18CSC302J- Computer Networks

Unit-3

Syllabus

1. DNS- DNS in the Internet, DNS Resolution, DNS Messages
2. TELNET – SSH
3. **FTP – TFTP**
4. WWW Architecture, Documents
5. HTTP, HTTP Request and Reply,
6. DHCP Operation, DHCP Configuration
7. SMTP, POP3, IMAP, MIME

Learning Resources

1. Douglas E. Comer, Internetworking with TCP/IP, Principles, protocols, and architecture,Vol 1 5th Edition,2006 ISBN: 0131876716, ISBN: 978-0131876712

FILE TRANSFER

FTP & TFTP

FILE TRANSFER

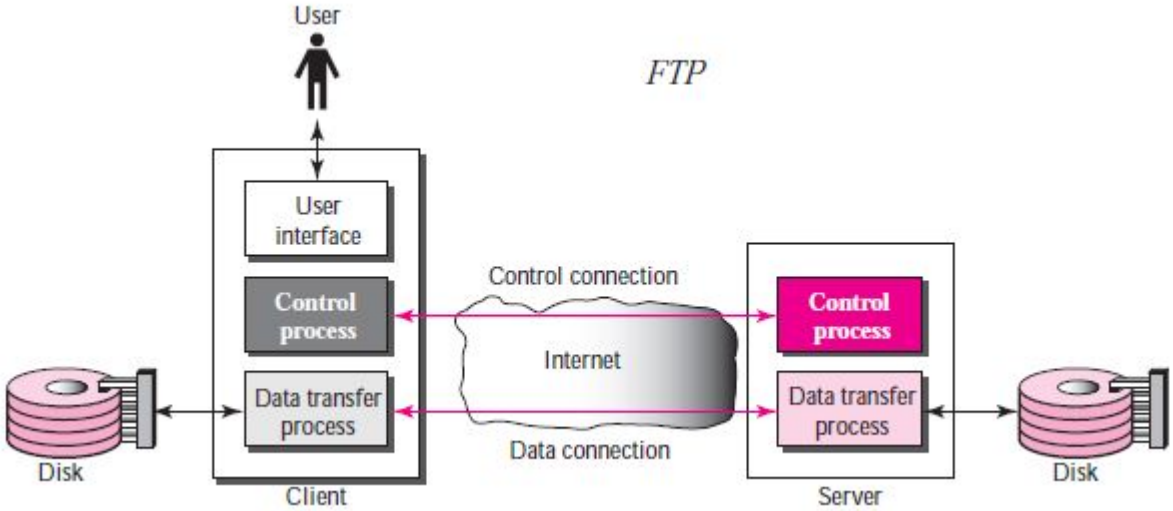
- Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment.
- As a matter of fact, the greatest volume of data exchange in the Internet today is due to file transfer.
- Two protocols involved in transferring files:
    1. File Transfer Protocol (FTP) and
    2. Trivial File Transfer Protocol (TFTP)

FTP

- File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.
- Transferring files from one system to another seems simple and straightforward, some problems must be dealt with first.
  - For example, two systems may use different file name conventions.
  - Two systems may have different ways to represent text and data.
  - Two systems may have different directory structures.
- All of these problems have been solved by FTP in a very simple and elegant approach.
- FTP differs from other client-server applications in that it establishes two connections between the hosts.
- One connection is used for data transfer, the other for control information (commands and responses).
- Separation of commands and data transfer makes FTP more efficient.
- The control connection uses very simple rules of communication.

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



FTP-Basic model working

- The client has three components: user interface, client control process, and the client data transfer process.
- The server has two components: the server control process and the server data transfer process.
- The control connection is made between the control processes.
- The data connection is made between the data transfer processes.
- The control connection remains connected during the entire interactive FTP session.
- The data connection is opened and then closed for each file transferred.
- It opens each time commands that involve transferring files are used, and it closes when the file is transferred.
- In other words, when a user starts an FTP session, the control connection opens.
- While the control connection is open, the data connection can be opened and closed multiple times if several files are transferred.

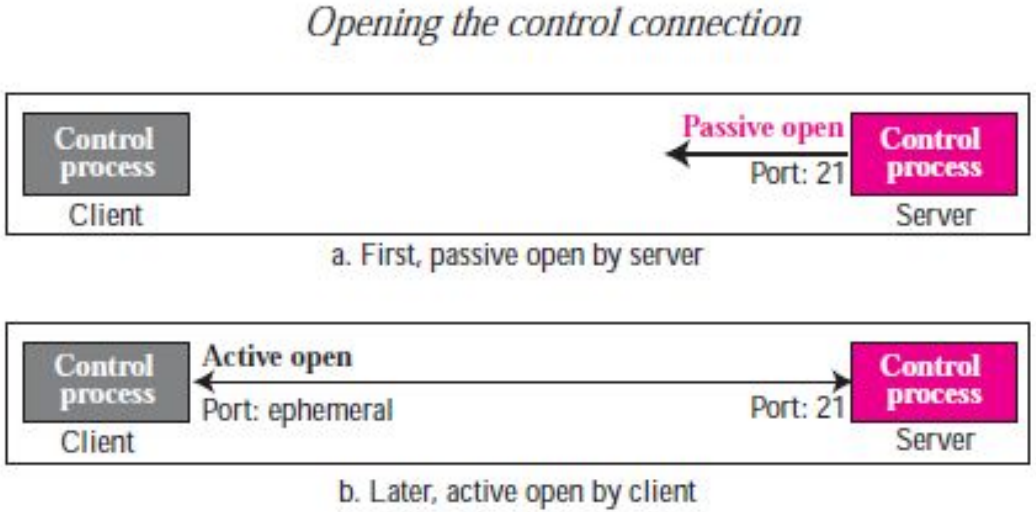
1. Connections

- The two FTP connections, control and data, use different strategies and different port numbers.

