

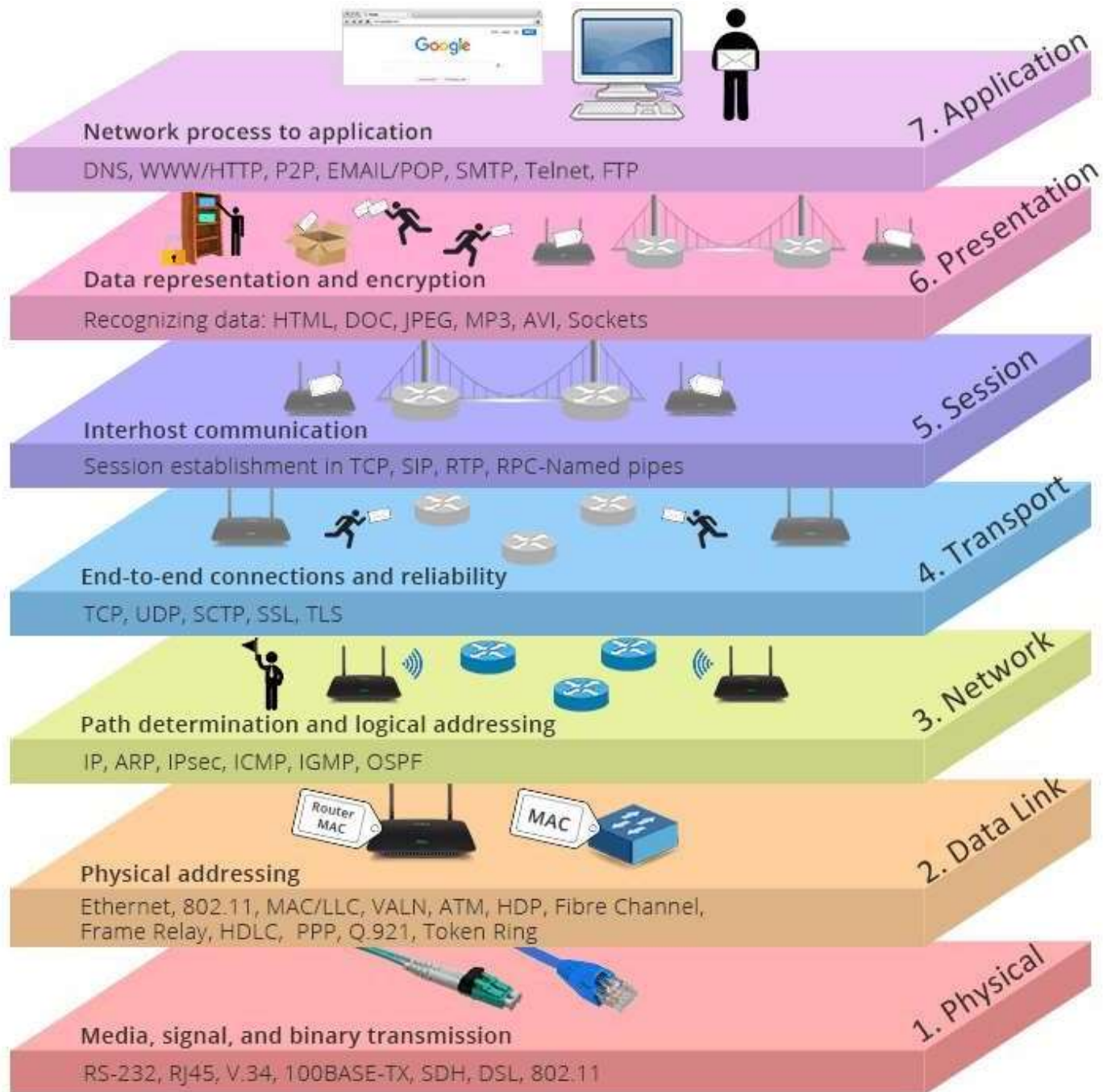
# **UNIT - V**

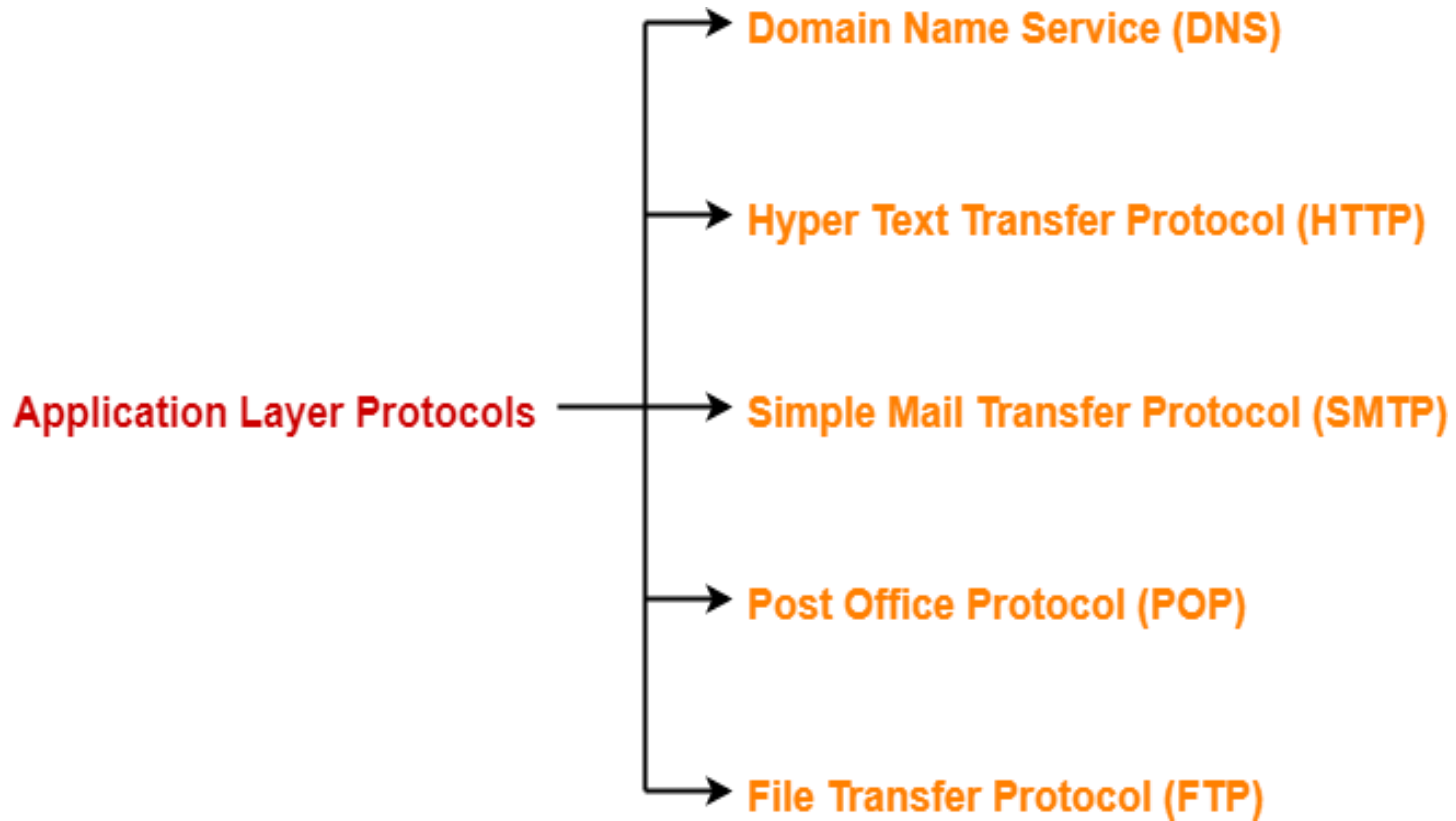
## **The Application Layer**

# Contents

## Application Layer-

- Introduction
- Providing services
- Applications layer paradigms: Client server model, HTTP, E-mail, WWW, TELNET, DNS
- RSA algorithm





# Application Layer

- The application layer is responsible for providing services to the user.
- The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, file access and transfer, access to system resources, surfing the world wide web, and network management.

## Application Layer Services

- 1. Mail Services:** This layer provides the basis for E-mail forwarding and storage.
- 2. Network Virtual Terminal:** It allows a user to log on to a remote host. The application creates software emulation of a terminal at the remote host. User's computer talks to the software terminal which in turn talks to the host and vice versa. Then the remote host believes it is communicating with one of its own terminals and allows user to log on.
- 3. Directory Services:** This layer provides access for global information about various services.
- 4. File Transfer, Access and Management (FTAM):** It is a standard mechanism to access files and manages it. Users can access files in a remote computer and manage it. They can also retrieve files from a remote computer.

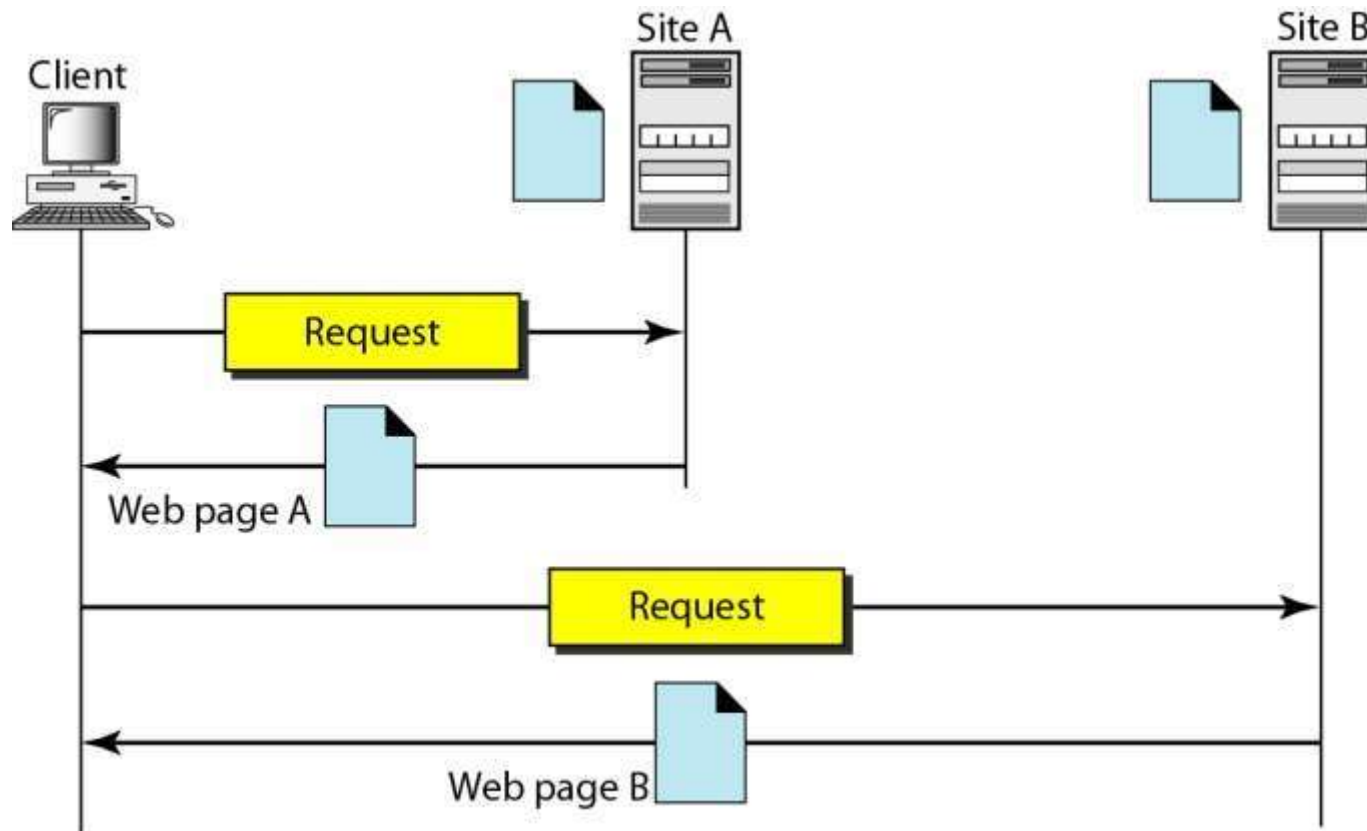
## Client-Server Model:

Two remote application processes can communicate mainly in two different fashions:

- **Peer-to-peer:** Both remote processes are executing at same level and they exchange data using some shared resource.
- **Client-Server:** One remote process acts as a Client and requests some resource from another application process acting as Server.

In client-server model, any process can act as Server or Client. It is not the type of machine, size of the machine, or its computing power which makes it server; it is the ability of serving request that makes a machine a server.

The **WWW** today is a distributed client/server service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called sites as shown in fig.





## **Client(Browser)**

- A variety of vendors offer commercial browsers that interpret and display a Web document ,and all use nearly the same architecture.
- Each browser usually consists of three parts: a controller, client protocol, 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. The interpreter can be HTML, Java, or JavaScript, depending on the type of document

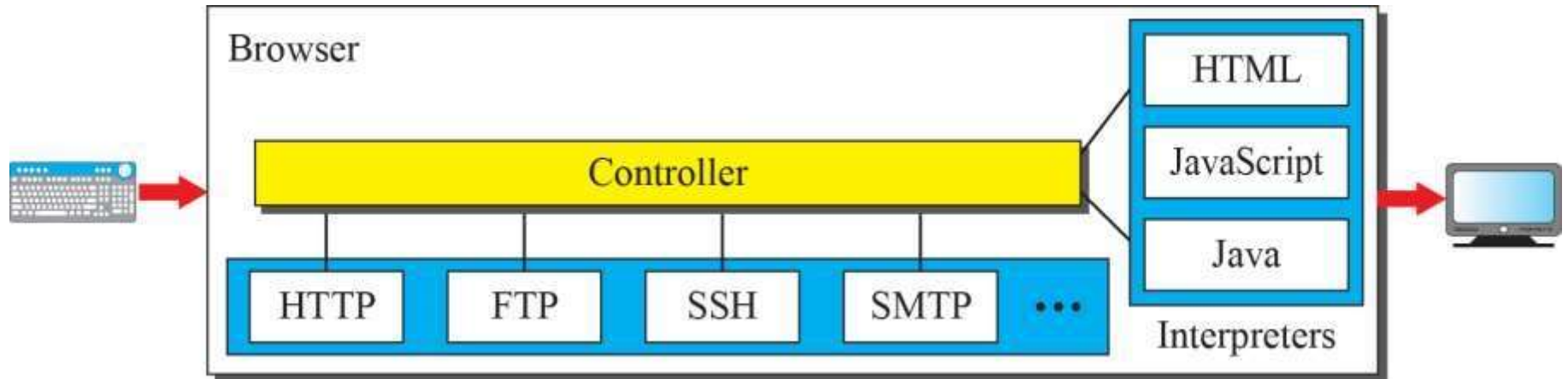
The client protocol can be one of the protocols described previously such as FTP or HTTP.

## **Server**

The Web page is stored at the server. Each time a client request arrives, the corresponding document is sent to the client. To improve efficiency, servers normally store requested files in a cache in memory; memory is faster to access than disk. A server can also become more efficient through multithreading or multiprocessing. In this case, a server can answer more than one

# *Browser*

---



## Uniform Resource Locator

- A client that wants to access a Web page needs the address. To facilitate the access of documents distributed throughout the world, HTTP uses locators. The uniform resource locator (URL) is a standard for specifying any kind of information on the Internet. The URL defines four things: protocol, host computer, port, and path.
- The protocol is the client/server program used to retrieve the document. Many different protocols can retrieve a document; among them are FTP or HTTP. The most common today is HTTP.



