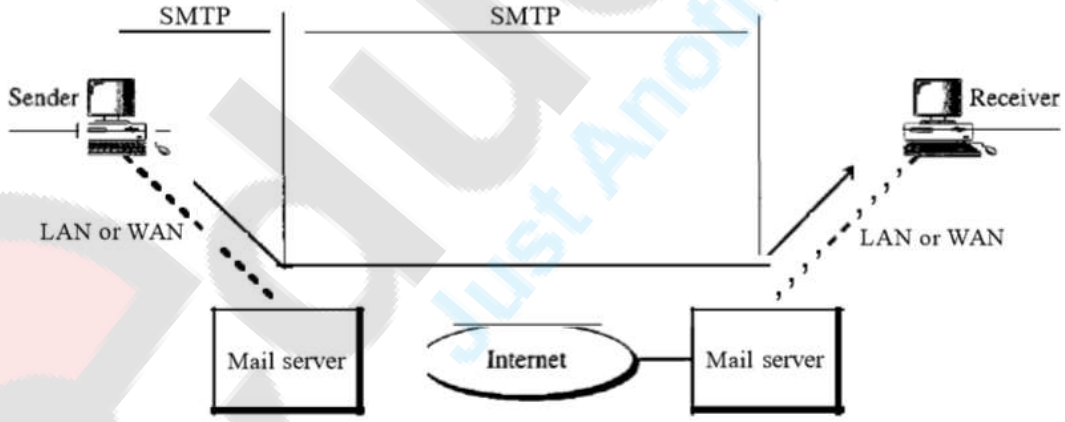
Unit-4

1) Short Note on SMTP? "(Nov 2013)"

Answer: -

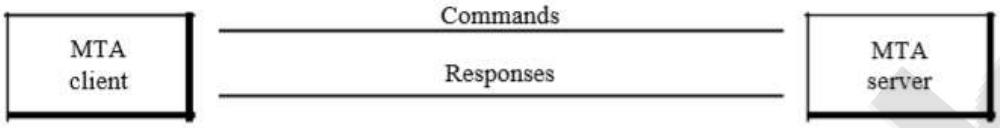
1) Message Transfer Agent: SMTP
2) The actual mail transfer is done through message transfer agents. To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA.
3) The formal protocol that defines the MTA client and server in the Internet is called the Simple Mail Transfer Protocol (SMTP).
4) As we said before, two pairs of MTA client/server programs are used in the most common situation (fourth scenario). Figure shows the range of the SMTP protocol in this scenario.
5) SMTP is used two times, between the sender and the sender's mail server and between the two mail servers. As we will see shortly, another protocol is needed between the mail server and the receiver.
6) SMTP simply defines how commands and responses must be sent back and forth. Each network is free to choose a software package for implementation. We discuss the mechanism of mail transfer by SMTP in the remainder of the section.

SMTP range



7) Commands and Responses:- SMTP uses commands and responses to transfer messages between an MTA client and an MTA server.

Commands and responses



8) Each command or reply is terminated by a two-character (carriage return and line feed) end-of-line token.
9) **Commands:** -Commands are sent from the client to the server. The format of a command is shown in Figure. It consists of a keyword followed by zero or more arguments. SMTP defines 14 commands. The first five are mandatory; every implementation must support these five commands. The next three are often used and highly recommended. The last six are seldom used.
10) **Responses**: -Responses are sent from the server to the client. A response is a three digit code that may be followed by additional textual information. Table lists some of the responses.

Table 26.7 *Commands*

Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPTTO	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

Table 26.8 *Responses*

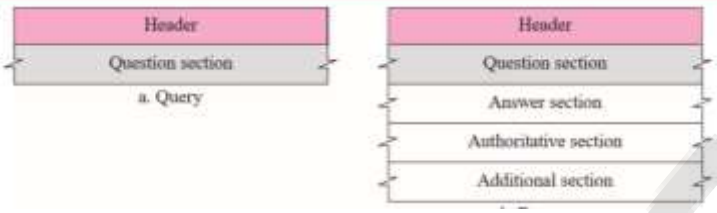
Code	*Description*
	Positive Completion Reply
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
	Positive Intermediate Reply
354	Start mail input
	Transient Negative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

2) List types of messages in DNS. Explain?

Answer: -

1) DNS has two types of messages: query and response. Both types have the same format. The query message consists of a header and question records; the response message consists of a header, question records, answer records, authoritative records, and additional records.

Figure 19.14 *Query and response messages*



2) **Header**: Both query and response messages have the same header format with some fields set to zero for the query messages. The header is 12 bytes.

Figure 19.15 *Header format*

3) The header fields are as follows:

A) Identification: This is a 16-bit field used by the client to match the response with the query. The client uses a different identification number each time it sends a query. The server duplicates this number in the corresponding response.

B) Flags: This is a 16-bit field consisting of the subfields.

Figure 19.16 *Flags field*

4) A brief description of each flag subfield follows.

A) **QR (query/response).** This is a 1-bit subfield that defines the type of message. If it is 0, the message is a query. If it is 1, the message is a response.

B) **OpCode**. This is a 4-bit subfield that defines the type of query or response (0 if standard, 1 if inverse, and 2 if a server status request).

C) **AA (authoritative answer).** This is a 1-bit subfield. When it is set (value of 1) it means that the name server is an authoritative server. It is used only in a response message.

D) **TC (truncated).** This is a 1-bit subfield. When it is set (value of 1), it means that the response was more than 512 bytes and truncated to 512. It is used when DNS uses the services of UDP (see Section 19.8 on Encapsulation).

E) **RD (recursion desired).** This is a 1-bit subfield. When it is set (value of 1) it means the client desires a recursive answer. It is set in the query message and repeated in the response message.

F) **RA (recursion available).** This is a 1-bit subfield. When it is set in the response, it means that a recursive response is available. It is set only in the response message.

G) **Reserved.** This is a 3-bit subfield set to 000. h. rCode. This is a 4-bit field that shows the status of the error in the response. Of course, only an authoritative server can make such a judgment. Table 19.2 shows the possible values for this field.

Table 19.2 *Values of rCode*

Value	*Meaning*	*Value*	*Meaning*
0	No error	4	Query type not supported
1	Format error	5	Administratively prohibited
2	Problem at name server	6–15	Reserved
3	Domain reference problem		

❑ **Number of question records**. This is a 16-bit field containing the number of queries in the question section of the message.

❑ **Number of answer records**. This is a 16-bit field containing the number of answer records in the answer section of the response message. Its value is zero in the query message.

❑ **Number of authoritative records.** This is a 16-bit field containing the number of authoritative records in the authoritative section of a response message. Its value is zero in the query message.

❑ **Number of additional records**. This is a 16-bit field containing the number of additional records in the additional section of a response message. Its value is zero in the query message.

