4. Remote login - TELNET & File transfer, FTP, TFTP

DATE / /

TELNET

Concept

i) It is an abbreviation for TErminal NETwork.
ii) It is standard TCP/IP protocol for virtual termina service as proposed by ISO.
iii) It is a general purpose client server appln program It is related to several concepts

a) Time sharing Environment

TELNET was designed at a time when most os such as UNIX, where operating in a time sharing env.
- In such env. large computer supports multiple users.
- The interaction between a user & the computer occurs through a terminal which is usually combination of keyboard, monitor & mouse.
- In a time sharing env., all of the processing must be done by the central computer. When user types a character on the keyboard, the character usually sent to the compute & echoed to the monitor. Time sharing creates an env. in which each user has the illusion of dedicated compute the user can run a program, access the system resources, switch from one program to another & so on.

b) Login

- In a time sharing env., users are part of the system with some right to access resources. Each authorized user has an identification & probably a password.
- The user identification defines the user as part of the system, the user logs into the system with user id or login name
- The system also includes password checking to prevent an unauthorized user from accessing the resources.

Local Login

- When a user logs into a local time sharing system, it is called local login.
- As a user types at a terminal or at a workstation running a terminal emulator, the keystrokes are accepted by the terminal driver.
- The terminal driver passes the characteres to the OS.
- The OS in turn, interprets the combination of characters & invoke the desired application program or utility

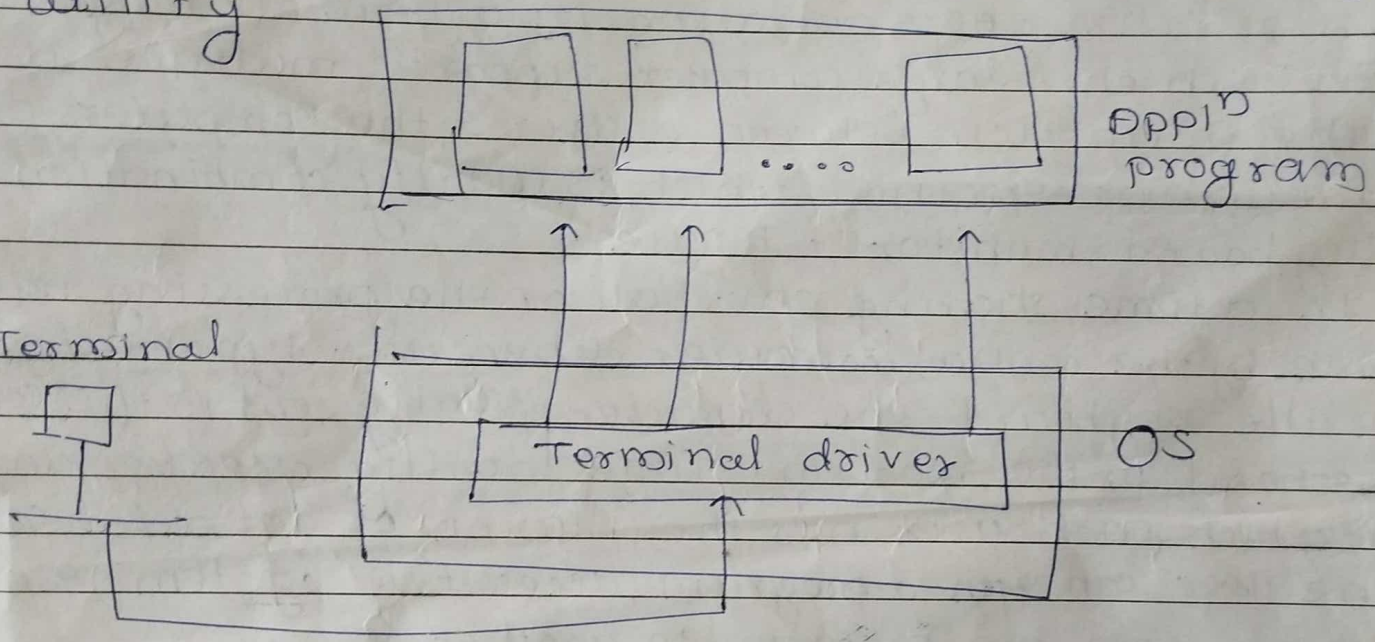


Fig → Local login

Remote Login

- When a user wants to access an appln program or utility located on a remote machine, he/she performs remote login.
- Here Telnet client & server programs are used.
- The user sends the keystrokes to the terminal driver where the local OS accepts the characters but does not interpret them.
- The characters are sent to the TELNET client, which transforms the characters to a universal character set called Network virtual Terminal (NVT)

GC

Client Server Model

characters & delivers them to the local TCP/IP stack.

