UNIT 5 : APPLICATION LAYER

WWW and HTTP – FTP – Email –Telnet –SSH – DNS – SNMP

1. INTRODUCTION

- The application layer is the highest layer in the protocol suite.
- The application layer provides services to the user.
- Communication is provided using a logical connection, which means that the two application layers assume that there is an imaginary direct connection through which they can send and receive messages.
- The application layer is the only layer that provides services to the Internet user
- The application layer exchange messages with their peers on other machines
- Applications need their own protocols. These applications are part of network protocol.
- Types of Application Protocols:

  Standard and Nonstandard Protocols

Standard Application-Layer Protocols

- There are several application-layer protocols that have been standardized and documented by the Internet authority.
- Each standard protocol is a pair of computer programs that interact with the user and the transport layer to provide a specific service to the user.
- Two very widely-used standardized application protocols:

  SMTP : Simple Mail Transfer Protocol is used to exchange electronic mail.

  HTTP : Hyper Text Transport Protocol is used to communicate between Web browsers and Web servers.

Nonstandard Application-Layer Protocols

- A programmer can create a nonstandard application-layer program if they can write two programs that provide service to the user by interacting with the transport layer.

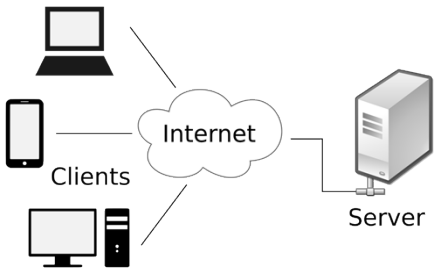
APPLICATION-LAYER PARADIGMS

Two paradigms have been developed for Application Layer

1. **Traditional Paradigm : Client-Server**
2. **New Paradigm : Peer-to-Peer**

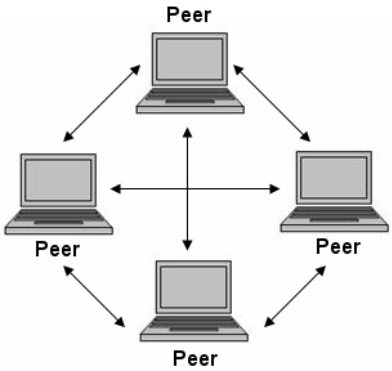
Client-Server Paradigm

- The **traditional paradigm** is called the client-server paradigm.
- It was the most popular Paradigm.
- In this paradigm, the service provider is an application program, called the server process; it runs continuously, waiting for another application program, called the client process, to make a connection through the Internet and ask for service.
- The server process must be running all the time; the client process is started when the client needs to receive service.
- There are normally some server processes that can provide a specific type of service, but there are many clients that request service from any of these server processes.



Peer-to-Peer(P2P) Paradigm

- A **new paradigm**, called the peer-to-peer paradigm has emerged to respond to the needs of some new applications.
- In this paradigm, there is no need for a server process to be running all the time and waiting for the client processes to connect.
- The responsibility is shared between peers.
- A computer connected to the Internet can provide service at one time and receive service at another time.
- A computer can even provide and receive services at the same time.

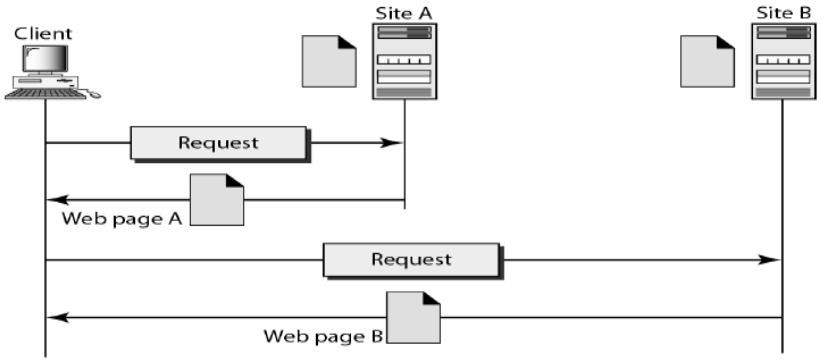


Mixed Paradigm

- An application may choose to use a mixture of the two paradigms by combining the advantages of both.
- For example, a light-load client-server communication can be used to find the address of the peer that can offer a service.
- When the address of the peer is found, the actual service can be received from the peer by using the peer-to-peer paradigm.

2. WWW (WORLD WIDE WEB)

- WWW is a distributed client/server service, in which a client (Browsers such as IE, Firefox, etc.) can access services at a server (Web server such as IIS, Apache).
- The service provided is distributed over many locations called sites.
- WWW was constructed originally by a small group of people led by Tim Berners Lee at CERN, in 1989 and in 1991 this was released to the world.
- A new protocol for the Internet and a system of document access to use it was proposed and named as WWW.



- This system allows document search and retrieval from any part of the Internet.
- The documents were having *Hypertext* as the content
- The units of information on the web can be referred to as pages, documents or resources.
- A document can contain text, images, sound and video, together called Hypermedia.
- Web is a vast collection of data, information, software and protocols , spread across the world in web servers, which are accessed by client machines by browsers through the Internet.

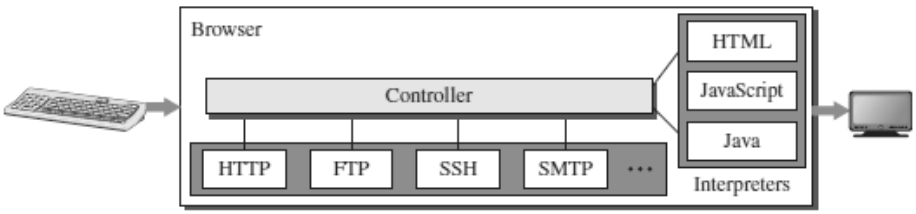
COMPONENTS OF THE WEB

Structural Components	Semantic Components
1. Web Clients/Browsers	1. Hyper Text Transfer Protocol (HTTP)
2. Web Servers	2. Hyper Text Markup Language (HTML)
3. Web Caches	3. eXtensible Markup Language (XML)
4. Internet	4. Uniform Resource Identifier (URI)

- Clients use browser application to send URL's via HTTP to servers requesting a Web page.
- Web pages constructed using HTML /XML and consist of text, graphics, sounds plus embedded files
- Servers (or caches) respond with requested Web page.
- Client's browser renders Web page returned by server
- Web Page is written using Hyper Text Markup Language (HTML)
- Displays text, graphics and sound in browser
- The entire system runs over standard networking protocols (TCP/IP, DNS)

WEB CLIENTS (BROWSERS)

- A browser is a software on the client on the web which initiates the communication with the server.
- Each browser usually consists of three parts: a controller, client protocols, and interpreters.
- The controller receives input from the keyboard or the mouse and uses the client programs to access the document. After the document has been accessed, the controller uses one of the interpreters to display the document on the screen.
- Examples are Internet Explorer, Mozilla FireFox, Netscape Navigator, Safari etc.

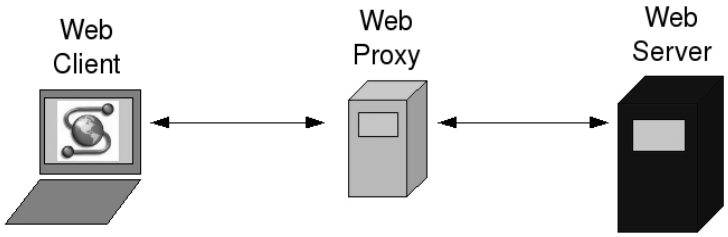


WEB SERVERS

- All the communication between the web client and a web server use the standard protocol called as HTTP.
- Web server informs its operating system to accept incoming network connections using a specific port on the machine.
- The server also runs as a background process.
- A client (browser) opens a connection to the server, sends a request, receives information from server and closes the connection.
- Web server monitors a communications port on its host machine, accepts the http commands through it and performs specified operations.
- HTTP commands include a URL specifying the host machine.
- The URL received is translated into either a filename or a program name, accordingly the requested file or the output of the program execution is sent back to the browser.

PROXY SERVER

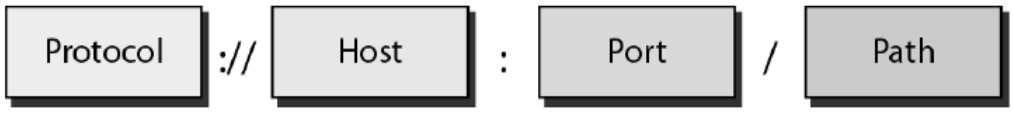
- A Proxy server is a computer that keeps copies of responses to recent requests.
- The web client sends a request to the proxy server.
- The proxy server checks its cache.
- If the response is not stored in the cache, the proxy server sends the request to the corresponding server.



- Incoming responses are sent to the proxy server and stored for future requests from other clients.
- The proxy server reduces the load on the original server, decreases traffic, and improves latency.
- However, to use the proxy server, the client must be configured to access the proxy instead of the target server.
- The proxy server acts as both server and client.
- When it receives a request from a client for which it has a response, it acts as a server and sends the response to the client.
- When it receives a request from a client for which it does not have a response, it first acts as a client and sends a request to the target server.
- When the response has been received, it acts again as a server and sends the response to the client.

URL - UNIFORM RESOURCE LOCATOR

- Uniform Resource Locator (URL), uniquely identify resources on the Internet
- URL provides information about its location on the Web
- When a user enters URL, browser forms a *request* message and sends it to the server.
- Web server retrieves the requested URL and sends back a *response* message.
- Web browser renders the response in HTML or appropriate format.
- Format : http://www.*domain_name*/*filename*
- Example : http://www.cs.hello.org/index.html



- The URL defines four parts - Method, Host computer, Port, and Path.
  - **Method:** The method is the protocol used to retrieve the document from a server. For example, HTTP.
  - **Host:** The host is the computer where the information is stored, and the computer is given an alias name. Web pages are mainly stored in the computers and the computers are given an alias name that begins with the characters "www". This field is not mandatory.
  - **Port:** The URL can also contain the port number of the server, but it's an optional field. If the port number is included, then it must come between the host and path and it should be separated from the host by a colon.
  - **Path:** Path is the pathname of the file where the information is stored. The path itself contain slashes that separate the directories from the subdirectories and files.

URL Paths

- The path of the document for a http protocol is same as that for a document or file or a directory in a client.
- In Unix the path components are separated by forward slashes (/) and in windows backward slashes (\).
- But an URL need not include all the directories in the path.
- A path which includes all the directories is a ***complete path***, else it is a ***partial path***.

URI - Uniform Resource Identifiers

- URI is a string that identifies resources such as document, image, service, etc.
- It is of the form *scheme*:*scheme-specific*
- Scheme *identifies* a resource type, such as mailto for mail address, file for file name, etc. and scheme-specific is a *resource* identifier.
- Example is mailto: abc123@gmail.com
- URI identifies a resource, whereas URL is used to locate a resource.

WEB DOCUMENTS

The documents in the WWW can be grouped into three broad categories:
Static, Dynamic and Active.

Static Documents

- Static documents are fixed-content documents that are created and stored in a server.
- The client can get a copy of the document only.
- In other words, the contents of the file are determined when the file is created, not when it is used.
- Of course, the contents in the server can be changed, but the user cannot change them.
- When a client accesses the document, a copy of the document is sent.
- The user can then use a browser to see the document.
- Static documents are prepared using one of several languages:
  1. HyperText Markup Language (HTML)
  2. Extensible Markup Language (XML)
  3. Extensible Style Language (XSL)
  4. Extensible Hypertext Markup Language (XHTML).

Dynamic Documents

- A dynamic document is created by a web server whenever a browser requests the document.
- When a request arrives, the web server runs an application program or a script that creates the dynamic document.

- The server returns the result of the program or script as a response to the browser that requested the document.
- Because a fresh document is created for each request, the contents of a dynamic document may vary from one request to another.
- A very simple example of a dynamic document is the retrieval of the time and date from a server.
- Time and date are kinds of information that are dynamic in that they change from moment to moment.
- Dynamic documents can be retrieved using one of several scripting languages:
    1. Common Gateway Interface (CGI)
    2. Java Server Pages (JSP)
    3. Active Server Pages (ASP)
    4. ColdFusion

Active Documents

- For many applications, we need a program or a script to be run at the client site. These are called active documents**.**
- For example, suppose we want to run a program that creates animated graphics on the screen or a program that interacts with the user.
- The program definitely needs to be run at the client site where the animation or interaction takes place.
- When a browser requests an active document, the server sends a copy of the document or a script.
- The document is then run at the client (browser) site.
- Active documents can be created using one of several languages:
    1. Java Applet – A program written in Java on the server. It is compiled and ready to be run. The document is in bytecode format.
    2. Java Script - Download and run the script at the client site.

---

3. HTTP (HYPERTEXT TRANSFER PROTOCOL)

- The HyperText Transfer Protocol (HTTP) is used to define how the client-server programs can be written to retrieve web pages from the Web.
- It is a protocol used to access the data on the World Wide Web (WWW).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
- HTTP is a *stateless* request/response protocol that governs client/server communication.
- An HTTP client sends a request; an HTTP server returns a response.
- The server uses the port number 80; the client uses a temporary port number.
- HTTP uses the services of TCP , a connection-oriented and reliable protocol.
- HTTP is a text-oriented protocol. It contains *embedded* URL known as links.

- When hypertext is clicked, browser opens a new connection, retrieves file from the server and displays the file.
- Each HTTP message has the general form

  START_LINE <CRLF>
  MESSAGE_HEADER <CRLF>
  <CRLF> MESSAGE_BODY <CRLF>

  where <CRLF> stands for carriage-return-line-feed.