a) Control Connection

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

Connections

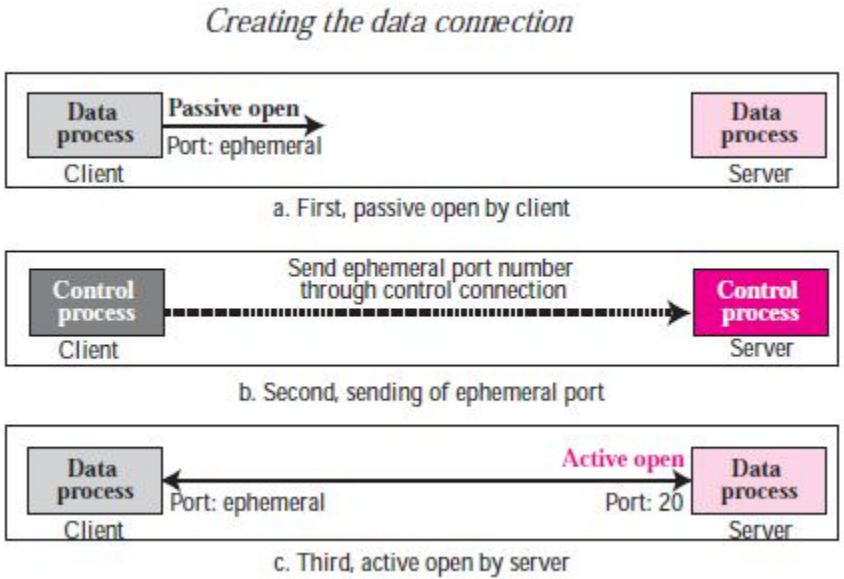


Connections

b) Data Connection

- The data connection uses the well-known port 20 at the server site.
- However, the creation of a data connection is different from what we have seen so far.
- The following shows how FTP creates a data connection:
  1. The client, not the server, issues a passive open using an ephemeral port. This must be done by the client because it is the client that issues the commands for transferring files.
  2. The client sends this port number to the server using the PORT command
  3. The server receives the port number and issues an active open using the wellknown port 20 and the received ephemeral port number.
- The steps for creating the initial data connection are shown in Figure next slide

Connections



2. Communication

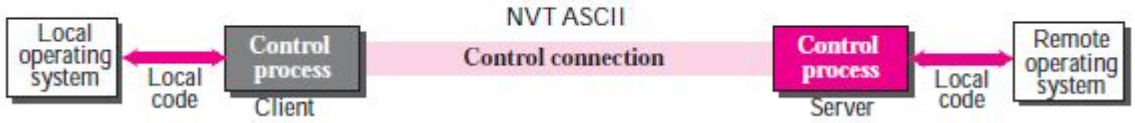
- The FTP client and server, which run on different computers, must communicate with each other.
- These two computers may use different operating systems, different character sets, different file structures, and different file formats.
- FTP must make this heterogeneity compatible.
- FTP has two different approaches, one for the control connection and one for the data connection.

Communication

a) Communication over Control Connection

- FTP uses the same approach as TELNET or SMTP to communicate across the control connection.
- It uses the NVT ASCII character set
- Communication is achieved through commands and responses.
- This simple method is adequate for the control connection because we send one command (or response) at a time.
- Each command or response is only one short line so we need not worry about file format or file structure.
- Each line is terminated with a two-character (carriage return and line feed) end-of-line token.

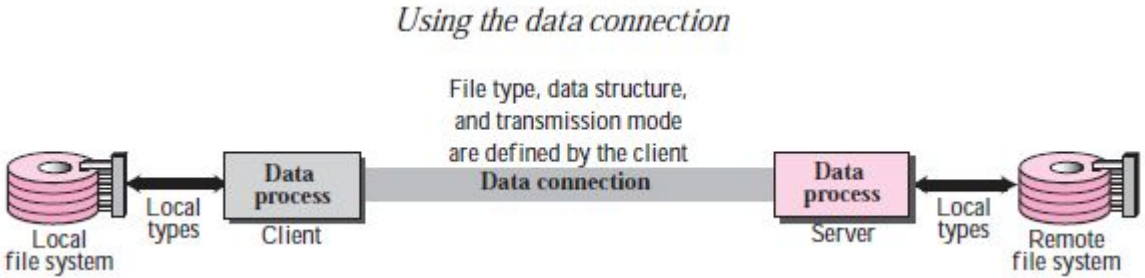
Using the control connection



Communication

b) Communication over Data Connection

- The purpose and implementation of the data connection are different from that of the control connection.
- We want to transfer files through the data connection.
- The client must define the type of file to be transferred, the structure of the data, and the transmission mode.
- Before sending the file through the data connection, we prepare for transmission through the control connection.
- The heterogeneity problem is resolved by defining three attributes of communication: file type, data structure, and transmission mode



Communication

File Type: FTP can transfer one of the following file types across the data connection:

❑ ASCII file.

- This is the default format for transferring text files.
- Each character is encoded using NVT ASCII.
- The sender transforms the file from its own representation into NVT ASCII characters and the receiver transforms the NVT ASCII characters to its own representation.

❑ EBCDIC file.

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

❑ Image file.

- This is the default format for transferring binary files.
- The file is sent as continuous streams of bits without any interpretation or encoding.
- This is mostly used to transfer binary files such as compiled programs.

Communication

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

a. Nonprint.

- This is the default format for transferring a text file.
- The file contains no vertical specifications for printing.
- This means that the file cannot be printed without further processing because there are no characters to be interpreted for vertical movement of the print head.
- This format is used for files that will be stored and processed later.

b. TELNET.

- In this format the file contains 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.

Communication

Data Structure : FTP can transfer a file across the data connection using one of the following interpretations about the structure of the data:

❑ File structure (default).

- The file has no structure.
- It is a continuous stream of bytes.

❑ Record structure.

- The file is divided into records.
- This can be used only with text files.

❑ Page structure.

- The file is divided into pages, with each page having a page number and a page header.
- The pages can be stored and accessed randomly or sequentially.

Communication

Transmission Mode : FTP can transfer a file across the data connection using one of the following three transmission modes:

❑ Stream mode.

- This is the default mode.
- Data are delivered from FTP to TCP as a continuous stream of bytes.
- TCP is responsible for chopping data into segments of appropriate size.
- If the data is simply a stream of bytes (file structure), no end-of file is needed.
- If the data are divided into records (record structure), each record will have a 1-byte end-of-record (EOR) character and the end of the file will have a 1-byte end-of-file (EOF) character.

❑ Block mode.

- Data can be delivered from FTP to TCP in blocks.
- In this case, each block is preceded by a 3-byte header.
- The first byte is called the block descriptor; the next two bytes define the size of the block in bytes.

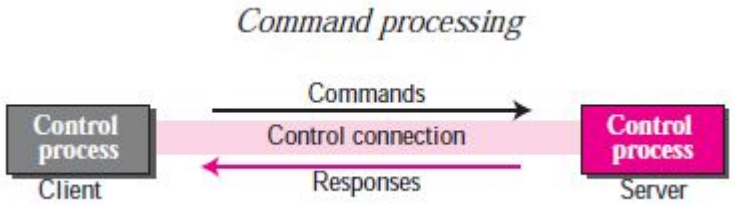
Communication

❑ Compressed mode.

- If the file is big, the data can be compressed.
- The compression method normally used is run-length encoding.
- In this method, consecutive appearances of a data unit are replaced by one occurrence and the number of repetitions.
- In a text file, this is usually spaces (blanks).
- In a binary file, null characters are usually compressed.

3. Command Processing

- FTP uses the control connection to establish a communication between the client control process and the server control process.
- During this communication, the commands are sent from the client to the server and the responses are sent from the server to the client.



