UNIT 2
APPLICATION LAYER
LH-4

CONTENTS

- Introduction; Application Architectures (Host-Based Architectures, Client-Based Architectures, Client-Server Architectures, Cloud Computing Architectures, Peer-to-Peer Architectures, Choosing Architectures); World Wide Web (Working of WWW, HTTP Request and Response); Electronic Mail (Working of Email, SMTP Packet, Multipurpose Internet Mail Extension); Other Applications (Telnet, Instant Messaging, Videoconferencing).

Introduction

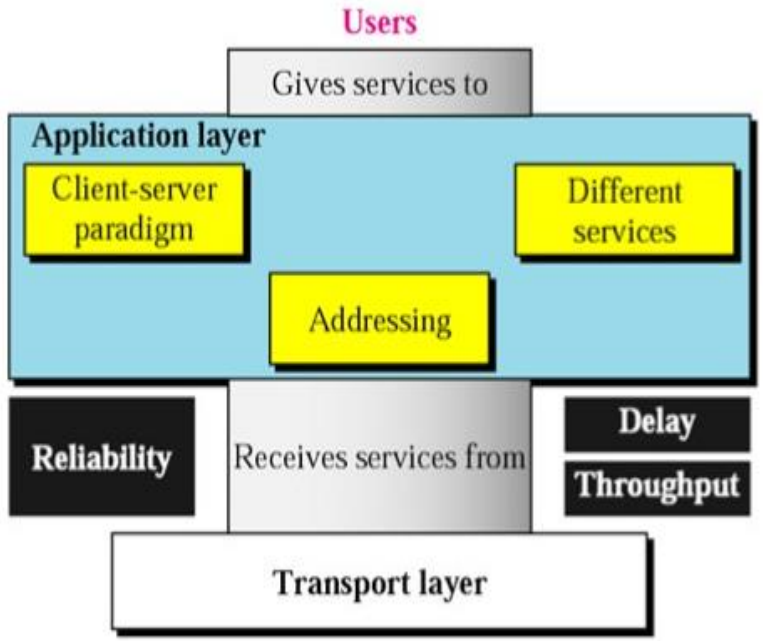
- The application layer is the top-most layer of OSI model.
- It provides services directly to user applications.
- It enables the to access the network.
- It provides user interfaces and support for services such as email, remote file access and transfer, shared database management and other types of distributed information services.
- The functions of the application layer are:

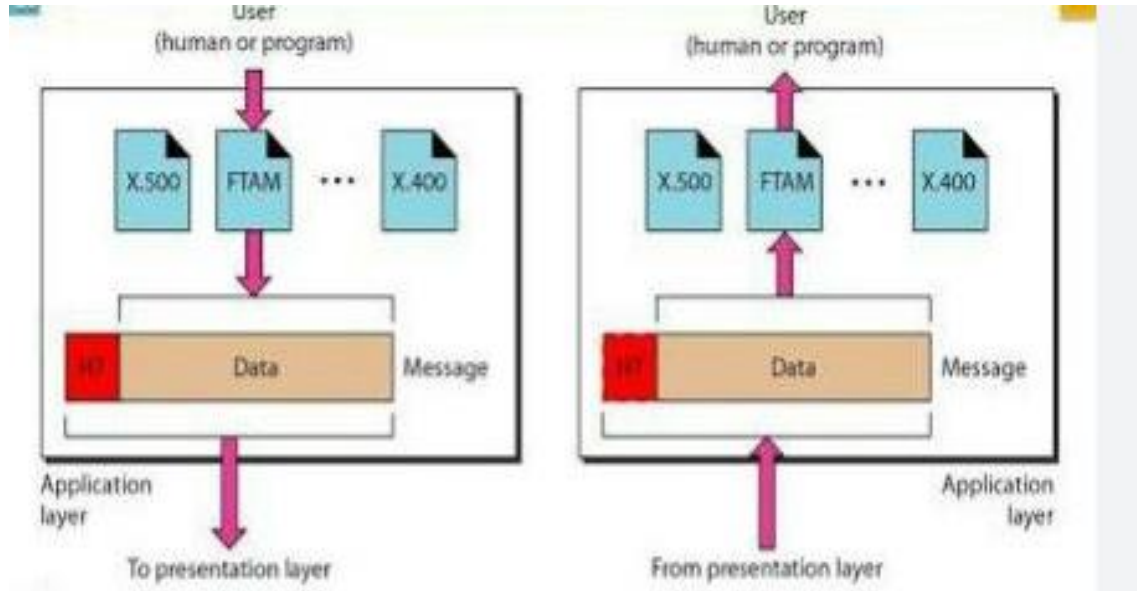
- It facilitates the user to use the services of the network.