Features of HTTP

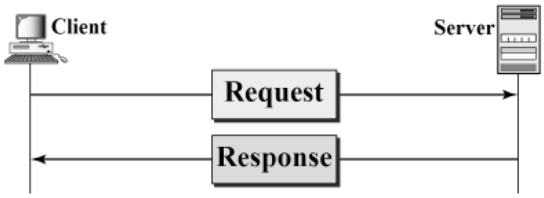
- **Connectionless protocol:**
  HTTP is a connectionless protocol. HTTP client initiates a request and waits for a response from the server. When the server receives the request, the server processes the request and sends back the response to the HTTP client after which the client disconnects the connection. The connection between client and server exist only during the current request and response time only.

- **Media independent:**
  HTTP protocol is a media independent as data can be sent as long as both the client and server know how to handle the data content. It is required for both the client and server to specify the content type in MIME-type header.

- **Stateless:**
  HTTP is a stateless protocol as both the client and server know each other only during the current request. Due to this nature of the protocol, both the client and server do not retain the information between various requests of the web pages.

HTTP REQUEST AND RESPONSE MESSAGES

- The HTTP protocol defines the format of the request and response messages.



- **Request Message:** The request message is sent by the client that consists of a request line, headers, and sometimes a body.
- **Response Message:** The response message is sent by the server to the client that consists of a status line, headers, and sometimes a body.

HTTP REQUEST MESSAGE

Request Line
Request Header : Value
Body (optional)

- The first line in a request message is called a *request line*.
- After the request line, we can have zero or more *request header* lines.
- The *body* is an optional one. It contains the comment to be sent or the file to be published on the website when the method is PUT or POST.

Request Line

- There are three fields in this request line - M*ethod, URL* and V*ersion*.
- The Method field defines the request types.
- The URL field defines the address and name of the corresponding web page.
- The Version field gives the version of the protocol; the most current version of HTTP is 1.1.
- Some of the Method types are

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
PUT	Sends a document from the client to the server
POST	Sends some information from the client to the server
TRACE	Echoes the incoming request
DELETE	Removes the web page
CONNECT	Reserved
OPTIONS	Inquires about available options

Request Header

- Each request header line sends additional information from the client to the server.
- Each header line has a header name, a colon, a space, and a header value.
- The value field defines the values associated with each header name.
- Headers defined for request message include

Header	*Description*
User-agent	Identifies the client program
Accept	Shows the media format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
Host	Shows the host and port number of the client
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Cookie	Returns the cookie to the server
If-Modified-Since	If the file is modified since a specific date

Body

- The *body* can be present in a request message. It is optional.
- Usually, it contains the comment to be sent or the file to be published on the website when the method is PUT or POST.

Conditional Request

- A client can add a condition in its request.
- In this case, the server will send the requested web page if the condition is met or inform the client otherwise.
- One of the most common conditions imposed by the client is the time and date the web page is modified.
- The client can send the header line *If-Modified-Since* with the request to tell the server that it needs the page only if it is modified after a certain point in time.

HTTP RESPONSE MESSAGE

Status Line
Response Header : Value
Body

- The first line in a request message is called a status *line*.
- After the request line, we can have zero or more *response header* lines.
- The *body* is an optional one. The body is present unless the response is an error message

Status Line

- The Status line contains three fields - *HTTP version , Status code, Status phrase*
- The first field defines the version of HTTP protocol, currently 1.1.
- The status code field defines the status of the request. It classifies the HTTP result. It consists of three digits.

  1*xx*–Informational, 2*xx*– Success, 3*xx*–Redirection,
  4*xx*–Client error, 5*xx*–Server error
- The Status phrase field gives brief description about status code in text form.
- Some of the Status codes are

Code	Phrase	Description
100	Continue	Initial request received, client to continue process
200	OK	Request is successful
301	Moved permanently	Requested URL is no longer in use
404	Not found	Document not found
500	Internal server error	An error such as a crash, at the server site

Response Header

- Each header provides additional information to the client.
- Each header line has a header name, a colon, a space, and a header value.
- Some of the response headers are:

Response Header	Description
Content-type	specifies the MIME type
Expires	date and time up to which the document is valid
Last-modified	date and time when the document was last updated
Location	specifies location of the created or moved document

Body

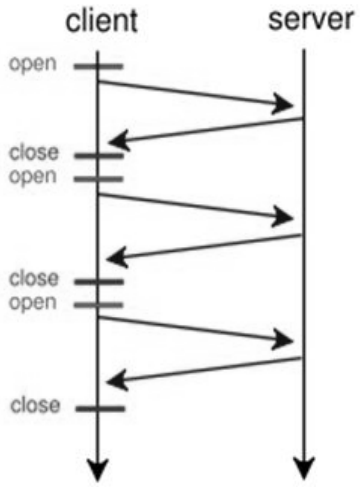
- The body contains the document to be sent from the server to the client.
- The body is present unless the response is an error message.

HTTP CONNECTIONS

- HTTP Clients and Servers exchange multiple messages over the same TCP connection.
- If some of the objects are located on the same server, we have two choices: to retrieve each object using a new TCP connection or to make a TCP connection and retrieve them all.
- The first method is referred to as a *non-persistent connection,* the second as a *persistent connection*.
- HTTP 1.0 uses *non-persistent* connections and HTTP 1.1 uses *persistent* connections .

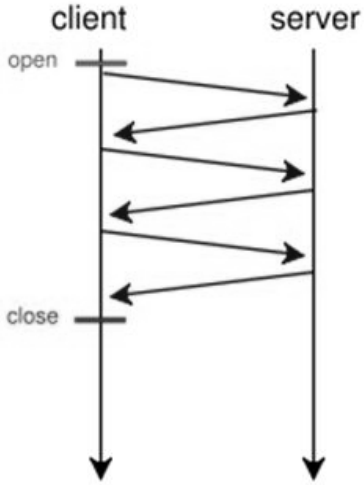
NON-PERSISTENT CONNECTIONS

- In a non-persistent connection, one TCP connection is made for each request/response.
- Only one object can be sent over a single TCP connection
- The client opens a TCP connection and sends a request.
- The server sends the response and closes the connection.
- The client reads the data until it encounters an end-of-file marker.
- It then closes the connection.



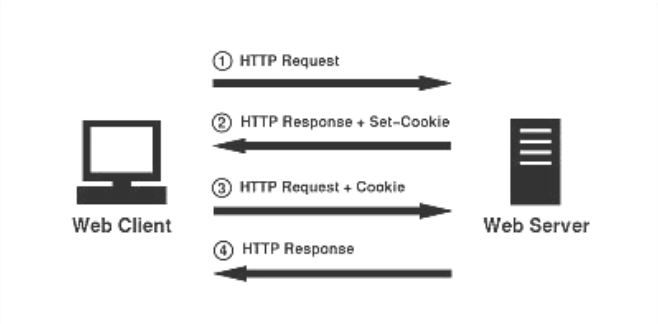
PERSISTENT CONNECTIONS

- HTTP version 1.1 specifies a persistent connection by default.
- Multiple objects can be sent over a single TCP connection.
- In a persistent connection, the server leaves the connection open for more requests after sending a response.
- The server can close the connection at the request of a client or if a time-out has been reached.
- Time and resources are saved using persistent connections. Only one set of buffers and variables needs to be set for the connection at each site.
- The round trip time for connection establishment and connection termination is saved.



HTTP COOKIES

- An **HTTP cookie** (also called **web cookie**, **Internet cookie**, **browser cookie**, or simply **cookie**) is a small piece of data sent from a website and stored on the user's computer by the user's web browser while the user is browsing.
- HTTP is stateless , Cookies are used to add State.
- Cookies were designed to be a reliable mechanism for websites to remember stateful information (such as items added in the shopping cart in an online store) or to record the user's browsing activity (including clicking particular buttons, logging in, or recording which pages were visited in the past).
- They can also be used to remember arbitrary pieces of information that the user previously entered into form fields such as names, addresses, passwords, and credit card numbers.



Components of Cookie

A cookie consists of the following components:

1. Name
2. Value
3. Zero or more attributes (name/value pairs). Attributes store information such as the cookie's expiration, domain, and flags

Creating and Storing Cookies

The creation and storing of cookies depend on the implementation; however, the principle is the same.

1. When a server receives a request from a client, it stores information about the client in a file or a string. The information may include the domain name of the client, the contents of the cookie (information the server has gathered about the client such as name, registration number, and so on), a timestamp, and other information depending on the implementation.
2. The server includes the cookie in the response that it sends to the client.
3. When the client receives the response, the browser stores the cookie in the cookie directory, which is sorted by the server domain name.

Using Cookies

- When a client sends a request to a server, the browser looks in the cookie directory to see if it can find a cookie sent by that server.
- If found, the cookie is included in the request.
- When the server receives the request, it knows that this is an old client, not a new one.
- The contents of the cookie are never read by the browser or disclosed to the user. It is a cookie *made* by the server and *eaten* by the server.

Types of Cookies

1.Authentication cookies

These are the most common method used by web servers to know whether the user is logged in or not, and which account they are logged in with. Without such a mechanism, the site would not know whether to send a page containing sensitive information, or require the user to authenticate themselves by logging in.

2.Tracking cookies

These are commonly used as ways to compile individuals browsing histories.

3.Session cookie

A session cookie exists only in temporary memory while the user navigates the website. Web browsers normally delete session cookies when the user closes the browser.

4.Persistent cookie

Instead of expiring when the web browser is closed as session cookies do, a persistent cookie expires at a specific date or after a specific length of time. This means that, for the cookie's entire lifespan , its information will be transmitted to the server every time the user visits the website that it belongs to, or every time the user views a resource belonging to that website from another website.

HTTP CACHING

- HTTP Caching enables the client to retrieve document *faster* and reduces load on the server.
- HTTP Caching is implemented at Proxy server, ISP router and Browser.
- Server sets *expiration* date (Expires header) for each page, beyond which it is not cached.
- HTTP Cache document is returned to client only if it is an *updated* copy by checking against If-Modified-Since header.
- If cache document is *out-of-date*, then request is forwarded to the server and response is cached along the way.
- A web page will not be cached if *no-cache* directive is specified.

HTTP SECURITY

- HTTP does not provide security.
- However HTTP can be run over the Secure Socket Layer (SSL).
- In this case, HTTP is referred to as HTTPS.
- HTTPS provides confidentiality, client and server authentication, and data integrity.

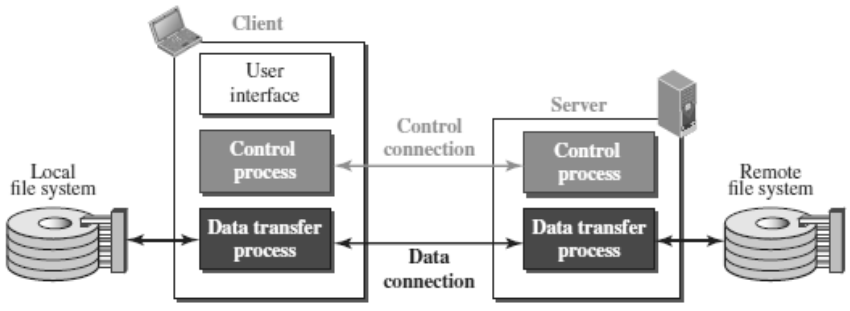
4. FTP (FILE TRANSFER PROTOCOL)

- FTP stands for File transfer protocol.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- It is also used for downloading the files to computer from other servers.
- Although we can transfer files using HTTP, FTP is a better choice to transfer large files or to transfer files using different formats.

FTP OBJECTIVES

- It provides the sharing of files.
- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

FTP MECHANISM



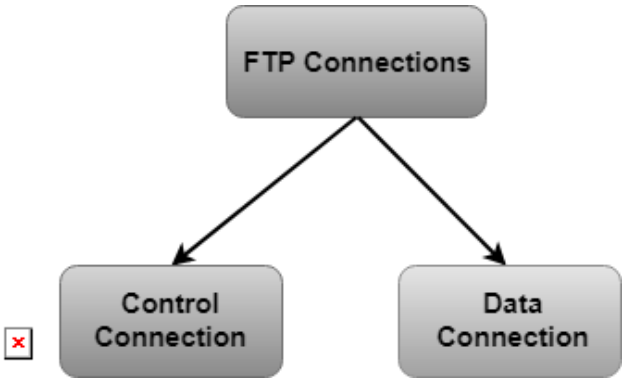
- The above figure shows the basic model of the FTP.
- The FTP client has three components:
  - user interface, control process, and data transfer process.
- The server has two components:
  - server control process and server data transfer process.

FTP CONNECTIONS

- There are two types of connections in FTP -

Control Connection and Data Connection.

- The two connections in FTP have different lifetimes.
- The control connection remains connected during the entire interactive FTP session.
- The data connection is opened and then closed for each file transfer activity. 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.
- FTP uses two well-known TCP ports:
  - Port 21 is used for the control connection
  - Port 20 is used for the data connection.



- **Control Connection:**
  - The control connection uses very simple rules for communication.
  - Through control connection, we can transfer a line of command or line of response at a time.
  - The control connection is made between the control processes.
  - The control connection remains connected during the entire interactive FTP session.

- **Data Connection:**
  - The Data Connection uses very complex rules as data types may vary.
  - The data connection is made between data transfer processes.
  - The data connection opens when a command comes for transferring the files and closes when the file is transferred.

FTP COMMUNICATION

- FTP Communication is achieved through commands and responses.
- FTP Commands are sent from the client to the server
- FTP responses are sent from the server to the client.
- FTP Commands are in the form of ASCII uppercase, which may or may not be followed by an argument.
- Some of the most common commands are

Command	*Description*
ABOR	Abort the previous command
CDUP	Change to parent directory
CWD	Change to another directory
DELE	Delete a file
LIST	List subdirectories or files
MKD	Create a new directory
PASS	Password
PASV	Server chooses a port
PORT	Client chooses a port
PWD	Display name of current directory
QUIT	Log out of the system
RETR	Retrieve files; files are transferred from server to client
RMD	Delete a directory
RNFR	Identify a file to be renamed
RNTO	Rename the file
STOR	Store files; file(s) are transferred from client to server
STRU	Define data organization (**F**: file, **R**: record, or **P**: page)
TYPE	Default file type (**A**: ASCII, **E**: EBCDIC, **I**: image)
USER	User information
MODE	Define transmission mode (**S**: stream, **B**: block, or **C**: compressed

- 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 parameter.