Command Processing

Commands

- Commands, which are sent from the FTP client control process, are in the form of ASCII uppercase, which may or may not be followed by an argument.
- Commands are roughly divide into six groups:
  - Access commands
  - File management commands
  - Data formatting commands
  - Port defining commands
  - File transferring commands and
  - Miscellaneous commands

Command Processing

a) Access commands

- These commands let the user access the remote system.
- Below table lists common commands in this group.

Access commands

Command	*Argument(s)*	*Description*
USER	User id	User information
PASS	User password	Password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of the system
ABOR		Abort the previous command

Command Processing

b) File management commands.

- These commands let the user access the file system on the remote computer.
- They allow the user to navigate through the directory structure, create new directories, delete files, and so on.
- Some common commands in this group.

File management commands

Command	*Argument(s)*	*Description*
CWD	Directory name	Change to another directory
CDUP		Change to parent directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
NLIST	Directory name	List subdirectories or files without attributes
MKD	Directory name	Create a new directory
PWD		Display name of current directory
RMD	Directory name	Delete a directory
RNFR	File name (old)	Identify a file to be renamed
RNTO	File name (new)	Rename the file
SMNT	File system name	Mount a file system

Command Processing

c) Data formatting commands

- These commands let the user define the data structure, file type, and transmission mode.
- The defined format is then used by the file transfer commands.

Data formatting commands

Command	*Argument(s)*	*Description*
TYPE	**A** (ASCII), **E** (EBCDIC), **I** (Image), **N** (Nonprint), or **T** (TELNET)	Define file type
STRU	**F** (File), **R** (Record), or **P** (Page)	Define organization of data
MODE	**S** (Stream), **B** (Block), or **C** (Compressed)	Define transmission mode

Command Processing

d) Port defining commands.

- These commands define the port number for the data connection on the client site.
- There are two methods to do this.
- In the first method, using the PORT command, the client can choose an ephemeral port number and send it to the server using a passive open.
- The server uses that port number and creates an active open.
- In the second method, using the PASV command, the client just asks the server to first choose a port number.
- The server does a passive open on that port and sends the port number in the response
- The client issues an active open using that port number.

Port defining commands

Command	*Argument(s)*	*Description*
PORT	6-digit identifier	Client chooses a port
PASV		Server chooses a port

Command Processing

e)File transfer commands.

- These commands actually let the user transfer files.

File transfer commands

Command	*Argument(s)*	*Description*
RETR	File name(s)	Retrieve files; file(s) are transferred from server to client
STOR	File name(s)	Store files; file(s) are transferred from client to server
APPE	File name(s)	Similar to STOR, but if file exists, data must be appended to it
STOU	File name(s)	Same as STOR, but file name will be unique in the directory
ALLO	File name(s)	Allocate storage space for files at the server
REST	File name(s)	Position file marker at a specified data point
STAT	File name(s)	Return status of files

Command Processing

f) Miscellaneous commands.

- These commands deliver information to the FTP user at the client site.

Miscellaneous commands

Command	*Argument(s)*	*Description*
HELP		Ask information about the server
NOOP		Check if server is alive
SITE	Commands	Specify the site-specific commands
SYST		Ask about operating system used by the server

Command Processing

Responses

- Every FTP command generates at least one response.
- A response has two parts: a three digit number followed by text.
- The numeric part defines the code; the text part defines needed parameters or extra explanations.
- We represent the three digits as xyz.
- The meaning of each digit is described below.

First Digit

- The first digit defines the status of the command.
- One of five digits can be used in this position:

Command Processing

First Digit

- The first digit defines the status of the command.
- One of five digits can be used in this position:

❑ 1yz (positive preliminary reply).

- The action has started. The server will send another reply before accepting another command.

❑ 2yz (positive completion reply).

- The action has been completed. The server will accept another command.

❑ 3yz (positive intermediate reply).

- The command has been accepted, but further information is needed.

❑ 4yz (transient negative completion reply).

- The action did not take place, but the error is temporary. The same command can be sent later.

❑ 5yz (permanent negative completion reply).

- The command was not accepted and should not be retried again.

Command Processing

Second Digit

- The second digit also defines the status of the command.
- One of six digits can be used in this position:
  - ❑ x0z (syntax).
  - ❑ x1z (information).
  - ❑ x2z (connections).
  - ❑ x3z (authentication and accounting).
  - ❑ x4z (unspecified).
  - ❑ x5z (file system).

Third Digit

- The third digit provides additional information.
- Table shows list of possible responses (using all three digits).

Command Processing

Third Digit

Responses

Code	*Description*
Positive Preliminary Reply	
120	Service will be ready shortly
125	Data connection open; data transfer will start shortly
150	File status is OK; data connection will be open shortly
Positive Completion Reply	
200	Command OK
211	System status or help reply
212	Directory status
213	File status
214	Help message
215	Naming the system type (operating system)
220	Service ready
221	Service closing
225	Data connection open
226	Closing data connection
227	Entering passive mode; server sends its IP address and port number
230	User login OK
250	Request file action OK
Positive Intermediate Reply	
331	User name OK; password is needed
332	Need account for logging
350	The file action is pending; more information needed

Command Processing

Third Digit

Responses (continued)

Code	*Description*
Transient Negative Completion Reply	
425	Cannot open data connection
426	Connection closed; transfer aborted
450	File action not taken; file not available
451	Action aborted; local error
452	Action aborted; insufficient storage
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 parameter not implemented
530	User not logged in
532	Need account for storing file
550	Action is not done; file unavailable
552	Requested action aborted; exceeded storage allocation
553	Requested action not taken; file name not allowed

4. File Transfer

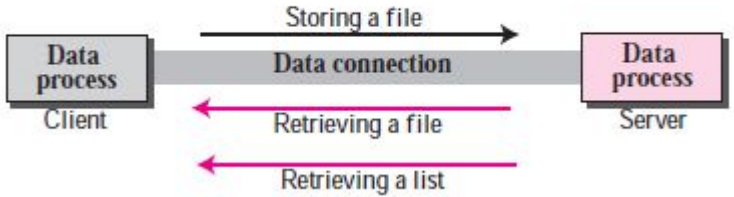
- File transfer occurs over the data connection under the control of the commands sentover the control connection.
- However, we should remember that file transfer in FTP means one of three things

❑ A file is to be copied from the server to the client (download). This is called retrieving a file. It is done under the supervision of the RETR command.

❑ A file is to be copied from the client to the server (upload). This is called storing a file. It is done under the supervision of the STOR command.