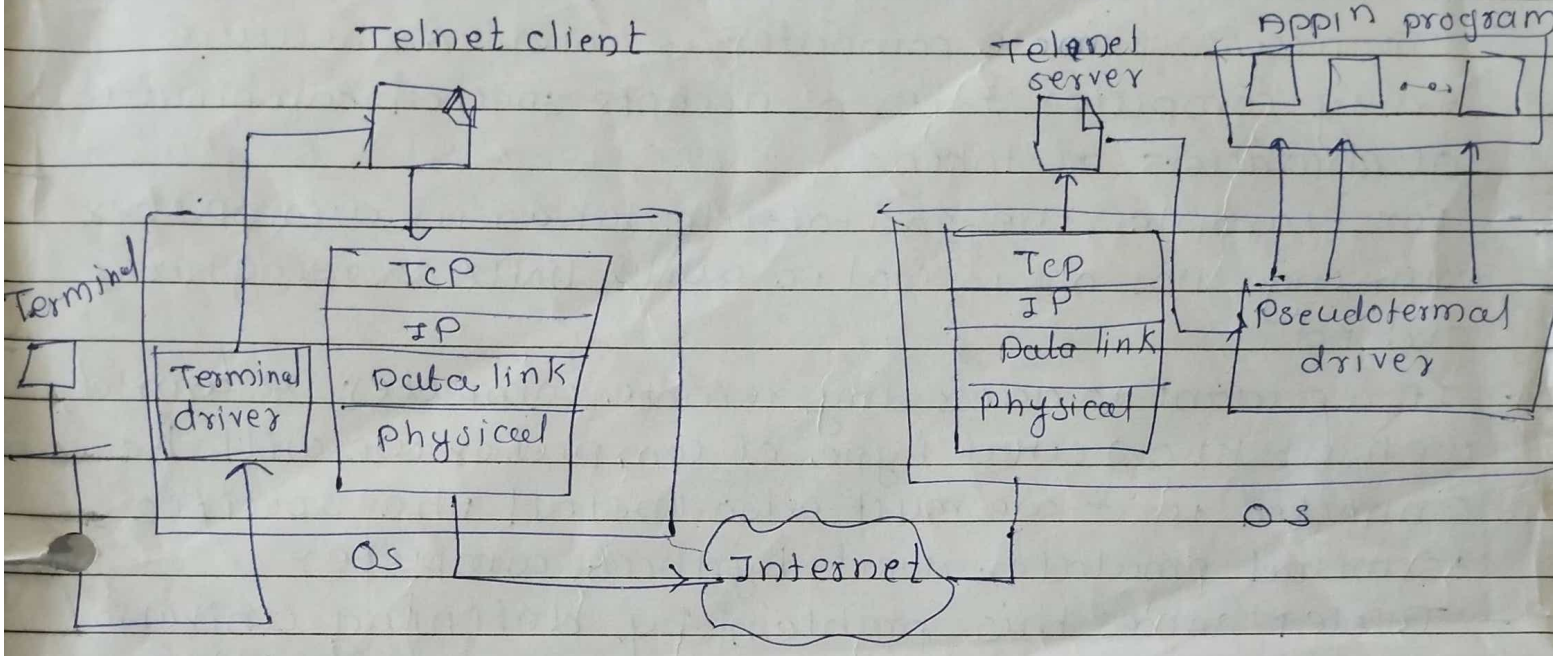


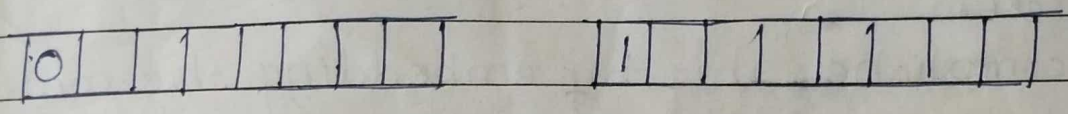
Fig → Remote login.

- The commands or text, in NVT form, travel through the Internet & arrive at the TCP/IP stack at the remote machine.
- The characters are delivered to the OS & passed to the Telnet server, which changes the characters to the corresponding characters understandable by remote computer.
- The character can't be passed directly to the OS because remote OS is not designed to receive characters from a TELNET server.
- It is designed to receive characters from a terminal driver. The soln is to add a piece of sw called pseudoterminal driver, which pretends that characters are coming from a terminal.
- The OS then passes the characters to the appropriate appln programs.

DATE / /

NVT character set

It uses two set of characters, one for data & one for control. Both are 8 bit bytes.



Data character control character

Fig → Format of data & control character

Data character

- 8 bit character set in which seven lowest order bits are same as US ASCII & highest order bit is 0
- Although it is possible to send an 8 bit ASCII, this must first be agreed upon between client & server using option negotiation

control character

- To send control characters between computers NVT uses 8 bit character set in which highest order bit is set to 1

Some NVT control character

character	meaning
EOF	End of file
EOR	End of Record
SE	subOption End
NOP	Not operation
DM	Data mark
BRK	Break
AYT	Are You There
AO	Abort output
EL	Erase Line
EC	Erase character

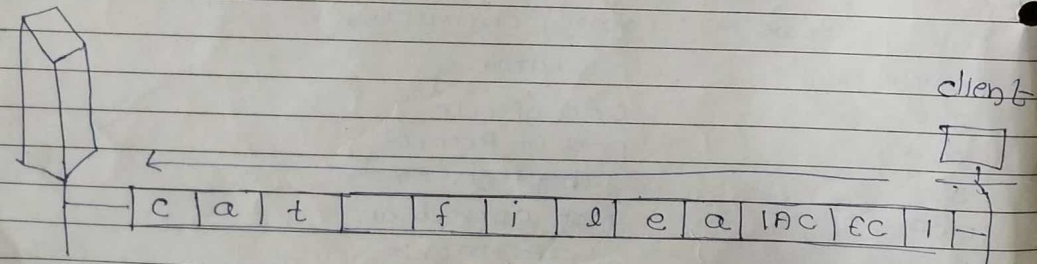
Embedding

- Telnet uses only one TCP connection. The server uses the well known port 23 & client uses an ephemeral port.
- The same connection is used for sending both data & control characters.
- TELNET accomplishes this by embedding the control characters in the data stream. However to distinguish data from control characters, each sequence of control characters is preceded by special control character called interpret as control (IAC).
- e.g. imagine a user wants a server to display file (file1) on a remote server. She types:

cat file1

In which cat is a Unix command that displays the content of the file on the screen. However, the name of the file has been mistyped (file a instead of file1). The user uses the backspace key to correct this situation.

cat file a <backspace>1



Fig→ An example of embedding.

- However, in the default implementation of TELNET, the user can't edit locally, the editing is done at the remote server.
- The backspace character is translated into two remote characters (IAC EC), which is embedded in the data & sent to the remote server. What is sent to the server is shown in above figure.

Options

- TELNETS lets the client & server negotiate options before or during the use of the service.
- Options are extra features available to user with more sophisticated terminal.
- Users with simpler terminals can use default features.
- Table shows some common options.

code	option	meaning
0	Binary	Interpret as 8 bit binary transmissio
1	Echo	Echo the data received on one side to the other
3	Supress go-ahead	Supress go-ahead signals after data
5	Status	Request the status TELNET
6	Timing mark	Define timing marks
24	Terminal type	Set terminal type
32	Terminal speed	Set terminal speed
34	Line mode	change to line mode

Binary →

- This option allows the receiver to interpret every 8-bit character received, except IAC as binary data, when IAC is received, next characters are interpreted as commands.
- However, if two consecutive IAC characters are received, the 1st is discarded & 2nd is interpreted as data.

Echo

This option allows the server to echo data received from the client. This means that every character sent by client to the server will be echoed back to the screen of the client terminal.

- In this case, user terminal usually does not echo characters when they are typed but waits until it receives them from the server.

Supress go-ahead

This option supressed go-ahead (GA) character

Status

This option allows the user or the process running on the client machine to get the status of the options being enabled at server site.

Timing mark

This option allows one party to issue a timing mark that indicates all previously received data has been processed.

Terminal type

Allows the client to send it's terminal type

Terminal speed

Allows client to send it's terminal speed.

Line mode

Allows the client to switch to line mode.

Option Negotiation

Four control characters are used

Table → NVT character set for option negotiation

character	code	Meaning 1	Meaning 2	Meaning 3
WILL	251	Offering to enable	Accepting to enable	
WONT	252	Rejecting to enable	Offering to disable	
DO	253	Approving to enable	Requesting to enable	
DONT	254	Disapproving to enable	Approving to disable	

Enabling an option

- Some options can only be enabled by server, some only by client & some by both.
- An option is enabled either through an offer or a request.

Offer to Enable

- A party can offer to enable an option if it has right to do so.
- The offering party sends WILL command which means "will I enable the option?" The other party sends either the DO command, which means "please do," or DONT command which means "please don't" see fig.