- The host is the computer on which the information is located, although the name of the computer can be an alias. Web pages are usually stored in computers, and computers are given alias names that usually begin with the characters "www".
- The URL can optionally contain the port number of the server. If the port is included, it is inserted between the host and the path, and it is separated from the host by a colon.
- Path is the pathname of the file where the information is located. Note that the path can itself contain slashes that, in the UNIX operating system, separate the directories from the subdirectories and files.

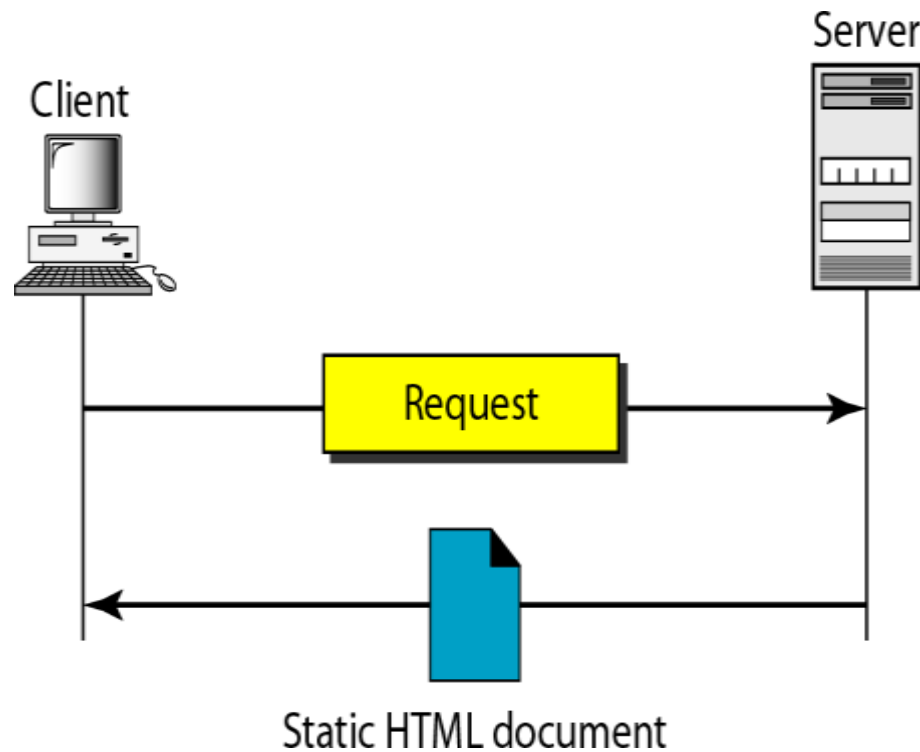


## Web Documents

*The documents in the WWW can be grouped into three broad categories: **static**, **dynamic**, and **active**. The category is based on the time at which the contents of the document are determined.*

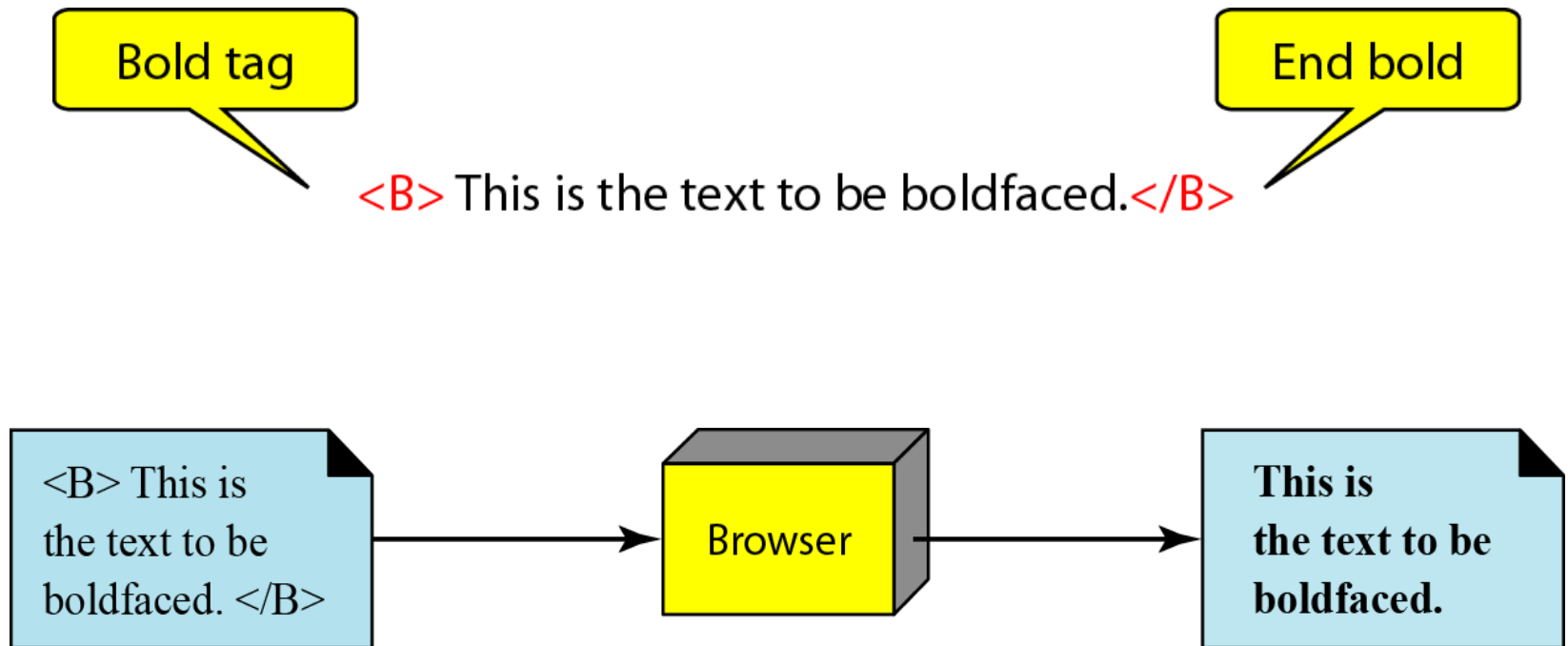
## Static Documents

Static documents are fixed-content documents that are created and stored in a server. The client can get only a copy of the document. When a client accesses the document, a copy of the document is sent. The user can then use a browsing program to display the document.



# *HTML*

Hypertext Markup Language (HTML) is a language for creating Web pages.



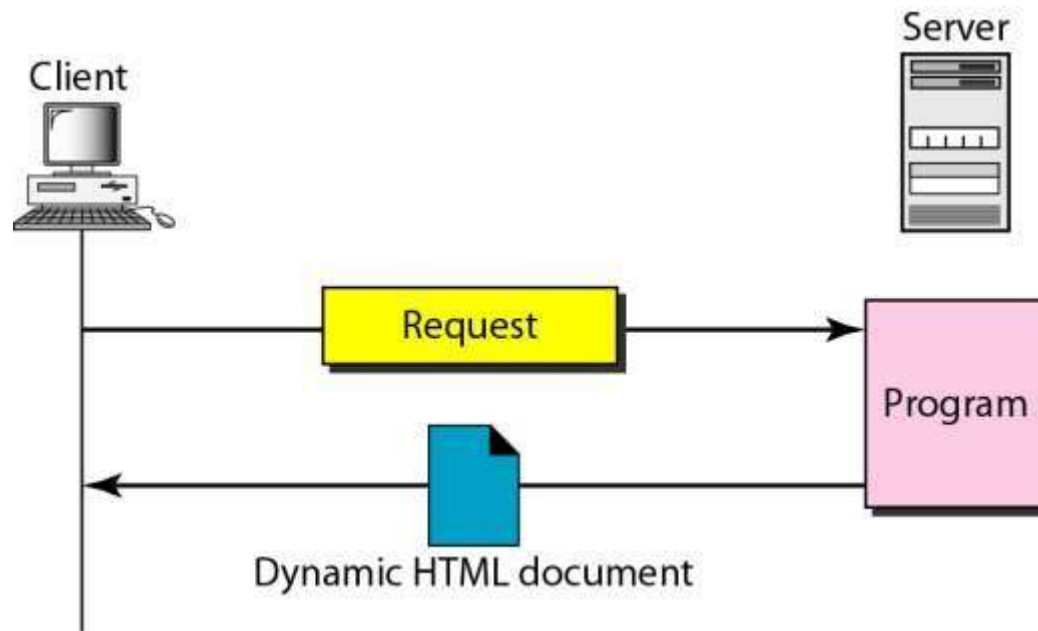


## Dynamic Documents

- 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 output of the program or script as a response to the browser that requested the document.
- 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. The client can ask the server to run a program such as the date program in UNIX and send the result of the program to the client.

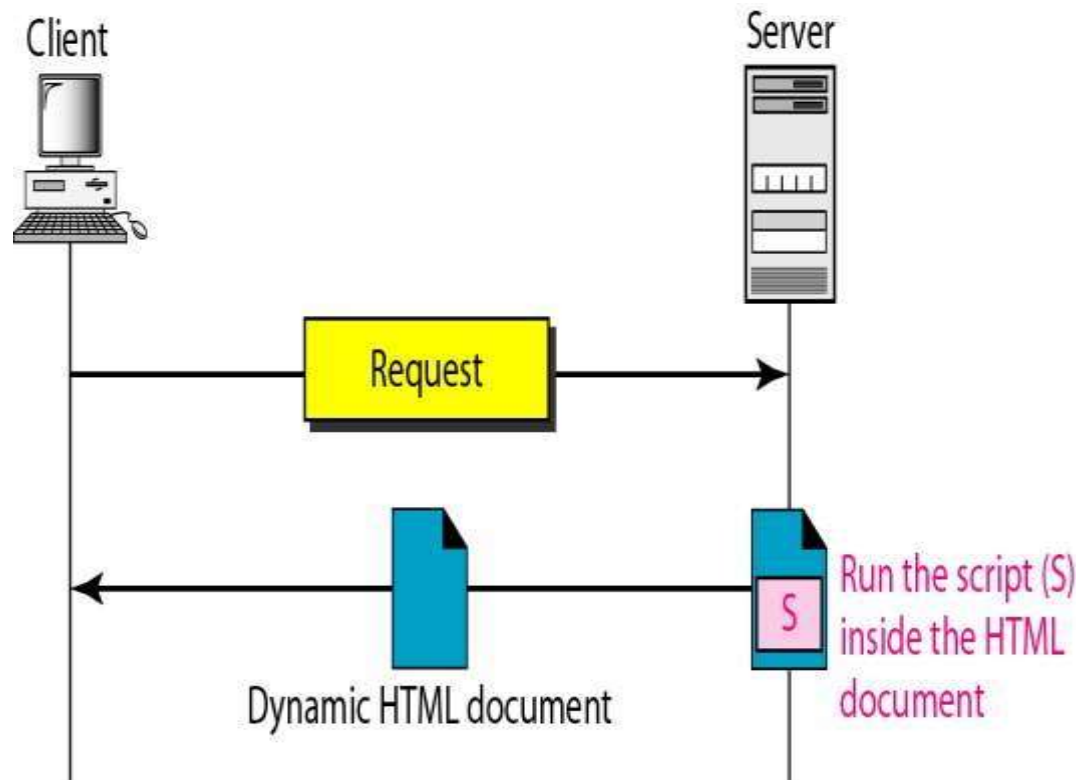
# Common Gateway Interface (CGI)

- The Common Gateway Interface (CGI) is a technology that creates and handles dynamic documents.



*Dynamic document using CGI*

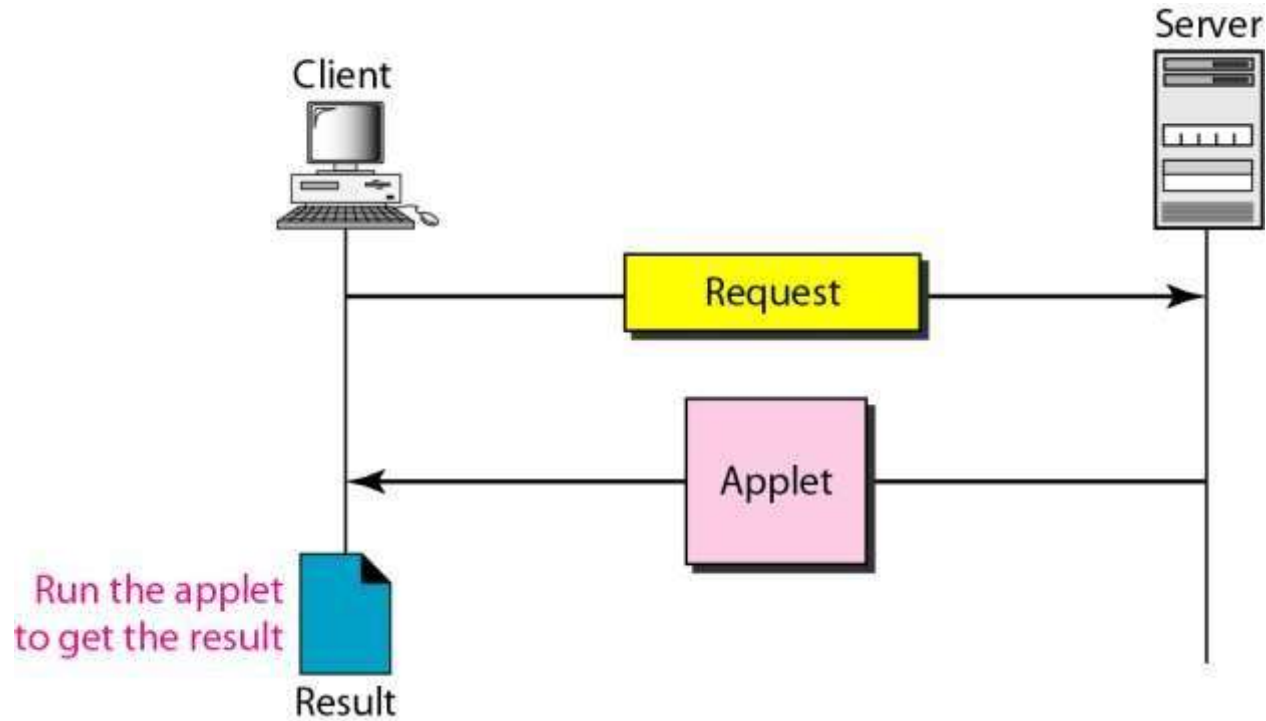
## *Dynamic document using server-site script*



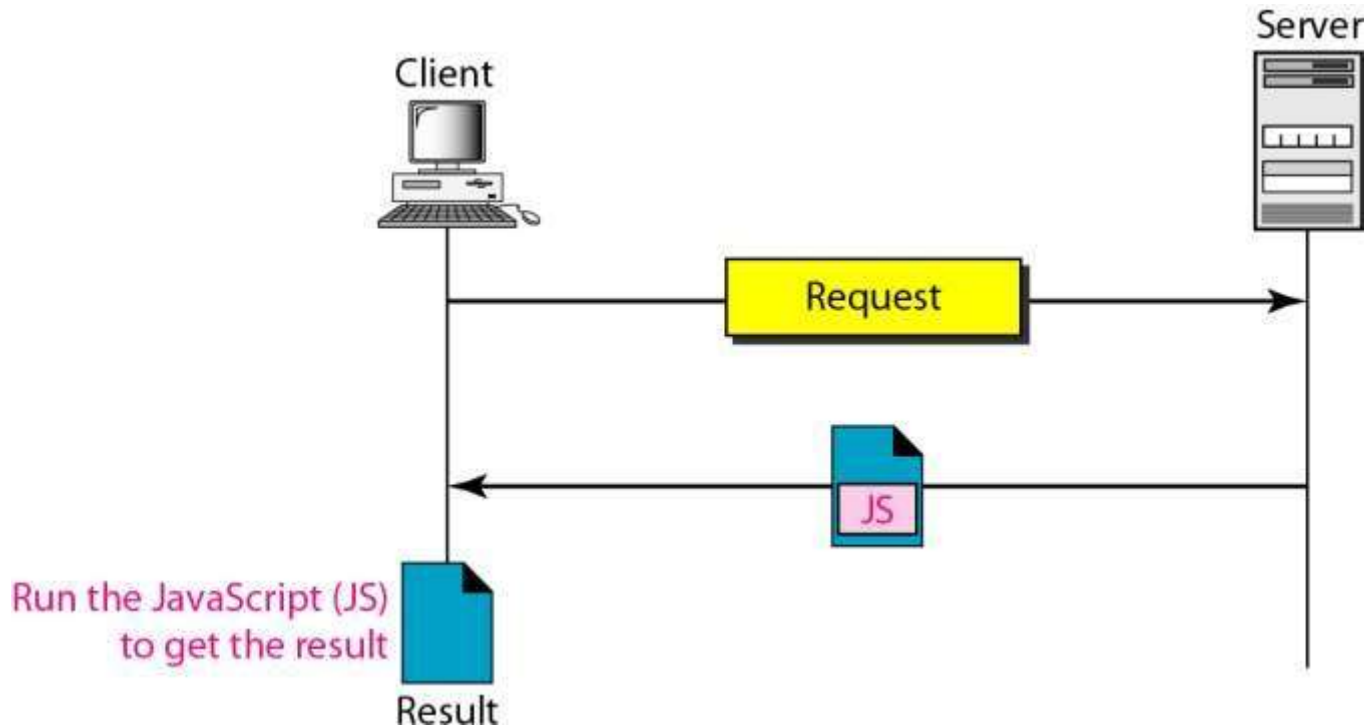
Dynamic documents are sometimes referred to as server-site dynamic documents.