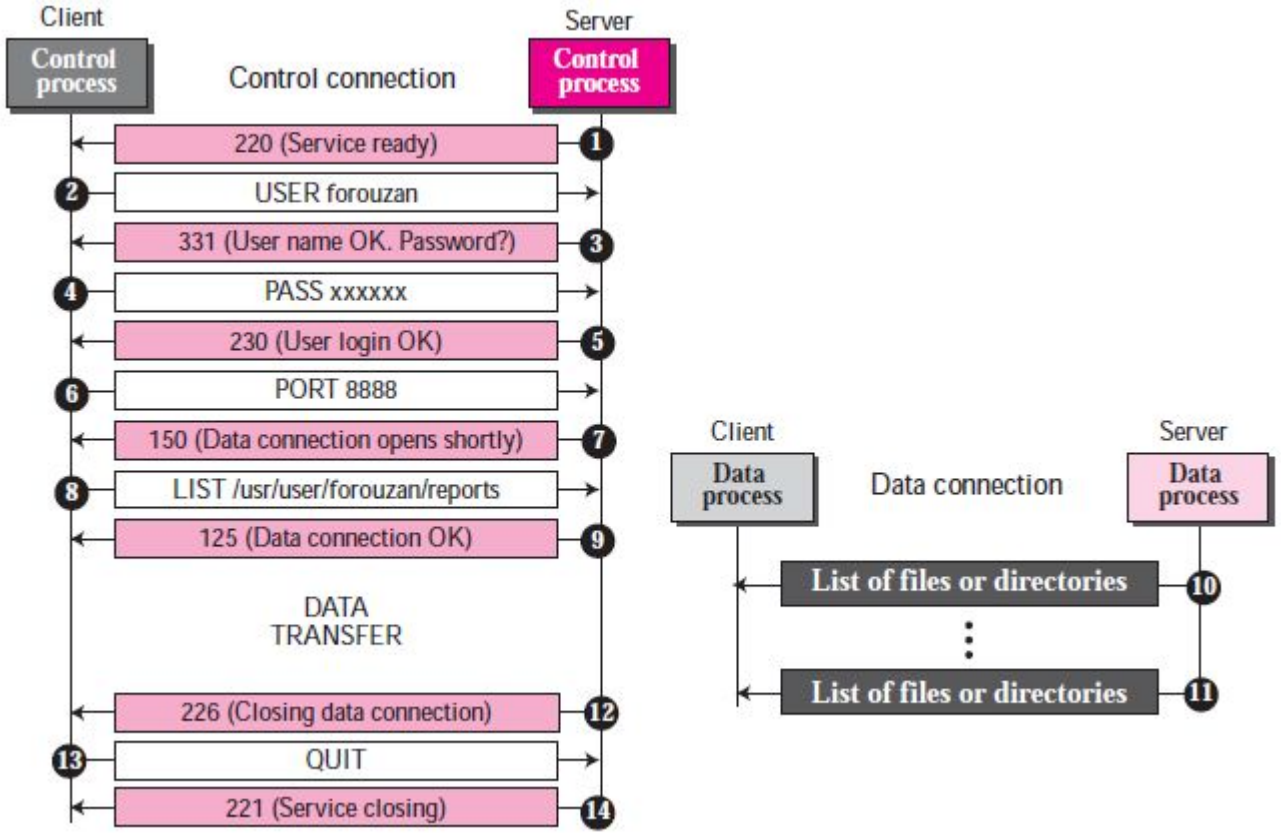
❑ A list of directory or file names is to be sent from the server to the client. This is done under the supervision of the LIST command. Also FTP treats a list of directory or file names as a file. It is sent over the data connection.

File transfer



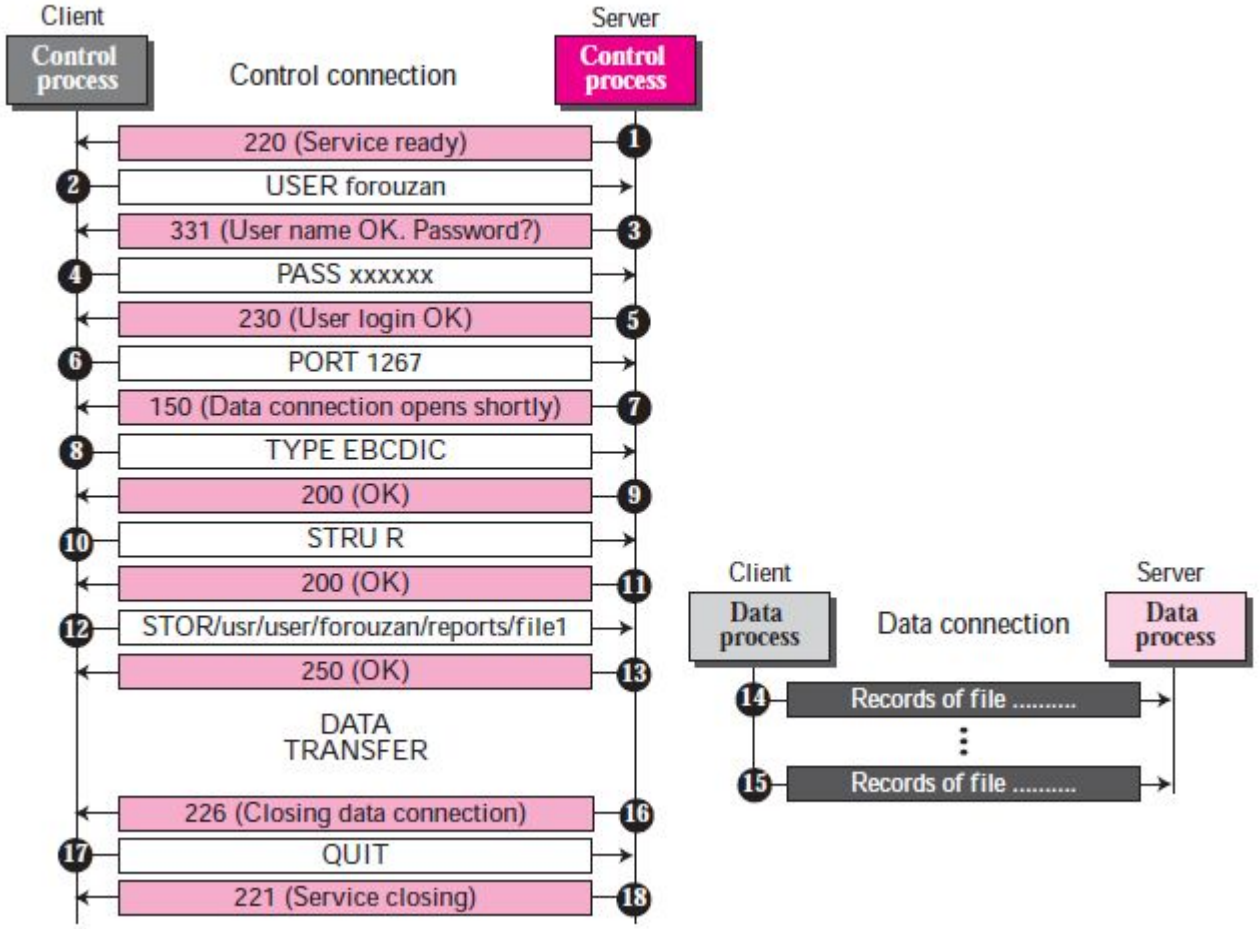
File Transfer

Example of using FTP for retrieving a list of items in a directory.