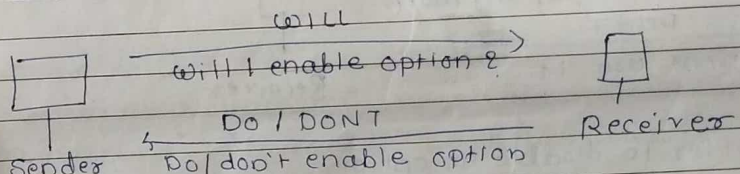


Fig → Offer to enable an option

Request to Enable

- A party can request from other party enabling of an option
- The request can be accepted or refused by other party
- The requesting party sends the DO command, which means "please to enable the option".
- The other party sends either WILL command which means 'I will', or the WONT command which means "I won't."

Disabling an option

- An option that has been enabled can disabled by one of the parties.
- An option is disabled either through an offer or request

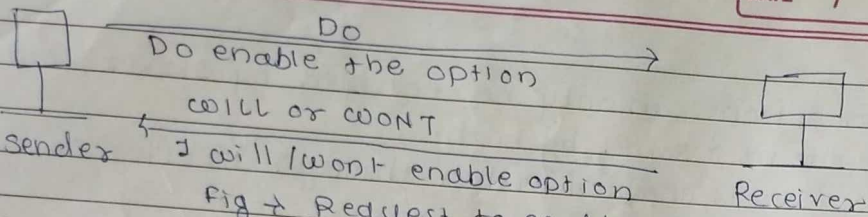


Fig → Request to enable option

Offer to Disable

- A party can offer to disable an option
- The other party must approve the offering, it cannot be disapproved.
- The offering party sends WONT command which means I won't use this option anymore. The answer must be the DONT command which means "Don't use it anymore".

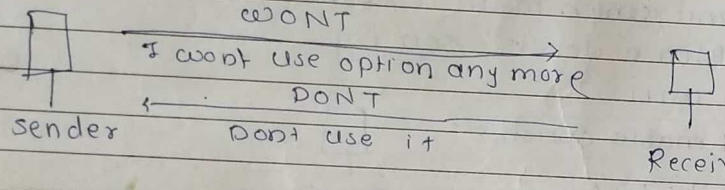


Fig → offer to disable an option.

Request to Disable

- A party can request from another party disabling of an option
- The other party must accept request, it can't be rejected.
- The requesting party sends the DONT command, which means "please don't use this option anymore.
- The answer must be WONT command which means 'I wont use it anymore.

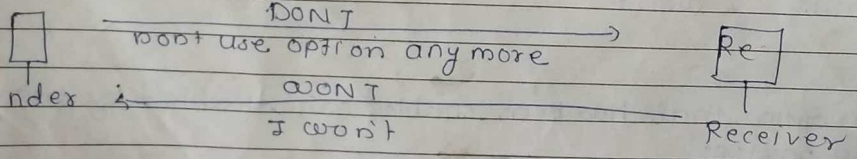
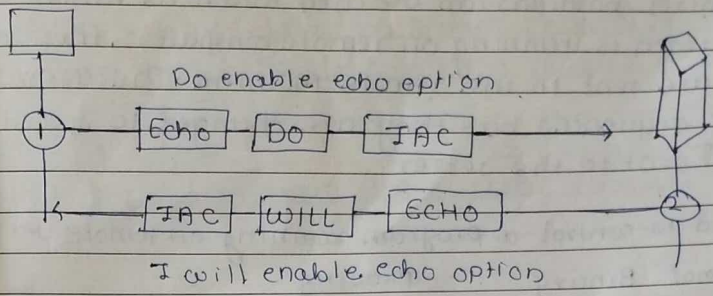


Figure → Request to disable an option.

Example



- The client should request from server that sends the character to IAC, DO & ECHO. The server accepts the request & enables option. It informs the client by sending three character approval: IAC, WILL & ECHO.

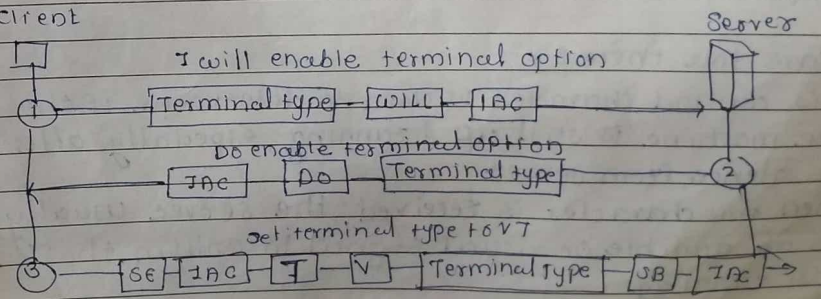
Suboption Negotiation

- Some options require additional info., e.g to define type or speed of a terminal, negotiation includes string or no. to define type or speed. In either case, two suboption characters indicated in table are needed for suboption negotiation.

NVT character set for suboption negotiation

character	Decimal	Binary	meaning
SE	240	11110000	suboption end
SB	250	11111010	suboption begin

Example of sub-option negotiation



Controlling the server

- Some control characters can be used to control remote server.
- When program is running on remote computer, these control characters are sent to the remote machine. The user still types same sequences, but they are changed to special characters & sent to the server.

characters used to control a program running on remote server

character	Decimal	Binary	maning
IP	244	11110100	Interrupt process
AO	245	11110101	Abort output
AYT	246	11110110	Are you there
EC	247	11110111	Erase last character
EL	248	11111000	Erase line.

IP

- If the program is running on remote machine, the appropriate function should be called by the os of the remote machine.
- The TELNET defines IP control character that is read & interpreted as appropriate command for invoking the interrupting function in remote machine.

AO (Abort output)

- Same as IP, but it allows the process to continue without creating output.
- This is useful if the process has another effect in addition to creating output. The user wants this effect but not the output.

AYT (Are you there?)

- This control character is used to determine if the remote machine is still up & running, especially after a long silence from server.
- When this character is received, the server usually sends an audible or visual signal to confirm that it is running.

EC (Erase character)

- When user sends data from keyboard to the local machine, the delete / backspace character can erase last character typed. To do same in remote machine.

EL (Erase Line)

This is used to erase the current line in the remote host.

Out-of-Band Signaling

- To make control characters effective in special situation TELNET uses out-of-Band signaling.
- In out-of-Band signaling control characters are preceded by IAC & sent to the remote process.
- Imagine situation in which an application program running at server site has gone into an infinite loop & does not accept any more input data.
- The user wants to interupt application program, but the program does not read data from buffer.
- The TCP at server site has found that buffer is full & has sent segment specifying that the client window size should be zero.
- When TELNET process (client / server) wants to send any out-of-band sequence of characters to other process (client/server) it embeds the sequence in data stream & inserts a special character called DM (data mark).
- However, to inform the other party, it creates TCP segment with urgent bit set & urgent pointer pointing to the DM character.
- When receiving process receives data, it reads the data & discards any data preceding control character.
- When it reaches the DM character, the remaining data are handled normally.
- In other words DM character is used as synchronization character that switches the receiving process from the urgent mode to the normal mode & resynchronizes two ends.