# Active Documents

For many applications, we need a program or a script to be run at the client site. These are called active documents



# Active document using client-site script



Active documents are sometimes referred to as client-site dynamic documents.



# *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. 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, which, as discussed before, is a connection - oriented and reliable protocol.*

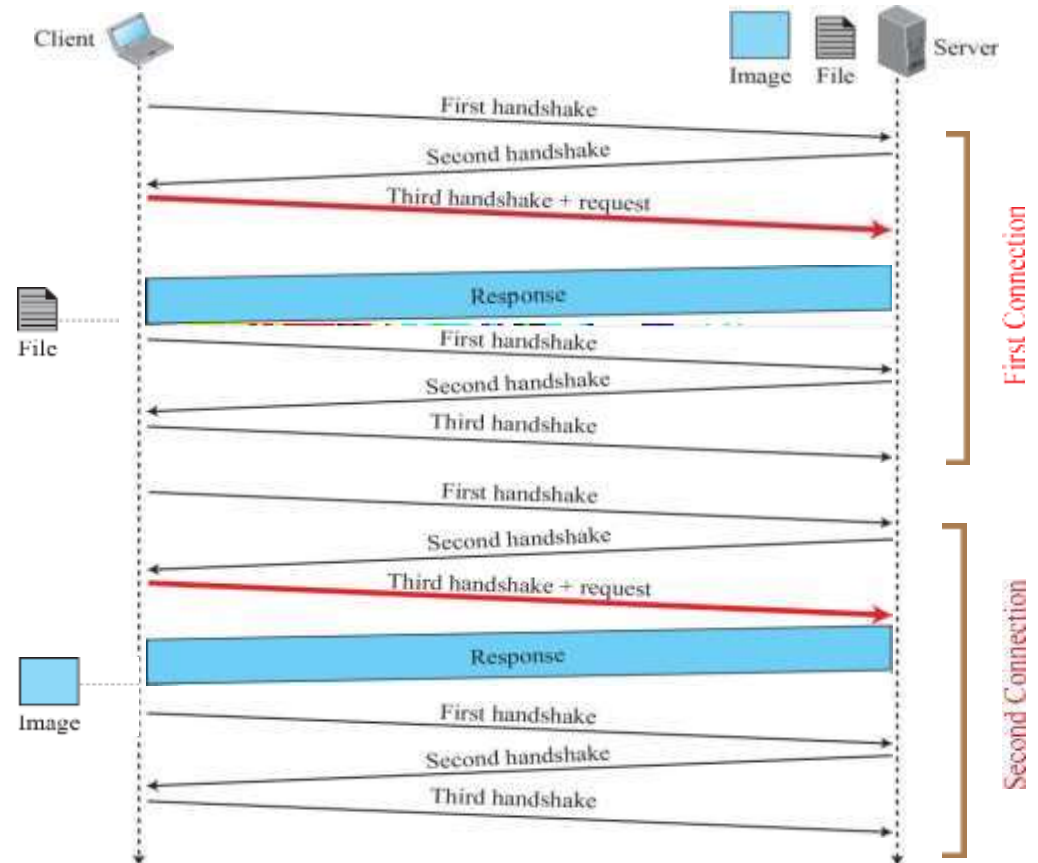
## *HTTP versions*

- HTTP 1.0 uses non persistent HTTP. At most one object can be sent over a single TCP connection.*
- HTTP 1.1 uses persistent HTTP. In this version, multiple objects can be sent over a single TCP connection.*

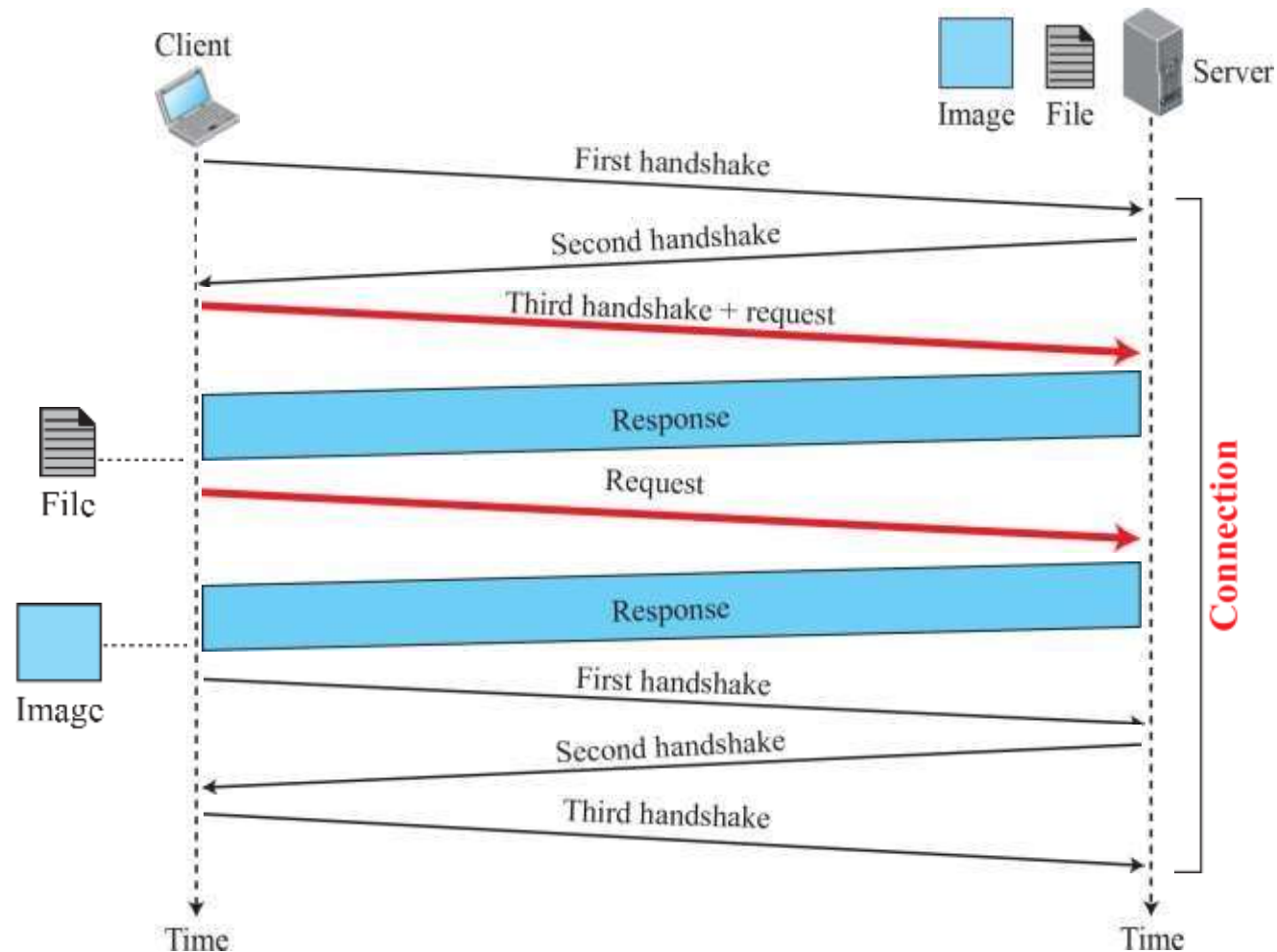
# Example

Below figure shows an example of a *nonpersistent* connection. The client needs to access a file that contains one link to an image. The text file and image are located on the same server. Here we need two connections. For each connection, TCP requires at least three handshake messages to establish the connection, but the request can be sent with the third one. After the connection is established, the object can be transferred. After receiving an object, another three handshake messages are needed to terminate the connection.

**Figure:**

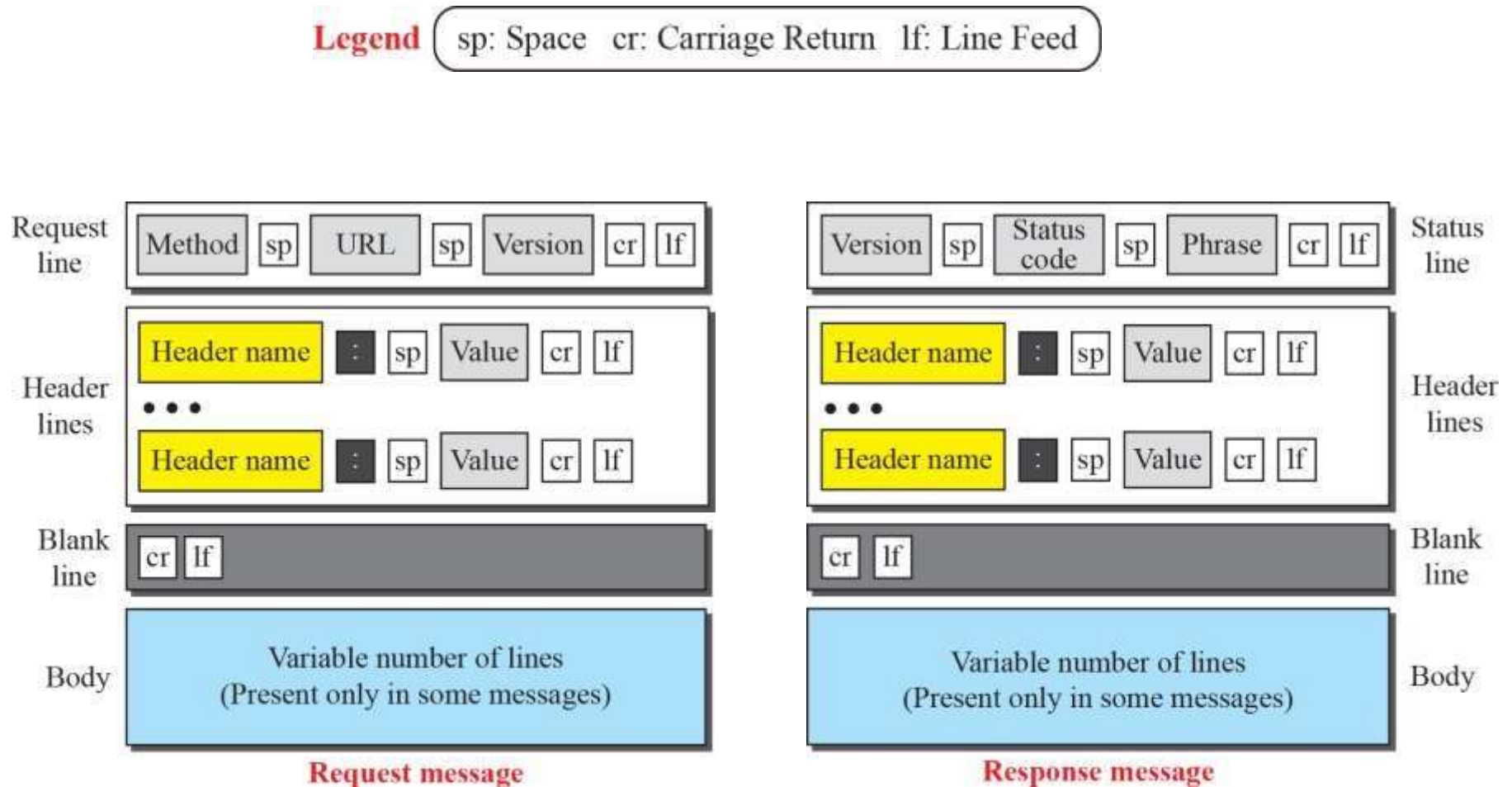


using a persistent connection only one connection establishment and connection termination is used, but the request for the image is sent separately.