- It is used to develop network-based applications.

- It provides user services like user login, naming network devices, formatting messages, and e-mails, transfer of files etc.

- It is also concerned with error handling and recovery of the message as a whole


Users
Gives services to
Application layer
Client-server paradigm
Different services
Addressing
Reliability
Receives services from
Delay
Throughput
Transport layer


User (human or program)
X.500
FTAM
X.400
HT
Data
Message
Application layer
To presentation layer
User (human or program)
X.500
FTAM
X.400
HT
Data
Message
Application layer
From presentation layer

Application layer protocols

- Hyper Text Transfer Protocol, HTTP: It is the underlying protocol for world wide web. It defines how hypermedia messages are formatted and transmitted.
- File Transfer Protocol, FTP: It is a client-server based protocol for transfer of files between client and server over the network.
- Simple Mail Transfer Protocol, SMTP: It lays down the rules and semantics for sending and receiving electronic mails (e-mails).
- Domain Name System, DNS: It is a naming system for devices in networks. It provides services for translating domain names to IP addresses.
- TELNET: It provides bi-directional text-oriented services for remote login to the hosts over the network.
- Simple Network Management Protocol, SNMP: It is for managing, monitoring the network and for organizing information about the networked devices.

- **Functionalities of the Application layer**

Specific **functionalities of the Application layer** are as follows:

1. Network Virtual terminal

- The application layer is the software version of a physical terminal and this layer permitted to a user to log on to a remote host.
- For this, an application creates a software emulation of a terminal at the remote host. By this user's computer can communicate with the software terminal, which in turn, communicates with the host.
- It is shown that the remote host is communicating with one of its terminals, so it allows the user to log on.

2. File Transfer, Access, and Management (FTAM)

- An application permits a user to access files in a remote computer, to retrieve files from a computer and to manage files on a remote computer.
- FTAM is concerned with a hierarchical virtual file in terms of file attributes, file structure and the types of operations performed on the files and their attributes.

3. Addressing

- To achieve communication between client and server system, there is a need for addressing.
- When a request is sent from the client side to the server side, this request contains the server address and its own address.
- The server answered to the client request, this request contains the destination address, i.e., client address. DNS is used to achieve this type of addressing.

4. Mail Services

- Email forwarding and storage of e-mails provided by an application layer.

5. Directory Services

- A distributed database is contained by an application that provides access for global information about various objects and services.

6. Authentication

- It provides authentication to occur between devices for an extra layer of security and it authenticates the sender or receiver's message or both.