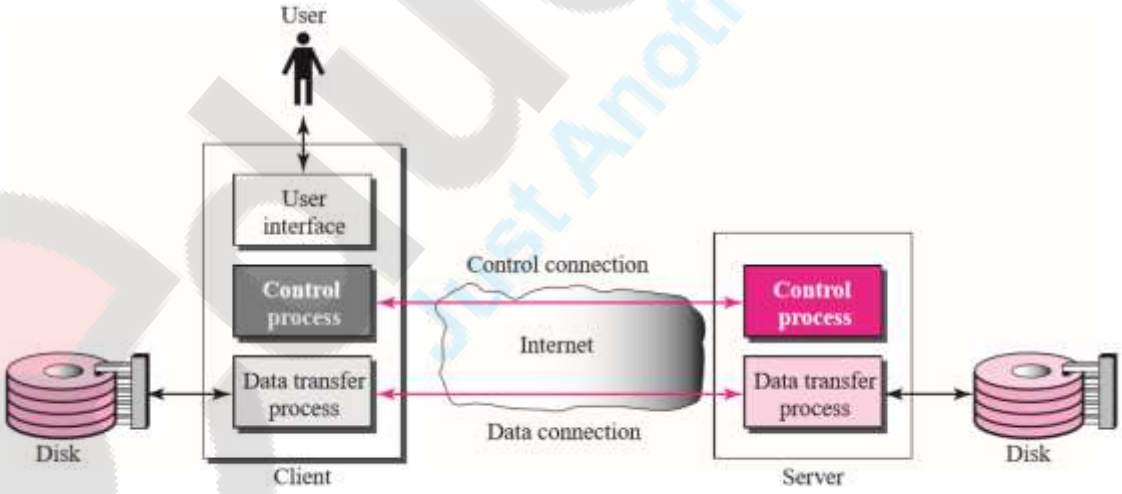
5) **Question Section** This is a section consisting of one or more question records. It is present on both query and response messages. We will discuss the question records in a following section.
6) **Answers Section** This is a section consisting of one or more resource records. It is present only on response messages. This section includes the answer from the server to the client (resolver). We will discuss resource records in a following section.
7) **Authoritative Section** This is a section consisting of one or more resource records. It is present only on response messages. This section gives information (domain name) about one or more authoritative servers for the query.
8) **Additional Information Section** This is a section consisting of one or more resource records. It is present only on response messages. This section provides additional information that may help the resolver. For example, a server may give the domain

name of an authoritative server to the resolver in the authoritative section, and include the IP address of the same authoritative server in the additional information section.

3) Short Note on FTP?

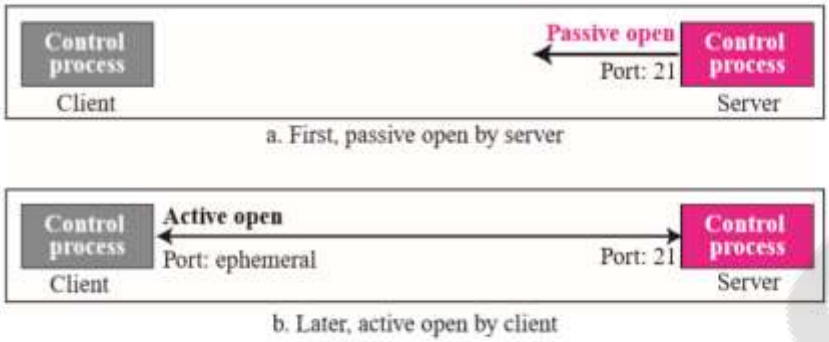
Answer: -

1) File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.
2) 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.
3) FTP uses two well-known TCP ports: **Port 21** is used for the control connection, and **port 20** is used for the data connection.
4) 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.
5) 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.



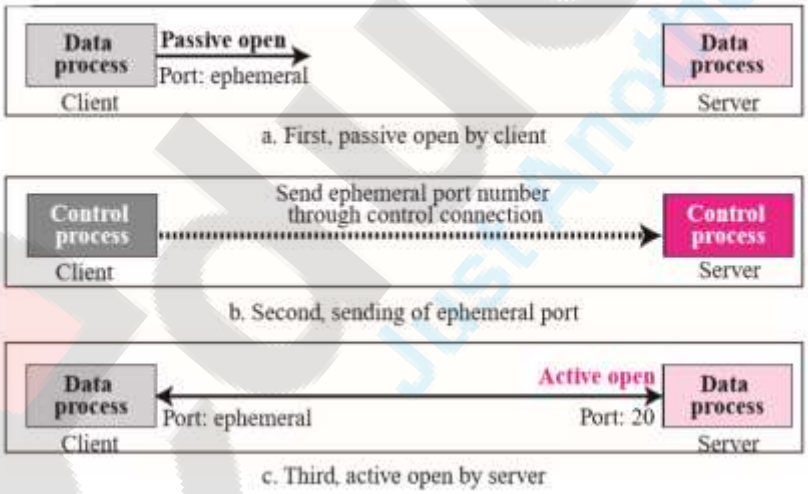
6) **Control Connection**

Opening the control connection



7) 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 minimizing delay because this is an interactive connection between a user (human) and a server.
8) Data Connection

Creating the data connection



9) 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 (we will discuss this command shortly.

3. The server receives the port number and issues an active open using the well-known port 20 and the received ephemeral port number.

4) Short Note on MIME?

Answer: -

1) MIME Electronic mail has a simple structure. Its simplicity, however, comes with a price. It can send messages only in NVT 7-bit ASCII format. In other words, it has some limitations. It cannot be used for languages other than English (such as French, German, Hebrew, Russian, Chinese, and Japanese). Also, it cannot be used to send binary files or video or audio data.
2) **Multipurpose Internet Mail Extensions (MIME)** is a supplementary protocol that allows non-ASCII data to be sent through e-mail. MIME transforms non-ASCII data at the sender site to NVT ASCII data and delivers it to the client MTA to be sent through the Internet. The message at the receiving site is transformed back to the original data.
3) We can think of MIME as a set of software functions that transforms non-ASCII data to ASCII data and vice versa, as shown in Figure –

4) **MIME Headers** – MIME defines five headers that can be added to the original e-mail header section to define the transformation parameters:
   a) MIME-Version: - This header defines the version of MIME used. The current version is 1.1.
   b) Content-Type: - This header defines the type of data used in the body of the message. The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters. MIME allows seven different types of data, listed in Table –

Type	Subtype	Description
Text	Plain	Unformatted
	HTML	HTML format (see Appendix E)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
Message	RFC822	Body is an encapsulated 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 (eight-bit bytes)

c) Content-Transfer-Encoding: - Content-Transfer-Encoding – This header defines the method used to encode the messages into 0s and 1s for transport: Content-Transfer-Encoding: <type> The five types of encoding methods are listed in Table

Type	Description
7bit	NVT ASCII characters and short lines
8bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data are encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters are encoded as an equal sign plus an ASCII code

d) Content-Id: - This header uniquely identifies the whole message in a multiple message environment.

e) Content-Description: - This header defines whether the body is image, audio, or video.