**Figure :** *Formats of the request and response messages*



## Table: Methods

---

<i>Method</i>	<i>Action</i>
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

---

## **Table :** Request Header Names

---

<i>Header</i>	<i>Description</i>
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 (explained later)
If-Modified-Since	If the file is modified since a specific date

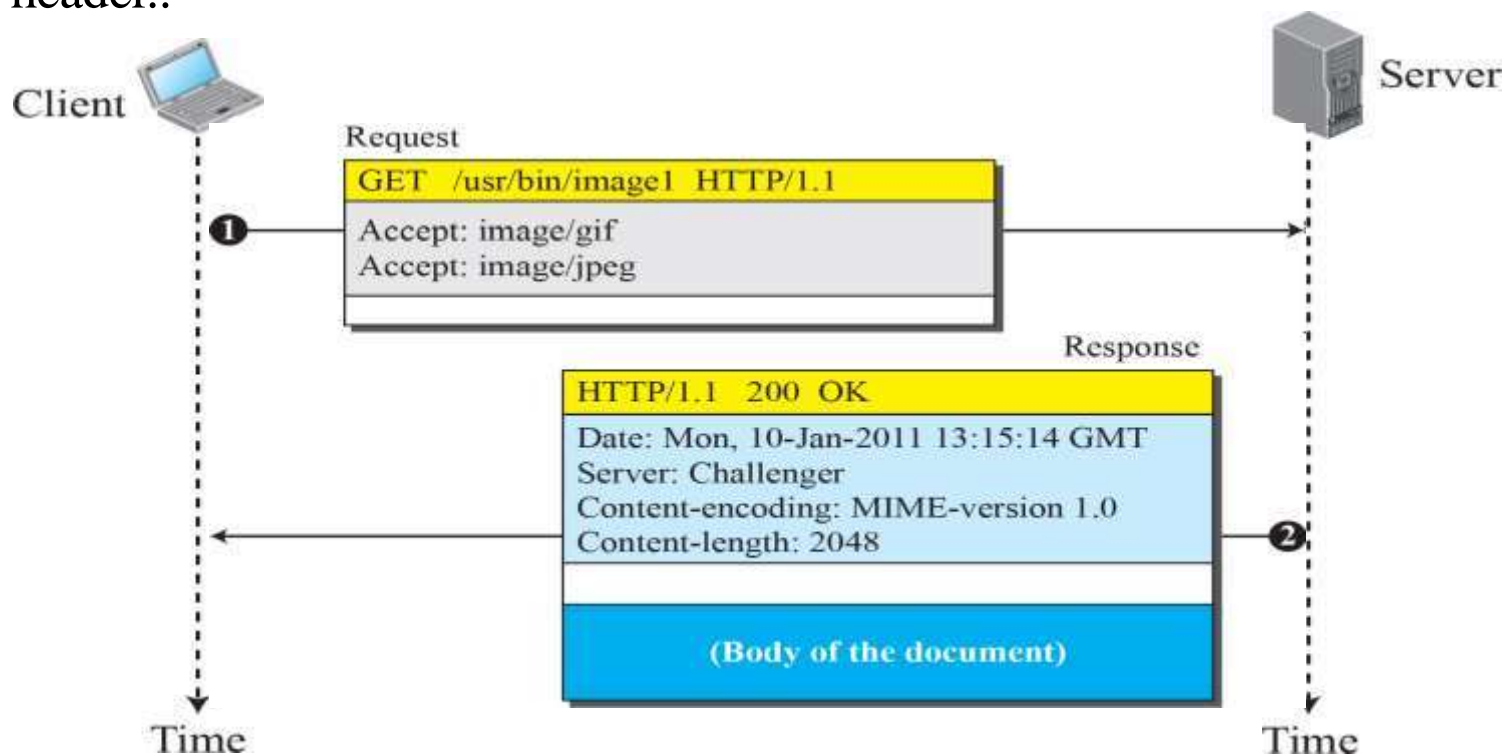
---

**Table :** Response Header Names

<i>Header</i>	<i>Description</i>
Date	Shows the current date
Upgrade	Specifies the preferred communication protocol
Server	Gives information about the server
Set-Cookie	The server asks the client to save a cookie
Content-Encoding	Specifies the encoding scheme
Content-Language	Specifies the language
Content-Length	Shows the length of the document
Content-Type	Specifies the media type
Location	To ask the client to send the request to another site
Accept-Ranges	The server will accept the requested byte-ranges
Last-modified	Gives the date and time of the last change

# Example

This example retrieves a document (see below figure). We use the GET method to retrieve an image with the path `/usr/bin/image26`. The request line shows the method (GET), the URL, and the HTTP version (26.1). The header has two lines that show that the client can accept images in the GIF or JPEG format. The request does not have a body. The response message contains the status line and four lines of header. The header lines define the date, server, content encoding (MIME version, which will be described in electronic mail), and length of the document. The body of the document follows the header..





# ELECTRONIC MAIL

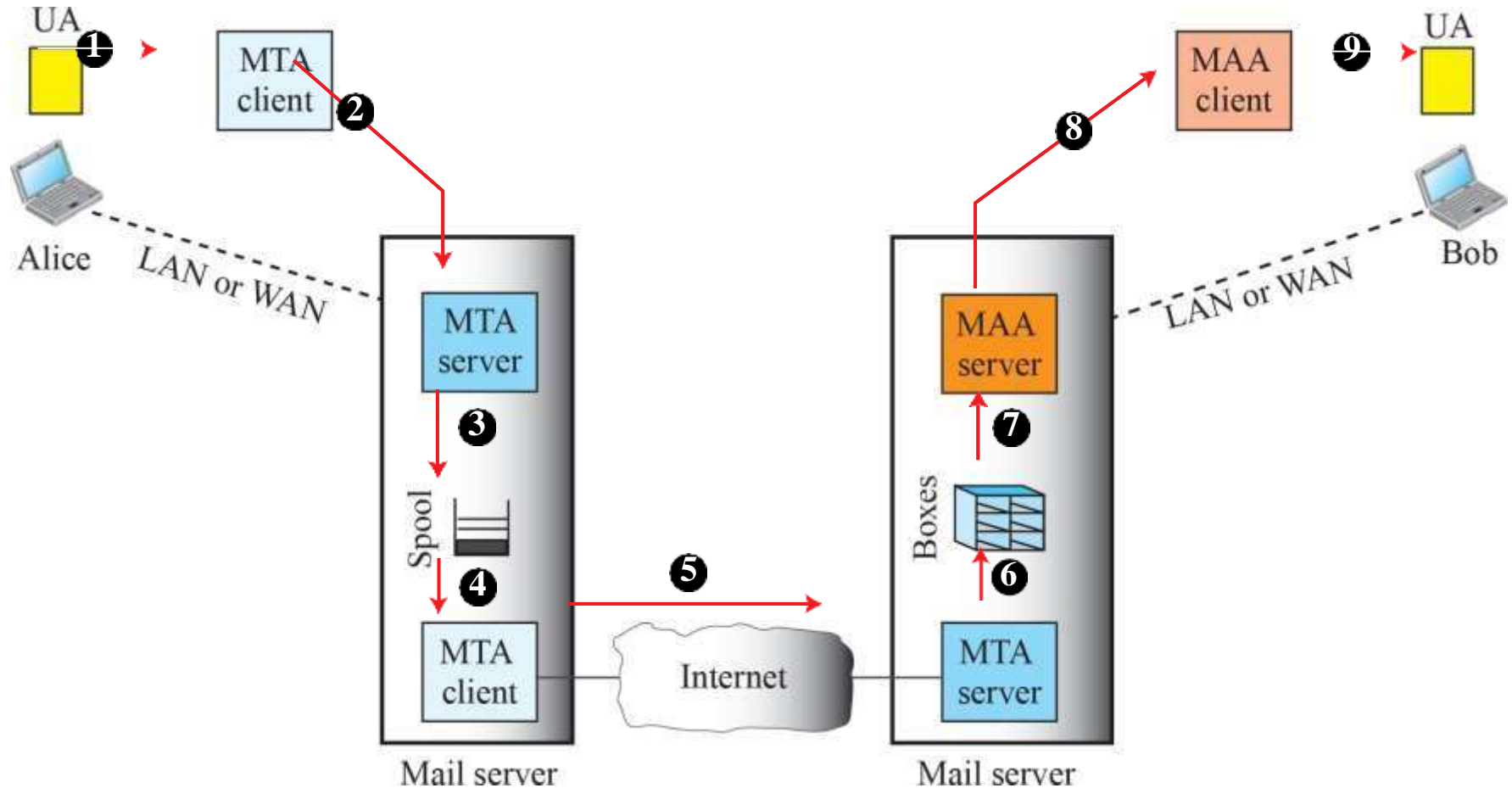
Electronic mail (or e-mail) allows users to exchange messages. The nature of this application is different from other applications discussed so far. This means that the idea of client/server programming should be implemented in another way: using some intermediate computers (servers).

## Architecture


To explain the architecture of e-mail, we give a common scenario, as shown in below figure Another possibility is the case in which Alice or Bob is directly connected to the corresponding mail server, in which LAN or WAN connection is not required, but this variation in the scenario does not affect our discussion.

**Figure : Common scenario**

UA: user agent  
MTA: message transfer agent  
MAA: message access agent



**Figure :** *Format of an e-mail*

<p>Behrouz Forouzan 20122 Olive Street Bellbury, CA 91000</p> <p>Firouz Mosharraf 1400 Los Gatos Street San Louis, CA 91005</p>	
<p>Behrouz Forouzan 20122 Olive Street Bellbury, CA 91000 Jan. 10, 2011</p> <p>Subject: Network</p> <p>Dear Mr. Mosharraf We want to inform you that our network is working properly after the last repair.</p> <p>Yours truly, Behrouz Forouzan</p>	

**Postal mail**

	<p><b>Mail From:</b> forouzan@some.com <b>RCPT To:</b> mosharraf@aNetwork.com</p>	Envelope
Header	<p>From: Behrouz Forouzan To: Firouz Mosharraf Date: 1/10/2011 Subject: Network</p>	
Body	<p>Dear Mr. Mosharraf We want to inform you that our network is working properly after the last repair.</p> <p>Yours truly, Behrouz Forouzan</p>	Message

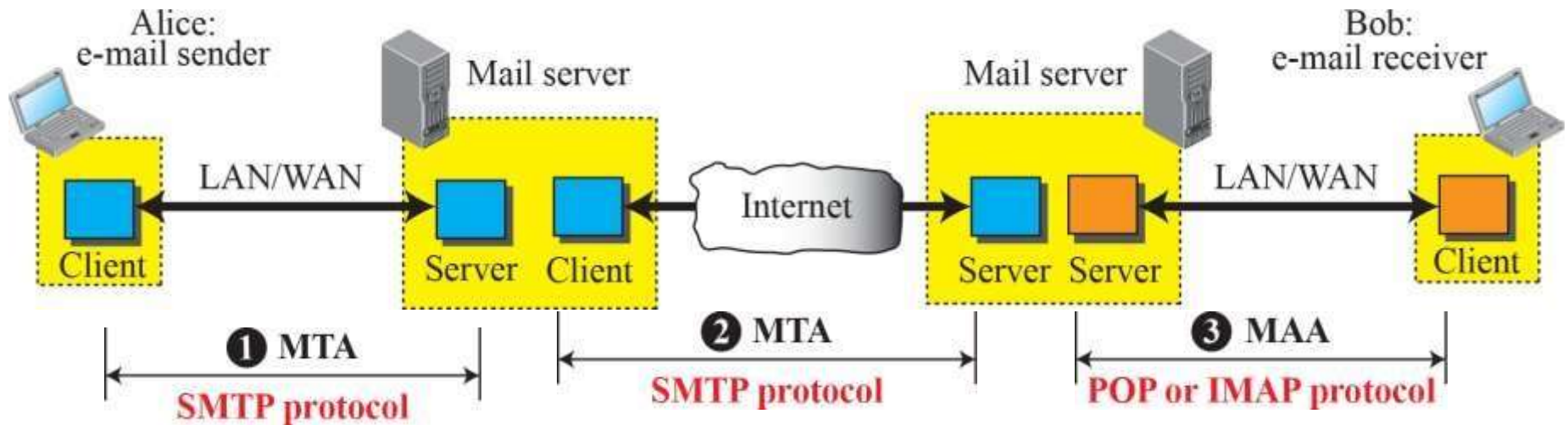
**Electronic mail**



**Figure : E-mail address**



**Figure : Protocols used in electronic mail**



POP3 downloads the email from a server to a single computer, then deletes the email from the server. On the other hand, IMAP stores the message on a server and synchronizes the message across multiple devices.

# ***MIME*(Multipurpose Internet Mail Extension)**

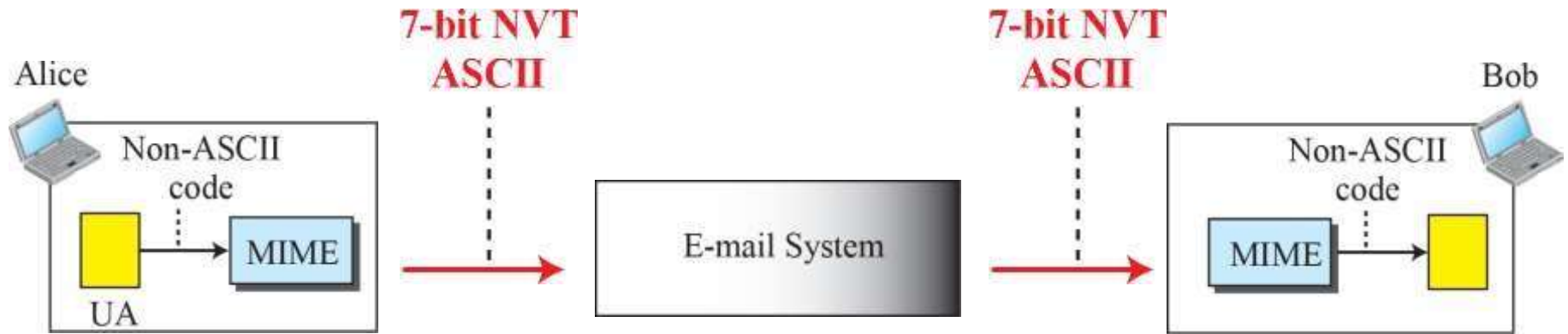
- **MIME** is a standard which was proposed by Bell Communications in 1991 in order to expand limited capabilities of email.
- MIME is a kind of add on or a supplementary protocol which allows non-ASCII data to be sent through SMTP. It allows the users to exchange different kinds of data files on the Internet: audio, video, images, application programs as well.

## **Why do we need MIME?**

Limitations of Simple Mail Transfer Protocol (SMTP):

- SMTP has a very simple structure
- It only send messages in NVT 7-bit ASCII format.
- It cannot be used for languages that do not support 7-bit ASCII format such as- French, German, Russian, Chinese and Japanese, etc. so it cannot be transmitted using SMTP. So, in order to make SMTP more broad we use MIME.
- It cannot be used to send binary files or video or audio data.

**Figure :** *MIME*



## Working of MIME –

Suppose a user wants to send an email through user agent and it is in a non-ASCII format so there is a MIME protocol which converts it into 7-bit NVT ASCII format. Message is transferred through e-mail system to the other side in 7-bit format now MIME protocol again converts it back into non-ASCII code and now the user agent of receiver side reads it and then information is finally read by the receiver.

MIME header is basically inserted at the beginning of any e-mail transfer.

**MIME Version** – Defines version of MIME protocol.

**Content Type** – Type of data used in the body of message. They are of different types like text data (plain, HTML), audio content or video content.

**Content Type Encoding** – It defines the method used for encoding the message. Like 7-bit encoding, 8-bit encoding, etc.

**Content Id** – It is used for uniquely identifying the message.

**Content description** – It defines whether the body is actually image, video or audio.

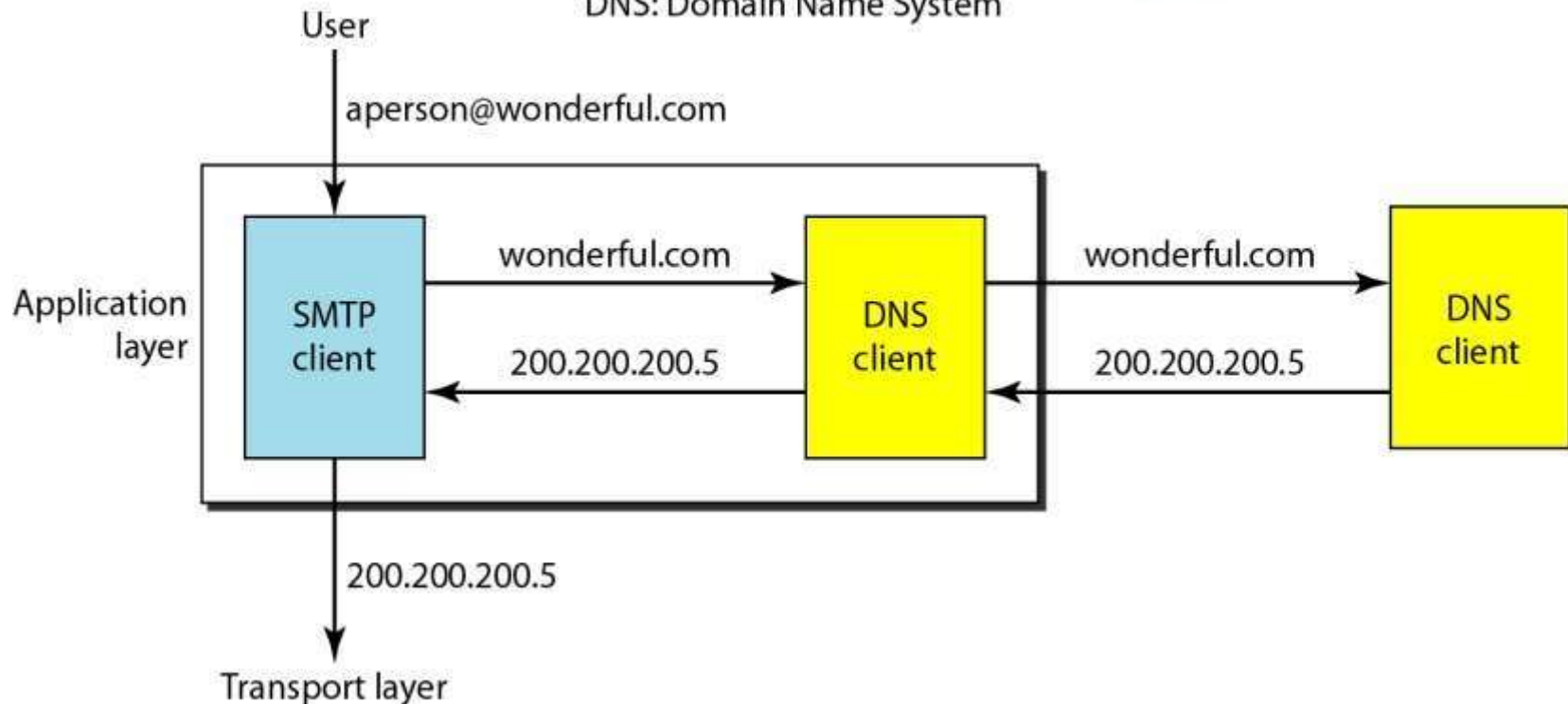
**MIME headers**

E-mail header
MIME-Version: 1.1 Content-Type: type/subtype Content-Transfer-Encoding: encoding type Content-ID: message ID Content-Description: textual explanation of nontextual contents
E-mail body

# DNS (Domain Name System)

To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name.