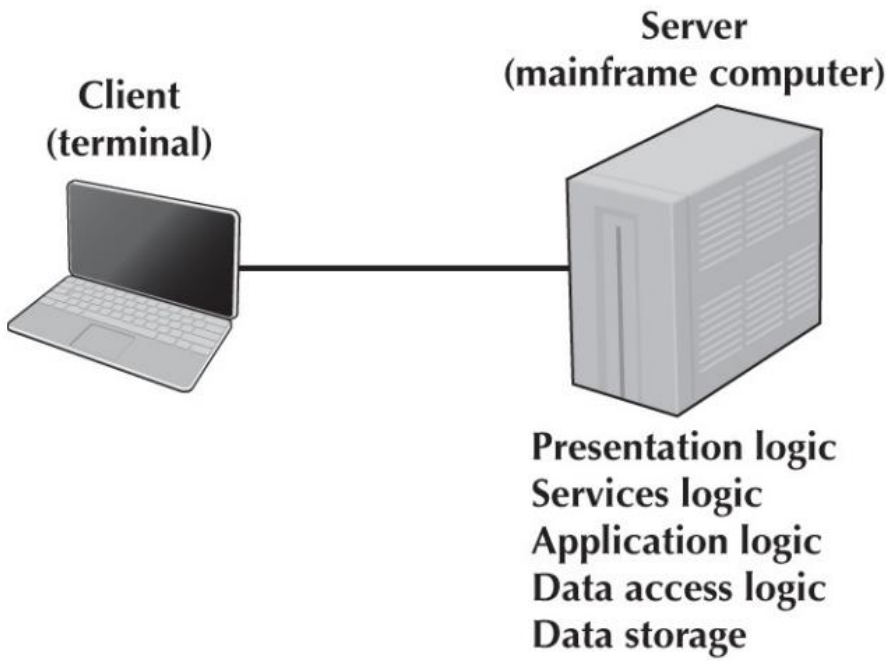
Application Architecture

- It should be clear that to use the Internet, two application programs are needed to interact with each other: one running on a computer somewhere in the world, the other running on another computer somewhere else in the world.
- An application architecture is the way in which the functions of the application layer software are spread among the clients and servers in the network.
- The work done by any application program can be divided into four general functions which are the basic building blocks of any application.
- **Data storage:** Most application programs require data to be stored and retrieved. whether it is a small file such as a memo produced by a word processor or a large database such as an organization's accounting records.
- **Data access logic**: the processing required to access data, which often means database queries in SQL (structured query language).
- **Application logic (sometimes called business logic),** which also can be simple or complex, depending on the application.
- **Presentation logic**, the presentation of information to the user and the acceptance of the user's commands.

- There are many ways in which these four functions can be allocated between the client computers and the servers in a network. There are four fundamental application architectures in use today.
- In **host-based architectures,** the server (or host computer) performs virtually all of the work.
- In **client-based architectures,** the client computers perform most of the work.
- In **client-server architectures,** the work is shared between the servers and clients.
- In **peer-to-peer architectures,** computers are both clients and servers and thus share the work. The client-server architecture is the dominant application architecture.

- **Host Based Architectures:**

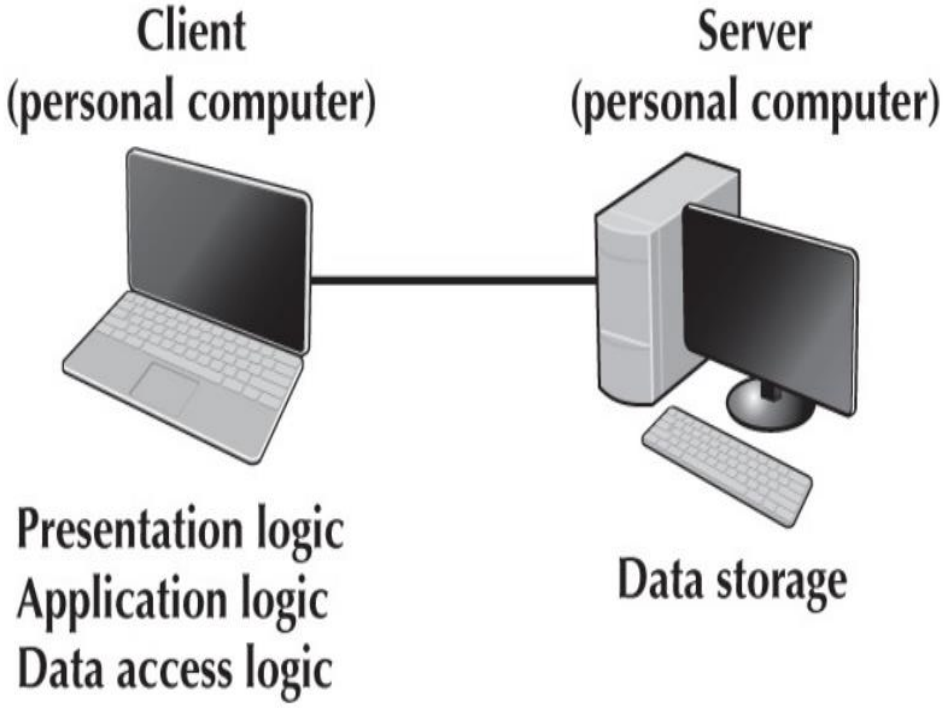
The very first data communications networks developed in the 1960s were host-based, with the server (usually a large mainframe computer) performing all functions. The clients (usually terminals) enabled users to send and receive messages to and from the host computer. The clients merely captured keystrokes, sent them to the server for processing, and accepted instructions from the server on what to display.