Figure – shows the MIME headers.

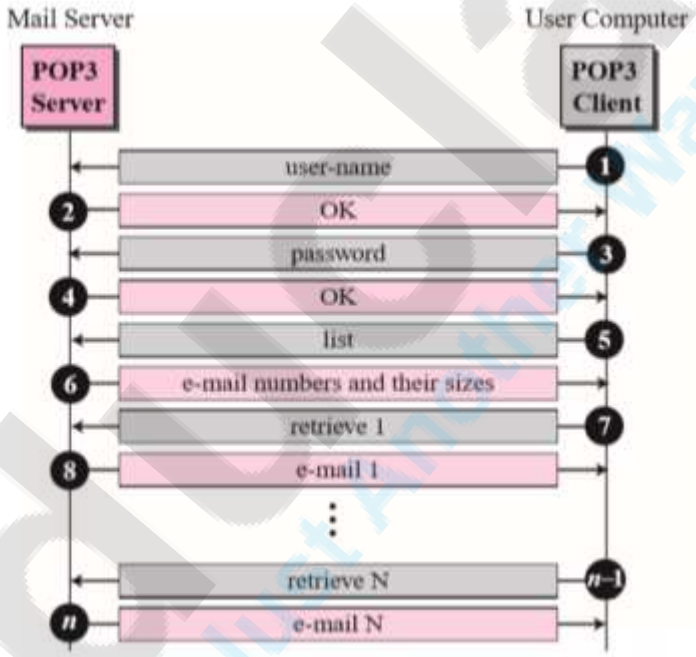
5) Short Note on POP3?

Answer: -

1) Post Office Protocol, version 3 (POP3) is simple and limited in functionality. The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server. Mail access starts with the client when the user needs to download its e-mail from the mailbox on the mail server. The client opens a connection to the server on TCP port 110. It then sends its user name and password to access the mailbox. The user can then list and retrieve the mail messages, one by one. Figure shows an example of downloading using POP3. ‘
2) POP3 begins when the user agent (the client) opens a TCP connection to the mail server (the server) on port 110. With the TCP connection established, POP3 progresses through three phases: authorization, transaction, and update. During the first phase, authorization, the user agent sends a username and a password (in the

clear) to authenticate the user. During the second phase, transaction, the user agent retrieves messages; also during this phase, the user agent can mark messages for deletion, remove deletion marks, and obtain mail statistics. The third phase, update, occurs after the client has issued the quit command, ending the POP3 session; at this time, the mail server deletes the messages that were marked for deletion.

3) In a POP3 transaction, the user agent issues commands, and the server responds to each command with a reply. There are two possible responses: +OK (sometimes followed by server-to-client data), used by the server to indicate that the previous command was fine; and -ERR, used by the server to indicate that something was wrong with the previous command.
4) The authorization phase has two principal commands: user <username> and pass <password>. To illustrate these two commands, we suggest that you Telnet directly into a POP3 server, using port 110, and issue these commands.



5) POP3 has two modes: the delete mode and the keep mode.
6) In the delete mode, the mail is deleted from the mailbox after each retrieval. In the keep mode, the mail remains in the mailbox after retrieval. The delete mode is normally used when the user is working at her permanent computer and can save and organize the received mail after reading or replying.
7) The keep mode is normally used when the user accesses her mail away from her primary computer (e.g., a laptop). The mail is read but kept in the system for later retrieval and organizing.
8) A problem with this download-and-delete mode is that the recipient, Bob, may be nomadic and may want to access his mail messages from multiple machines, for example, his office PC, his home PC, and his portable computer. The downloadand-

delete mode partitions Bob's mail messages over these three machines; in particular, if Bob first reads a message on his office PC, he will not be able to reread the message from his portable at home later in the evening. In the download-and keep mode, the user agent leaves the messages on the mail server after downloading them. In this case, Bob can reread messages from different machines; he can access a message from work and access it again later in the week from home.

9) During a POP3 session between a user agent and the mail server, the POP3 server maintains some state information; in particular, it keeps track of which user messages have been marked deleted. However, the POP3 server does not carry state information across POP3 sessions. This lack of state information across sessions greatly simplifies the implementation of a POP3 server.

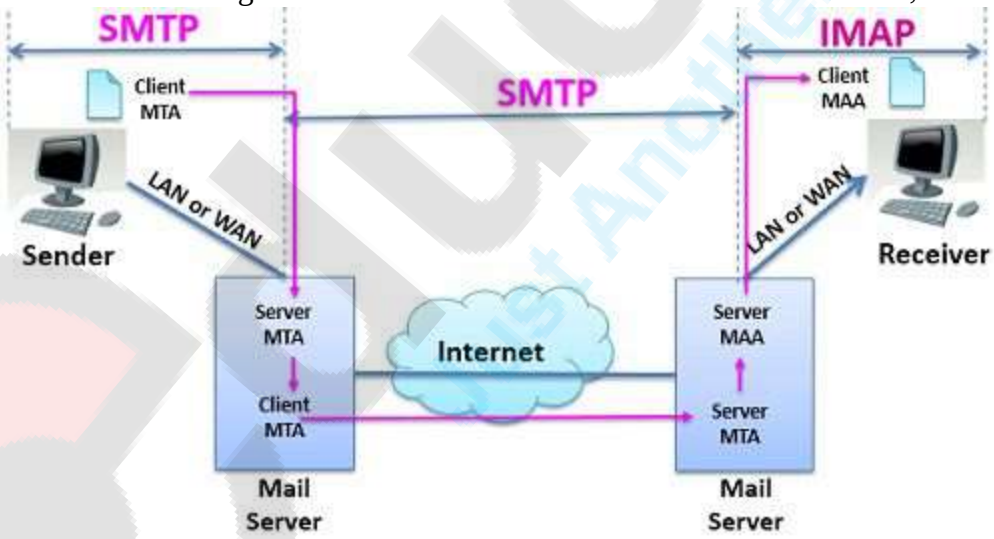
6) Short Note on IMAP?

Answer: -

1) Definition of IMAP

2) Internet Mail Accessing Protocol (IMAP) is also a **mail accessing agent** like POP3. But it is more powerful, has more features and is more complex than POP3. The POP3 protocol was found deficient in many ways. So IMAP is introduced to overcome these deficiencies.

3) POP3 does not allow a user to organize mails on the mailbox. The user cannot create different folders on the server. The user cannot partially check the content of emails before downloading them. The user has to download an email to read it, in POP.