SMTP: Simple Mail Transfer Protocol (e-mail)  
DNS: Domain Name System



## **NAME SPACE**

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.

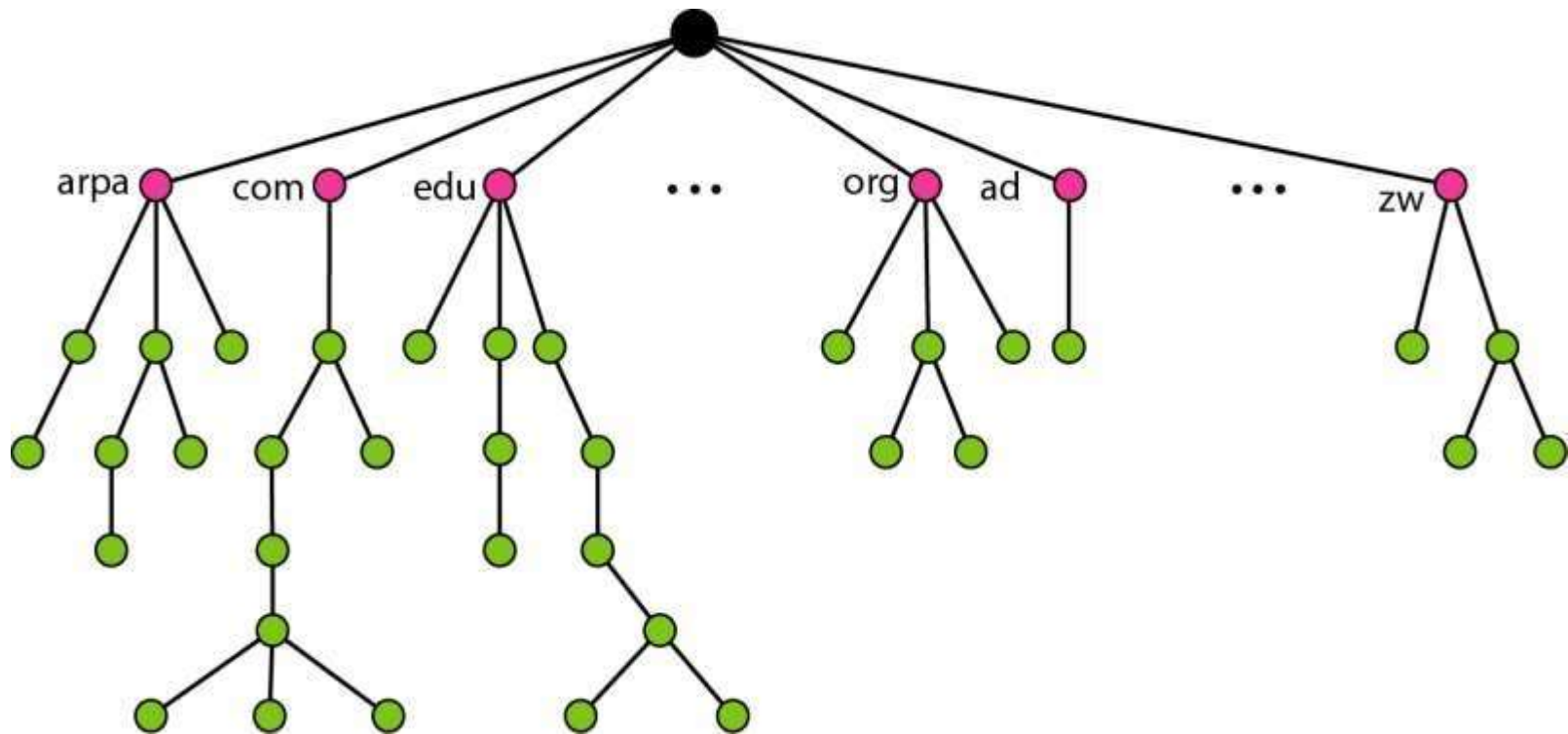
### **Hierarchical Name Space**

In a hierarchical name space, each name is made of several parts. The first part can define the nature of the organization, the second part can define the name of an organization, the third part can define departments in the organization, and so on.

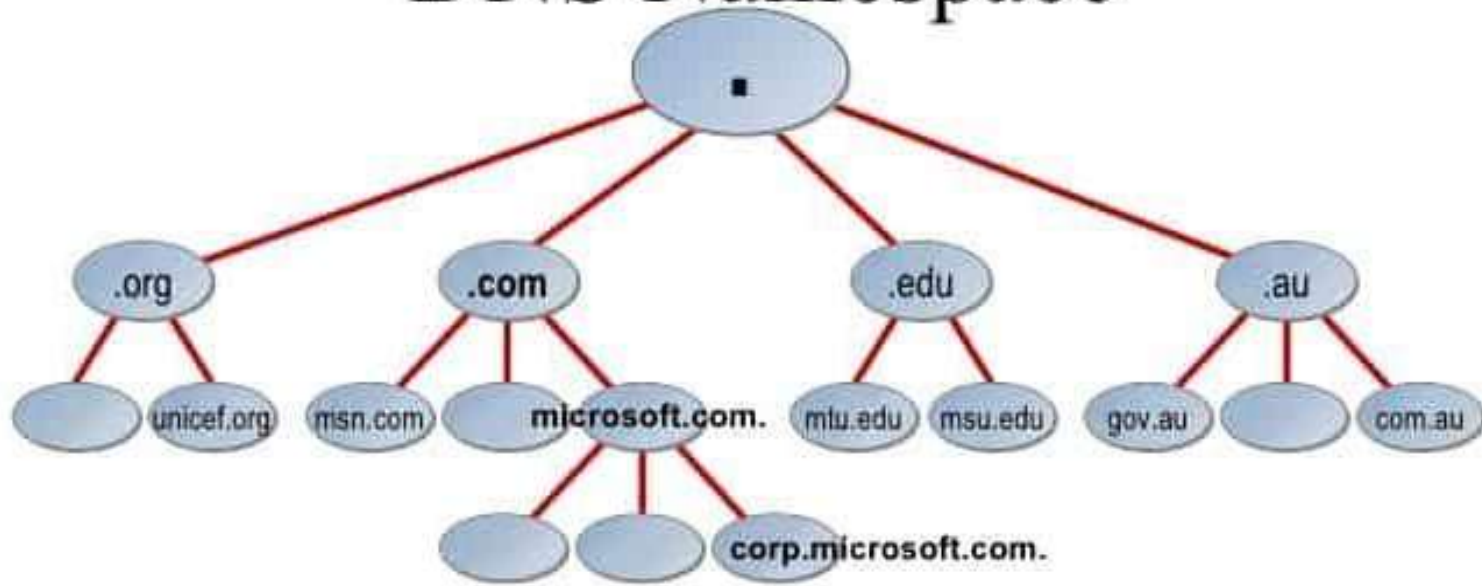
**Example:** challenger.jhda.edu, challenger.berkeley.edu, and challenger.smart.com

# DOMAIN NAME SPACE

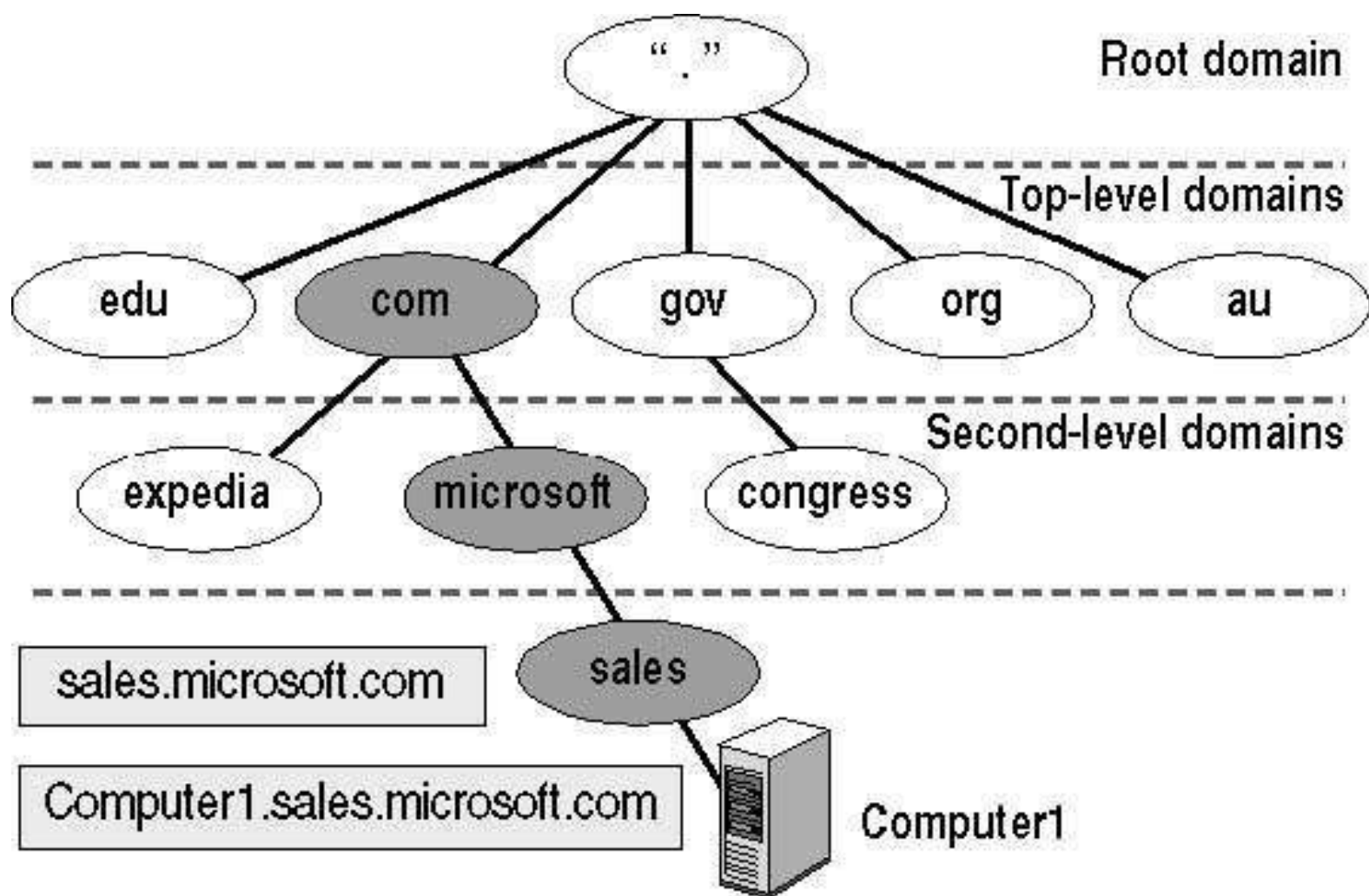
To have a hierarchical name space, a domain name space was designed. In this design the names are defined in an inverted-tree structure with the root at the top. The tree can have only 128 levels: level 0 (root) to level 127.



# DNS Namespace







## Label

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 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. Below Figure shows some domain names

FQDN



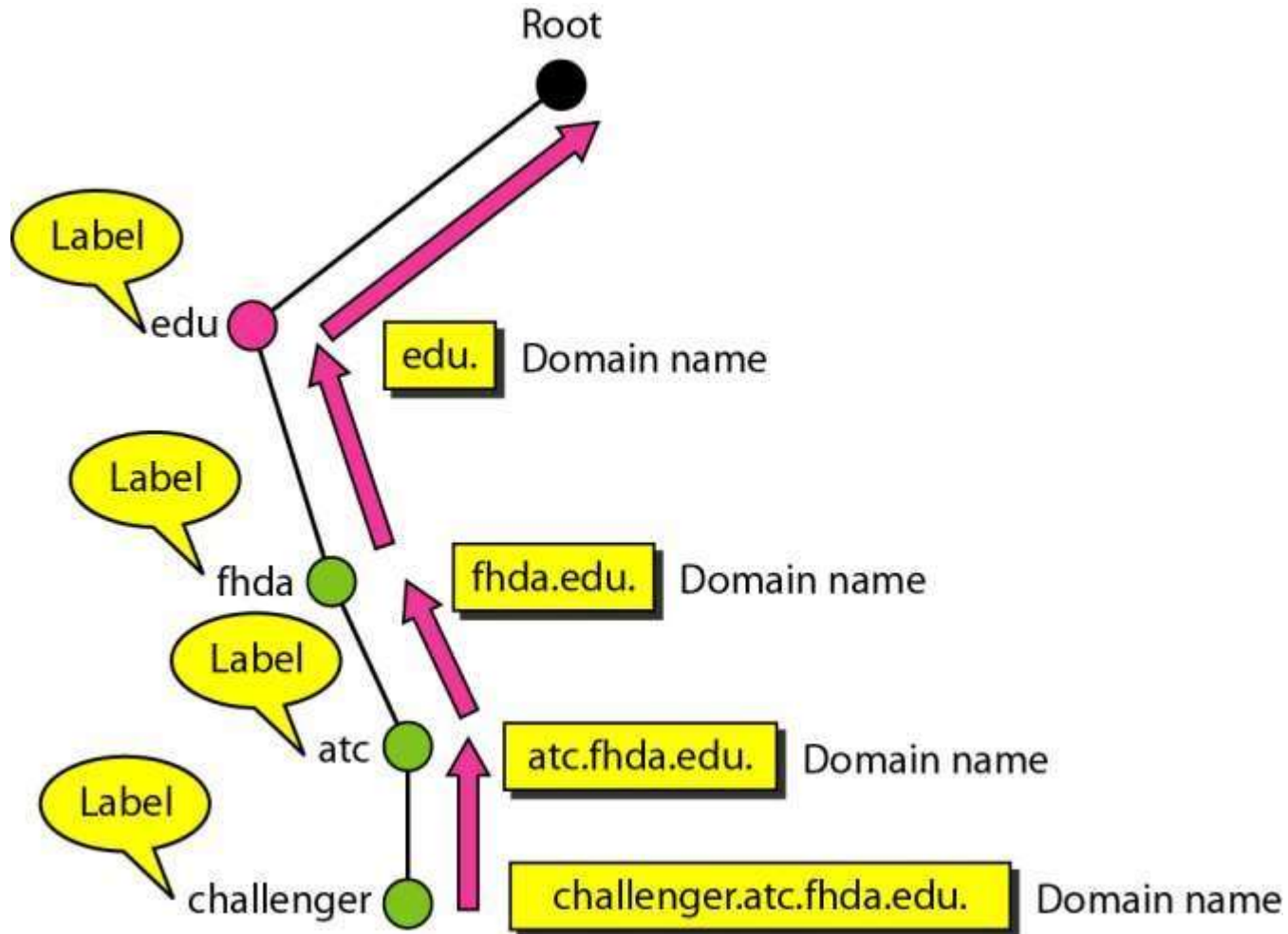
challenger.atc.fhda.edu.  
cs.hmme.com.  
www.funny.int.