Code	*Description*	*Code*	*Description*
125	Data connection open	**250**	Request file action OK
150	File status OK	**331**	User name OK; password is needed
200	Command OK	**425**	Cannot open data connection
220	Service ready	**450**	File action not taken; file not available
221	Service closing	**452**	Action aborted; insufficient storage
225	Data connection open	**500**	Syntax error; unrecognized command
226	Closing data connection	**501**	Syntax error in parameters or arguments
230	User login OK	**530**	User not logged in

FTP FILE TYPE

- FTP can transfer one of the following file types across the data connection: ASCII file, EBCDIC file, or image file.

FTP DATA STRUCTURE

- FTP can transfer a file across the data connection using one of the following data structure : *file structure, record structure,* or *page structure*.
- The file structure format is the default one and has no structure. It is a continuous stream of bytes.
- In the record structure, the file is divided into *records*. This can be used only with text files.
- In the 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.

FTP TRANSMISSION MODE

- FTP can transfer a file across the data connection using one of the following three transmission modes: *stream mode, block mode,* or *compressed mode*.
- The stream mode is the default mode; data are delivered from FTP to TCP as a continuous stream of bytes.
- In the block mode, data can be delivered from FTP to TCP in blocks.
- In the compressed mode, data can be compressed and delivered from FTP to TCP.

FTP FILE TRANSFER

- File transfer occurs over the data connection under the control of the commands sent over the control connection.
- File transfer in FTP means one of three things:
  - retrieving a file (server to client)
  - storing a file (client to server)
  - directory listing (server to client).

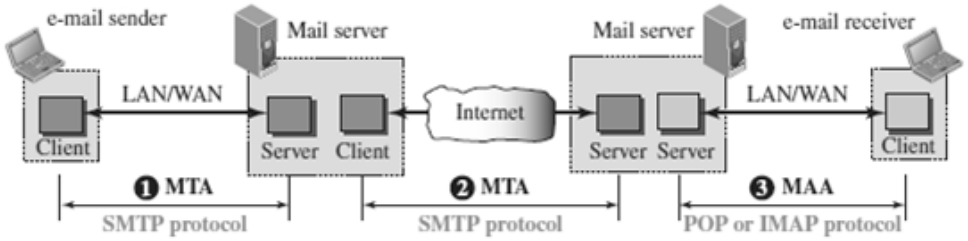
FTP SECURITY

- FTP requires a password, the password is sent in plaintext which is unencrypted. This 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.

---

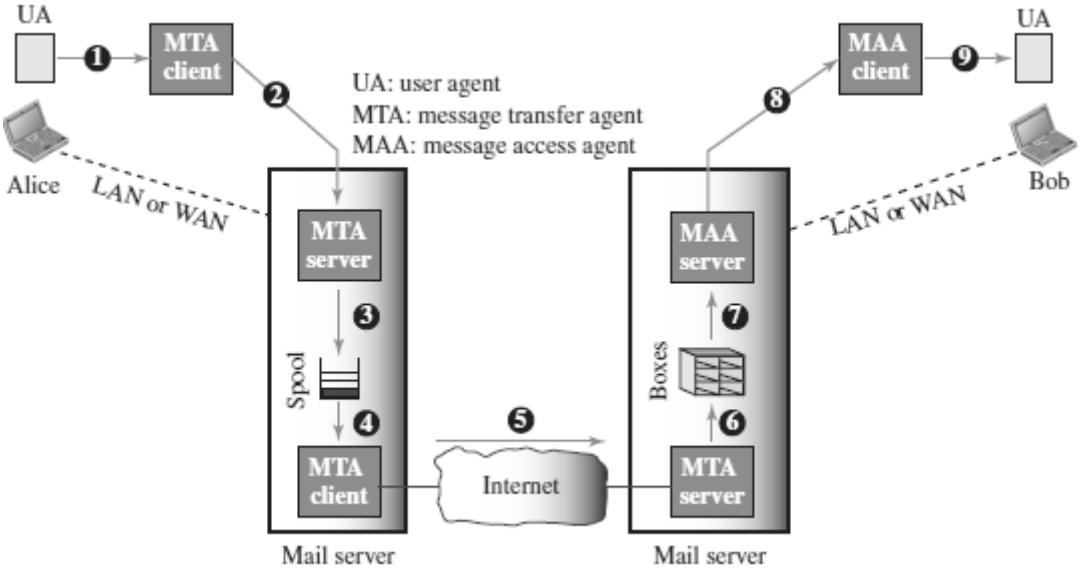
5. EMAIL (SMTP, MIME, IMAP, POP)

- One of the most popular Internet services is electronic mail (E-mail).
- Email is one of the oldest network applications.
- The **three main components of an Email** are
  1. User Agent (UA)
  2. Messsage Transfer Agent (MTA) – SMTP
  3. Messsage Access Agent (MAA) - IMAP , POP



- When the sender and the receiver of an e-mail are on the same system, we need only two User Agents and no Message Transfer Agent
- When the sender and the receiver of an e-mail are on different system, we need two UA, two pairs of MTA (client and server), and two MAA (client and server).

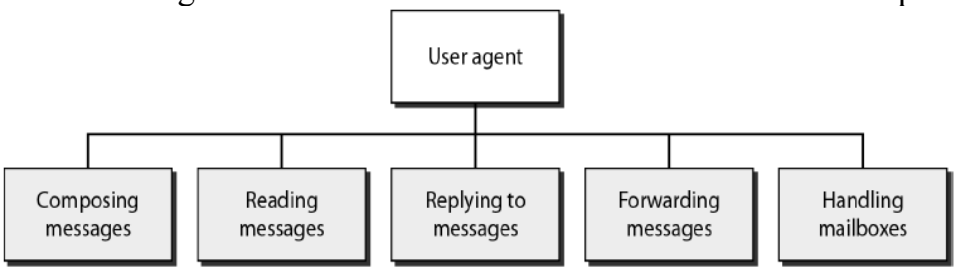
WORKING OF EMAIL



- When Alice needs to send a message to Bob, she runs a UA program to prepare the message and send it to her mail server.
- The mail server at her site uses a queue (spool) to store messages waiting to be sent. The message, however, needs to be sent through the Internet from Alice's site to Bob's site using an MTA.
- Here two message transfer agents are needed: one client and one server.
- The server needs to run all the time because it does not know when a client will ask for a connection.
- The client can be triggered by the system when there is a message in the queue to be sent.
- The user agent at the Bob site allows Bob to read the received message.
- Bob later uses an MAA client to retrieve the message from an MAA server running on the second server.

USER AGENT (UA)

- The first component of an electronic mail system is the user agent (UA).
- It provides service to the user to make the process of sending and receiving a message easier.
- A user agent is a software package that composes, reads, replies to, and forwards messages. It also handles local mailboxes on the user computers.



- There are two types of user agents: **Command-driven and GUI-based**.

Command driven

- Command driven user agents belong to the early days of electronic mail.
- A command-driven user agent normally accepts a one character command from the keyboard to perform its task.
- Some examples of command driven user agents are *mail, pine,* and *elm*.

GUI-based

- Modern user agents are GUI-based.
- They allow the user to interact with the software by using both the keyboard and the mouse.
- They have graphical components such as icons, menu bars, and windows that make the services easy to access.
- Some examples of GUI-based user agents are *Eudora* and *Outlook***.**

MESSAGE TRANSFER AGENT (MTA)

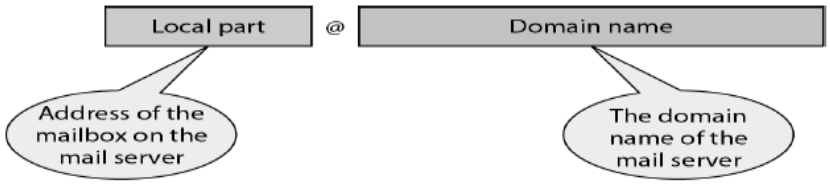
- The actual mail transfer is done through message transfer agents (MTA).
- To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA.
- The formal protocol that defines the MTA client and server in the Internet is called Simple Mail Transfer Protocol (SMTP).

MESSAGE ACCESS AGENT (MAA)

- MAA is a software that pulls messages out of a mailbox.
- POP3 and IMAP4 are examples of MAA.

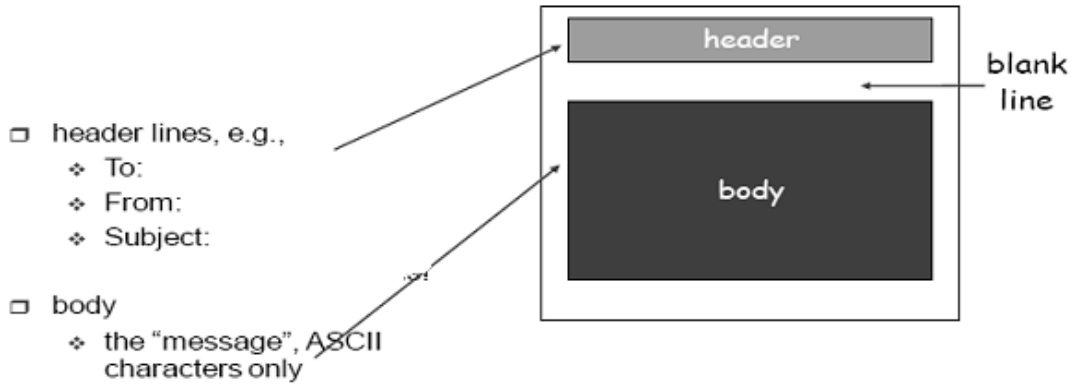
ADDRESS FORMAT OF EMAIL

- E-mail address is *userid @ domain* where *domain* is hostname of the *mail server*.



MESSAGE FORMAT OF EMAIL

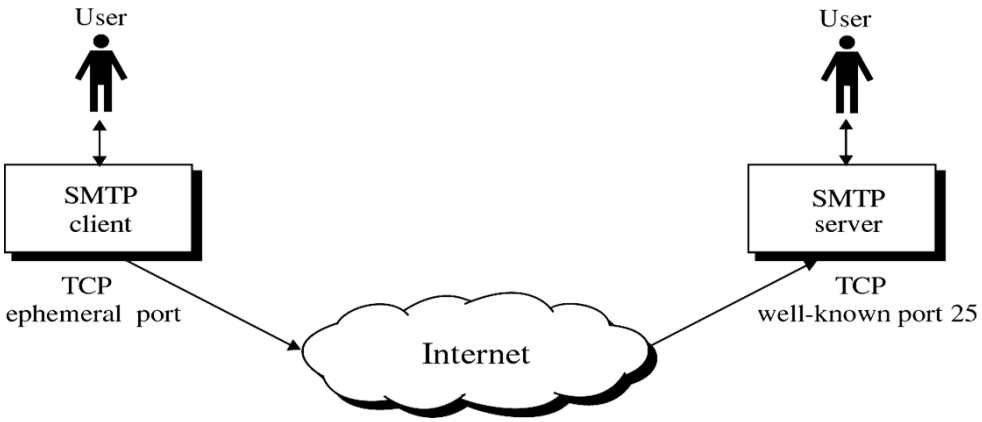
- Email message consists of two parts namely *header* and *body*.
- Each header line contains *type* and *value* separated by a colon (:).
- Some header contents are:
  - **From:** identifier sender of the message.
  - **To:** mail address of the recipient(s).
  - **Subject:** says about purpose of the message.
  - **Date:** timestamp of when the message was transmitted.
- Header is separated from the body by a *blank* line.
- Body contains the *actual* message.



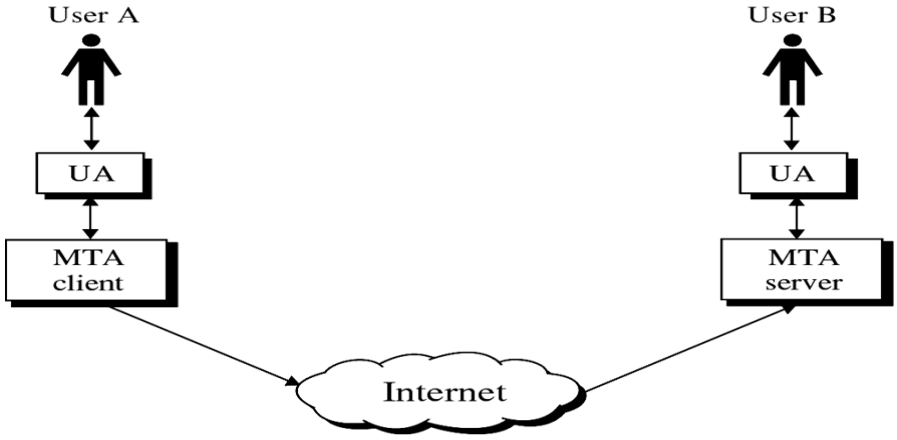
- Email was extended in 1993 to carry many different types of data: audio, video, images, Word documents, and so on.
- This extended version is known as **MIME**(Multipurpose Mail Extension).