4) IMAP is used to access the mail from the mailbox at the mail server. Using IMAP the user can check the **email header** before downloading it. The user is able to check the content of the email for a **particular string of character** that too before downloading the email.

5) In case, the bandwidth is limited, using IMAP the user can **partially download** the mail. It is useful in case the email contains multimedia with high bandwidth requirement. The user can create, delete or rename the mailboxes on the server. The user can also create a hierarchy of these mailboxes in a folder. This is how IMAP is more powerful than POP3 protocol.

7) Short Note on HTTP?

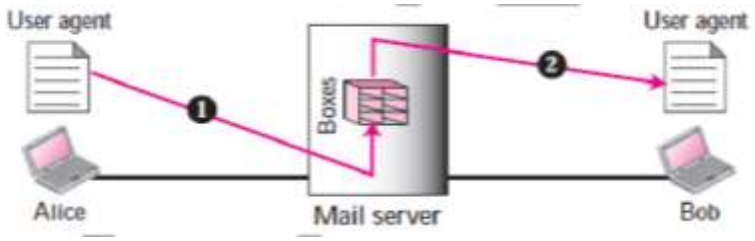
Answer: -

1) HTTP is a **Hyper Text Transfer Protocol**. It helps in accessing data from the **World Wide Web**. HTTP works similar to the combine functions of FTP and SMTP. Similar to the functioning of **FTP** because like FTP, it transfers file using service of **TCP**. But it uses only one TCP connection i.e. **data connection**, no separate Control Connection is used in HTTP. HTTP uses services of TCP on port no **80**.
2) HTTP is similar to **SMTP** because the data transferred between client and server appears like SMTP **messages**. But HTTP messages are not destined to the humans for reading; they are interpreted and read by the web server and web browser. Unlike SMTP messages, HTTP messages are delivered immediately instead of storing and then forwarding.
3) The commands from the client side are sent in a **request message** to the web server. The web server sends the requested content in a **response message**. The HTTP does not provide any security, to enable security it is run over the **Secure Socket layer**.

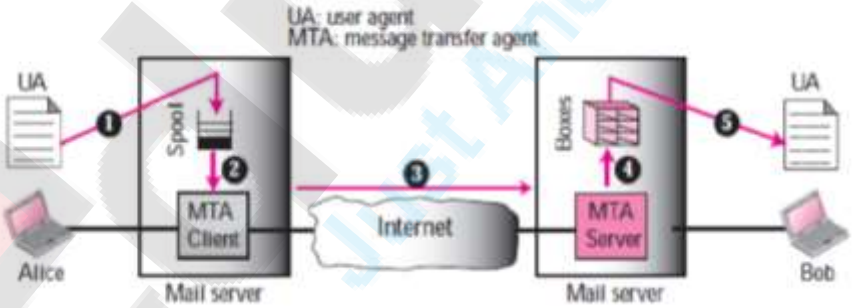
8) Explain architecture of E-mail and give four scenario?

Answer: -

1) **First Scenario** –
   A) In the first scenario, the sender and the receiver of the e-mail are users (or application programs) on the same mail server; they are directly connected to a shared mail server. The administrator has created one mailbox for each user where the received messages are stored.
   B) A mailbox is part of a local hard drive, a special file with permission restrictions. Only the owner of the mailbox has access to it. When Alice needs to send a message to Bob, she runs a user agent (UA) program to prepare the message and store it in Bob's mailbox.
   C) The message has the sender and recipient mailbox addresses (names of files). Bob can retrieve and read the contents of his mailbox at his convenience using a user agent. Figure – shows the concept.
   D) This is similar to the traditional memo exchange between employees in an office. There is a mail room where each employee has a mailbox with his or her name on it.
   E) When Alice needs to send a memo to Bob, she writes the memo and inserts it into Bob's mailbox. When Bob checks his mailbox, he finds Alice's memo and reads it.
   F) **When the sender and the receiver of an e-mail are on the same mail server, we need only two user agents.**



2) **Second Scenario** –
   A) In the second scenario, the sender and the receiver of the e-mail are users (or application programs) on two different mail servers. The message needs to be sent over the Internet. Here we need user agents (UAs) and message transfer agents (MTAs) as shown in Figure –
   B) Alice needs to use a user agent program to send her message to the mail server at own site. The mail server at her site uses a queue (spool) to store messages waiting to be sent. Bob also needs a user agent program to retrieve messages stored in the mailbox of the system at his site.
   C) The message, however, needs to be sent through the Internet from Alice's site to Bob's site. Here two message transfer agents are needed: one client and one server. Like most client-server programs on the Internet, the server needs to run all of the time because it does not know when a client will ask for a connection.
   D) The client, on the other hand, can be triggered by the system when there is a message in the queue to be sent.
   E) When the sender and the receiver of an e-mail are on different mail servers, we need two UAs and a pair of MTAs (client and server).



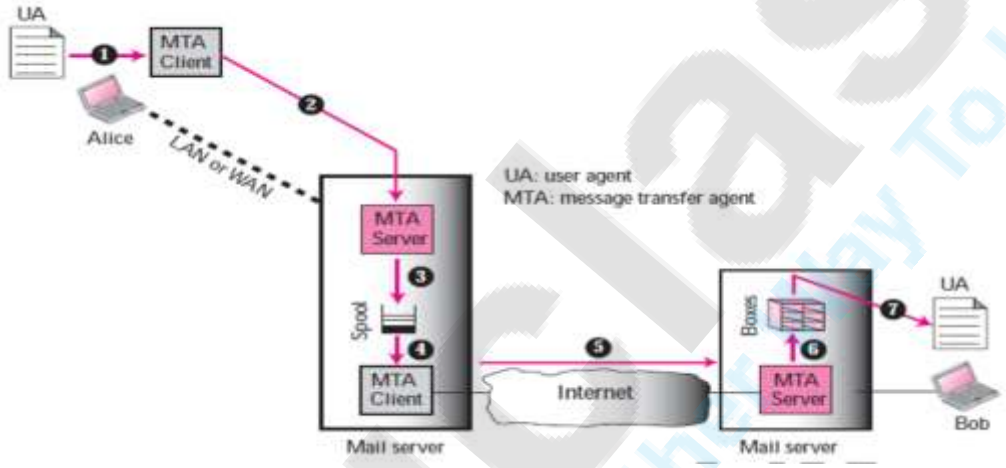
3) **Third Scenario**
   A) Figure – shows the third scenario. Bob, as in the second scenario, is directly connected to his mail server. Alice, however, is separated from her mail server. Alice is either connected to the mail server via a point-to-point WAN—such as a dial-up modem, a DSL, or a cable modem—or she is connected to a LAN in an organization that uses one mail server for handling e-mails; all users need to send their messages to this mail server.
   B) Alice still needs a user agent to prepare her message. She then needs to send the message through the LAN or WAN. This can be done through a pair of message

transfer agents (client and server). Whenever Alice has a message to send, she calls the user agent which, in turn, calls the MTA client.