- This very simple architecture often works very well. Application software is developed and stored on the one server along with all data. If you've ever used a terminal you've used a host-based application. There is one point of control, because all messages flow through the one central server. In theory, there are economies of scale, because all computer resources are centralized.
- There are some fundamental problems with host-based networks. The server must process all messages. As the demands for more and more network applications grow, many servers become overloaded and unable to quickly process all the users' demands. Prioritizing users' access becomes difficult. Response time becomes slower, and network managers are required to spend increasingly more money to upgrade the server.

- **Client Based Architectures:**

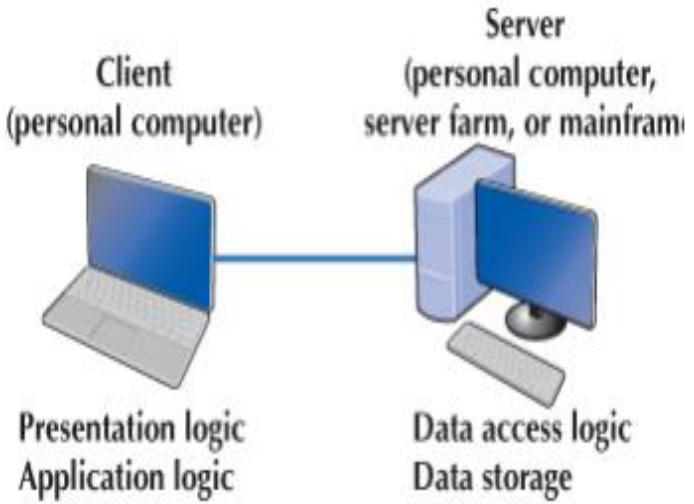
In the late 1980s, there was an explosion in the use of personal computers. Today, more than 90% of most organizations' total computer processing power now resides on personal computers, not in centralized mainframe computers. Part of this expansion was fueled by a number of low-cost, highly popular applications such as word processors, spreadsheets, and presentation graphics programs. It was also fueled in part by managers' frustrations with application software on host mainframe computers. Most mainframe software is not as easy to use as personal computer software, is far more expensive, and can take years to develop.



- With client-based architectures, the clients are personal computers on a LAN, and the server is usually another personal computer on the same network. The application software on the client computers is responsible for the presentation logic, the application logic, and the data access logic; the server simply stores the data.
- This simple architecture often works very well. If you've ever used a word processor and stored your document file on a server (or written a program in Visual Basic or C that runs on your computer but stores data on a server), you've used a client-based architecture.
- The fundamental problem in client-based networks is that all data on the server must travel to the client for processing. For example, suppose the user wishes to display a list of all employees with company life insurance. All the data in the database (or all the indices) must travel from the server where the database is stored over the network circuit to the client, which then examines each record to see if it matches the data requested by the user. This can overload the network circuits because far more data are transmitted from the server to the client than the client actually needs.

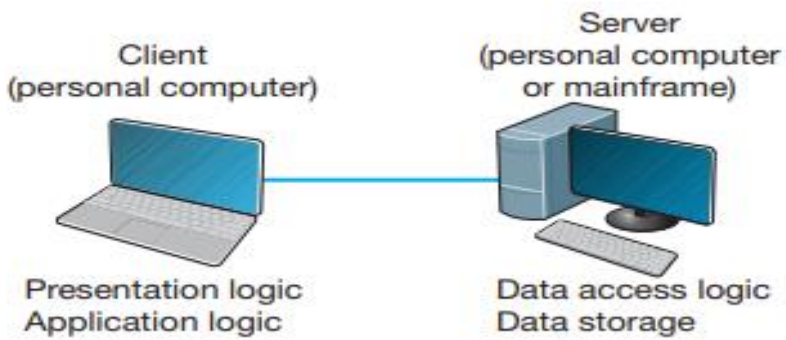
- **Client-Server Architecture:**

- Most applications written today use client–server architectures. Client– server architectures attempt to balance the processing between the client and the server by having both do some of the logic. In these networks, the client is responsible for the presentation logic, whereas the server is responsible for the data access logic and data storage. The application logic may either reside on the client, reside on the server, or be split between both.