File Transfer

Example of how an image (binary) file is stored.



5. Anonymous FTP

- To use FTP, a user needs an account (user name) and a password on the remote server.
- Some sites have a set of files available for public access.
- To access these files, a user does not need to have an account or password.
- Instead, the user can use anonymous as the user name and guest as the password.
- User access to the system is very limited.
- Some sites allow anonymous users only a subset of commands.
- For example, most sites allow the user to copy some files, but do not allow navigation through the directories.

6. Security for FTP

- The FTP protocol was designed when the security was not a big issue.
- Although FTP requires a password, the password is sent in plaintext (unencrypted), which means it can be intercepted and used by an attacker.
- The data transfer connection also transfers data in plaintext, which is insecure.
- To be secure, one can add a Secure Socket Layer between the FTP application layer and the TCP layer.
- In this case FTP is called SSL-FTP.

Security for FTP

The sftp Program

- Another way to transfer files using a secure channel is to use another independent protocol called sftp (secure file transfer protocol).
- This is actually a program in Unix called sftp that is part of the SSH protocol
- When SSH has established a secure connection between an SSH client and an SSH server, one of the application programs that can use this connection (multiplexing) is sftp.
- In other words, sftp is part of the application component of the SSH.
- The sftp program is an interactive program that can work like FTP and uses a set of interface commands to transfer files between the SSH client and SSH server.

FTP-SUMMARY

- Connections
  - Control Connection
  - Data Connection
- Communication
  - Communication over Control Connection
  - Communication over Data Connection
    - File Type
    - Data Structure
    - Transmission Mode
- Command Processing
  - Commands
  - Responses
- File Transfer
- Anonymous FTP
- Security for FTP
- The sftp Program

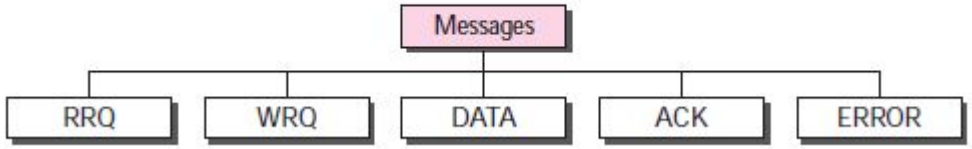
TFTP

- There are occasions when we need to simply copy a file without the need for all of the features of the FTP protocol.
- For example, when a diskless workstation or a router is booted, we need to download the bootstrap and configuration files.
- We just need a protocol that quickly copies the files.
- **Trivial File Transfer Protocol (TFTP)** is designed for these types of file transfer.
- It is so simple that the software package can fit into the read-only memory of a diskless workstation.
- It can be used at bootstrap time.
- The reason that it fits on ROM is that it requires only basic IP and UDP.
- However, there is no security for TFTP. TFTP can read or write a file for the client.
- Reading means copying a file from the server site to the client site.
- Writing means copying a file from the client site to the server site.
- TFTP uses the services of UDP on the well-known port 69.

1. Messages

- There are five types of TFTP messages, RRQ, WRQ, DATA, ACK, and ERROR,

Message categories

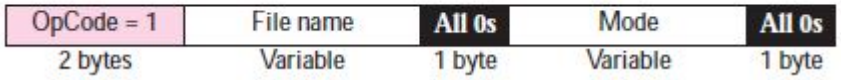


Messages

a. RRQ

- The **read request (RRQ) message** is used by the client to establish a connection for reading data from the server

RRQ format



The RRQ message fields are as follows:

❑ OpCode.

- The first field is a 2-byte operation code. The value is 1 for the RRQ message.

❑ File name.

- The next field is a variable-size string (encoded in ASCII) that defines the name of the file. Since the file name varies in length, termination is signaled by a 1-byte field of 0s.

❑ Mode.

- The next field is another variable-size string defining the transfer mode.
- The mode field is terminated by another 1-byte field of 0s.
- The mode can be one of two strings: “netascii” (for an ASCII file) or “octet” (for a binary file).
- The file name and mode fields can be in upper- or lowercase, or a combination of both.

Messages

b. WRQ

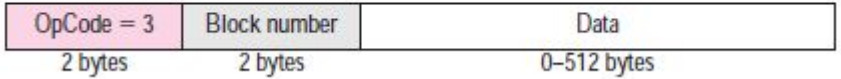
- The **write request (WRQ) message** is used by the client to establish a connection for writing data to the server.

WRQ format

c. DATA

- The data (DATA) message is used by the client or the server to send blocks of data.

DATA format



Messages

❑ OpCode.

- The first field is a 2-byte operation code. The value is 3 for the DATA message.

❑ Block number.

- This is a 2-byte field containing the block number.
- The sender of the data (client or server) uses this field for sequencing.
- All blocks are numbered sequentially starting with 1.
- The block number is necessary for acknowledgment.

❑ Data.

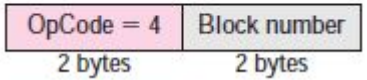
- This block must be exactly 512 bytes in all DATA messages except the last block, which must be between 0 and 511 bytes.
- A non-512 byte block is used as a signal that the sender has sent all the data.
- In other words, it is used as an end-of-file indicator.
- If the data in the file happens to be an exact multiple of 512 bytes, the sender must send one extra block of zero bytes to show the end of transmission

Messages

d. ACK

- The acknowledge (ACK) message is used by the client or server to acknowledge the receipt of a data block. The message is only 4 bytes long.

ACK format



The ACK message fields are as follows:

❑ OpCode.

- The first field is a 2-byte operation code. The value is 4 for the ACK message.

❑ Block number.

- The next field is a 2-byte field containing the number of the block received.

- The ACK message can also be a response to a WRQ.
- It is sent by the server to indicate that it is ready to receive data from the client.

Messages

e. ERROR

- The ERROR message is used by the client or the server when a connection cannot be established or when there is a problem during data transmission.
- It can be sent as a negative response to RRQ or WRQ.
- It can also be used if the next block cannot be transferred during the actual data transfer phase.
- The error message is not used to declare a damaged or duplicated message.
- These problems are resolved by error-control mechanisms

ERROR format

Messages

e. ERROR

The ERROR message fields are as follows:

❑ OpCode.

- The first field is a 2-byte operation code. The value is 5 for the ERROR message.