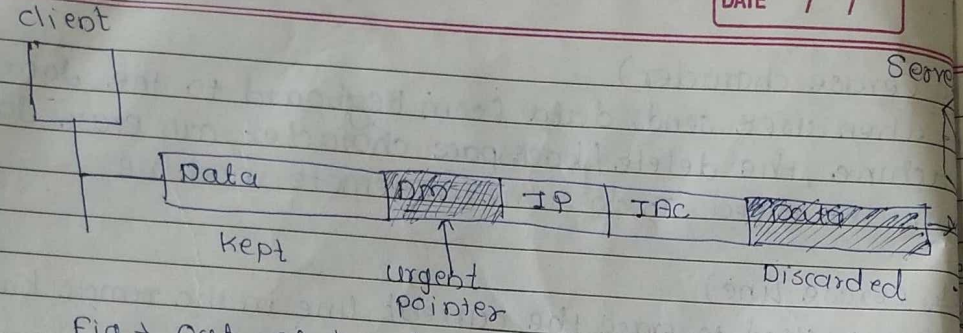


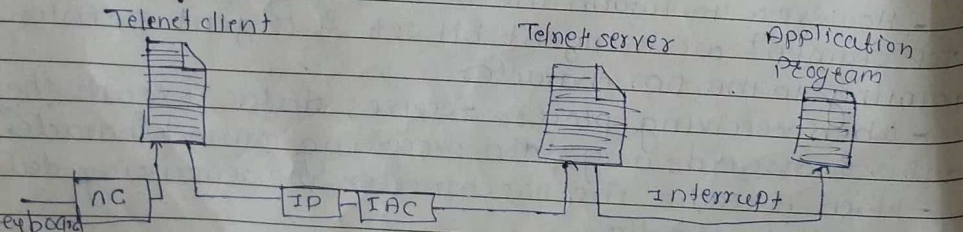
Fig → Out-of-band-signaling

- In this way, control character (IP) is delivered out-of-band to the OS, which uses the appropriate function to interrupt the running application program.

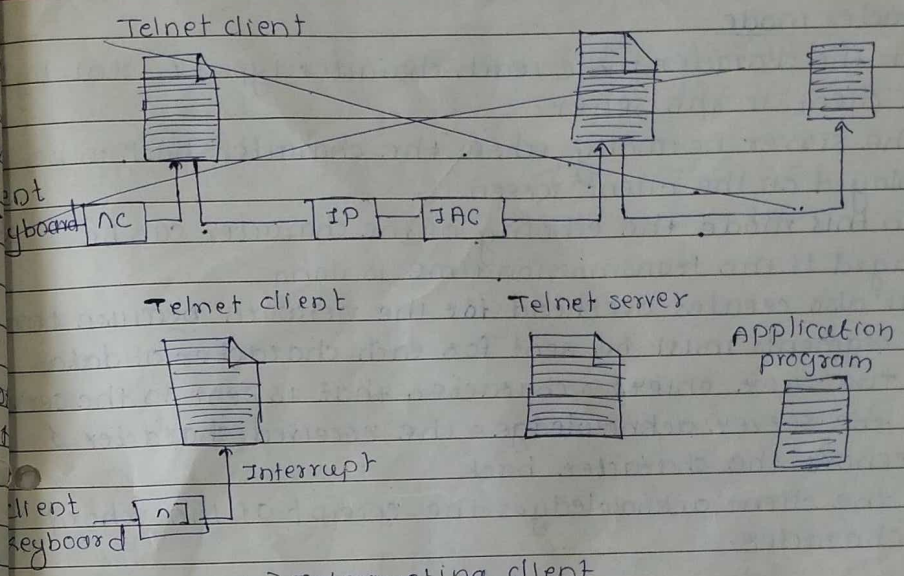
Escape character

- A character typed by the user is normally sent to server.
- However, sometimes the user wants characters interpreted by client instead of the server.
- In this case the user can use escape character, normally Ctrl+]
- Following fig. compares the interruption of an application program at the remote site with interruption of the client process at local site using escape character.
- The TELNET prompt is displayed after this escape character.

Fig → Two different interruptions



a) Interrupting application program



b) Interrupting client

Modes of operation

- Most TELNET implementations operate in one of three default mode, character mode or line mode.

Default mode

- The default mode is used if no other modes are invoked through option negotiation.
- In this mode, the echoing is done by client.
- The user types a character & client echoes the characters on the screen but does not send it until whole line is completed.
- After sending whole line to the server, the client waits for the GA-go ahead command from server before accepting new line from the server.
- The operation is half duplex. Half duplex operation is not efficient when the TCP connection itself is full duplex & so this mode is becoming ~~obsolete~~ obsolete.

Character mode

- In the character mode, each character typed is sent by the client to the server.
- The server normally echoes the character back to be displayed on the client screen.
- In this mode the echoing of the character can be delayed if the transmission time is long
- It also creates overhead for the networks because three TCP segments must be sent for each character of data.

1) The user enters a character that is sent to the server
2) the server acknowledges the received character & echoes the character back
3) The client acknowledges the receipt of the echoed character.

③ Line mode

- A new mode has been proposed to compensate for the deficiencies of the default mode & the character mode.
- In this mode, called the line mode, line editing is done by the client.
- The client then sends the whole line to the server.
- Although line mode looks like default mode, it is not.
- The default mode operates in the half duplex mode.
- The line mode is full duplex with the client sending one line after another, without need for an intervenning characters from the server.

User Interface

- The normal user does not use TELNET command. It has defines an interface with user friendly commands.
- The interface is responsible for translating the user friendly commands to the commands in the protocol.

command	meaning
open	connect to remote computer
close	close the connection
display	show the operating command
mode	chang to line or character mode
set	set the operating parameter
status	Display the status information
send	Send special characters
quit	Exit Telnet

- The telnet client can use SSH

* FTP -

① File Transfer protocol (FTP) is the standard mechanism provided by TCP/IP for copying file from one host to another.

② Although transferring files from one system to another seems simple.

③ For ex - 2 systems may use different file name conventions.

④ Two systems may have different ways to represent text and data.

⑤ Two systems may have different directory structures.

⑥ All of these problems have been solved by FTP in very simple approach.

⑦ FTP differs from other client-server applications in that it establishes two connections bet^n hosts.

⑧ one connection is used for data transfer, the other for control information.

⑨ The control connection uses very simple rules of communication.

⑩ we need to transfer only line of command or line of response at time.

⑪ FTP uses 2 well-known TCP ports : port 21 is used for control connection and port 20 is used for data connection.

Port 21 - control conn.
20 - data conn.