SIMPLE MAIL TRANSFER PROTOCOL (SMTP)

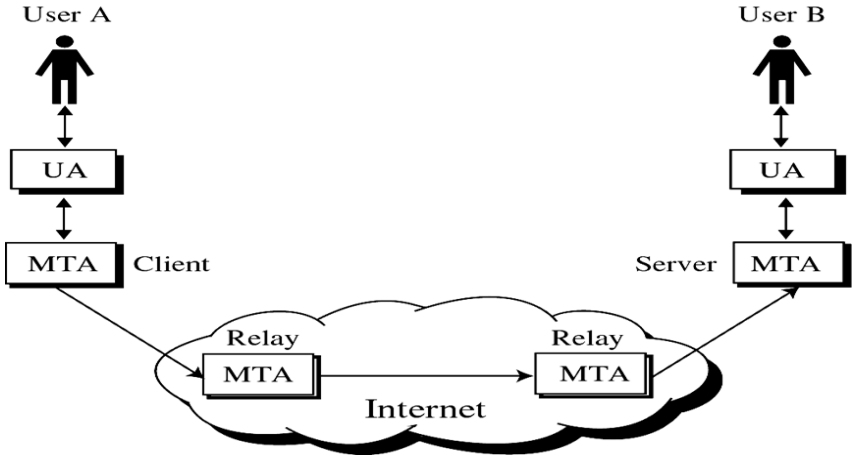
- SMTP is the standard protocol for transferring mail between hosts in the TCP/IP protocol suite.
- SMTP is not concerned with the format or content of messages themselves.
- SMTP uses information written on the *envelope* of the mail (message header), but does not look at the *contents* (message body) of the envelope.



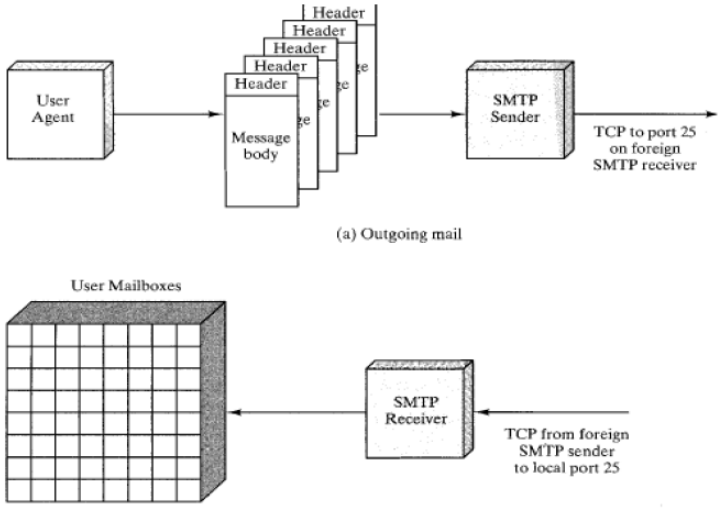
- SMTP clients and servers have two main components
  - **User Agents(UA)** – Prepares the message, encloses it in an envelope.
  - **Mail Transfer Agent (MTA)** – Transfers the mail across the internet



- SMTP also allows the use of Relays allowing other MTAs to relay the mail.



SMTP MAIL FLOW



- To begin, mail is created by a user-agent program in response to user input.
- Each created message consists of a header that includes the recipient's email address and other information, and a message body containing the message to be sent.
- These messages are then queued in some fashion and provided as input to an SMTP Sender program.

SMTP COMMANDS AND RESPONSES

- The operation of SMTP consists of a series of commands and responses exchanged between the SMTP sender and SMTP receiver.
- The initiative is with the SMTP sender, who establishes the TCP connection.
- Once the connection is established, the SMTP sender sends commands over the connection to the receiver.
- The command is from an MTA client to an MTA server; the response is from an MTA server to the MTA client.

SMTP Commands

- Commands are sent from the client to the server. It consists of a keyword followed by zero or more arguments. SMTP defines 14 commands.

SMTP commands

Keyword	*Argument(s)*	*Description*
HELO	Sender's host name	Identifies itself
MAIL FROM	Sender of the message	Identifies the sender of the message
RCPT TO	Intended recipient	Identifies the recipient of the message
DATA	Body of the mail	Sends the actual message
QUIT		Terminates the message
RSET		Aborts the current mail transaction
VRFY	Name of recipient	Verifies the address of the recipient
NOOP		Checks the status of the recipient
TURN		Switches the sender and the recipient
EXPN	Mailing list	Asks the recipient to expand the mailing list
HELP	Command name	Asks the recipient to send information about the command sent as the argument
SEND FROM	Intended recipient	Specifies that the mail be delivered only to the terminal of the recipient, and not to the mailbox
SMOL FROM	Intended recipient	Specifies that the mail be delivered to the terminal *or* the mailbox of the recipient
SMAL FROM	Intended recipient	Specifies that the mail be delivered to the terminal *and* the mailbox of the recipient

SMTP 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.

SMTP 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

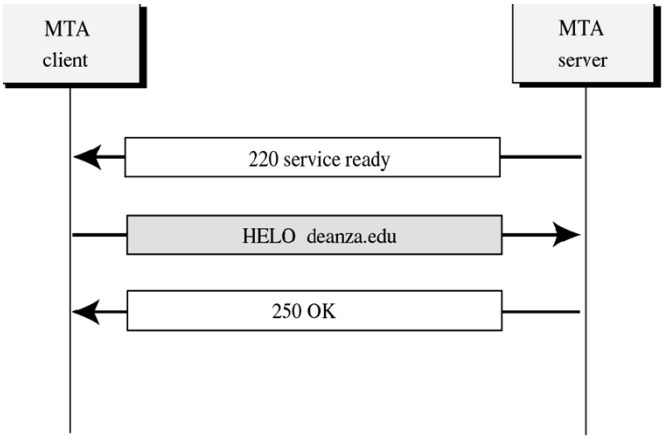
SMTP OPERATIONS

Basic SMTP operation occurs in three phases:

1. Connection Setup
2. Mail Transfer
3. Connection Termination

Connection Setup

- An SMTP sender will attempt to set up a TCP connection with a target host when it has one or more mail messages to deliver to that host.
- The sequence is quite simple:
  1. The sender opens a TCP connection with the receiver.
  2. Once the connection is established, the receiver identifies itself with "Service Ready".
  3. The sender identifies itself with the HELO command.
  4. The receiver accepts the sender's identification with "OK".
  5. If the mail service on the destination is unavailable, the destination host returns a "Service Not Available" reply in step 2, and the process is terminated.

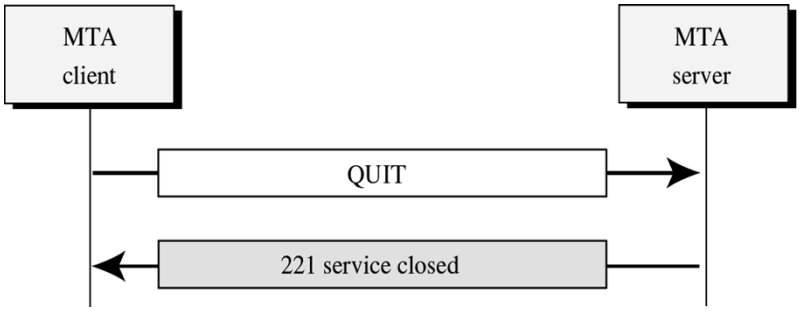


Mail Transfer

- Once a connection has been established, the SMTP sender may send one or more messages to the SMTP receiver.
- There are three logical phases to the transfer of a message:
  1. A MAIL command identifies the originator of the message.
  2. One or more RCPT commands identify the recipients for this message.
  3. A DATA command transfers the message text.

Connection Termination

- The SMTP sender closes the connection in two steps.
- First, the sender sends a QUIT command and waits for a reply.
- The second step is to initiate a TCP close operation for the TCP connection.
- The receiver initiates its TCP close after sending its reply to the QUIT command.



LIMITATIONS OF SMTP

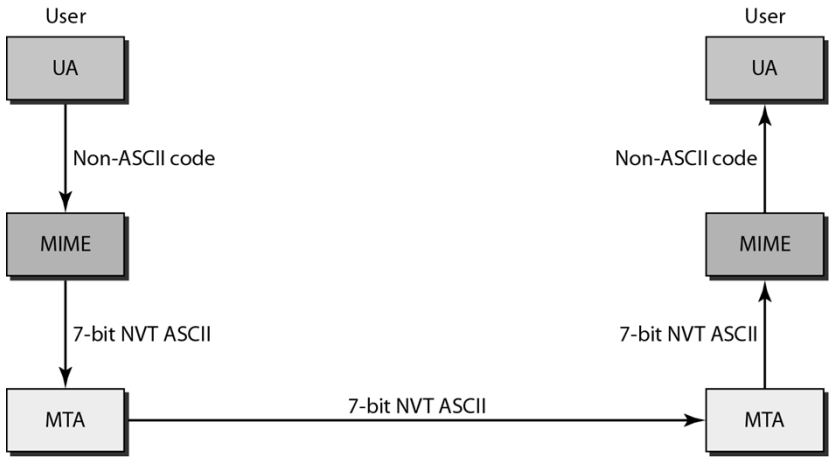
- SMTP cannot transmit executable files or other binary objects.
- SMTP cannot transmit text data that includes national language characters, as these are represented by 8-bit codes with values of 128 decimal or higher, and SMTP is limited to 7-bit ASCII.
- SMTP servers may reject mail message over a certain size.
- SMTP gateways that translate between ASCII and the character code EBCDIC do not use a consistent set of mappings, resulting in translation problems.
- Some SMTP implementations do not adhere completely to the SMTP standards defined.
- Common problems include the following:
  1. Deletion, addition, or recording of carriage return and linefeed.
  2. Truncating or wrapping lines longer than 76 characters.
  3. Removal of trailing white space (tab and space characters).
  4. Padding of lines in a message to the same length.
  5. Conversion of tab characters into multiple-space characters.

MULTIPURPOSE INTERNET MAIL EXTENSION (MIME)

- SMTP provides a basic email service, while MIME adds multimedia capability to SMTP.
- MIME is an extension to SMTP and is used to overcome the problems and limitations of SMTP.
- Email system was designed to send messages only in *ASCII* format.
  - Languages such as French, Chinese, etc., are not supported.
  - Image, audio and video files cannot be sent.
- MIME adds the following features to email service:

- Be able to send multiple attachments with a single message;
- Unlimited message length;
- Use of character sets other than ASCII code;
- Use of rich text (layouts, fonts, colors, etc)
- Binary attachments (executables, images, audio or video files, etc.), which may be divided if needed.

➢ MIME is a protocol that *converts* non-ASCII data to 7-bit NVT(Network Virtual Terminal) ASCII and vice-versa.

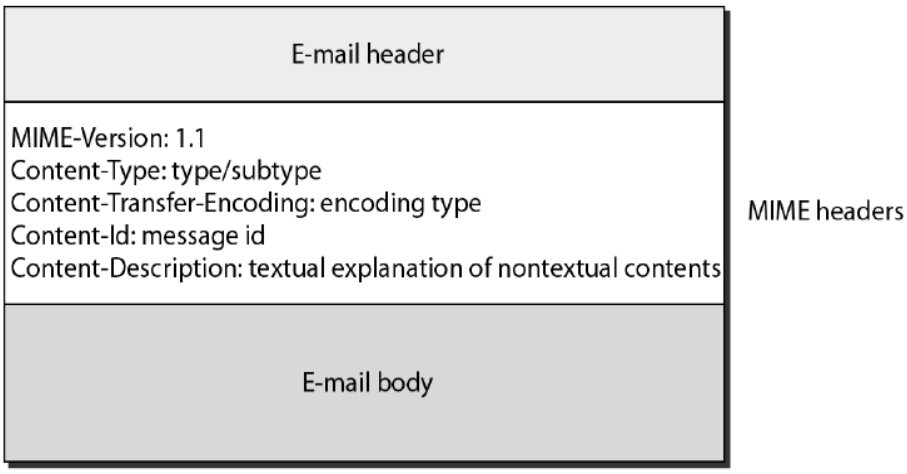


MIME HEADERS

➢ Using headers, MIME describes the type of message content and the encoding used.

➢ *Headers* defined in MIME are:

- MIME-Version- current version, i.e., 1.1
- Content-Type - message type (text/html, image/jpeg, application/pdf)
- Content-Transfer-Encoding - message encoding scheme (eg base64).
- Content-Id - unique identifier for the message.
- Content-Description - describes type of the message body.



MIME CONTENT TYPES

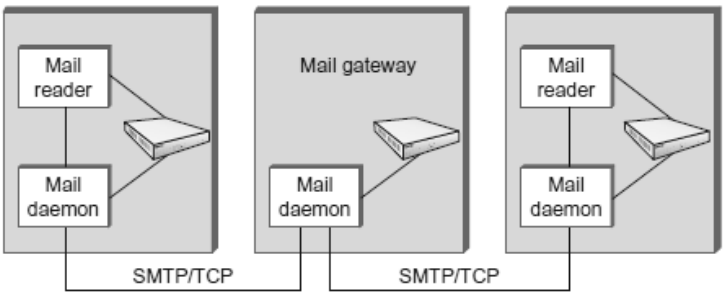
- There are seven different major types of content and a total of 14 subtypes.
- In general, a content type declares the general type of data, and the subtype specifies a particular format for that type of data.
- MIME also defines a multipart type that says how a message carrying more than one data type is structured.
- This is like a programming language that defines both base types (e.g., integers and floats) and compound types (e.g., structures and arrays).
- One possible multipart subtype is mixed, which says that the message contains a set of independent data pieces in a specified order.
- Each piece then has its own header line that describes the type of that piece.
- The table below lists the MIME content types:

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

ENCODING FORMATS OF MIME

- MIME uses various encoding formats to convert binary data into the ASCII character set.
- To transfer binary data, MIME offers five encoding formats which can be used in the header transfer-encoding:
  - ***7-bit*** : 7-bit text format (for messages without accented characters);
  - ***8-bit*** : 8-bit text format;
  - ***quoted-printable*** : Quoted-Printable format, recommended for messages which use a 7-bit alphabet (such as when there are accent marks);
  - ***base-64*** : Base 64, for sending binary files as attachments;
  - ***binary*** : binary format; not recommended.
- Since MIME is very open, it can use third-party encoding formats such as:
  - ***BinHex*** : A proprietary format belonging to Apple
  - ***Uuencode*** : for UNIX-to-UNIX encoding
  - ***Xencode*** : for binary-to-text encoding

MESSAGE TRANSFER IN MIME



- MTA is a mail daemon (sendmail) active on hosts having mailbox, used to send an email.
- Mail passes through a sequence of *gateways* before it reaches the recipient mail server.
- Each gateway stores and forwards the mail using Simple mail transfer protocol (SMTP).
- SMTP defines communication between MTAs over TCP on port 25.
- In an SMTP session, sending MTA is *client* and receiver is *server*. In each exchange:
- Client posts a command (HELO, MAIL, RCPT, DATA, QUIT, VRFY, etc.)
- Server responds with a code (250, 550, 354, 221, 251 etc) and an explanation.
- Client is identified using HELO command and verified by the server
- Client forwards message to server, if server is willing to accept.
- Message is terminated by a line with only single period (.) in it.
- Eventually client terminates the connection.