❑ Error number.

- This 2-byte field defines the type of error.

❑ Error data.

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

Error numbers and their meanings

Number	*Meaning*	*Number*	*Meaning*
0	Not defined	5	Unknown port number
1	File not found	6	File already exists
2	Access violation	7	No such user
3	Disk full or quota exceeded		
4	Illegal operation		

2. Connection

- **TFTP uses UDP services.**
- Because there is no provision for connection establishment and termination in UDP, UDP transfers each block of data encapsulated in an independent user datagram.
- In TFTP, however, we do not want to transfer only one block of data; we do not want to transfer the file as independent blocks either.
- We need connections for the blocks of data being transferred if they all belong to the same file.
- TFTP uses RRQ, WRQ, ACK, and ERROR messages to establish connection.
- It uses the DATA message with a block of data of fewer than 512 bytes (0–511) to terminate connection.

Connection

a) Connection Establishment

- Connection establishment for reading files is different from connection establishment for writing files

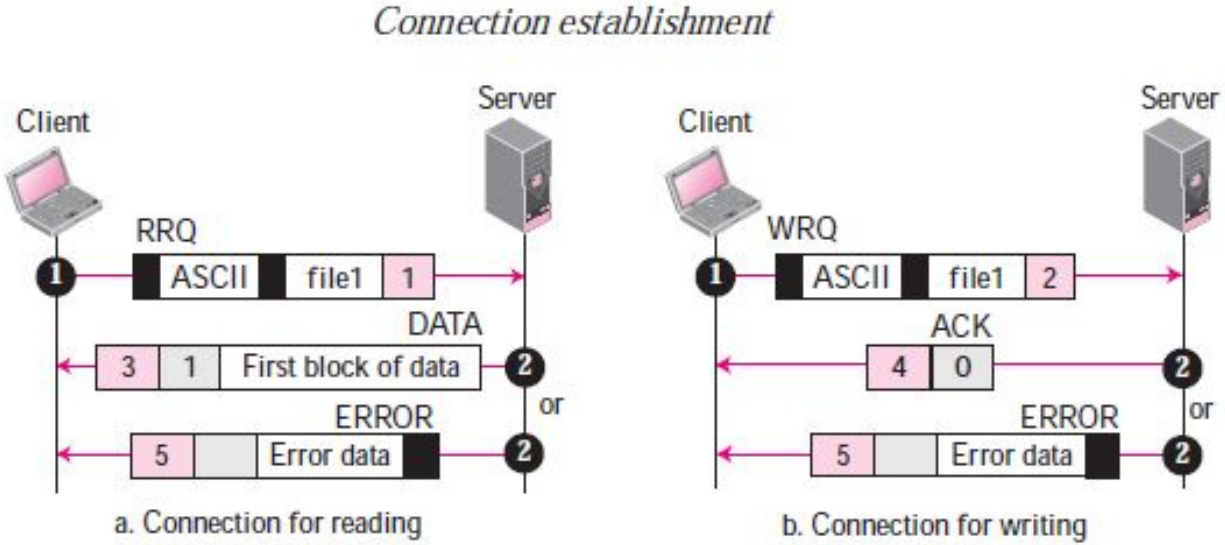
❑ Reading.

- To establish a connection for reading, the TFTP client sends the RRQ message.
- The name of the file and the transmission mode is defined in this message.
- If the server can transfer the file, it responds positively with a DATA message containing the first block of data.
- If there is a problem, such as difficulty in opening the file or permission restriction, the server responds negatively by sending an ERROR message.

❑ Writing.

- To establish a connection for writing, the TFTP client uses the WRQ message.
- The name of the file and the transmission mode is defined in this message.
- If the server can accept a copy of the file, it responds positively with an ACK message using a value of 0 for the block number.
- If there is any problem, the server responds negatively by sending an ERROR message.

Connection


Connection establishment
Client
Server
RRQ
ASCII
file1
1
DATA
3
1
First block of data
2
or
ERROR
5
Error data
2
a. Connection for reading
Client
Server
WRQ
ASCII
file1
2
ACK
4
0
2
or
ERROR
5
Error data
2
b. Connection for writing

Connection

b) Connection Termination

- After the entire file is transferred, the connection must be terminated.
- As mentioned previously, TFTP does not have a special message for termination.
- Termination is accomplished by sending the last block of data, which is less than 512 bytes.

3. Data Transfer

- The data transfer phase occurs between connection establishment and termination.
- TFTP uses the services of UDP, which is unreliable.
- The file is divided into blocks of data, in which each block except the last one is exactly 512 bytes.
- The last block must be between 0 and 511 bytes.
- TFTP can transfer data in ASCII or binary format.
- UDP does not have any mechanism for flow and error control.
- TFTP has to create a flow- and error-control mechanism to transfer a file made of continuous blocks of data.

Data Transfer

a) Flow Control

- TFTP sends a block of data using the DATA message and waits for an ACK message.
- If the sender receives an acknowledgment before the time-out, it sends the next block.
- Thus, flow control is achieved by numbering the data blocks and waiting for an ACK before the next data block is sent.

Retrieve a File

- When the client wants to retrieve (read) a file, it sends the RRQ message.
- The server responds with a DATA message sending the first block of data (if there is no problem) with a block number of 1.

Store a File

- When the client wants to store (write) a file, it sends the WRQ message.
- The server responds with an ACK message (if there is no problem) using 0 for the block number.
- After receiving this acknowledgment, the client sends the first data block with a block number of 1.

Data Transfer

b) Error Control

- The TFTP error-control mechanism is different from those of other protocols.
- It is symmetric, which means that the sender and the receiver both use time-outs.
- The sender uses a time-out for data messages; the receiver uses a time-out for acknowledgment messages.
- If a data message is lost, the sender retransmits it after time-out expiration.
- If an acknowledgment is lost, the receiver retransmits it after time-out expiration.
- This guarantees a smooth operation.
- Error control is needed in four situations: a damaged message, a lost message, a lost acknowledgment, or a duplicated message.

Data Transfer

Damaged Message

- There is no negative acknowledgment.
- If a block of data is damaged, it is detected by the receiver and the block is discarded.
- The sender waits for the acknowledgment and does not receive it within the time-out period.
- The block is then sent again; there is no checksum field in the DATA message of TFTP.
- The receiver can detect data corruption is through the checksum field of the UDP user datagram.

Lost Message

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

Lost Acknowledgment

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

Duplicate Message

- Duplication of blocks can be detected by the receiver through block number.
- If a block is duplicated, it is simply discarded by the receiver.