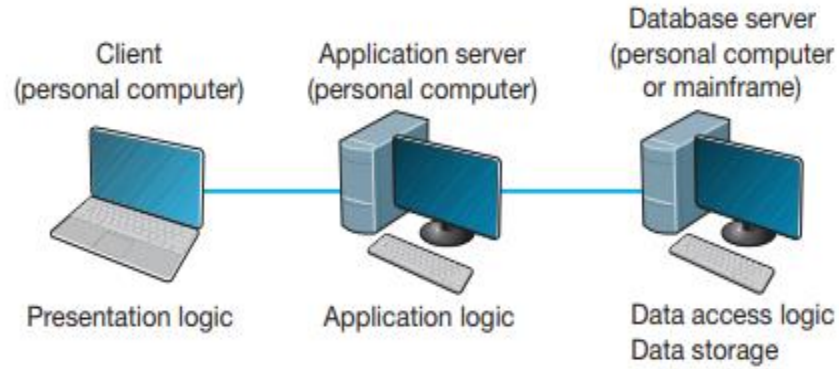


- Figure shows the simplest case, with the presentation logic and application logic on the client and the data access logic and data storage on the server.
- In this case, the client software accepts user requests and performs the application logic that produces database requests that are transmitted to the server.
- The server software accepts the database requests, performs the data access logic, and transmits the results to the client. The client software accepts the results and presents them to the user. When you used a Web browser to get pages from a Web server, you used a client– server architecture. Likewise, if you've ever written a program that uses SQL to talk to a database on a server, you've used a client–server architecture.

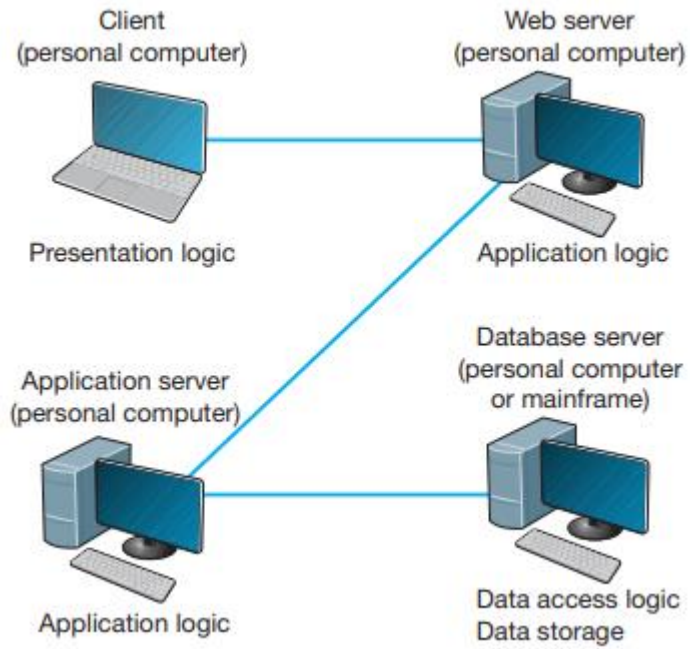
- **Types of Client-Server Architecture:**
- There are many ways in which the application logic can be partitioned between the client and the server.
- **2-tier architecture:** In 2-tier architecture, the presentation and business logic layers are stored on the client while the data layer is stored on a server. So as long as the code of an application is fully executed on the client and some of the data is being stored in a remote database, that application fits the 2-tier architecture criteria. A desktop application that requires you to log into an online account is a good example.
- **3-tier architecture:** The presentation layer is stored on the client, the business logic layer is stored on one server, and the data layer is stored on another server. For example, you use a smartphone app's user interface to interact with an app while an application server executes most of the app's code and a database server stores the data.
- **N-tier architecture:** In N-tier architecture, the presentation and data layers are left untouched, compared to the 3-tier architecture. The difference is that N-tier architecture splits the business logic layer into multiple layers to improve performance, management, and stability.



Two – Tier



Three– Tier



N– Tier

- Another way of classifying client server architectures is by examining how much of the application logic is placed on the client computer.
- **Thin-client:** A thin client is a lightweight computer that relies on the resources of the host computer. It works by connecting with a remote server, where applications and data are stored. They don't have storage capability to store user's data, so the date stores at a remote server from which the thin client is connected to. In terms of security, they are more secure than thick clients as they have fewer security threats. System management is much easier in thin clients.
- **Thick-client**: The thick client relies lightly upon the server and provides rich functionality. The majority of data processing is performed by thick clients. They are also called as heavy or fat clients. In terms of security, they are less secure than thin client as they have more security threats. They are not dependent upon the applications of server as they have their own software applications and operating system.