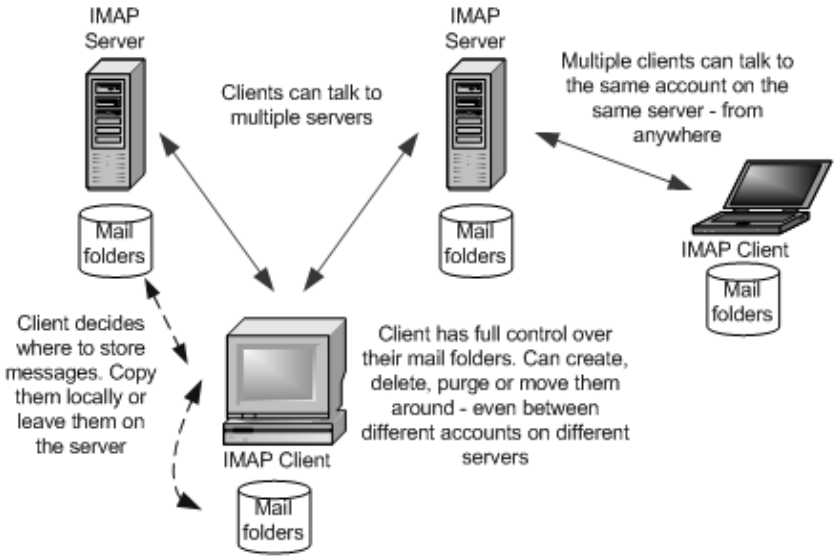
IMAP (INTERNET MAIL ACCESS PROTOCOL)

- IMAP is an Application Layer Internet protocol that allows an e-mail client to access e-mail on a remote mail server.
- It is a method of accessing electronic mail messages that are kept on a possibly shared mail server.
- IMAP is a more capable wire protocol.
- IMAP is similar to SMTP in many ways.
- IMAP is a client/server protocol running over TCP on port 143.
- IMAP allows multiple clients simultaneously connected to the same mailbox, and through flags stored on the server, different clients accessing the same mailbox at the same or different times can detect state changes made by other clients.
- In other words, it permits a "client" email program to access remote message stores as if they were local.
- For example, email stored on an IMAP server can be manipulated from a desktop computer at home, a workstation at the office, and a notebook computer while travelling, without the need to transfer messages or files back and forth between these computers.
- IMAP can support email serving in three modes:

- ***Offline***
- ***Online***

  Users may connect to the server, look at what email is available, and access it online. This looks to the user very much like having local spool files, but they're on the mail server.
- ***Disconnected operation***

  A mail client connects to the server, can make a "cache" copy of selected messages, and disconnects from the server. The user can then work on the messages offline, and connect to the server later and resynchronize the server status with the cache.



OPERATION OF IMAP

- The mail transfer begins with the client authenticating the user and identifying the mailbox they want to access.
- ***Client Commands***

  LOGIN, AUTHENTICATE, SELECT, EXAMINE, CLOSE, and LOGOUT
- ***Server Responses***

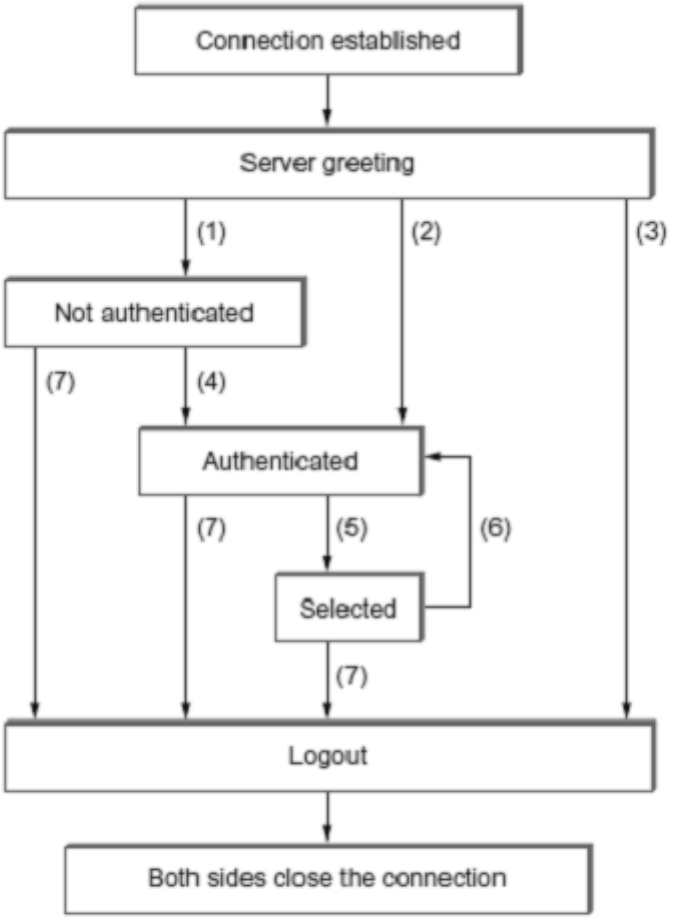
  OK, NO (no permission), BAD (incorrect command),
- When user wishes to FETCH a message, server responds in MIME format.
- Message *attributes* such as size are also exchanged.
- *Flags* are used by client to report user actions.

  SEEN, ANSWERED, DELETED, RECENT

IMAP4

- The latest version is IMAP4. IMAP4 is more powerful and more complex.
- IMAP4 provides the following extra functions:
  - A user can check the e-mail header prior to downloading.
  - A user can search the contents of the e-mail for a specific string of characters prior to downloading.

- A user can partially download e-mail. This is especially useful if bandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.
- A user can create, delete, or rename mailboxes on the mail server.
- A user can create a hierarchy of mailboxes in a folder for e-mail storage.



(1) Connection without preauthentication (OK greeting)
(2) Preauthenticated connection (PREAUTH greeting)
(3) Rejected connection (BYE greeting)
(4) Successful LOGIN or AUTHENTICATE command
(5) Successful SELECT or EXAMINE command
(6) CLOSE command, or failed SELECT or EXAMINE command
(7) LOGOUT command, server shutdown, or connection closed

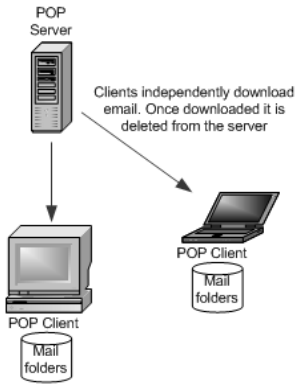
ADVANTAGES OF IMAP

- With IMAP, the primary storage is on the server, not on the local machine.
- Email being put away for storage can be foldered on local disk, or can be foldered on the IMAP server.
- The protocol allows full user of remote folders, including a remote folder hierarchy and multiple inboxes.
- It keeps track of explicit status of messages, and allows for user-defined status.

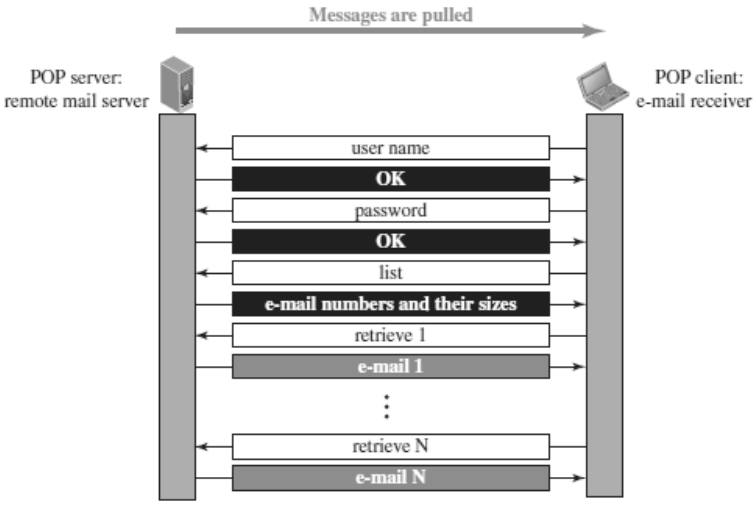
- Supports new mail notification explicitly.
- Extensible for non-email data, like netnews, document storage, etc.
- Selective fetching of individual MIME body parts.
- Server-based search to minimize data transfer.
- Servers may have extensions that can be negotiated.

POST OFFICE PROTOCOL (POP3)

- Post Office Protocol (POP3) is an application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection.
- There are two versions of POP.
  - The first, called *POP2,* became a standard in the mid-80's and requires SMTP to send messages.
  - The current version, POP3, can be used with or without SMTP. POP3 uses TCP/IP port 110.
- POP is a much simpler protocol, making implementation easier.
- POP supports offline access to the messages, thus requires less internet usage time
- POP does not allow search facility.
- In order to access the messages, it is necessary to download them.
- It allows only one mailbox to be created on server.
- It is not suitable for accessing non mail data.
- POP mail moves the message from the email server onto the local computer, although there is usually an option to leave the messages on the email server as well.
- POP treats the mailbox as one store, and has no concept of folders.
- POP works in two modes namely, ***delete* and *keep*** mode.
  - In ***delete mode***, mail is *deleted* from the mailbox after retrieval. The delete mode is normally used when the user is working at their permanent computer and can save and organize the received mail after reading or replying.
  - In ***keep mode***, mail after reading is *kept* in mailbox for later retrieval. The keep mode is normally used when the user accesses her mail away from their primary computer .



- POP3 client is *installed* on the recipient computer and POP server on the mail server.
- Client *opens* a connection to the server using TCP on port 110.
- Client sends username and password to *access* mailbox and to retrieve messages.



POP3 Commands

POP commands are generally abbreviated into codes of three or four letters
The following describes some of the POP commands:

1. **UID -** This command opens the connection
2. **STAT -** It is used to display number of messages currently in the mailbox
3. **LIST -** It is used to get the summary of messages
4. **RETR -**This command helps to select a mailbox to access the messages
5. **DELE -** It is used to delete a message
6. **RSET -** It is used to reset the session to its initial state
7. **QUIT -** It is used to log off the session

DIFFERENCE BETWEEN POP AND IMAP

SNo.	POP	IMAP
1	Generally used to support single client.	Designed to handle multiple clients.
2	Messages are accessed offline.	Messages are accessed online although it also supports offline mode.
3	POP does not allow search facility.	IMAP offers ability to search emails.
4	All the messages have to be downloaded.	It allows selective transfer of messages to the client.
5	Only one mailbox can be created on the server.	Multiple mailboxes can be created on the server.
6	Not suitable for accessing non-mail data.	Suitable for accessing non-mail data i.e. attachment.

7	POP commands are generally abbreviated into codes of three or four letters. Eg. STAT.	IMAP commands are not abbreviated, they are full. Eg. STATUS.
8	It requires minimum use of server resources.	Clients are totally dependent on server.
9	Mails once downloaded cannot be accessed from some other location.	Allows mails to be accessed from multiple locations.
10	The e-mails are not downloaded automatically.	Users can view the headings and sender of e-mails and then decide to download.
11	POP requires less internet usage time.	IMAP requires more internet usage time.

Advantages of IMAP over POP

- IMAP is more powerful and more complex than POP.
- User can *check* the e-mail header prior to downloading.
- User can *search* e-mail for a specific string of characters prior to downloading.
- User can download *partially,* very useful in case of limited bandwidth.
- User can create, delete, or rename *mailboxes* on the mail server.

6. TELNET (TERMINAL NETWORK)

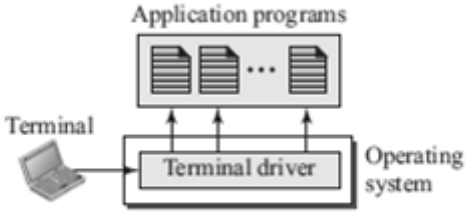
- TELNET is the original remote logging protocol, based on client-server program.
- Telnet provides a connection to the remote computer in such a way that a local terminal appears to be at the remote side.
- TELNET allows us to explain the issues and challenges related to the concept of remote logging.
- Network administrators often use TELNET for diagnostic and debugging purposes.
- TELNET requires a logging name and password.
- It is vulnerable to hacking because it sends all data including the password in plaintext (not encrypted).
- A hacker can eavesdrop and obtain the logging name and password.Because of this security issue, the use of TELNET has diminished.