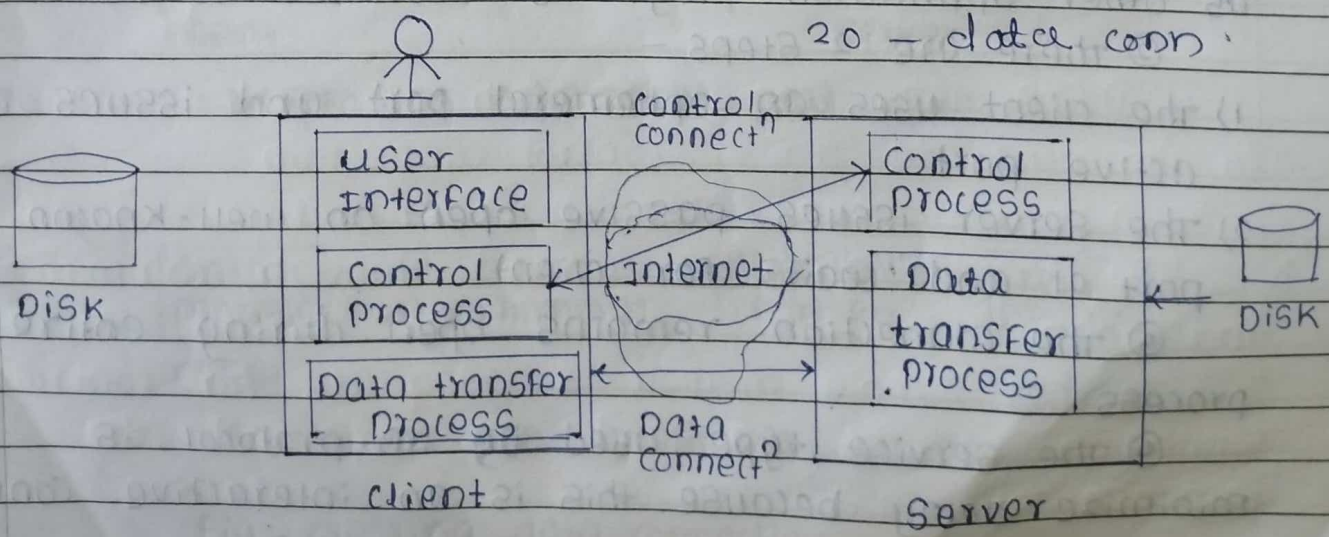


Fig:- FTP

The client has 3 components –
1) User interface
2) Client control process
3) Client data transfer process.

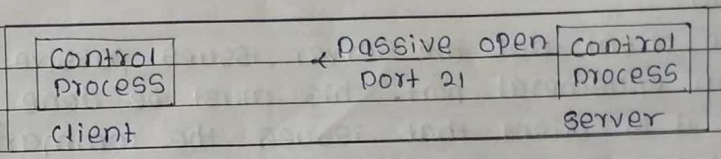
The server has 2 components –
1) Server control process
2) Server data transfer process.

The control connection is made between the control processes.
The data connection is made between data transfer processes.
The control connection remains connected during entire interactive FTP session.
The data connection is opened and then closed for each file transferred.

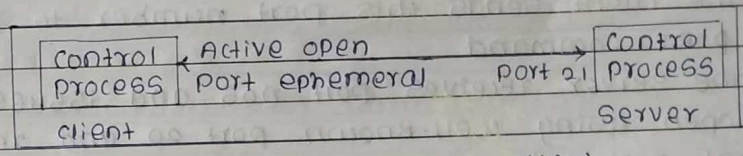
* Connections –
The 2 FTP connections, control and data use different strategies and different port no's.

① Control connection –

① The control connection is created in same way as other application programs described so far.
② There are 2 steps –
1) The client uses an ephemeral port and issues an active port.
2) The server issues passive open on well-known port 21 and waits for client.
③ The connection remains open during entire process.
④ The service type, used by IP protocol, is minimize delay because this is an interactive connect between user and server.
⑤ The user types commands and expects to receive responses without significant delay.



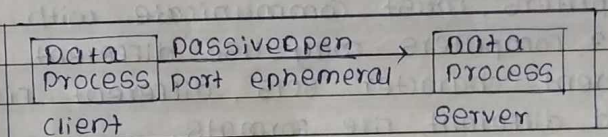
a) passive open by server



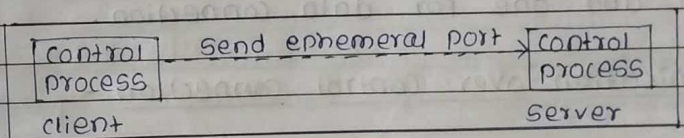
b) Later active open by client.

Fig :- opening control connection.

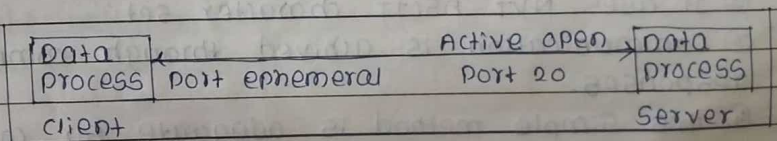
② Data connection –



a) passive open by client.



b) Sending ephemeral port.



c) Active open by server.

Fig :- Creating data connection.

① The data connection uses well-known port 20 at server site.

② The following shows how FTP creates data connection.

1) The client, not the server, issues passive open using an ephemeral port. This must be done by client because it is client that issues the commands for transferring files.

2) The client sends this port number to server using PORT command.

3) The server receives port no's and issues an active open using well-known port 20 and received ephemeral port no's.

* Communications -

① The FTP client and server, which run on different computers, must communicate with each other.

② These 2 computers may use different operating systems, different character sets, different file structures and different file formats.

③ FTP must make this heterogeneity compatible.

④ FTP has 2 different approaches, one for control connection and one for data connection.

1] Communication over control connection -

① FTP uses same approach as TELNET (or) SMTP to communicate across control connection.

② It uses NVT ASCII character set.

③ Communication is achived through commands and responses.

④ This simple method is adequate for control connection because we send one command (or) response at time.

⑤ Each command or response is only one short line so we need not worry about file format or file structure.

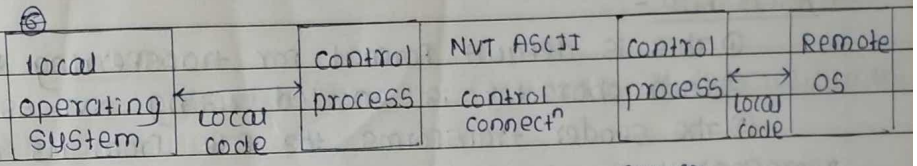


Fig - Using control connection.

2] Communication over Data Connection -

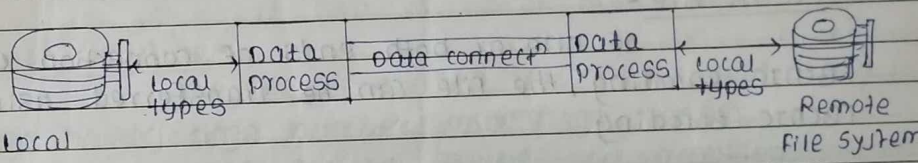


Fig :- Using data connection.

① The purpose and implementation of data connection are different from that of the control connection.

② We want to transfer files through the data connection.

③ The client must define type of file to be transferred, the structure of data and transmission mode.

④ Before sending file through data connection, we prepare for transmission through control connection.