C) The MTA client establishes a connection with the MTA server on the system, which is running all the time. The system at Alice's site queues all messages received. It then uses an MTA client to send the messages to the system at Bob's site; the system receives the message and stores it in Bob's mailbox.

D) At his convenience, Bob uses his user agent to retrieve the message and reads it. Note that we need two pairs of MTA client-server programs.

E) 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).

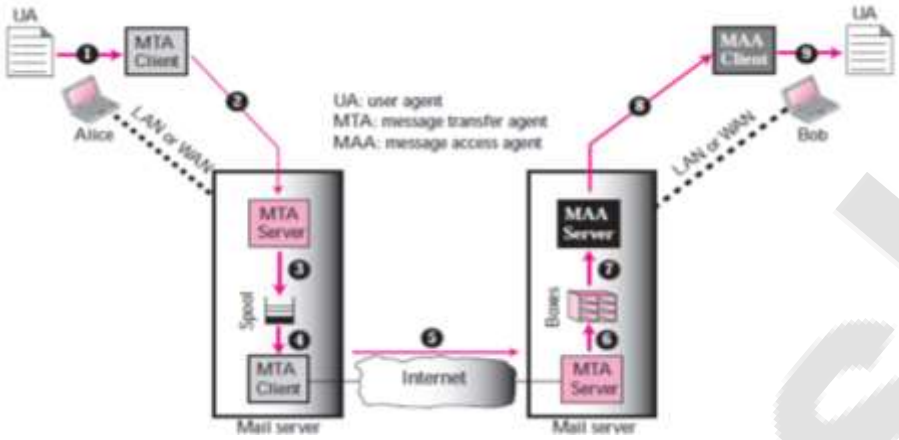


4) Fourth Scenario –

A) In the fourth and most common scenario, Bob is also connected to his mail server by a WAN or a LAN. After the message has arrived at Bob's mail server, Bob needs to retrieve it. Here, we need another set of client-server agents, which we call message access agents (MAAs). Bob uses an MAA client to retrieve his messages.

B) The client sends a request to the MAA server, which is running all the time, and requests the transfer of the messages. The situation is shown in Figure –

C) 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 (client and server), and a pair of MAAs (client and server). This is the most common situation today.



9) Short Note on Telnet?

Answer: -

1) Telnet, defined in RFC 854, is a popular application-layer protocol used for remote login. It runs over TCP and is designed to work between any pair of hosts.
2) Telnet is an interactive application. We discuss a Telnet example here, as it nicely illustrates TCP sequence and acknowledgment numbers. We note that many users now prefer to use the SSH protocol rather than Telnet, since data sent in a Telnet connection (including passwords!) is not encrypted, making Telnet vulnerable to eavesdropping attacks.
3) Host A initiates a Telnet session with Host B. Because Host A initiates the session, it is labelled the client, and Host B is labelled the server. Each character typed by the user (at the client) will be sent to the remote host; the remote host will send back a copy of each character, which will be displayed on the Telnet user's screen. This "echo back" is used to ensure that characters seen by the Telnet user have already been received and processed at the remote site. Each character thus traverses the network twice between the time the user hits the key and the time the character is displayed on the user's monitor.

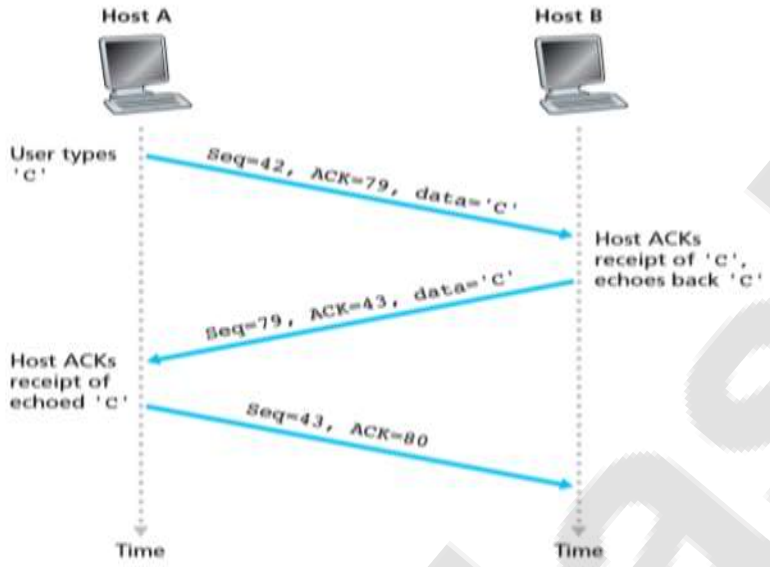


Figure 3.31 ♦ Sequence and acknowledgment numbers for a simple Telnet application over TCP

4) The user types a single letter, 'C,'and then grabs a coffee. Let's examine the TCP segments that are sent between the client and server. As shown in Figure 3.31, we suppose the starting sequence numbers are 42 and 79 for the client and server, respectively. Recall that the sequence number of a segment is the sequence number of the first byte in the data field. Thus, the first segment sent from the client will have sequence number 42; the first segment sent from the server will have sequence number 79. Recall that the acknowledgment number is the sequence number of the next byte of data that the host is waiting for. After the TCP connection is established but before any data is sent, the client is waiting for byte 79 and the server is waiting for byte 42.
5) The first segment is sent from the client to the server, containing the 1-byte ASCII representation of the letter 'C' in its data field. This first segment also has 42 in its sequence number field, as we just described. Also, because the client has not yet received any data from the server, this first segment will have 79 in its acknowledgment number field.
6) The second segment is sent from the server to the client. It serves a dual purpose. First it provides an acknowledgment of the data the server has received. By putting 43 in the acknowledgment field, the server is telling the client that it has successfully received everything up through byte 42 and is now waiting for bytes 43 onward. The second purpose of this segment is to echo back the letter 'C.' Thus; the second segment has the ASCII representation of 'C'in its data field. This second segment has the sequence number 79, the initial sequence number of the server-to client data flow of this TCP connection, as this is the very first byte of data that the server is sending. Note that the acknowledgment for client-to-server data is carried in a segment

carrying server-to-client data; this acknowledgment is said to be piggybacked on the server-to-client data segment.