TYPES OF TELNET LOGGING

There are two types of TELNET logging:

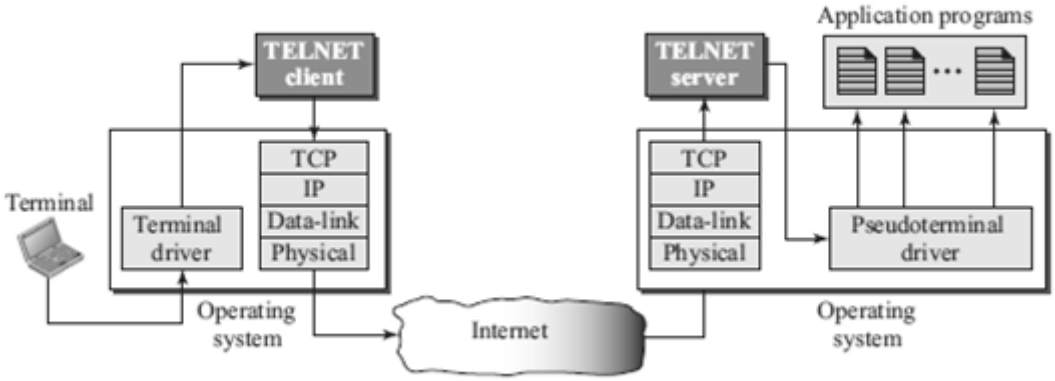
Local Logging and Remote Logging

Local Login



- When a user logs into a local system, it is called local logging.
- As a user types at a terminal or at a workstation running a terminal emulator, the keystrokes are accepted by the terminal driver.
- The terminal driver passes the characters to the operating system.
- The operating system, in turn, interprets the combination of characters and invokes the desired application program or utility.

Remote Logging



- When a user wants to access an application program or utility located on a remote machine, they perform remote logging.
- Remote Logging uses TELNET client and TELENT server programs.
- The user sends the keystrokes to the terminal driver where the local operating system accepts the characters but does not interpret them.
- The characters are sent to the TELNET client, which transforms the characters into a universal character set called Network Virtual Terminal (NVT) characters and delivers them to the local TCP/IP stack.
- The commands or text, in NVT form, travel through the Internet and arrive at the TCP/IP stack at the remote machine.
- The characters are delivered to the operating system and passed to the TELNET server, which changes the characters to the corresponding characters understandable by the remote computer.
- The characters cannot be passed directly to the operating system because the remote operating system is not designed to receive characters from a TELNET server; it is designed to receive characters from a terminal driver.

- A piece of software called pseudoterminal driver, is added to this,which pretends that the characters are coming from a terminal.
- The operating system then passes the characters to the appropriate application program.

TELENT OPTIONS

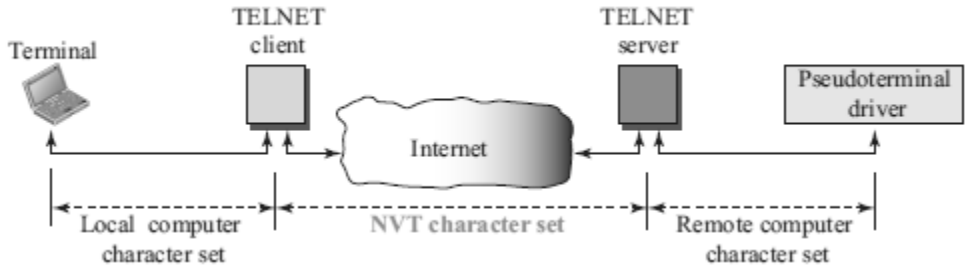
- TELNET lets the client and server negotiate options before or during the use of the service.
- Options are extra features available to a user with a more sophisticated terminal.
- Users with simpler terminals can use default features.

TELENT COMMANDS

Command	*Meaning*	*Command*	*Meaning*
open	Connect to a remote computer	**set**	Set the operating parameters
close	Close the connection	**status**	Display the status information
display	Show the operating parameters	**send**	Send special characters
mode	Change to line or character mode	**quit**	Exit TELNET

NETWORK VIRTUAL TERMINAL (NVT)

- The mechanism to access a remote computer is complex.
- We are dealing with heterogeneous systems.
- This is because every computer and its operating system accepts a special combination of characters as tokens.
- For example, the end-of-file token in a computer running the DOS operating system is Ctrl+z, while the UNIX operating system recognizes Ctrl+d.
- If we want to access any remote computer in the world, we must first know what type of computer we will be connected to, and we must also install the specific terminal emulator used by that computer.
- TELNET solves this problem by defining a universal interface called the Network Virtual Terminal (NVT) character set.
- Via this interface, the client TELNET translates characters (data or commands) that come from the local terminal into NVT form and delivers them to the network.
- The server TELNET, on the other hand, translates data and commands from NVT form into the form acceptable by the remote computer.



NVT Character Format

- NVT uses two sets of characters, one for data and one for control.
- For data, NVT normally uses what is called NVT ASCII. This is an 8-bit character set in which the seven lowest order bits are the same as ASCII and the highest order bit is 0.
- To send control characters between computers , NVT uses an 8-bit character set in which the highest order bit is set to 1.

NVT character format

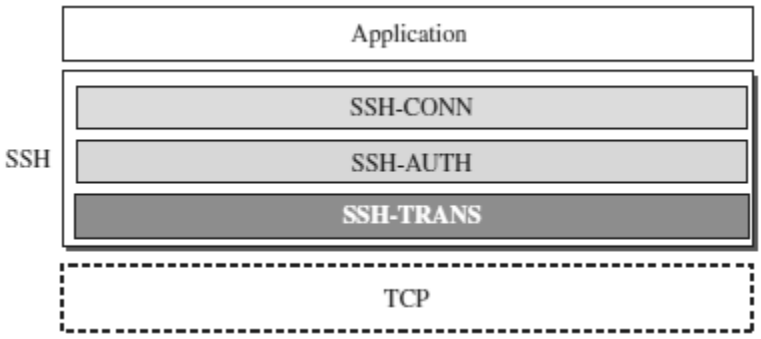
7. SSH (SECURE SHELL)

- **Secure Shell (SSH)** is a secure application program that can be used today for several purposes such as remote logging and file transfer, it was originally designed to replace TELNET.
- There are two versions of SSH: SSH-1 and SSH-2, which are totally incompatible. The first version, SSH-1, is now deprecated because of security flaws in it.

SSH COMPONENTS

SSH is an application-layer protocol with three components:

1. SSH Transport-Layer Protocol (SSH-TRANS)
2. SSH Authentication Protocol (SSH-AUTH)
3. SSH Connection Protocol (SSH-CONN)



SSH Transport-Layer Protocol (SSH-TRANS)

- SSH first uses a protocol that creates a secured channel on top of the TCP.
- This new layer is an independent protocol referred to as SSH-TRANS.
- When the procedure implementing this protocol is called, the client and server first use the TCP protocol to establish an insecure connection.
- Then they exchange several security parameters to establish a secure channel on top of the TCP.

Services provided by this protocol:

1. Privacy or confidentiality of the message exchanged
2. Data integrity, which means that it is guaranteed that the messages exchanged between the client and server are not changed by an intruder
3. Server authentication, which means that the client is now sure that the server is the one that it claims to be
4. Compression of the messages, which improves the efficiency of the system and makes attack more difficult

SSH Authentication Protocol (SSH-AUTH)

- After a secure channel is established between the client and the server and the server is authenticated for the client.
- SSH can call another procedure that can authenticate the client for the server.
- This layer defines a number of authentication tools similar to the ones used in SSL.
- Authentication starts with the client, which sends a request message to the server.
- The request includes the user name, server name, the method of authentication, and the required data.
- The server responds with either a success message, which confirms that the client is authenticated, or a failed message, which means that the process needs to be repeated with a new request message.

SSH Connection Protocol (SSH-CONN)

- After the secured channel is established and both server and client are authenticated for each other, SSH can call a piece of software that implements the third protocol, SSHCONN.
- One of the services provided by the SSH-CONN protocol is multiplexing.
- SSH-CONN takes the secure channel established by the two previous protocols and lets the client create multiple logical channels over it.
- Each channel can be used for a different purpose, such as remote logging, file transfer, and so on.

SSH APPLICATIONS

SSH is a general-purpose protocol that provides a secure connection between a client and server.

SSH for Remote Logging

- Several free and commercial applications use SSH for remote logging.
- Among them, we can mention PuTTy, by Simon Tatham, which is a client SSH program that can be used for remote logging.
- Another application program is Tectia, which can be used on several platforms.

SSH for File Transfer

- One of the application programs that is built on top of SSH for file transfer is the *Secure File Transfer Program* (*sftp*).

- The *sftp* application program uses one of the channels provided by the SSH to transfer files.
- Another common application is called *Secure Copy* (*scp*).
- This application uses the same format as the UNIX copy command, *cp,* to copy files.

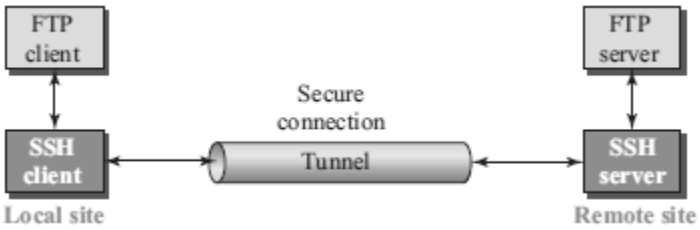
Port Forwarding

- One of the interesting services provided by the SSH protocol is port forwarding.
- We can use the secured channels available in SSH to access an application program that does not provide security services.
- Applications such as TELNET and Simple Mail Transfer Protocol (SMTP),can use the services of the SSH port forwarding mechanism.
- The SSH port forwarding mechanism creates a tunnel through which the messages belonging to other protocols can travel.
- For this reason, this mechanism is sometimes referred to as SSH *tunneling*.

SSH PACKET FORMAT

- The length field defines the length of the packet but does not include the padding.
- The Padding field is added to the packet to make the attack on the security provision more difficult.
- The type field designates the type of the packet used in different SSH protocols.
- The data field is the data transferred by the packet in different protocols.
- The CRC field is used for error detection.

SECURING FTP APPLICATIONS USING SSH

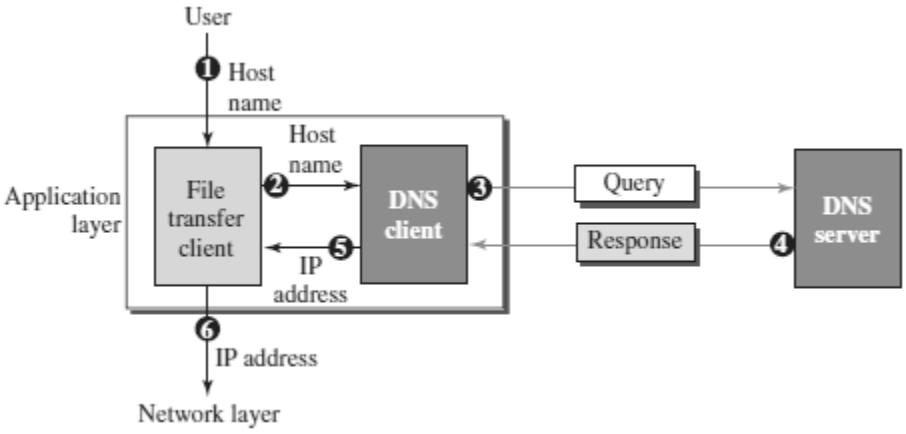


- The FTP client can use the SSH client on the local site to make a secure connection
- with the SSH server on the remote site.
- Any request from the FTP client to the FTP server is carried through the tunnel provided by the SSH client and server.
- Any response from the FTP server to the FTP client is also carried through the tunnel provided by the SSH client and server.

8. DNS (DOMAIN NAME SYSTEM)

- Domain Name System was designed in 1984.
- DNS is used for name-to-address mapping.
- The DNS provides the protocol which allows clients and servers to communicate with each other.
- Eg: Host name like www.yahoo.com is translated into numerical IP addresses like 207.174.77.131
- Domain Name System (DNS) is a distributed database used by TCP/IP applications to map between hostnames and IP addresses and to provide electronic mail routing information.
- Each site maintains its own database of information and runs a server program that other systems across the Internet can query.

WORKING OF DNS



The following six steps shows the working of a DNS. It maps the host name to an IP address:

1. The user passes the host name to the file transfer client.
2. The file transfer client passes the host name to the DNS client.
3. Each computer, after being booted, knows the address of one DNS server. The DNS client sends a message to a DNS server with a query that gives the file transfer server name using the known IP address of the DNS server.
4. The DNS server responds with the IP address of the desired file transfer server.
5. The DNS server passes the IP address to the file transfer client.
6. The file transfer client now uses the received IP address to access the file transfer server.

NAME SPACE

- To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP address.

- The names must be unique because the addresses are unique.
- A name space that maps each address to a unique name can be organized in two ways: ***flat (*or) *hierarchical.***

Flat Name Space

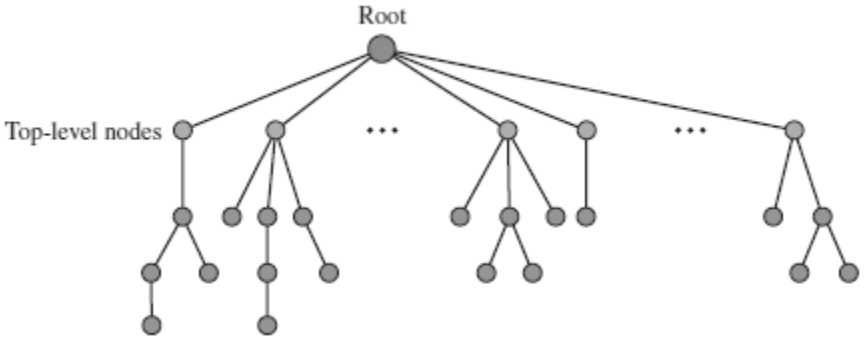
- In a flat name space, a name is assigned to an address.
- A name in this space is a sequence of characters without structure.
- The main disadvantage of a flat name space is that it cannot be used in a large system such as Internet because it must be centrally controlled to avoid ambiguity and duplication.