PQDN



challenger.atc.fhda.edu  
cs.hmme  
www

# Domain names and labels

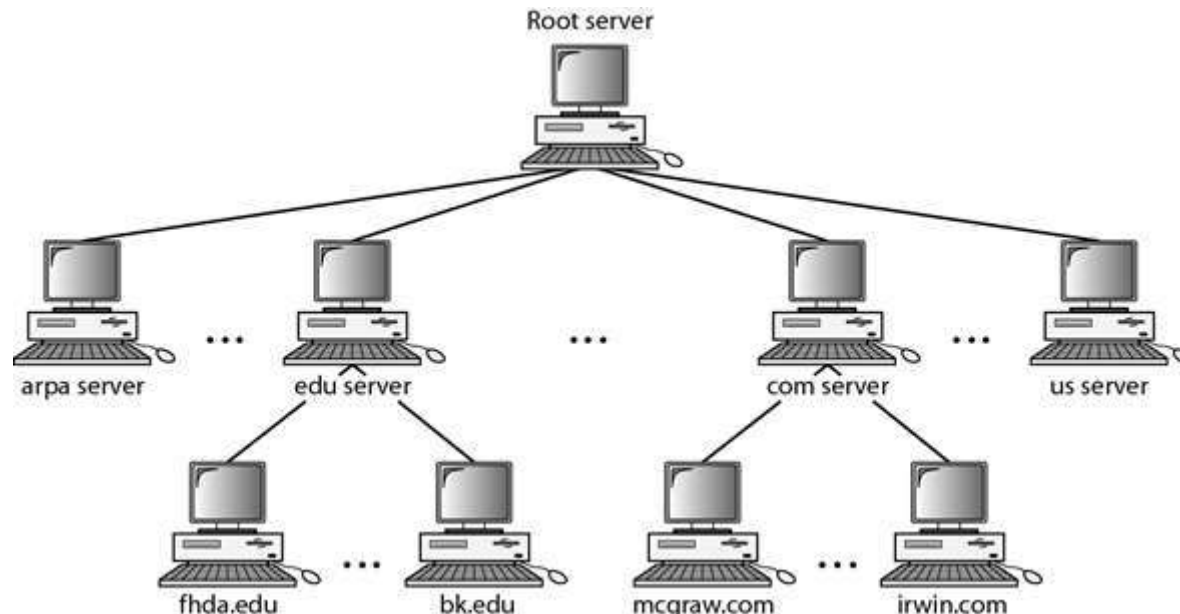


# DISTRIBUTION OF NAME SPACE:

The information contained in the domain name space must be stored. However, it is very inefficient and also unreliable to have just one computer store such a huge amount of information. In this section, we discuss the distribution of the domain name space

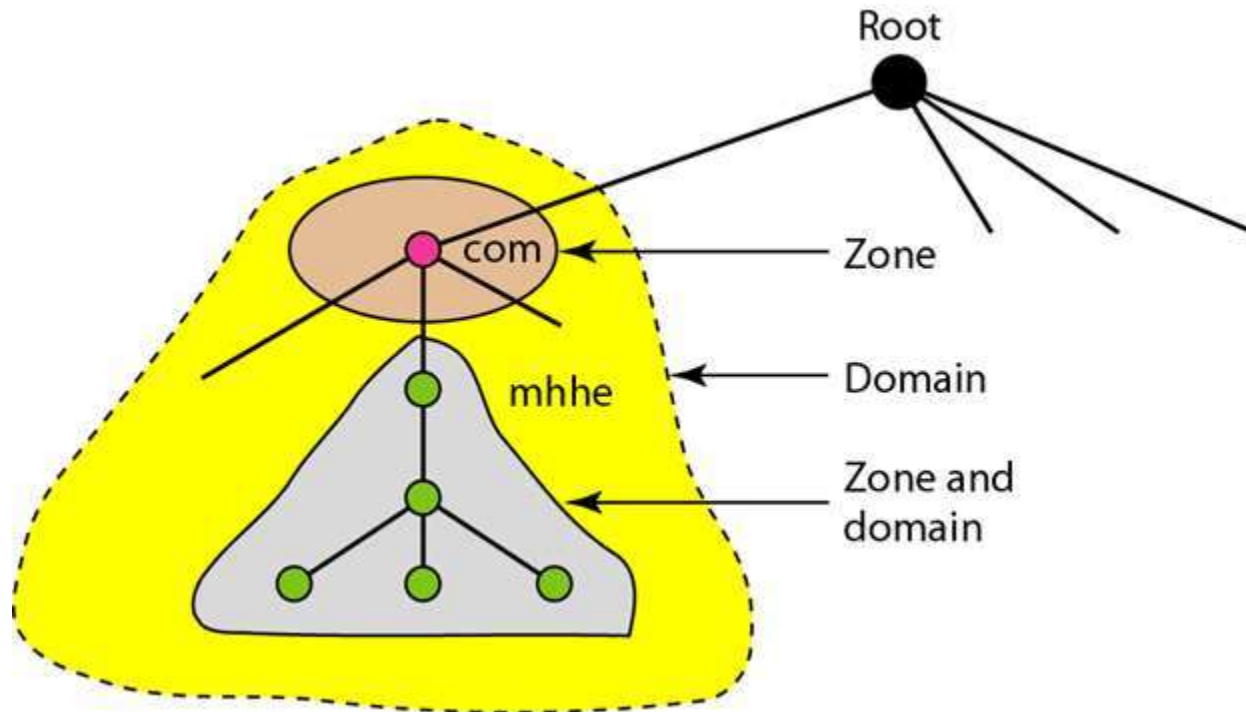
## 1 Hierarchy of Name Servers

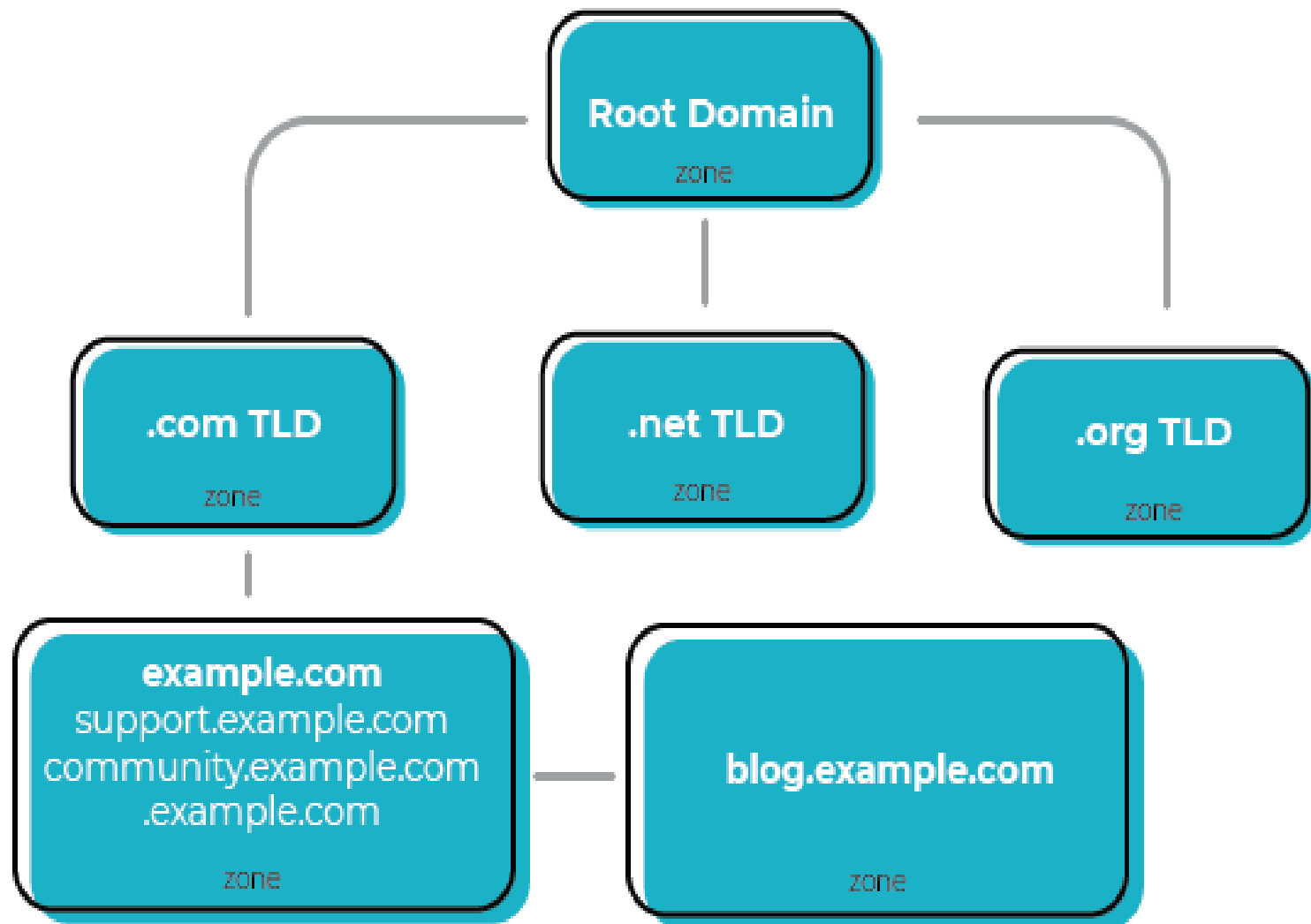
distribute the information among many computers called DNS servers. we let the root stand alone and create as many domains (subtrees) as there are first-level nodes



## 2 Zone

Since the complete domain name hierarchy cannot be stored on a single server, it is divided among many servers. What a server is responsible for or has authority over is called a zone. We can define a zone as a contiguous part of the entire tree





### **3 Root Server**

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

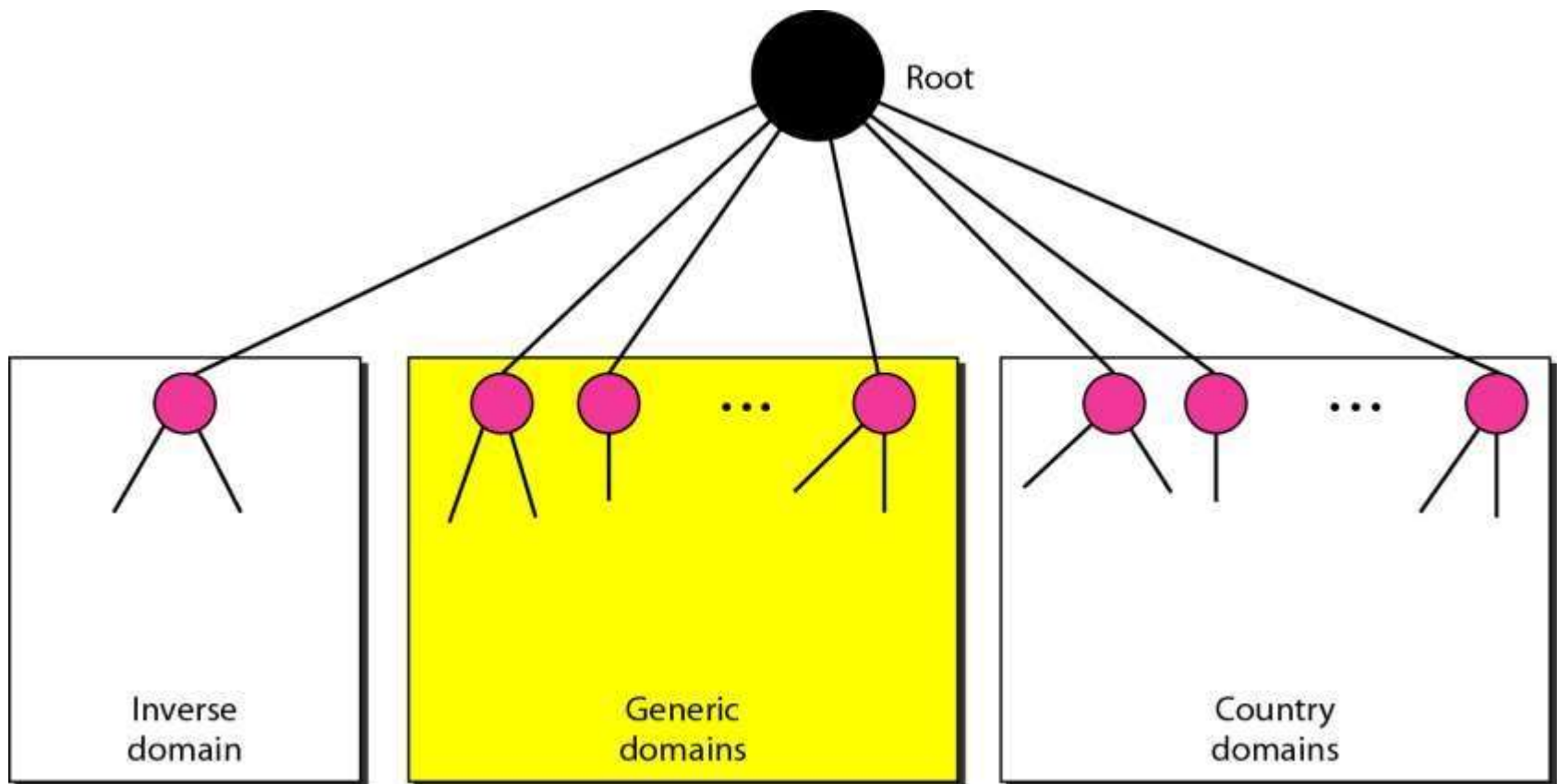
### **4 Primary and Secondary Servers**

A primary server is a server that stores a file about the zone for which it is an authority. It is responsible for creating, maintaining, and updating the zone file. It stores the zone file on a local disk

A secondary server is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. The secondary server neither creates nor updates the zone files