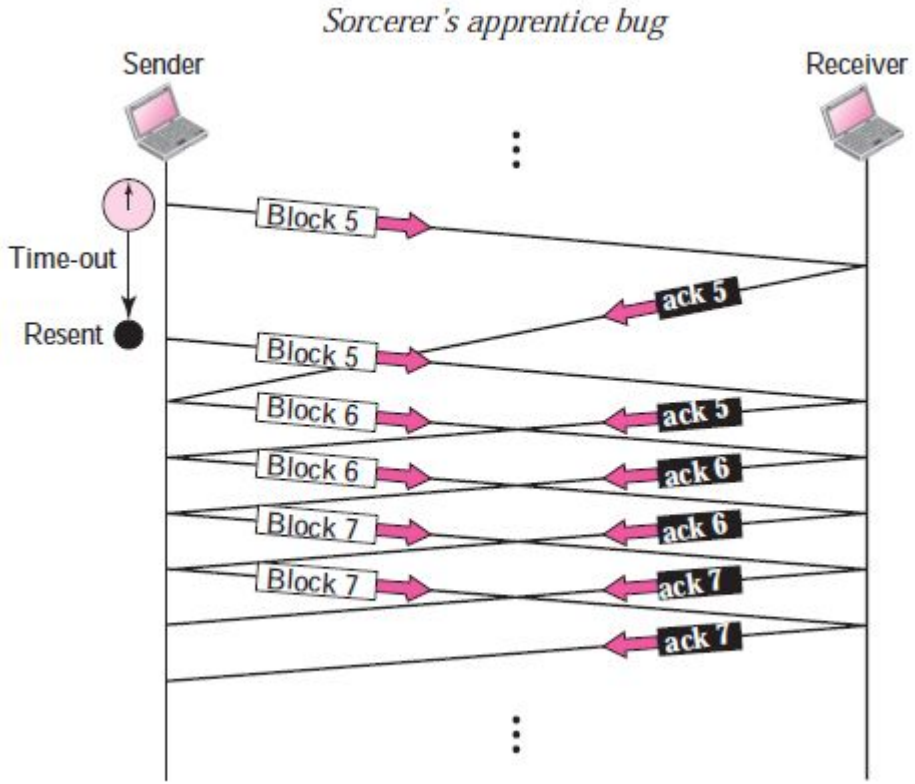
Data Transfer

c) Sorcerer's Apprentice Bug

- Although the flow- and error-control mechanism is symmetric in TFTP, it can lead to a problem known as the **sorcerer's apprentice bug**, named for the cartoon character who inadvertently conjures up a mop that continuously replicates itself.
- This will happen if the ACK message for a packet is not lost but delayed.
- In this situation, every succeeding block is sent twice and every succeeding acknowledgment is received twice.

Data Transfer

- The acknowledgment for the fifth block is delayed.
- After the time-out expiration, the sender retransmits the fifth block, which will be acknowledged by the receiver again.
- The sender receives two acknowledgments for the fifth block, which triggers it to send the sixth block twice.
- The receiver receives the sixth block twice and again sends two acknowledgments, which results in sending the seventh block twice. And so on.



4. UDP Ports

- When a process uses the services of UDP, the server process issues a passive open on the well-known port and waits for the client process to issue an active open on an ephemeral port.
- After the connection is established, the client and server communicate using these two ports.
- TFTP follows a different set of steps because the communication between a client TFTP and a server TFTP can be quite lengthy (seconds or even minutes).
- If a TFTP server uses the well-known port 69 to communicate with a single client, no other clients can use these services during that time.
- The solution to this problem, is to use the well-known port for the initial connection and an ephemeral port for the remaining communication.
- The steps are as follows:
  1. The server passively opens the connection using the well-known port 69.
  2. A client actively opens a connection using an ephemeral port for the source port and the well-known port 69 for the destination port. This is done through the RRQ message or the WRQ message.

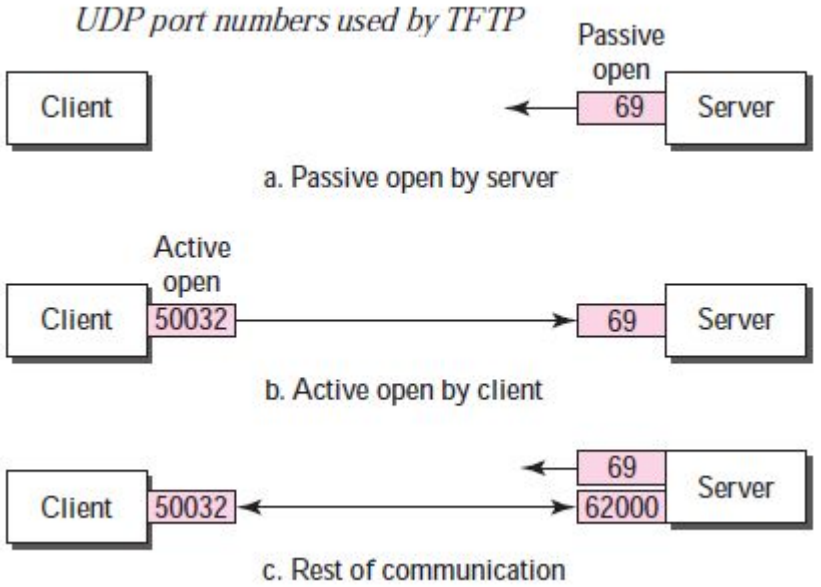
UDP Ports

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

It sends the DATA or ACK or ERROR message using these ports.

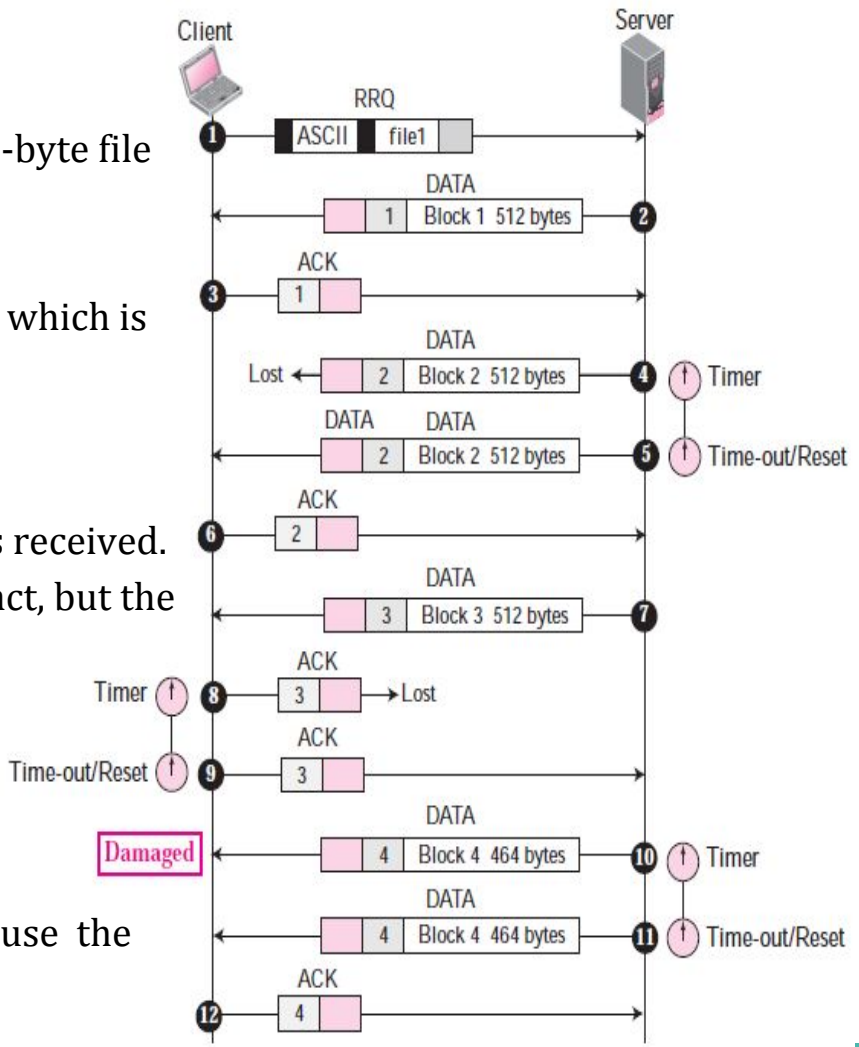
This frees the wellknown port (69) for use by other clients.