⑤ The heterogeneity problem is resolved by defining 3 attributes of communication -

1) File type
2) Data structure
3) Transmission mode

① File Type - FTP can transfer one of the following file types across data connection.

1) ASCII File -
① This is default format for transferring text
② Each character is encoded using NVT ASCII.
③ The sender transforms the file from its own representation into NVT ASCII character & receiver transforms the NVT ASCII characters to its own representation.

2) EBCDIC File -
If one or both ends of connection use EBCDIC encoding, the file can be transferred using EBCDIC encoding.

3) Image File -
① This is default format for transferring binary files.
② The file is sent as continuous streams of bits without any interpretation or encoding.

If file is encoded in ASCII or EBCDIC, another attribute must be added to define printability of file.

a) Nonprint -
This is default format for transferring text file. The file contains no vertical specifications for printing.

b) TELNET -
In this format the file contain NVT ASCII vertical characters such as CR (carriage return), LF (line feed), NL (new line) and VT (vertical tab).
The file is printable after transfer.

② Data Structure -
FTP can transfer file across the data connection using one of the following structure.

1) File Structure -
The file has no structure. It is continuous stream of bytes.

2) Record Structure -
The file is divided into records. this can be used only with text files.

3) page structure -
The file is divided into pages, with each page having page number and page header.
This pages can be stored and accessed randomly (or) sequentially.

③ Transmission mode -

1) Stream mode -
① This is default mode.
② Data are delivered from FTP to TCP as a continuous stream of bytes.

2) Block mode -
① Data can be delivered from FTP to TCP in blocks
② In this case, each block is preceded by a 3-byte header.
③ The first byte is called block descriptor; nety 2 bytes define size of block in bytes.

3) Compressed mode -
① If file is big, the data can be compressed
② The compression method normally used is

run-length encoding.

③ In this method, consecutive appearance of data unit are replaced by one occurrance and number of repetitions.

* Command processing -

① FTP uses control connection to establish a communication between client control process & server control process.

② During this communication, the commands are sent from client to server and responses are sent from server to client.

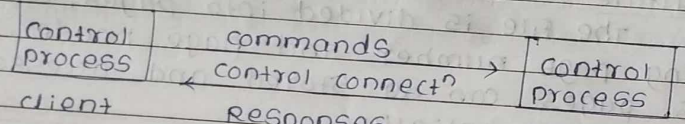


Fig :- Command processing

Commands -

① Commands, which are sent from FTP client control process, are in the form of ASCII uppercase which may or may not be followed by an argument.

② We can roughly divide commands into six groups.

1) Access commands
2) File management commands
3) Data Formatting commands
4) port defining commands
5) File transferring commands
6) miscellaneous commands.

1) Access commands - These commands let user access the remote system

Command	Argument	Description
USER	user id	User information
PASS	user password	password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of system
ABOR		Abort previous command.

2) File management commands -

These commands let user access file system on remote computer.

Command	Argument	Description
CWD	Directory name	change to another directory
CDUP		change to parent directory
DELE	File name	Delete file
LIST	Directory name	List subdirectories or files
MKD	Directory name	Create new directory
RMD	Directory name	Delete directory
RNTO	File name (new)	Rename the file
SMNT	File sys. name	mount file system.

3) port defining commands -

① These commands define port nos for data connection on client site.

② There are 2 methods to do this.

③ In first method, using PORT command, the client can choose an ephemeral port number and send it to server using passive open.

④ The server uses that port no's and creates an active open.

⑤ In second method, using PASV command, client just asks server to first choose port no's.

⑥ The server does passive open on that port and sends port no's in response.

Command	Argument	Description
PORT	6-digit identifier	Client chooses port
PASV		Server chooses port

4) Data formatting commands -

These commands let user define data structure, file type, and transmission mode.

Command	Argument	Description
TYPE	A (ASCII), E (EBCDIC), I (Image), N (Nonprint), T (TELNET)	Define file type
STRU	F (File), R (Record), P (page)	
MODE	S (Stream), B (Block), C (Compressed)	

5) Miscellaneous commands -

Command	Argument	Description
HELP		Ask info. about server
NOOP		Check if server is alive
SITE	Commands	Specify site-specific commands.

6) File transfer commands -

These commands actually let user transfer files.

Command	Arguments
RETR	File name
STOR	File name
APPE	〃
STOU	〃
ALLO	〃
REST	〃
STAT	〃

* File Transfer -

① File transfer occurs over data connection under control of commands sent over control connection.

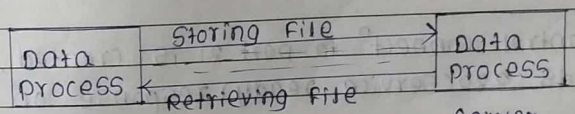
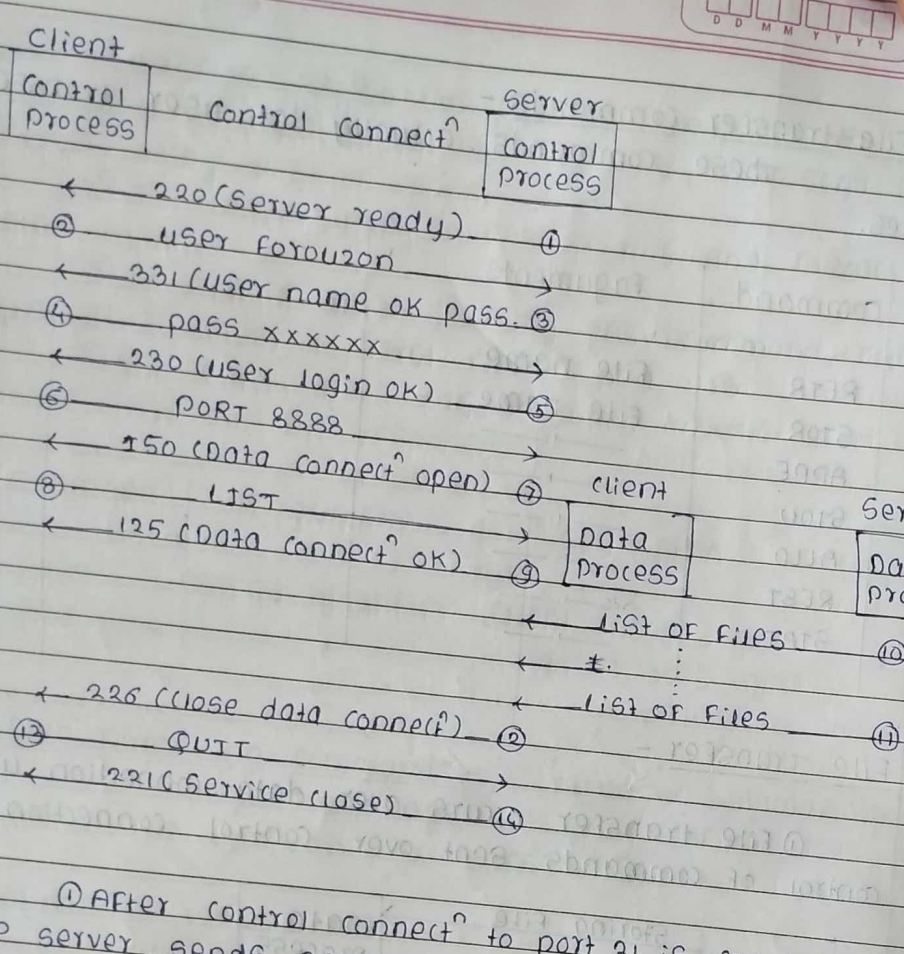


Fig :- File Transfer.