# DNS IN THE INTERNET

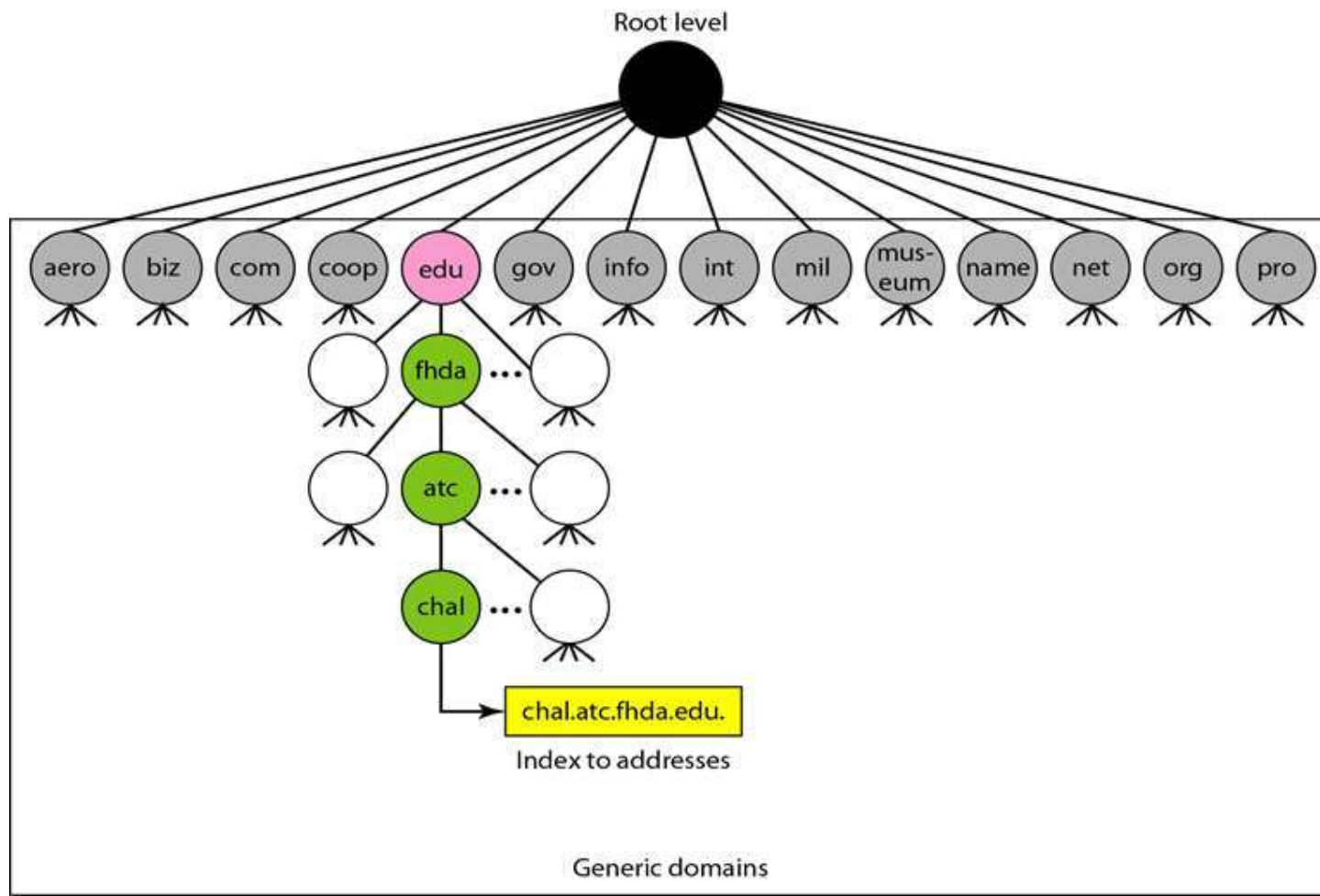
DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain





# 1 Generic Domains

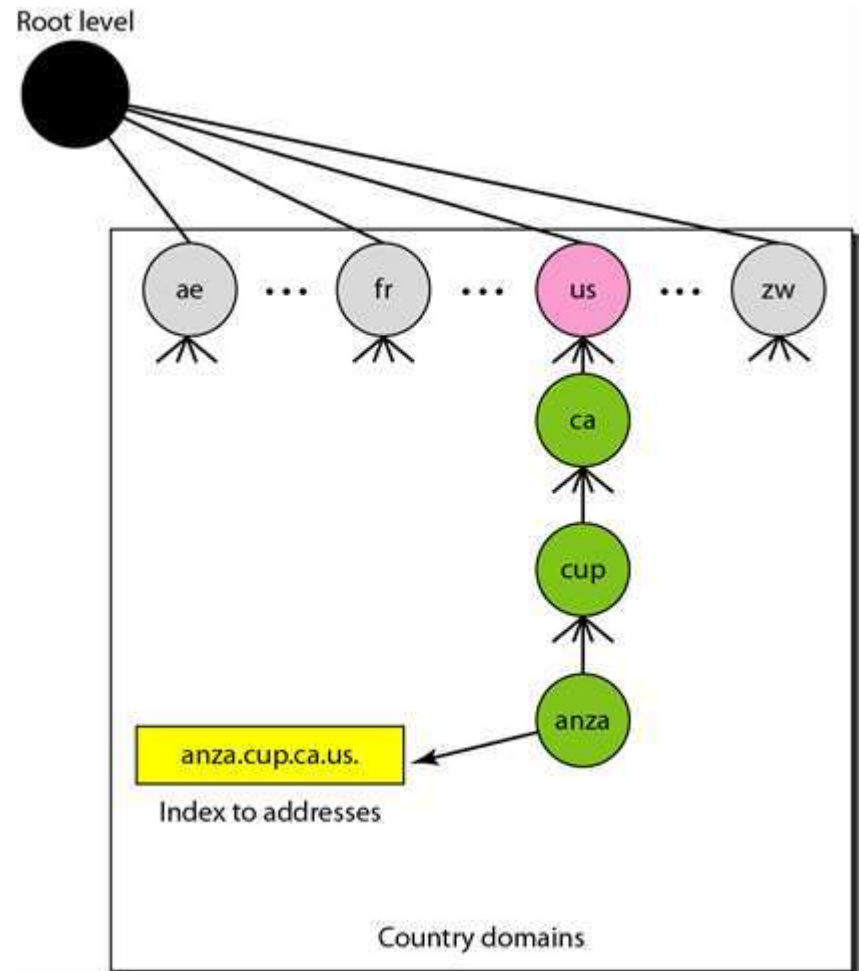
The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database



<i>Label</i>	<i>Description</i>
<b>aero</b>	Airlines and aerospace companies
<b>biz</b>	Businesses or firms (similar to “com”)
<b>com</b>	Commercial organizations
<b>coop</b>	Cooperative business organizations
<b>edu</b>	Educational institutions
<b>gov</b>	Government institutions
<b>info</b>	Information service providers
<b>int</b>	International organizations
<b>mil</b>	Military groups
<b>museum</b>	Museums and other nonprofit organizations
<b>name</b>	Personal names (individuals)
<b>net</b>	Network support centers
<b>org</b>	Nonprofit organizations
<b>pro</b>	Professional individual organizations

## 2 Country Domains

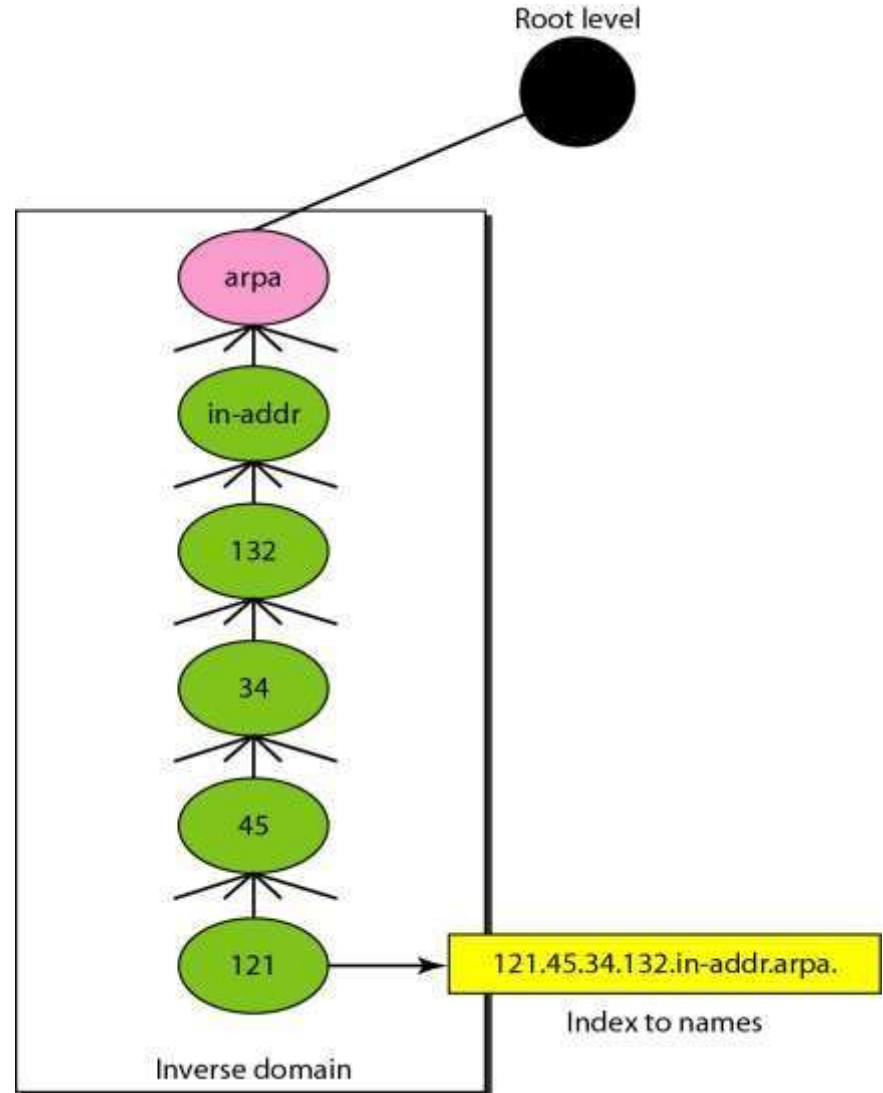
The country domains section uses two-character country abbreviations (e.g., us for United States). Second labels can be organizational, or they can be more specific, national designations. The United States, for example, uses state abbreviations as a subdivision of us (e.g., ca.us.).



### **3 Inverse Domain**

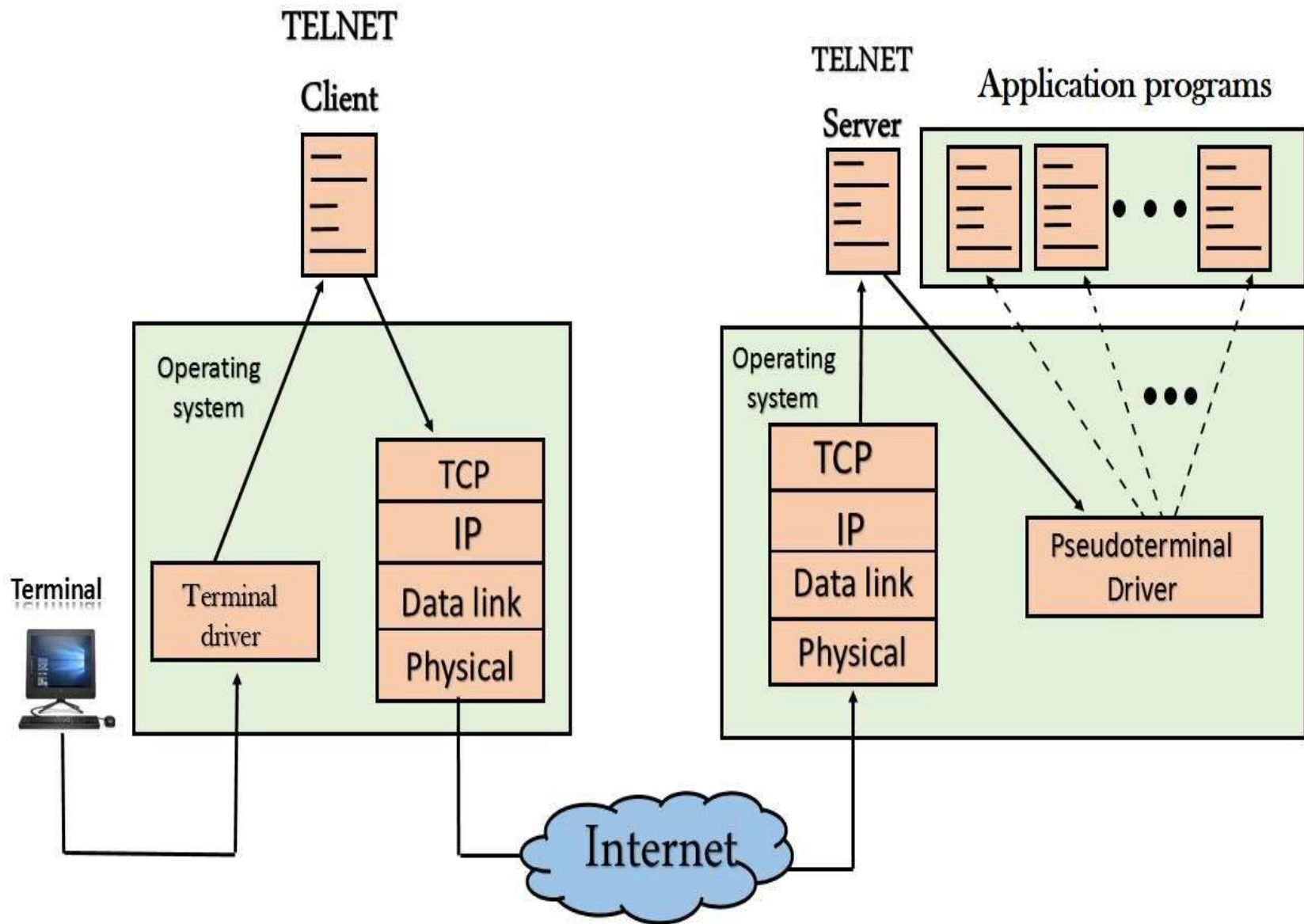
The inverse domain is used to map an address to a name.

When the server has received a request from the client, and the server contains the files of only authorized clients. To determine whether the client is on the authorized list or not, it sends a query to the DNS server and ask for mapping an address to the name.



# Introduction to TELNET

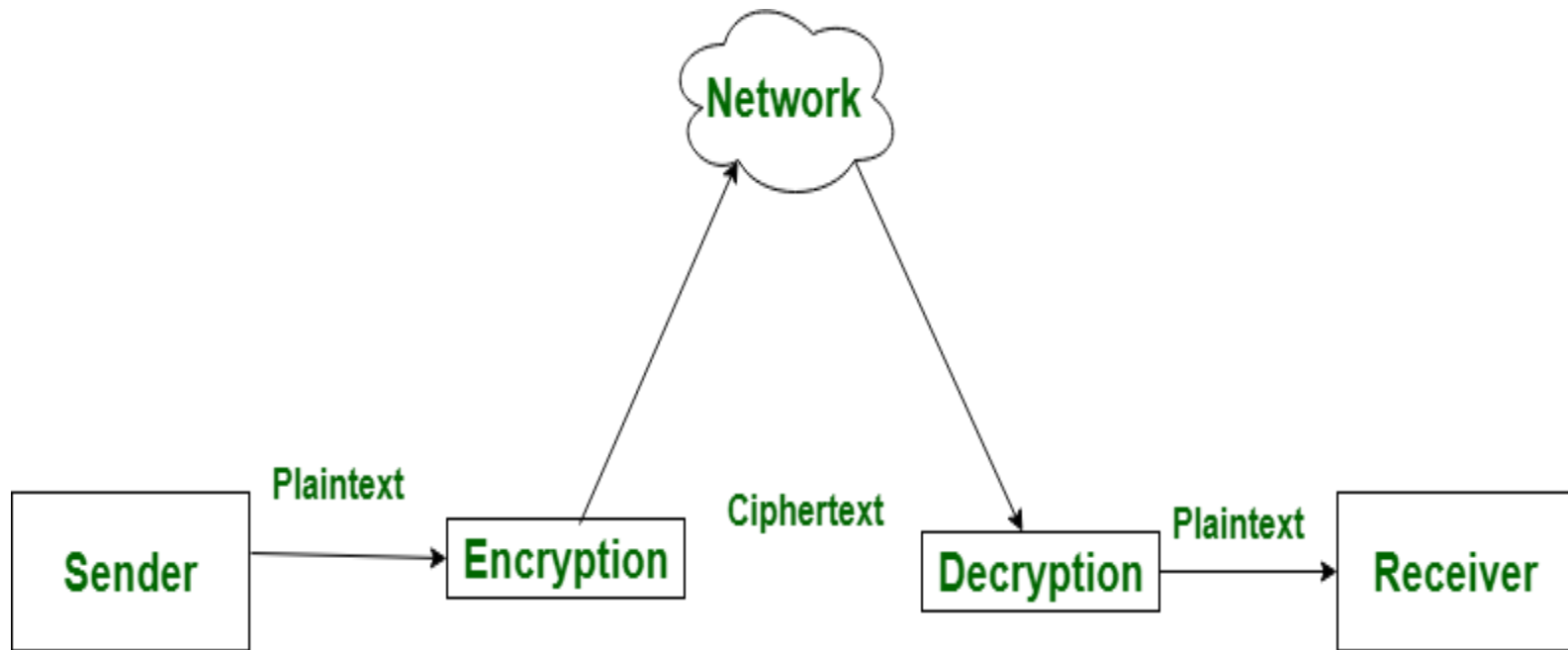
- **TELNET** stands for **TErminaLNET**work. It is a type of protocol that enables one computer to connect to local computer.
- It is used as a standard **TCP/IP protocol** for virtual terminal service which is given by **ISO**.
- Computer which starts connection known as the **local computer**.
- Computer which is being connected to i.e. which accepts the connection known as **remote computer**.