When the client receives the first message from the server, it uses its own ephemeral port and the ephemeral port sent by the server for future communication.



5. TFTP Example

- The client wants to retrieve a copy of the contents of a 2,000-byte file called file1.
- The client sends an RRQ message.
- The server sends the first block, carrying the first 512 bytes, which is received intact and acknowledged.
- The first two messages are the connection establishment.
- The second block, carrying the second 512 bytes, is lost.
- After the time-out, the server retransmits the block, which is received.
- The third block, carrying the third 512 bytes, is received intact, but the ack is lost.
- After the time-out, the receiver retransmits the ack.
- The last block, carrying the remaining 464 bytes, is received damaged, so the client simply discards it.
- After the time-out, the server retransmits the block.
- This message is considered the connection termination because the block carries fewer than 512 bytes.



6. TFTP Options

- An extension to the TFTP protocol that allows the appending of options to the RRQ and WRQ messages has been proposed.
- The options are mainly used to negotiate the size of the block and possibly the initial sequence number.
- Without the options the size of a block is 512 bytes except for the last block.
- The negotiation can define a size of block to be any number of bytes so long as the message can be encapsulated in a UDP user datagram.
- A new type of message, **option acknowledgment (OACK)**, to let the other party accept or reject the options, has also been proposed.

7. Security

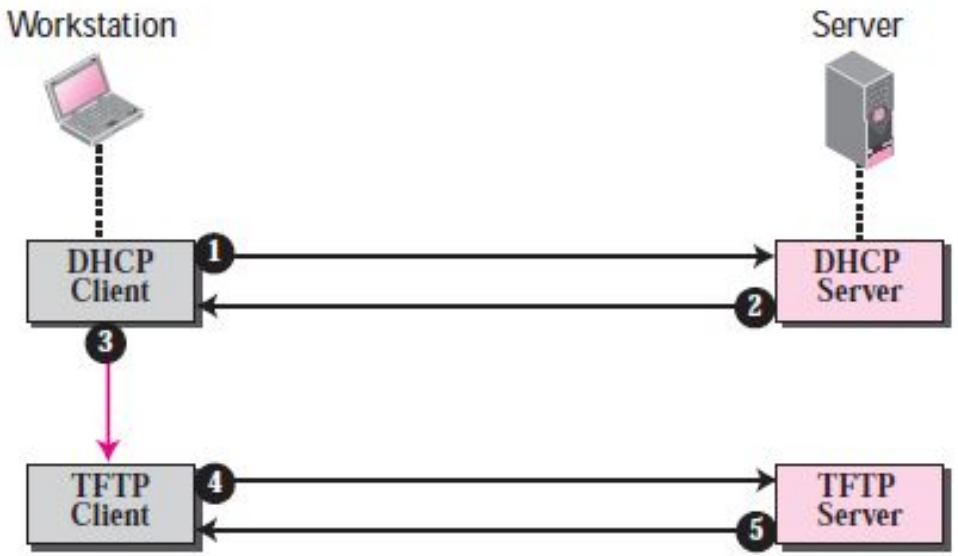
- One important point about TFTP is that there is no provision for security:
- There is no user identification or password.
- Today, however, precautions must be taken to prevent hackers from accessing files.
- One security measure is to limit the access of TFTP to noncritical files.
- One way to achieve minimal security is to implement security in the router close to a TFTP server, which would allow only certain hosts to access the server.

8. Applications

- TFTP is very useful for basic file transfer where security is not a big issue.
- It can be used to initialize devices such as bridges or routers.
- **Its main application is in conjunction with the DHCP.**
- TFTP requires only a small amount of memory and uses only the services of UDP and IP.
- It can easily be configured in ROM (or PROM).
- When the station is powered on, TFTP will be connected to a server and can download the configuration files from there.
- The powered-on station uses the DHCP client to get the name of the configuration file from the DHCP server.
- The station then passes the name of the file to the TFTP client to get the contents of the configuration file from the TFTP server.

Applications

Use of TFTP with DHCP



TFTP-SUMMARY

- **Messages- RRQ, WRQ, DATA, ACK, and ERROR**
- **Connection**
  - **Connection Establishment**
  - **Connection Termination**
- **Data Transfer**
  - **Flow control**
  - **Error control**
  - **Sorcerer's Apprentice Bug**
- **UDP Ports**
- **TFTP Example**
- **TFTP Options**
- **Security**
- **Applications**

Difference-FTP vs TFTP

- File transfer Protocol
- Authentication is required
- Uses TCP service-Connection oriented
- Establishes two connection
  Data (TCP port-21)
  Control (TCP port-20)
- More complex

- Trivial file transfer protocol
- No authentication is required
- Uses UDP service-connection less
- Establishes single connection (UDP Port -69)
- Less complex
- Very simple

References (finishing slides covering references for all the topics)

1. Douglas E. Comer, Internetworking with TCP/IP, Principles, protocols, and architecture,Vol 1 5th Edition,2006 ISBN: 0131876716, ISBN: 978-0131876712 **(Ref 2 in syllabus)**
2. https://slideplayer.com/slide/13911208/
3. http://www.csun.edu/~jeffw/Semesters/2006Fall/COMP429/Presentations/Ch25-FTP.pdf
4. https://study.com/academy/lesson/testing-an-ftp-connection.html
5. www.afternerd.com/blog/smtp