Hierarchical Name Space

- In a hierarchical name space, each name is made of several parts.
- The first part can define the organization, the second part can define the name, the third part can define departments, and so on.
- In this case, the authority to assign and control the name spaces can be decentralized.
- A central authority can assign the part of the name that defines the nature of the organization and the name.
- The responsibility for the rest of the name can be given to the organization itself. Suffixes can be added to the name to define host or resources.
- The management of the organization need not worry that the prefix chosen for a host is taken by another organization because even if part of an address is the same, the whole address is different.
- The names are unique without the need to be assigned by a central authority.
- The central authority controls only part of the name, not the whole name.

DOMAIN NAME SPACE

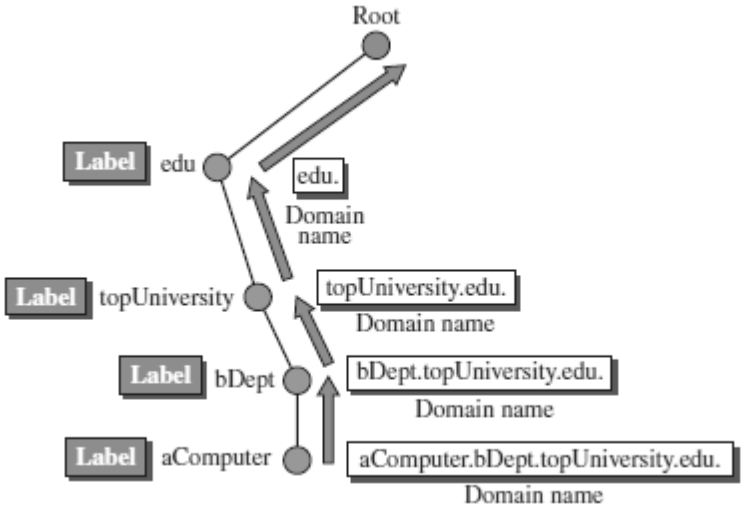
- To have a hierarchical name space, a domain name space was designed. In this design, the names are defined in an inverted-tree structure with the root at the top.
- Each node in the tree has a label, which is a string with a maximum of 63 characters.
- The root label is a null string.
- DNS requires that children of a node have different labels, which guarantees the uniqueness of the domain names.



- Each node in the tree has a **label**, which is a string with a maximum of 63 characters.
- The root label is a null string (empty string). DNS requires that children of a node (nodes that branch from the same node) have different labels, which guarantees the uniqueness of the domain names.

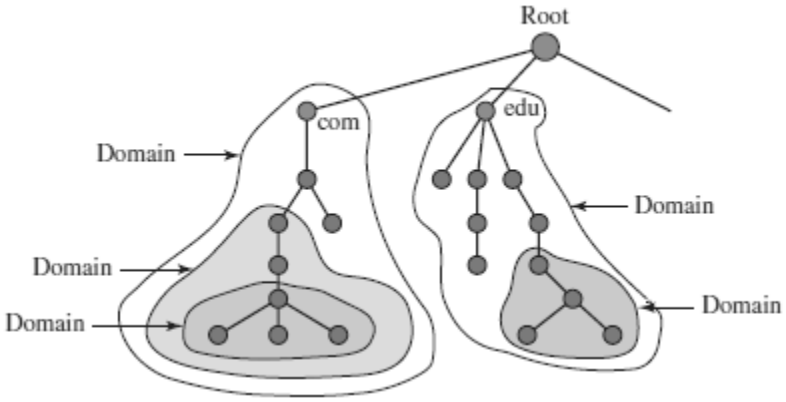
Domain Name

- Each node in the tree has a label called as domain name.
- A full domain name is a sequence of labels separated by dots (**.**)
- The domain names are always read from the node up to the root.
- The last label is the label of the root (null).
- This means that a full domain name always ends in a null label, which means the last character is a dot because the null string is nothing.
- If a label is terminated by a null string, it is called a ***fully qualified domain name (FQDN)***.
- If a label is not terminated by a null string, it is called a ***partially qualified domain name (PQDN).***



Domain

- A domain is a subtree of the domain name space.
- The name of the domain is the domain name of the node at the top of the sub-tree.
- A domain may itself be divided into domains.

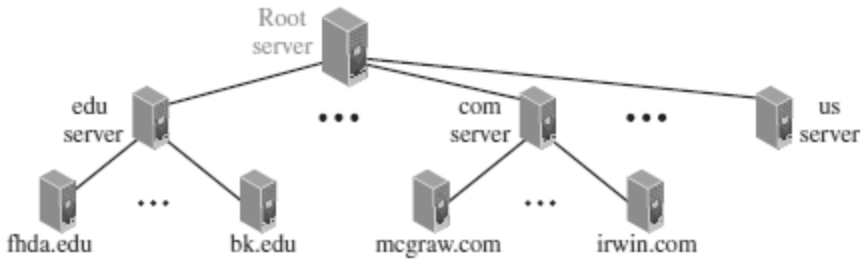


DISTRIBUTION OF NAME SPACE

- The information contained in the domain name space must be stored.
- But it is very inefficient and also not reliable 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 reliable because any failure makes the data inaccessible.
- The solution to these problems is to distribute the information among many computers called ***DNS servers***.

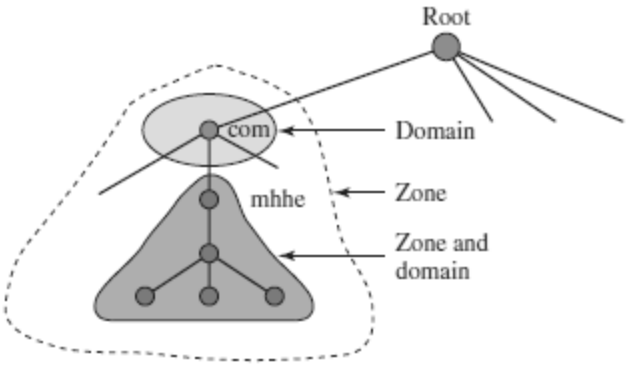
HIERARCHY OF NAME SERVERS

- The way to distribute information among DNS servers is to divide the whole space into many domains based on the first level.
- Let the root stand-alone and create as many domains as there are first level nodes.
- Because a domain created this way could be very large,
- DNS allows domains to be divided further into smaller domains.
- Thus we have a hierarchy of servers in the same way that we have a hierarchy of names.



ZONE

- What a server is responsible for, or has authority over, is called a ***zone***.
- The server makes a database called a *zone* file and keeps all the information for every node under that domain.
- If a server accepts responsibility for a domain and does not divide the domains into smaller domains, the domain and zone refer to the same thing.
- But if a server divides its domain into sub domains and delegates parts of its authority to other servers, domain and zone refer to different things.
- The information about the nodes in the sub domains is stored in the servers at the lower levels, with the original server keeping some sort of references to these lower level servers.
- But still, 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.



ROOT SERVER

- A root sever 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.
- Currently there are more than 13 root servers, each covering the whole domain name space.
- The servers are distributed all around the world.

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.
  - Primary Servers are responsible for creating, maintaining, and updating the zone file.
  - Primary Server stores the zone file on a local disc.
- 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 disc.
- If updating is required, it must be done by the primary server, which sends the updated version to the secondary.

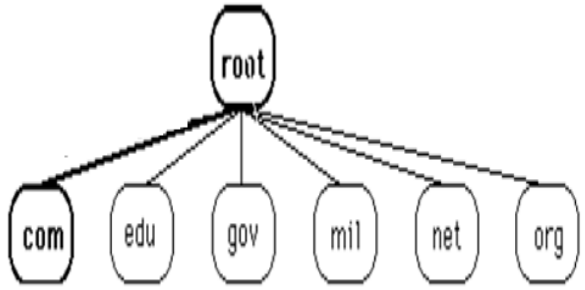
- A primary server loads all information from the disk file; the secondary server loads all information from the primary server.

DNS IN THE INTERNET

- DNS is a protocol that can be used in different platforms.
- In the Internet, the domain name space (tree) is divided into three different sections - ***Generic domains, Country domains, and Inverse domain***.

Generic Domains

- The generic domains define registered hosts according to their generic behavior.
- Each node in the tree defines a domain, which is an index to the domain name space database.
- The first level in the generic domains section allows seven possible three character levels.
- These levels describe the organization types as listed in following table**.**



.COM	Commercial Organizations
.EDU	Educational institutions
.GOV	Government institutions
.MIL	Military groups
.NET	Major network support centers
.INT	International Organizations
.ORG	Nonprofit Organizations

Country Domains

- The country domains section follows the same format as the generic domains but uses two characters for country abbreviations
- E.g.; ***in*** for ***India***, ***us*** for ***United States*** etc) in place of the three character organizational abbreviation at the first level.
- Second level labels can be organizational, or they can be more specific, national designation.
- India for example, uses state abbreviations as a subdivision of the country domain us. (e.g., ca.in.)

Inverse Domains

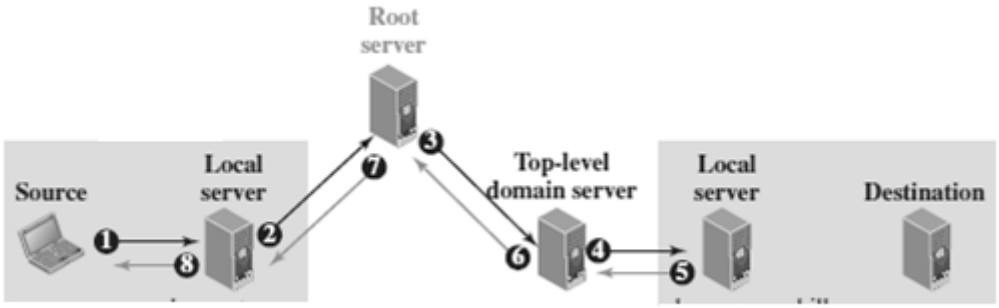
- Mapping an address to a name is called Inverse domain.

- The client can send an IP address to a server to be mapped to a domain name and it is called *PTR(Pointer) query*.
- To answer queries of this kind, DNS uses the inverse domain

DNS RESOLUTION

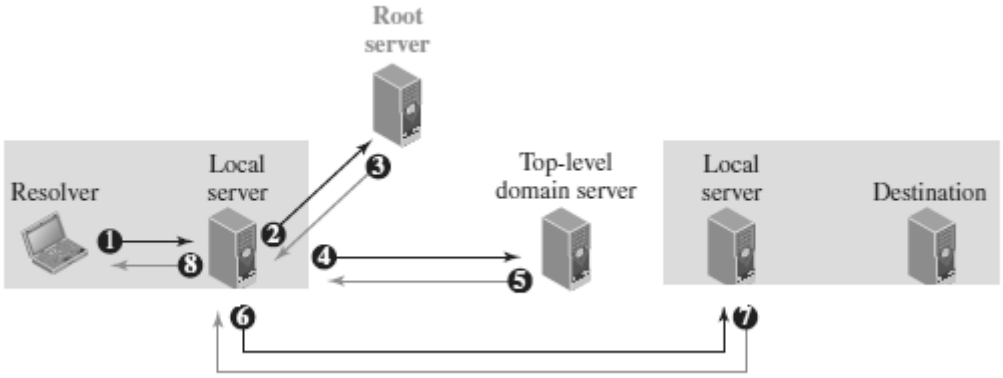
- Mapping a name to an address or an address to a name is called name address resolution.
- DNS is designed as a client server application.
- A host that needs to map an address to a name or a name to an address calls a DNS client named a ***Resolver***.
- The Resolver accesses the closest DNS server 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.
- A resolution can be either ***recursive or iterative***.

Recursive Resolution



- The application program on the source host calls the DNS resolver (client) to find the IP address of the destination host. The resolver, which does not know this address, sends the query to the local DNS server of the source (Event 1)
- The local server sends the query to a root DNS server (Event 2)
- The Root server sends the query to the top-level-DNS server(Event 3)
- The top-level DNS server knows only the IP address of the local DNS server at the destination. So it forwards the query to the local server, which knows the IP address of the destination host (Event 4)
- The IP address of the destination host is now sent back to the top-level DNS server(Event 5) then back to the root server (Event 6), then back to the source DNS server, which may cache it for the future queries (Event 7), and finally back to the source host (Event 8).

Iterative Resolution



- In iterative resolution**,** each server that does not know the mapping, sends the IP address of the next server back to the one that requested it.
- The iterative resolution takes place between two local servers.
- The original resolver gets the final answer from the destination local server.
- The messages shown by Events 2, 4, and 6 contain the same query.
- However, the message shown by Event 3 contains the IP address of the top-level domain server.
- The message shown by Event 5 contains the IP address of the destination local DNS server
- The message shown by Event 7 contains the IP address of the destination.
- When the Source local DNS server receives the IP address of the destination, it sends it to the resolver (Event 8).

DNS CACHING

- 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.
- 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 resolve the problem.
- However, to 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***.
- Caching speeds up resolution. Reduction of this search time would increase efficiency, 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.

