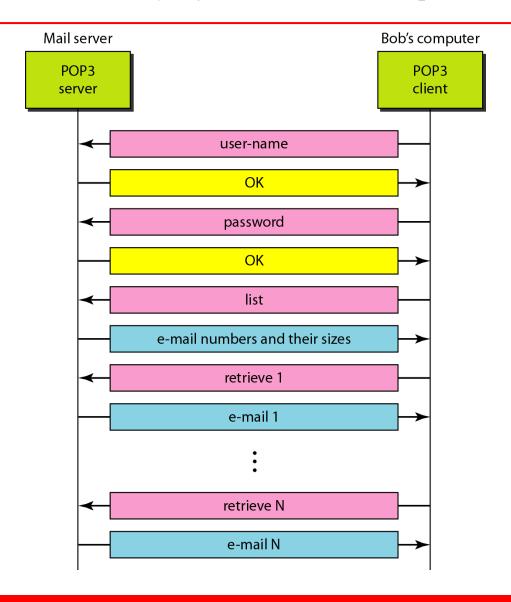


Figure 26.20 The exchange of commands and responses in POP3



26-3 FILE TRANSFER

Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment. As a matter of fact, the greatest volume of data exchange in the Internet today is due to file transfer.

Topics discussed in this section:

File Transfer Protocol (FTP) Anonymous FTP



FTP uses the services of TCP. It needs two TCP connections.

The well-known port 21 is used for the control connection and the well-known port 20 for the data connection.

Figure 26.21 *FTP*

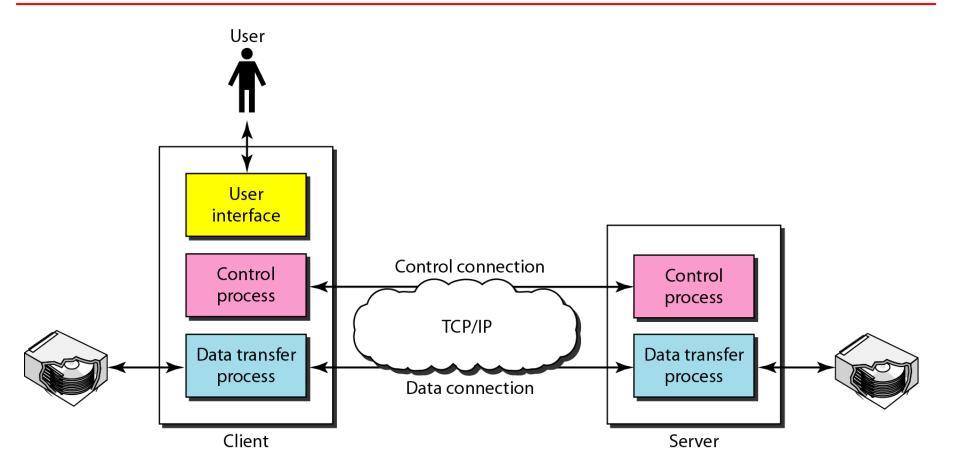


Figure 26.22 Using the control connection

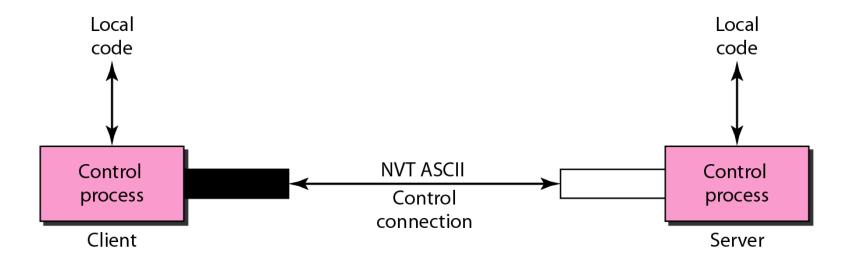
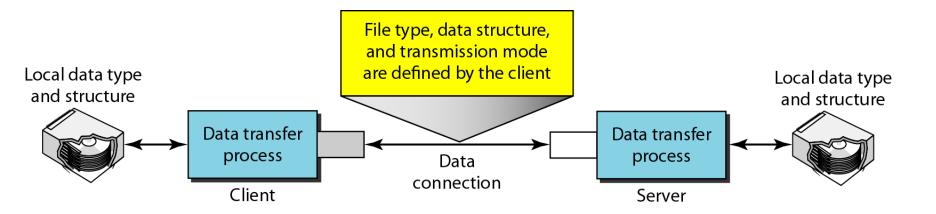


Figure 26.23 Using the data connection



Example 26.4

The following shows an actual FTP session for retrieving a list of items in a directory. The colored lines show the responses from the server control connection; the black lines show the commands sent by the client. The lines in white with a black background show data transfer.

- 1. After the control connection is created, the FTP server sends the 220 response.
- 2. The client sends its name.
- 3. The server responds with 331.

Example 26.4 (continued)

- 4. The client sends the password (not shown).
- 5. The server responds with 230 (user log-in is OK).
- 6. The client sends the list command (ls reports) to find the list of files on the directory named report.
- 7. Now the server responds with 150 and opens the data connection.
- 8. The server then sends the list of the files or directories on the data connection.
- 9. The client sends a QUIT command.
- 10. The server responds with 221.

Example 26.4 (continued)

\$ ftp voyager.deanza.fhda.edu

Connected to voyager.deanza.fhda.edu.

220 (vsFTPd 1.2.1)

530 Please login with USER and PASS.

Name (voyager.deanza.fhda.edu:forouzan): forouzan

331 Please specify the password.

Password:

230 Login successful.

Remote system type is UNIX.

Using binary mode to transfer files.

ftp> ls reports

227 Entering Passive Mode (153,18,17,11,238,169)

150 Here comes the directory listing.

drwxr-xr-x	2 3027	411	4096 Sep 24 2002 business
drwxr-xr-x	2 3027	411	4096 Sep 24 2002 personal
drwxr-xr-x	2 3027	411	4096 Sep 24 2002 school

226 Directory send OK.

ftp> quit

221 Goodbye.

Example 26.5

We show an example of anonymous FTP. We assume that some public data are available at internic.net.

\$ ftp internic.net

Connected to internic.net

220 Server ready

Name: anonymous

331 Guest login OK, send "guest" as password

Password: guest

continued on next slide

Example 26.5 (continued)

```
ftp > pwd
257 '/' is current directory
ftp > ls
200 OK
150 Opening ASCII mode
bin
ftp > close
221 Goodbye
ftp > quit
```