7) The third segment is sent from the client to the server. Its sole purpose is to acknowledge the data it has received from the server. (Recall that the second segment contained data—the letter 'C'—from the server to the client.) This segment has an empty data field (that is, the acknowledgment is not being piggybacked with any client-to-server data). The segment has 80 in the acknowledgment number field because the client has received the stream of bytes up through byte sequence number 79 and it is now waiting for bytes 80 onward. You might think it odd that this segment also has a sequence number since the segment contains no data. But because TCP has a sequence number field, the segment needs to have some sequence number.

10) Short Note on SNMP?

Answer: -

1) The Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using the TCPIIP protocol suite. It provides a set of fundamental operations for monitoring and maintaining an internet.
2) SNMP uses the concept of manager and agent. That is, a manager, usually a host, controls and monitors a set of agents, usually routers.
3) SNMP is an application-level protocol in which a few manager stations control a set of agents. The protocol is designed at the application level so that it can monitor devices made by different manufacturers and installed on different physical networks. In other words, SNMP frees management tasks from both the physical characteristics of the managed devices and the underlying networking technology. It can be used in a heterogeneous internet made of different LANs and WANs connected by routers made by different manufacturers.

4) **Managers and Agents**
   A) A management station, called a manager, is a host that runs the SNMP client program.
   B) A managed station, called an agent, is a router (or a host) that runs the SNMP server program. Management is achieved through simple interaction between a manager and an agent.

C) The agent keeps performance information in a database. The manager has access to the values in the database. For example, a router can store in appropriate variables the number of packets received and forwarded.

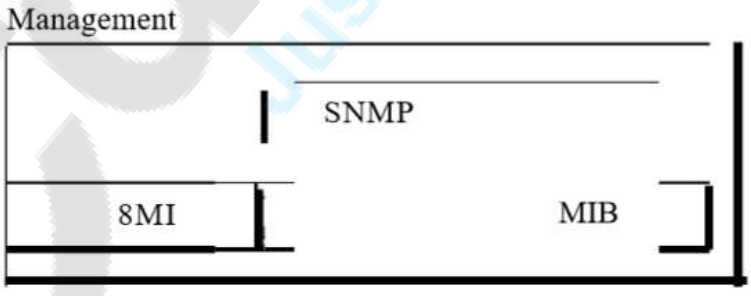
D) The manager can fetch and compare the values of these two variables to see if the router is congested or not. The manager can also make the router perform certain actions. For example, a router periodically checks the value of a reboot counter to see when it should reboot itself. It reboots itself, for example, if the value of the counter is O.

E) The manager can use this feature to reboot the agent remotely at any time. It simply sends a packet to force a 0 value in the counter. Agents can also contribute to the management process. The server program running on the agent can check the environment, and if it notices something unusual, it can send a warning message, called a trap, to the manager. In other words, management with SNMP is based on three basic ideas:

- A manager checks an agent by requesting information that reflects the behaviour of the agent.
- A manager forces an agent to perform a task by resetting values in the agent database.
- An agent contributes to the management process by warning the manager of an unusual situation.

5) **Management Components**: - SNMP uses two other protocols: Structure of Management Information (SMI) and Management Information Base (MIB). In other words, management on the Internet is done through the cooperation of the three protocols SNMP, SMI, and MIB.

Components ofnetwork management on the Internet



A) **Role of SNMP:** -SNMP has some very specific roles in network management. It defines the format of the packet to be sent from a manager to an agent and vice versa. It also interprets the result and creates statistics (often with the help of other management software). The packets exchanged contain the object (variable)

names and their status (values). SNMP is responsible for reading and changing these values.

B) **Role of SMI: -**To use SNMP, we need rules. We need rules for naming objects. This is particularly important because the objects in SNMP form a hierarchical structure (an object may have a parent object and some children objects). Part of a name can be inherited from the parent. We also need rules to define the type of the objects. What types of objects are handled by SNMP? Can SNMP handle simple types or structured types? How many simple types are available? What are the sizes of these types? What is the range of these types? In addition, how are each of these types encoded? We need these universal rules because we do not know the architecture of the computers that send, receive, or store these values. The sender may be a powerful computer in which an integer is stored as 8-byte data; the receiver may be a small computer that stores an integer as 4-byte data. SMI is a protocol that defines these rules. However, we must understand that SMI only defines the rules; it does not define how many objects are managed in an entity or which object uses which type. SMI is a collection of general rules to name objects and to list their types. The association of an object with the type is not done by SMI.

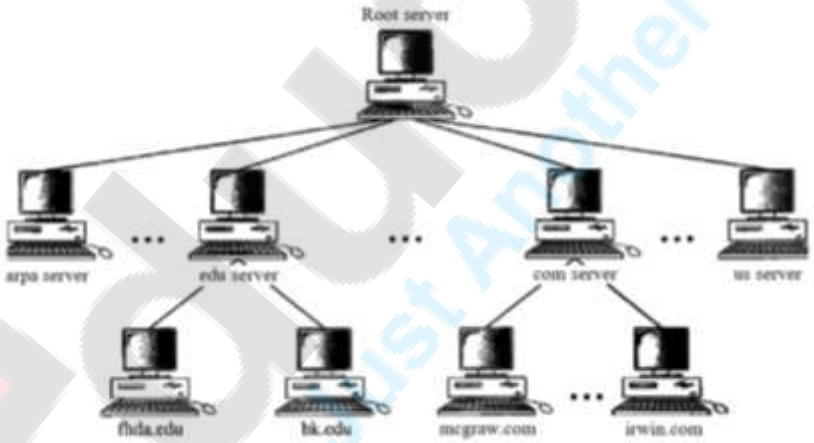
C) **Role of MIB**: -We hope it is clear that we need another protocol. For each entity to be managed, this protocol must define the number of objects, name them according to the rules defined by SMI, and associate a type to each named object. This protocol is MIB. MIB creates a set of objects defined for each entity similar to a database (mostly metadata in a database, names and types without values).

11) Explain Domain Naming System? Explain it all rule and components?

Answer: -

1) A user of an e-mail program may know the e-mail address of the recipient; however, the IP protocol needs the IP address. The DNS client program sends a request to a DNS server to map the e-mail address to the corresponding IP address.
2) TCPIIP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name.
3) The Internet was small; mapping was done by using a host file. The host file had only two columns: name and address. Every host could store the host file on its disk and update it periodically from a master host file. When a program or a user wanted to map a name to an address, the host consulted the host file and found the mapping.
4) It is impossible to have one single host file to relate every address with a name and vice versa. The host file would be too large to store in every host. In addition, it would be impossible to update all the host files every time there was a change.