— A file is to be copied from server to the client this is called retrieving a file.

— A file is to be copied from client to the server this is called storing a file.

— A list of directory or file names is to be sent from server to client. This is done under supervision of LIST command.



① After control connect^n to port 21 is created, the FTP server sends 220 (service ready) response on the control connect^n.

② Client sends USER command.

③ Server responds with 331 (user name is OK, password is required)

④ Client sends PASS command.

⑤ Server responds with 230 (user login is OK)

⑥ Client issues passive open on ephemeral port for data connect^n and sends PORT command.

⑦ It send response 150 (data connect^n will open)

⑧ Client sends LIST message.

⑨ Now server responds with 125 and opens data connect^n.

⑩ Server send list of files on data connection.

⑪ Server responds with 226 (closing data connect^n) over the control connect^n.

⑫ In our example, client sends QUIT command.

⑬ After receiving QUIT command, the server responds with 221 (service closing) and then closes control connect^n.

Anonymous FTP -

① To use FTP user needs an account & password on remote server.

② Some sites have set of files available for public access.

③ To access these files, user does not need to have an account or password.

④ Instead, the user can use anonymous as user name and guest the password.

⑤ For ex - most sites allow the user to copy some files, but do not allow navigation through directories.

* Security for FTP -

① The FTP protocol was designed when security was not big issue.

② Although FTP requires password, the password is sent in plaintext, which means it can intercepted and used by an attacker.

③ The data transfer connect^n also transfers data in plaintext, which is insecure.

④ To be secure, one can add ~~Sattle~~ Secure Socket Layer bet^n FTP application layer and TCP layer.

⑤ In this case FTP is called SSL-FTP.

* TFTP -

① There are occasions when we need to simply copy a file without need for all of the features of FTP protocol.

② For ex- when diskless workstation or router booted, we need to download bootstrap & configure files.

③ Here we do not need all of sophistication provided in FTP.

④ We just need protocol that quickly copies the files.

⑤ Trivial File Transfer protocol (TFTP) is designed for these types of file transfer.

⑥ It is so simple that software package can fit into ROM of diskless workstation.

⑦ It can be used at bootstrap time.

⑧ The reason that it fits on ROM is that it requires only basic IP and UDP.

⑨ However, there is no security for TFTP.

* TFTP msgs -

There are 5 types of TFTP messages.
RRQ, WRQ, DATA, ACK and ERROR

① RRQ -

The read request (RRQ) msg is used by client to establish connect^n for reading data from server.
Its format is below -

opcode = 1	File Name	All 0's	mode	All 0's
2 bytes	variable	1 byte	variable	1 byte

opcode -

The first field is 2-byte operation code. The value is 1 for RRQ msg.

File Name -

The next field is variable-size string that defines name of the file.

Since file name varies in length, termination is signaled by 1-byte field of 0's.

mode -

The next field is another variable-size string defining the transfer mode.

The mode field is terminated by another 1-byte field of 0's.

The mode can be one of two strings: "netascii" or "octet".

The file name and mode fields can be in upper or lowercase or combination of both.

② WRQ -

The write request (WRQ) msg is used by client to establish connect^n for writing data to server.

opcode = 2	File Name	All 0's	mode	All 0's
2 bytes	variable	1-byte	variable	1-byte

The format is same as RRQ except the opcode = 2.

③ DATA -

The DATA msg is used by client or server to send blocks of data.

opcode = 3	Block Number	Data
2-bytes	2-bytes	0-512 bytes

opcode -

The first field is 2-byte operation code. The value 3 for DATA msg.

Block Number -

① This is 2-byte field containing block number.
② The sender of data uses field for sequencing
③ All blocks are numbered sequentially starting with 1.
④ The block number is necessary for ack. as well see shortly.

Data -

This block must be exactly 512 bytes in all DATA msg except last block, which must be between 0 & 512 bytes.
A non-512 byte block is used as signal that sender has sent all data.

④ ACK -

The Acknowledge (ACK) msg is used by client or server to ack. the receipt of data block.

opcode = 4	Block Number
2 bytes	2 bytes

opcode -

The first field is 2-byte operation code. the value is 4 for ack msg

Block Number -

① The next field is 2-byte field containing no. of block received.
② The ack msg can also be response to WRQ.
③ It is sent by server to indicate that it is ready to receive data from client.
④ In this case value of block number field is 0.

⑤ ERROR -

① The ERROR msg is used by client or server when connect° cannot be established or when there is problem during data transmission.
② It can be sent negative response to RRQ or WRQ
③ It can also be used if next block cannot be transferred during actual data transfer phase.
④ The error msg is not used to declare damaged or duplicated msg

opcode = 5	Error Number	Error data	All 0's
2-bytes	2-bytes	variable	1-byte

opcode -

The first field is 2-bytes operation code. the value is 5 for ERROR msg.

Error Number -

This 2-bytes field defines type of error.

Error data -

This variable-byte field contains textual error data and is terminated by 1-byte field of 0's.

* Connection -

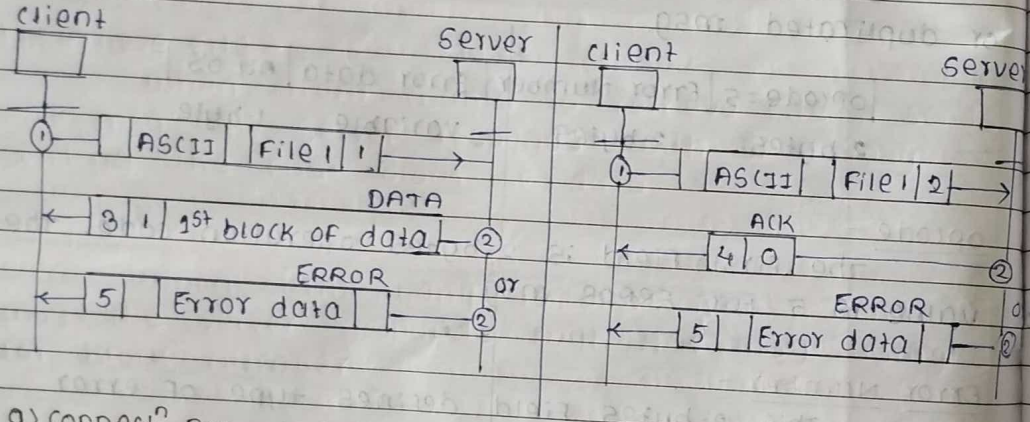
① TFTP uses UDP services.
② Because there is no provision for connection establishment and termination in UDP, UDP transfer each block of data encapsulated in independant user datagram.
③ In TFTP, however, we do not want to transfer only one block of data; we do not want to transfer file as independant blocks either.
④ We need connections for blocks of data being

transferred if they all belong to same file.