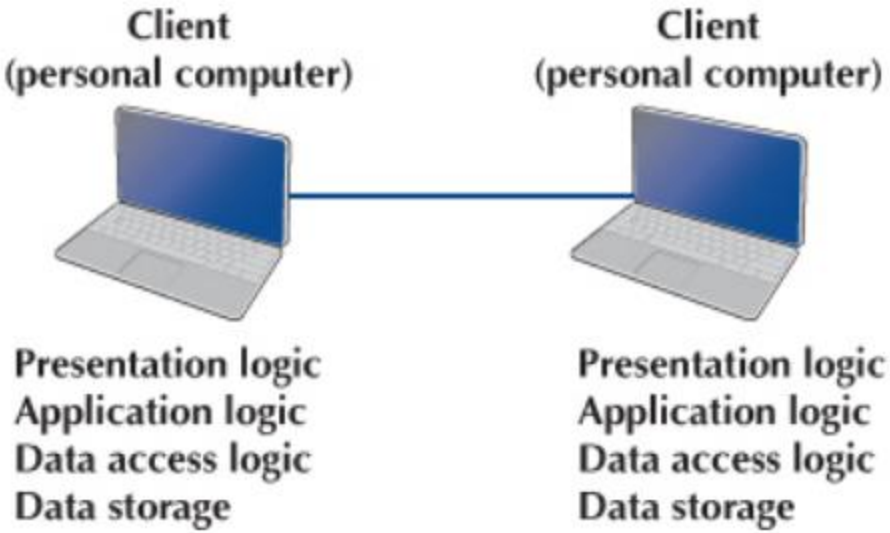
- **Cloud Computing Architectures**
- The traditional client–server architecture can be complicated and expensive to deploy. Every application has to be hosted on a server so that it can fulfill requests from potentially thousands of clients. An organization has hundreds of applications, so running a successful client–server architecture requires a variety of software and hardware and the skilled personnel who can build and maintain this architecture.
- Cloud computing architectures are different because they outsource part or all of the infrastructure to other firms that specialize in managing that infrastructure. There are three common cloud-based architecture models. Figure summarizes these three models and compares them to the client– server architecture

	Thin-Client Client-Server		Infrastructure as a Service		Platform as a Service		Software as a Service	
Who manages which parts	Internal	Outsourced	Internal	Outsourced	Internal	Outsourced	Internal	Outsourced
Application Logic	X		X		X			X
Data Storage	X		X		X			X
Data Access Logic	X		X			X		X
Operating System	X		X			X		X
Virtualization Software	X		X			X		X
Server Hardware	X			X		X		X
Storage Hardware	X			X		X		X
Network Hardware	X			X		X		X

- **Software as a Service (SaaS):**
- Saas is one of the three cloud computing models. With SaaS, an organization outsources the entire application to the cloud provider and uses it as any other application that is available via a browser (thin client). SaaS is based on multitenancy. This means that rather than having many copies of the same application, there is only one application that everybody shares, yet everybody can customize it for his or her specific needs. Popular SaaS offering the include Microsoft 365, Salesforce, and Google Workspace (also known as, G Suite).
- **Platform as a Service (PaaS):**
- PaaS is another of the three cloud computing models. What if there is an application you need but no cloud provider offers one you like? You can build your own application and manage your own data on the cloud infrastructure provided by your cloud supplier. This model is called Platform as a Service. The developers in your organization decide what programming language to use to develop the application of choice. The needed hardware and software infrastructure, called the platform, is rented from the cloud provider.

- **Infrastructure as a Service (laaS):**
- With laas, the cloud provider manages the hardware, including servers, storage, and networking components. The organization is responsible for all the software, including operating system (and virtualization software), database software, and its applications and data. IaaS is sometimes referred to also as Haas, or Hardware as a Service, because in this cloud model, only the hardware is provided; everything else is up to the organization This model allows a decrease in capital expenditures for hardware and maintaining the proper environment (e.g., cooling) and redundancy, and backups for data and applications. Providers of laas are Amazon Web Services, Microsoft Windows Azure, and so on.