# **RSA algorithm (Rivest-Shamir-Adleman) Algorithm**

- RSA algorithm is a public key encryption technique and is considered as the most secure way of encryption. It was invented by Rivest, Shamir and Adleman in year 1978 and hence name RSA algorithm.
- RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes that the Public Key is given to everyone and Private key is kept private.
- If the public key is used at encryption we have to use the private key of the same user in the decryption process.

- **Encryption** is the process of converting normal message (plaintext) into meaningless message (Ciphertext). Whereas **Decryption** is the process of converting meaningless message (Ciphertext) into its original form (Plaintext).





# RSAAAlgorithm

Here we need to find out both public and private keys

- The initial procedure begins with selection of two prime numbers namely  $p$  and  $q$ , and then calculating their product  $n$ , as shown –

$$n=p*q$$

- Then calculate  $\phi(n) = (p-1)*(q-1)$
- Assume  $e$  and  $d$  as Public and Private Keys.
- Assume  $e$  such that  $\gcd(e, \phi(n)) = 1$
- Assume  $d$  such that  $d*e \bmod \phi(n) = 1$

- Public Key. =  $\{e,n\}$
- Private key =  $\{d,n\}$
- After finding public and private keys, Encryption process starts i.e converting plain text to cipher text.
- Here we must consider the plain text must be less than n

## **Encryption Formula**

Consider a sender who sends the plain text message to someone whose public key is  $(e,n)$ . To encrypt the plain text message in the given scenario, use the following syntax -

$$C = P^e \bmod n$$

## **Decryption Formula**

$$P = C^d \bmod n$$

## RSA Algorithm – Example

- If  $p=3$  ,  $q=5$
- $n = p*q = 3*5 = 15$
- $\phi(n) = (p-1)*(q-1) = (3-1) * (5-1) = 8$
- Generating public key that is  $e$

$$\gcd(e, \phi(n)) = 1$$

3,5,7

$$\mathbf{e=3}$$

- Generating private key that is  $d$

$$d*e \bmod \phi(n) = 1$$

$$3*3 \bmod 8 = 1$$

$$\mathbf{d=3}$$

- Public Key. =  $\{e,n\} = \{3,15\}$
- Private key =  $\{d,n\} = \{3,15\}$

**ENCRYPTION – plain text must be less than  $n \rightarrow 4 < 15$**

$$\begin{aligned}C &= P^e \bmod n \\&= 4^3 \bmod 15 \\&= 64 \bmod 15\end{aligned}$$

**Cipher text = 4**

**DECRYPTION**

$$\begin{aligned}P &= C^e \bmod n \\&= 4^3 \bmod 15 \\&= 64 \bmod 15\end{aligned}$$

**Plain text = 4**

# **File Transfer Protocol**

- File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.
- Although transferring files from one system to another seems simple and straightforward.
- Before transferring, some problems must be dealt with first, such as:
  - 1) Two systems may use different file name conventions.
  - 2) Two systems may have different ways to represent text and data.
  - 3) Two systems may have different directory structures.
- All of these problems have been solved by FTP in a very simple and elegant approach.

# FILE TRANSFER

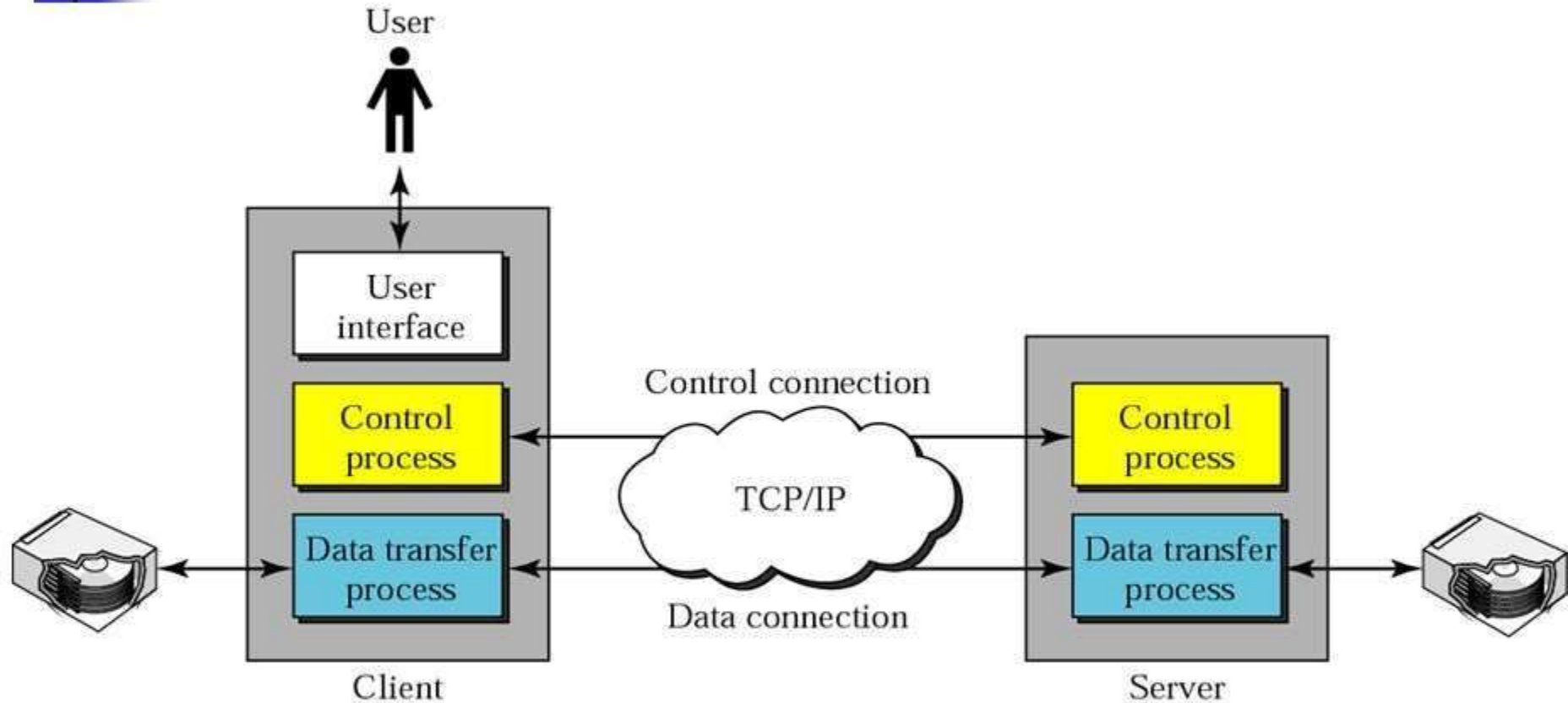
- FTP doesn't not really move, it copies files from one computer to another
- FTP differs from other client-server applications in that it establishes two connections between the hosts.
- One connection is used for **data transfer**, the other for **control information** (commands and responses).
- Separation of commands and data transfer makes FTP more efficient.
- We need to transfer only a line of command or a line of response at a time.

# FILE TRANSFER

- The control connection uses very simple rules of communication.
- The data connection, on the other hand, needs more complex rules due to the variety of data types transferred.
- FTP uses two well-known TCP ports:

Port 21 is used for the control connection,  
port 20 is used for the data connection.

**Figure 19.1** *FTP*



**Control connection stays connected during entire session.  
Data connection opened and closed for each file transferred.**



# Control Connection

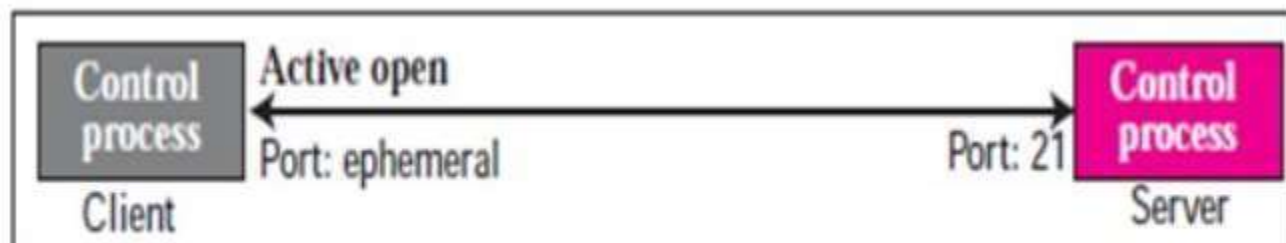
There are two steps:

- The server issues a passive open on the well-known port 21 and waits for a client.
- The client uses an ephemeral port and issues an active open. The connection remains open during the entire process.
- The service type, used by the IP protocol, is minimize delay because this is an interactive connection between a user (human) and a server.
- The user types commands and expects to receive responses without significant delay.

## *Opening the control connection*



a. First, passive open by server



b. Later, active open by client

## Data Connection

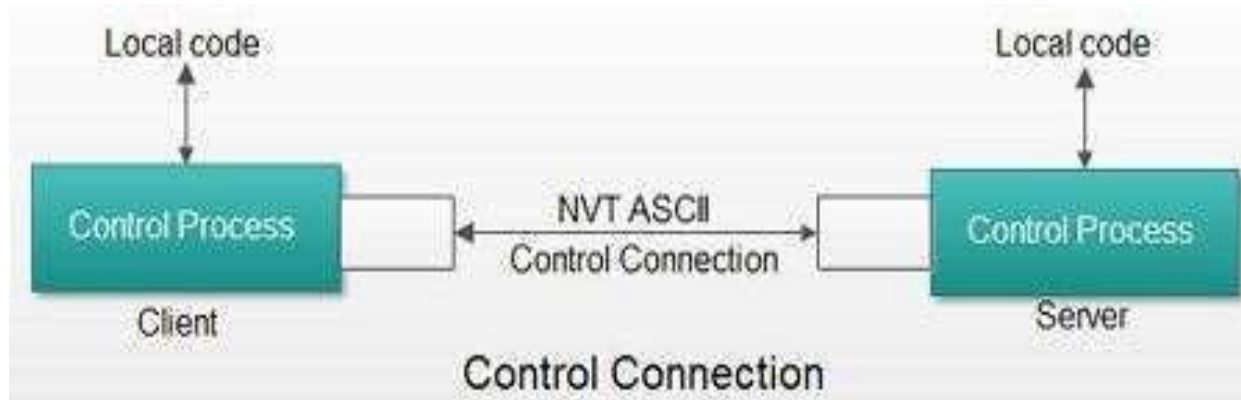
- The data connection uses the well-known port 20 at the server site.
- The following shows how FTP creates a data connection:
  - 1) The client, not the server, issues a passive open using an ephemeral port.
  - 2) The client sends this port number to the server using the PORT command.
  - 3) The server receives the port number and issues an active open using the well known port 20 and the received ephemeral port number.

## **Data Connection**

- The data connection is opened and then closed for each file transferred.
- It opens each time , commands that involve transferring files are used, and it closes when the file is transferred.
- While the control connection is open, the data connection can be opened and closed multiple times if several files are transferred.

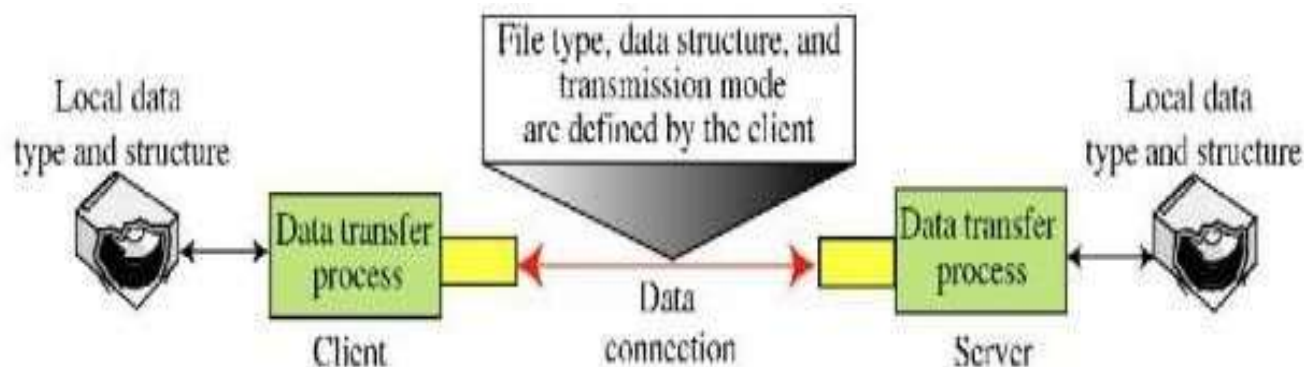
## Communication over Control Connection :

- FTP uses the same approach as SMTP to communicate across the control connection.
- It uses the 7-bit ASCII character set.
- Communication is achieved through commands and responses.
- Each command or response is only one short line, so we need not worry about file format or file structure.



## Communication over Data Connection :

- File transfer occurs over the data connection under the control of the commands sent over the control connection.
- We prepare for transmission through the control connection.
- The heterogeneity problem is resolved by defining three attributes of communication before sending the file through the data connection :
  - file type
  - data structure
  - transmission mode



**Figure 5.16 Data Connection**

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

**ASCII file.** - default format for transferring text files.

**EBCDIC file** - used If one or both ends of the connection use EBCDIC encoding (the file format used by IBM).

**image file** - default format for transferring binary files. The file is sent as continuous streams of bits without any interpretation or encoding.

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

**file structure:** the file is a continuous stream of bytes.

**record structure:** the file is divided into records. This can be used only with text files.

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

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

**Stream mode:**

This is the default mode. Data are delivered from FTP to TCP as a continuous stream of bytes.

TCP is responsible for chopping data into segments of appropriate size.

**Block mode:**

Data can be delivered from FTP to TCP in blocks.

Each block is preceded by a 3-byte header. The first byte is called the block descriptor; the next two bytes define the size of the block in bytes.

**Compressed mode:**

If the file is big, the data can be compressed.

The compression method normally used is run-length encoding.