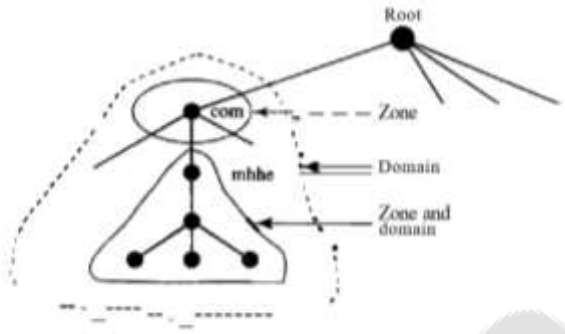
5) One solution would be to store the entire host file in a single computer and allow access to this centralized information to every computer that needs mapping. But we know that this would create a huge amount of traffic on the Internet.
6) Another solution, the one used today, is to divide this huge amount of information into smaller parts and store each part on a different computer. In this method, the host that needs mapping can contact the closest computer holding the needed information. This method is used by the Domain Name System (DNS).
7) **Distribution of Name Space**: - The information contained in the domain name space must be stored. However, it is very inefficient and also unreliable to have just one computer store such a huge amount of information. It is inefficient because responding to requests from all over the world places a heavy load on the system. It is not unreliable because any failure makes the data inaccessible.
   A) **Hierarchy of Name Servers:** - The solution to these problems is to distribute the information among many computers called DNS servers. One way to do this is to divide the whole space into many domains based on the first level. In other words, we let the root stand alone and create as many domains (subtrees) as there are first-level nodes. Because a domain created in this way could be very large, DNS allows domains to be divided further into smaller domains (subdomains). Each server can be responsible (authoritative) for either a large or a small domain.



   B) **Zone**: - If a server accepts responsibility for a domain and does not divide the domain into smaller domains, the domain and the zone refer to the same thing. The server makes a database called a zone file and keeps all the information for every node under that domain. The information about the nodes in the subdomains is stored in the servers at the lower levels, with the original server keeping some sort of reference to these lower-level servers. Of course the original server does not free itself from responsibility totally: It still has a zone, but the detailed information is kept by the lower-level servers. A server can also divide part of its domain and delegate responsibility but still keep part of the domain for

itself. In this case, its zone is made of detailed information for the part of the domain that is not delegated and references to those parts that are delegated.

Figure 25.7 *Zones and domains*



C) **Root Server:** - A root server is a server whose zone consists of the whole tree. A root server usually does not store any information about domains but delegates its authority to other servers, keeping references to those servers. There are several root servers, each covering the whole domain name space. The servers are distributed all around the world

D) **Primary and Secondary Servers:** - DNS defines two types of servers: primary and secondary. A primary server is a server that stores a file about the zone for which it is an authority. Itis responsible for creating, maintaining, and updating the zone file. It stores the zone file on a local disk. A secondary server is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. The secondary server neither creates nor updates the zone files. If updating is required, it must be done by the primary server, which sends the updated version to the secondary. The primary and secondary servers are both authoritative for the zones they serve. The idea is not to put the secondary server at a lower level of authority but to create redundancy for the data so that if one server fails, the other can continue serving clients. Note also that a server can be a primary server for a specific zone and a secondary server for another zone.

12) What is meant by resolution in DNS explain?

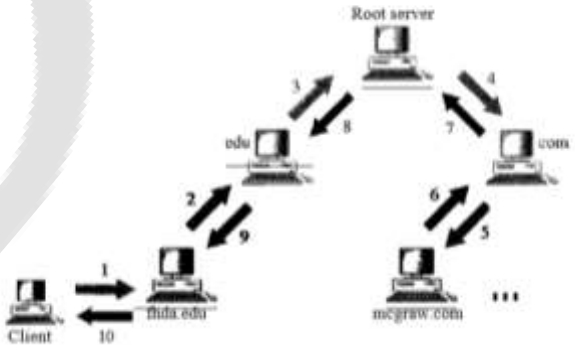
Answer: -

1) Mapping a name to an address or an address to a name is called name-address resolution.
2) **Resolver:** -
   A) DNS is designed as a client/server application. A host that needs to map an address to a name or a name to an address calls a DNS client called a resolver. The resolver accesses the closest DNS server with a mapping request. If the server has the information, it satisfies the resolver; otherwise, it either refers the resolver to other servers or asks other servers to provide the information. After the resolver

receives the mapping, it interprets the response to see if it is a real resolution or an error, and finally delivers the result to the process that requested it.

3) **Mapping Names to Addresses**
   A) The resolver gives a domain name to the server and asks for the corresponding address. In this case, the server checks the generic domains or the country domains to find the mapping. If the domain name is from the generic domains section, the resolver receives a domain name such as "chal.atc.jhda.edu.". The query is sent by the resolver to the local DNS server for resolution. If the local server cannot resolve the query, it either refers the resolver to other servers or asks other servers directly. If the domain name is from the country domains section, the resolver receives a domain name such as "ch.jhda.cu.ca.us.". The procedure is the same.
4) **Mapping Addresses to Names**
   A) A client can send an IP address to a server to be mapped to a domain name. As mentioned before, this is called a PTR query. To answer queries of this kind, DNS uses the inverse domain. However, in the request, the IP address is reversed and the two labels in-addr and arpa are appended to create a domain acceptable by the inverse domain section. For example, if the resolver receives the IF address 132.34.45.121, the resolver first inverts the address and then adds the two labels before sending. The domain name sent is "121.45.34.132.in-addr.arpa." which is received by the local DNS and resolved.
5) **Recursive Resolution**
   A) The client (resolver) can ask for a recursive answer from a name server. This means that the resolver expects the server to supply the final answer. If the server is the authority for the domain name, it checks its database and responds. If the server is not the authority, it sends the request to another server (the parent usually) and waits for the response. If the parent is the authority, it responds; otherwise, it sends the query to yet another server. When the query is finally resolved, the response travels back until it finally reaches the requesting client. This is called recursive resolution and is shown in Figure.

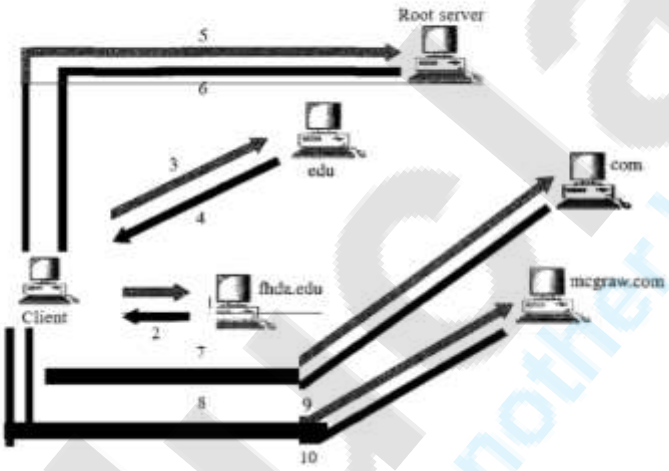
Figure 25.12 *Recursive resolution*



6) **Iterative Resolution**

A) If the client does not ask for a recursive answer, the mapping can be done iteratively. If the server is an authority for the name, it sends the answer. Ifit is not, it returns (to the client) the IP address of the server that it thinks can resolve the query. The client is responsible for repeating the query to this second server. If the newly addressed server can resolve the problem, it answers the query with the IP address; otherwise, it returns the IP address of a new server to the client. Now the client must repeat the query to the third server. This process is called iterative resolution because the client repeats the same query to multiple servers. In Figure 25.13 the client queries four servers before it gets an answer from the mcgraw.com server.