DNS RESOURCE RECORDS (RR)

- The zone information associated with a server is implemented as a set of *resource records*.
- In other words, a name server stores a database of resource records.
- A *resource record* is a 5-tuple structure :

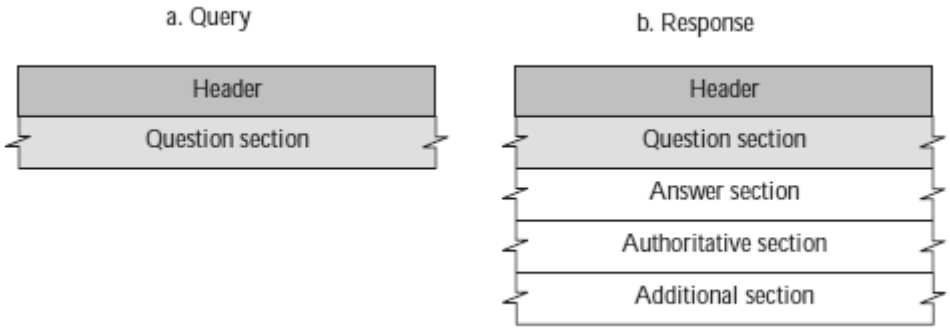
  (Domain Name, Type, Class, TTL, Value)
- The domain name identifies the resource record.
- The type defines how the value should be interpreted.
- The value defines the information kept about the domain name.
- The TTL defines the number of seconds for which the information is valid.
- The class defines the type of network

Types of Resource Records

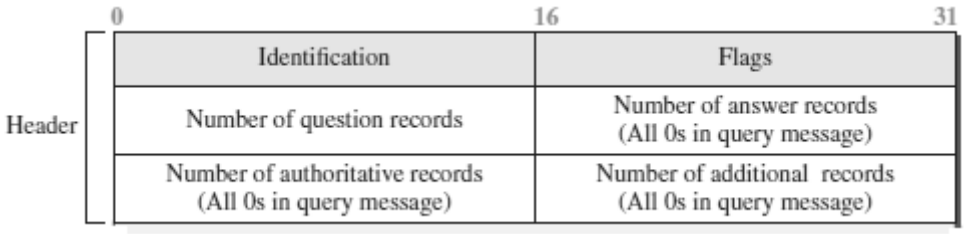
Type	*Interpretation of value*
A	A 32-bit IPv4 address
NS	Identifies the authoritative servers for a zone
CNAME	Defines an alias for the official name of a host
SOA	Marks the beginning of a zone
MX	Redirects mail to a mail server
AAAA	An IPv6 address

DNS MESSAGES

- ➢ DNS has two types of messages: query and response.
- ➢ Both types have the same format.
- ➢ The query message consists of a header and question section.
- ➢ The response message consists of a header, question section, answer section, authoritative section, and additional section .



- ➢ **Header**
  - Both query and response messages have the same header format with some fields set to zero for the query messages.
  - The header fields are as follows:



- The identification field is used by the client to match the response with the query.
- The flag field defines whether the message is a query or response. It also includes status of error.
- The next four fields in the header define the number of each record type in the message.

➢ **Question Section**
- The question section consists of one or more question records. It is present in both query and response messages.

➢ **Answer Section**
- The answer section consists of one or more resource records. It is present only in response messages.

➢ **Authoritative Section**
- The authoritative section gives information (domain name) about one or more authoritative servers for the query.

➢ **Additional Information Section**
- The additional information section provides additional information that may help the resolver.

DNS CONNECTIONS

➢ DNS can use either UDP or TCP.
➢ In both cases the well-known port used by the server is port 53.
➢ UDP is used when the size of the response message is less than 512 bytes because most UDP packages have a 512-byte packet size limit.
➢ If the size of the response message is more than 512 bytes, a TCP connection is used.

DNS REGISTRARS

➢ New domains are added to DNS through a *registrar*. A fee is charged.
➢ A registrar first verifies that the requested domain name is unique and then enters it into the DNS database.
➢ Today, there are many registrars; their names and addresses can be found at
http://www.intenic.net
➢ To register, the organization needs to give the name of its server and the IP address of the server.
➢ For example, a new commercial organization named *wonderful* with a server named *ws* and IP address 200.200.200.5, needs to give the following information to one of the registrars:
Domain name: *ws.wonderful.com* ***IP address:*** *200.200.200.5*

DDNS (DYNAMIC DOMAIN NAME SYSTEM)

- In DNS, when there is a change, such as adding a new host, removing a host, or changing an IP address, the change must be made to the DNS master file.
- The DNS master file must be updated dynamically.
- The ***Dynamic Domain Name System (DDNS)*** is used for this purpose.
- In DDNS, when a binding between a name and an address is determined, the information is sent to a primary DNS server.
- The primary server updates the zone.
- The secondary servers are notified either actively or passively.
- In active notification, the primary server sends a message to the secondary servers about the change in the zone, whereas in passive notification, the secondary servers periodically check for any changes.
- In either case, after being notified about the change, the secondary server requests information about the entire zone (called the *zone transfer*).
- To provide security and prevent unauthorized changes in the DNS records, DDNS can use an authentication mechanism.

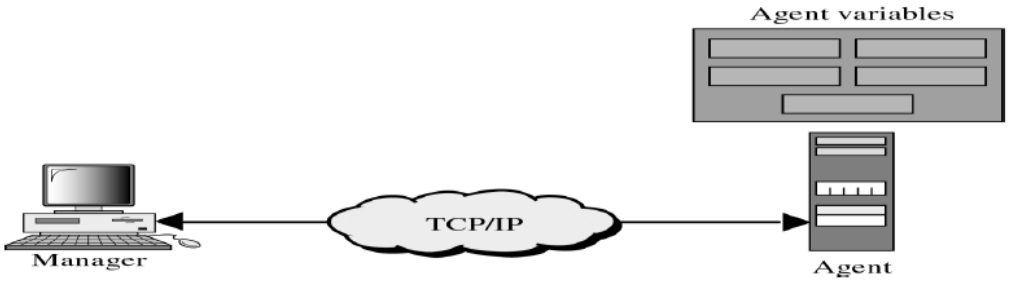
DNS SECURITY

- DNS is one of the most important systems in the Internet infrastructure; it provides crucial services to Internet users.
- Applications such as Web access or e-mail are heavily dependent on the proper operation of DNS.
- DNS can be attacked in several ways including:
  - ***Attack on Confidentiality*** - The attacker may read the response of a DNS ***server*** to find the nature or names of sites the user mostly accesses. This type of information can be used to find the user's profile. To prevent this attack, DNS messages need to be confidential.
  - ***Attack on authentication*** and ***integrity*** - The attacker may intercept the response of a DNS server and change it or create a totally new bogus ***response*** to direct the user to the site or domain the attacker wishes the user to access. This type of attack can be prevented using message origin authentication and message integrity.
  - ***Attack on denial-of-service*** - The attacker may flood the DNS server to overwhelm it or eventually crash it. This type of attack can be prevented using the provision against denial-of-service attack**.**

- To protect DNS, IETF has devised a technology named ***DNS Security (DNSSEC)*** that ***provides message origin authentication and message integrity*** using a security service called ***digital signature***.
- DNSSEC, however, ***does not provide confidentiality*** for the DNS messages.
- There is ***no specific protection against the denial-of-service attack*** in the specification of DNSSEC. However, the caching system protects the upper-level servers against this attack to some extent.

9. SNMP (SIMPLE NETWORK MANAGEMENT PROTOCOL)

- The **Simple Network Management Protocol (SNMP)** is a framework for managing devices in an internet using the TCP/IP protocol suite.
- SNMP is an application layer protocol that monitors and manages routers, distributed over a network.
- It provides a set of operations for monitoring and managing the internet.
- SNMP uses services of UDP on two well-known ports: 161 (Agent) and 162 (manager).
- SNMP uses the concept of ***manager*** and ***agent***.



SNMP MANAGER

- A manager is a host that runs the SNMP client program
- The manager has access to the values in the database kept by the agent.
- A manager checks the agent by requesting the information that reflects the behavior of the agent.
- A manager also forces the agent to perform a certain function by resetting values in the agent database.
- For example, a router can store in appropriate variables the number of packets received and forwarded.
- The manager can fetch and compare the values of these two variables to see if the router is congested or not.

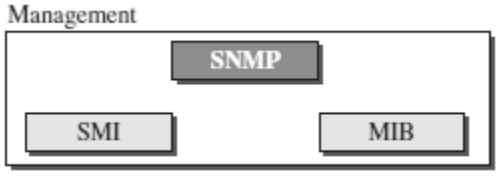
SNMP AGENT

- The agent is a router that runs the SNMP server program.
- The agent is used to keep the information in a database while the manager is used to access the values in the database.
- For example, a router can store the appropriate variables such as a number of packets received and forwarded while the manager can compare these variables to determine whether the router is congested or not.
- Agents can also contribute to the management process.
- A server program on the agent checks the environment, if something goes wrong, the agent sends a warning message to the manager.

SNMP MANAGEMENT COMPONENTS

- Management of the internet is achieved through simple interaction between a manager and agent.
- Management is achieved through the use of two protocols:

- Structure of Management Information (SMI)
- Management Information Base (MIB).



Structure of Management Information (SMI)

- To use SNMP, we need rules for naming objects.
- SMI is a protocol that defines these rules.
- SMI is a guideline for SNMP
- It emphasizes three attributes to handle an object: name, data type, and encoding method.
- Its functions are:
  - ❖ To name objects.
  - ❖ To define the type of data that can be stored in an object.
  - ❖ To show how to encode data for transmission over the network.

Name

- ✓ SMI requires that each managed object (such as a router, a variable in a router, a value,etc.) have a unique name. To name objects globally.
- ✓ SMI uses an ***object identifier*****,** which is a hierarchical identifier based on a tree structure.
- ✓ The tree structure starts with an unnamed root. Each object can be defined using a sequence of integers separated by dots.
- ✓ The tree structure can also define an object using a sequence of textual names separated by dots.

Type of data

- ✓ The second attribute of an object is the type of data stored in it.
- ✓ To define the data type, SMI uses **Abstract Syntax Notation One (ASN.1)** definitions.
- ✓ SMI has two broad categories of data types: *simple* and *structured.*
- ✓ The **simple data types** are atomic data types. Some of them are taken directly from ASN.1; some are added by SMI.
- ✓ SMI defines two **structured data types:** *sequence* and *sequence of*.
  - ▪ ***Sequence -*** A *sequence* data type is a combination of simple data types, not necessarily of the same type.
  - ▪ ***Sequence of -*** A *sequence of* data type is a combination of simple data types all of the same type or a combination of sequence data types all of the same type.

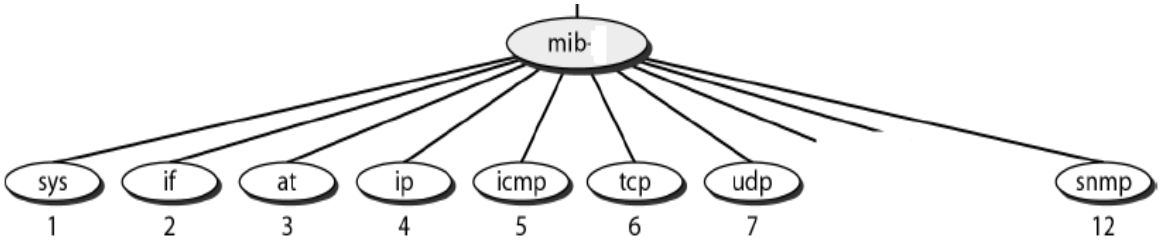
Encoding data

- ✓ SMI uses another standard, ***Basic Encoding Rules (BER),*** to encode data to be transmitted over the network.
- ✓ BER specifies that each piece of data be encoded in triplet format (TLV): tag, length, value

Management Information Base (MIB)

The Management Information Base (MIB) is the second component used in network management.

- Each agent has its own MIB, which is a *collection* of objects to be managed.
- MIB classifies objects under groups.



MIB Variables

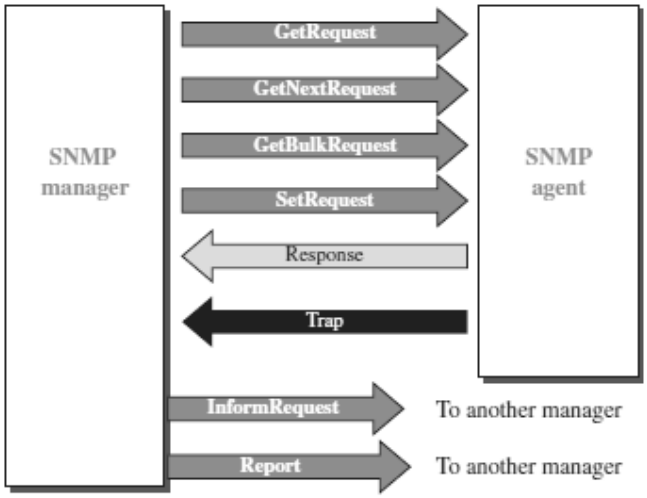
MIB variables are of two types namely *simple* and *table*.

- Simple variables are accessed using *group-id* followed by *variable-id* and 0
- Tables are ordered as *column-row* rules, i.e., column by column from top to bottom. Only *leaf* elements are accessible in a table type.

SNMP MESSAGES/PDU

SNMP is request/reply protocol that supports various operations using PDUs.
SNMP defines eight types of protocol data units (or PDUs):

GetRequest, GetNext-Request, GetBulkRequest, SetRequest, Response, Trap, InformRequest,* and *Report



GetRequest

- The GetRequest PDU is sent from the manager (client) to the agent (server) to retrieve the value of a variable or a set of variables.

GetNextRequest

- The GetNextRequest PDU is sent from the manager to the agent to retrieve the value of a variable.

GetBulkRequest

- The GetBulkRequest PDU is sent from the manager to the agent to retrieve a large amount of data. It can be used instead of multiple GetRequest and GetNextRequest PDUs.

SetRequest

- The SetRequest PDU is sent from the manager to the agent to set (store) a value in a variable.

Response

- The Response PDU is sent from an agent to a manager in response to GetRequest or GetNextRequest. It contains the value(s) of the variable(s) requested by the manager.

Trap

- The Trap PDU is sent from the agent to the manager to report an event. For example, if the agent is rebooted, it informs the manager and reports the time of rebooting.

InformRequest

- The InformRequest PDU is sent from one manager to another remote manager to get the value of some variables from agents under the control of the remote manager. The remote manager responds with a Response PDU.

Report

- The Report PDU is designed to report some types of errors between managers.