⑤ DATA msg with block of data of fewer than 512 bytes (0-511) to terminate connect^n.

Connection Establishment –

connection establishment for reading files is different from connect^n establishment for writing files.



a) connect^n for reading

b) connect^n for writing

Reading –

① To establish connect^n for reading, the TFTP client sends RRQ msg.

② the name of file and transmission mode is defined in this msg.

③ If server can transfer the file, it responds positively with DATA msg containing the first block of data.

④ If there is problem, such as difficulty in opening file or permission restriction, the server responds negatively by sending ERROR msg.

Writing –

① To establish connection for writing, the TFTP client uses WRQ msg.

② the name of file and transmission mode is defined in this msg.

③ If server can accept copy of file, it responds positively with an ACK msg using value of 0 for block number.

④ If there is any problem, the server responds negatively by sending ERROR msg.

Connection Termination –

① After entire file is transferred, the connection must be terminated.

② As mentioned previously, TFTP does not have special msg for termination.

③ Termination is accomplished by sending last block of data, which is less than 512 bytes.

* Data Transfer –

① The data transfer phase occurs bet^n connection establishment and termination.

② TFTP uses services of UDP, which is unreliable.

③ The file is divided into blocks of data, in which each block except last one is exactly 512 bytes.

④ The last block must be bet^n 0 and 511 bytes.

⑤ TFTP can transfer data in ASCII or binary format.

⑥ UDP does not have any mechanism for flow and error control.

⑦ TFTP has to create flow and error control mechanism to transfer file made of continuous blocks of data.

Flow control –

① TFTP sends block of data using DATA msg and waits for ACK msg.

② If sender receives ack before time-out, it sends next block.

③ Thus, flow control is achived by numbering data blocks and waiting for ack before next data is sent.

I] Retrieve file –

When client wants to retrieve file, it sends RRQ msg. The server responds with DATA msg sending the first block of data.

II] Store file –

When client wants to store file, it sends WRQ msg. The server responds with ACK msg using 0 for block number.

Error control –

① TFTP error-control mechanism is different from those of other protocols.

② It is symmetric, which means that sender and receiver both use time-outs.

③ The sender use time-out for data msg; the receiver uses time-out for ack. msg.

④ Error control is needed in 4 situation –

I] Damaged msg –

There is no -ve ack. If block of data is damaged, it is detected by receiver and block is discarded.

II] Lost msg –

If block is lost, it never reaches receiver and no ack is sent. The sender resends block after time-out.

III] Lost Ack –

If an ack is lost, we can have 2 situations, If timer of receiver matures before timer of the sender, the receiver retransmits ack; otherwise sender retransmits the data.

IV] Duplicate msg –

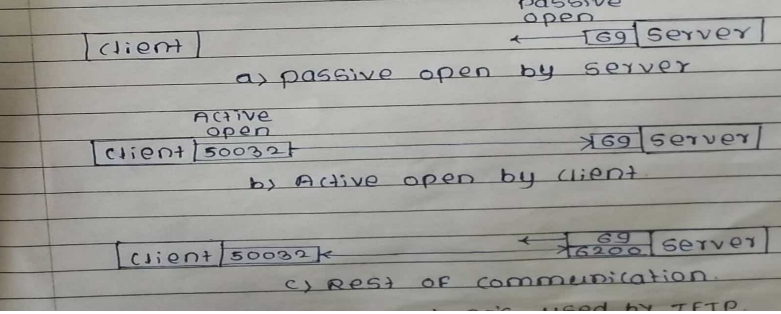
Duplication of blocks can be detected by receiver through block number.

If block is duplicated, it is simply discarded by receiver.

UDP ports –

① When process uses services of UDP, server process issues a passive open on well-known port and waits for client process to issue an active open on ephemeral port.

② After connect° is established, the client and server communicate using these ports.



Flow control -

① TFTP sends block of data using DATA msg and waits for ACK msg.

② If sender receives ack before time-out, it sends next block.

③ Thus, flow control is achived by numbering data blocks and waiting for ACK before next data is sent.

I] Retrieve file -

When client wants to retrieve file, it sends RRQ msg. The server responds with DATA msg sending the first block of data.

II] Store file -

When client wants to store file, it sends WRQ msg. The server responds with ACK msg using 0 for block number.

Error control -

① TFTP error-control mechanism is different from those of other protocols.

② It is symmetric, which means that sender and receiver both use time-outs.

③ The sender use time-out for data msg; the receiver uses time-out for ack. msg.

④ Error control is needed in 4 situation -

I] Damaged msg -

There is no -ve ack. If block of data is damaged, it is detected by receiver and block is discarded.

II] Lost msg -

If block is lost, it never reaches receiver and no ack is sent. The sender resends block after time-out.

III] Lost ACK -

If an ack is lost, we can have 2 situations, If timer of receiver matures before timer of the sender, the receiver retransmits ack; otherwise sender retransmits the data.

IV] Duplicate msg -

Duplication of blocks can be detected by receiver through block number.

If block is duplicated, it is simply discarded by receiver.

* UDP ports -

① When process uses services of UDP, server process issues a passive open on well-known port and waits for client process to issue an active open on ephemeral port.

② After connect is established, the client and server communicate using these ports.

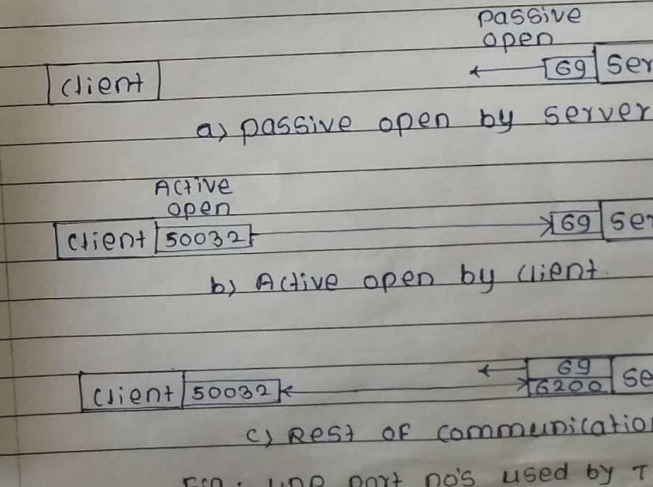


Fig :- UDP port no's used by TFTP.

Steps -

① The server passively opens communication using well-known port 69.

② A client actively opens connection using an ephemeral port for source port and well-known port 69 for destination port.

③ A server actively opens connection using new ephemeral port for source port and uses ephemeral port received from client as destination port.