Figure 25.13 *Iterative resolution*



7) **Caching**

A) Each time a server receives a query for a name that is not in its domain, it needs to search its database for a server IP address. Reduction of this search time would increase efficiency. DNS handles this with a mechanism called caching. When a server asks for a mapping from another server and receives the response, it stores this information in its cache memory before sending it to the client. If the same or another client asks for the same mapping, it can check its cache memory and solve the problem. Inform the client that the response is coming from the cache memory and not from an authoritative source, the server marks the response as unauthoritative.

B) Caching speeds up resolution, but it can also be problematic. If a server caches a mapping for a long time, it may send an outdated mapping to the client. To counter this, two techniques are used. First, the authoritative server always adds information to the mapping called time-to-live (TTL). It defines the time in seconds that the receiving server can cache the information. After that time, the mapping is invalid and any query must be sent again to the authoritative server.

Second, DNS requires that each server keep a TTL counter for each mapping it caches. The cache memory must be searched periodically, and those mappings with an expired TTL must be purged.

13) Describe the steps involved in communication between web and HTTP?

Answer: -

1) The steps of transferring a Web page from server to client for the case of non-persistent connections. The page consists of a base HTML file and 10 JPEG images, and that all 11 of these objects reside on the same server. Further suppose the URL for the base HTML file is
http://www.someSchool.edu/someDepartment/home.index
2) The HTTP client process initiates a TCP connection to the server www.someSchool.edu on port number 80, which is the default port number for HTTP. Associated with the TCP connection, there will be a socket at the client and a socket at the server.
3) The HTTP client sends an HTTP request message to the server via its socket. The request message includes the path name /someDepartment/home.index. (We will discuss HTTP messages in some detail below.)
4) The HTTP server process receives the request message via its socket, retrieves the object /someDepartment/home.index from its storage (RAM or disk), encapsulates the object in an HTTP response message, and sends the response message to the client via its socket.
5) The HTTP server process tells TCP to close the TCP connection. (But TCP doesn't actually terminate the connection until it knows for sure that the client has received the response message intact.)
6) The HTTP client receives the response message. The TCP connection terminates. The message indicates that the encapsulated object is an HTML file. The client extracts the file from the response message, examines the HTML file, and finds references to the 10 JPEG objects.
7) The first four steps are then repeated for each of the referenced JPEG objects
8) The browser receives the Web page, it displays the page to the user. Two different browsers may interpret (that is, display to the user) a Web page in somewhat different ways. HTTP has nothing to do with how a Web page is interpreted by a client. The HTTP specifications define only the communication protocol between the client HTTP program and the server HTTP program.
9) The steps above illustrate the use of **non-persistent connections,** where each TCP connection is closed after the server sends the object—the connection does not persist for other objects. Note that each TCP connection transports exactly one request message and one response message.
10) With persistent connections, the server leaves the TCP connection open after sending a response. Subsequent requests and responses between the same client and server

can be sent over the same connection. In particular, an entire Web page (in the example above, the base HTML file and the 10 images) can be sent over a single persistent TCP connection. Moreover, multiple Web pages residing on the same server can be sent from the server to the same client over a single persistent TCP connection. These requests for objects can be made back-to-back, without waiting for replies to pending requests (pipelining). Typically, the HTTP server closes a connection when it isn't used for a certain time (a configurable timeout interval). When the server receives the back-to-back requests, it sends the objects back-to-back. The default mode of HTTP uses persistent connections with pipelining.

14) Difference between

A) POP3 vs IMAP

Answer: -

BASIS FOR COMPARISON	POP3	IMAP
Basic	To read the mail ithas to be downloaded first.	The mail content can be checked partially before downloading.
Organize	The user can not organize mails in the mailbox of the mail server.	The user can organize the mails on the server.
Folder	The user can not create, delete or rename mailboxes on a mail server.	The user can create, delete or rename mailboxes on the mail server.
Content	A user can not search the content of mail for prior downloading.	A user can search the content of mail for specific string of character before downloading.
Partial Download	The user has to download the mail for accessing it.	The user can partially download the mail if bandwidth is limited.
Functions	POP3 is simple and has limited functions.	IMAP is more powerful, more complex and has more features over POP3.

B) SMPTP vs POP3

Answer: -

BASIS FOR COMPARISON	SMTP	POP3
Basic	It is message transfer agent.	It is message access agent.
Full form	Simple Mail Transfer Protocol.	Post Office Protocol version 3.
Implied	Between sender and sender mail server and between sender mail server and receiver mail server.	Between receiver and receiver mail server.
work	It transfers the mail from senders computer to the mail box present on receiver's mail server.	It allows to retrieve and organize mails from mailbox on receiver mail server to receiver's computer.

C) HTTP vs FTP

Answer: -

BASIS FOR COMPARISON	HTTP	FTP
Basic	HTTP is used to access websites.	FTP transfers file from one one host to another.
Connection	HTTP establishes data connection only.	FTP establishes two connection one for data and one for the control connection.
TCP ports	HTTP uses TCP's port number 80.	FTP uses TCP's port number 20 and 21.
URL	If you are using HTTP, http will appear in URL.	If you are using FTP, ftp will appear in URL.
Efficient	HTTP is efficient in transferring smaller files like web pages.	FTP is efficient in transferring larger files.
Authentication	HTTP does not require authentication.	FTP requires a password.
Data	The content transferred to a device using HTTP is not saved to the memory of that device.	The file transferred to the host device using FTP is saved in the memory of that host device.

D) FTP vs TFTP

Answer: -

BASIS FOR COMPARISON	FTP	TFTP
Abbreviation	File Transfer Protocol.	Trivial File Transfer Protocol.
Authentication	Authentication is required in FTP for communication between client and server.	No authentication is required in TFTP.
Service	FTP uses TCP service which is a connection-oriented service.	TFTP uses UDP service which is connection-less service.
Software	FTP software is larger than TFTP.	TFTP software is smaller than FTP and fits into readonly memory of the diskless workstation.
Connection	FTP establishes two connections one for data(TCP port no. 21) and one for control(TCP port no. 20).	TFTP establishes a single connection for its file transfer (UDP port no. 69).
Commands/Message	FTP have many commands.	TFTP have only five messages.
Complexity	FTP is more complex	TFTP is less complex.