- **Peer-to-Peer Architectures:**
- Peer-to-peer (P2P) architectures are very old, but their modern design became popular in the early 2000s with the rise of P2P file-sharing applications (e.g., Napster). With a P2P architecture, all computers act as both a client and a server. Therefore, all computers perform all four functions: presentation logic, application logic, data access logic, and data storage. With a P2P file-sharing application, a user uses the presentation, application, and data access logic installed on his or her computer to access the data stored on another computer in the network. With a P2P application-sharing network (e.g., grid computing such as seti.org), other users in the network can use others' computers to access application logic as well.



Peer-to-peer architecture

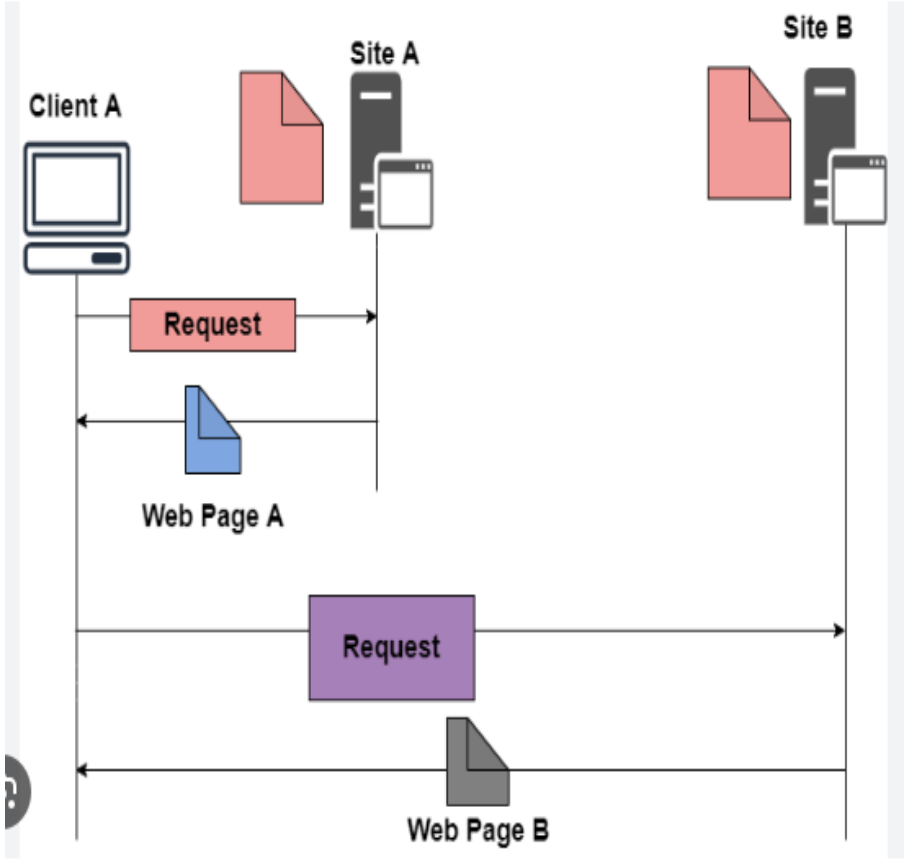
- The advantage of P2P networks is that the data can be installed anywhere on the network. They spread the storage throughout the network, even globally, so they can be very resilient to the failure of any one computer. The challenge is finding the data. There must be some central server that enables you to find the data you need, so P2P architectures often are combined with a client– server architecture. Security is a major concern in most P2P networks, so P2P architectures are not commonly used in organizations, except for specialized computing needs (e.g., grid computing).

- **Choosing Architectures**
- Each of the preceding architectures has certain costs and benefits, so how do you choose the “right” architecture? In many cases, the architecture is simply a given; the organization has a certain architecture, and one simply has to use it. In other cases, the organization is acquiring new equipment and writing new software and has the opportunity to develop a new architecture, at least in some part of the organization.
- Almost all new applications today are client–server applications. Client– server architectures provide the best scalability, the ability to increase (or decrease) the capacity of the servers to meet changing needs. For example, we can easily add or remove application servers or database servers depending on whether we need more or less capacity for application software or database software and storage.
- Client–server architectures are also the most reliable. We can use multiple servers to perform the same tasks, so that if one server fails, the remaining servers continue to operate and users don’t notice problems.

- Finally, client–server architectures are usually the cheapest because many tools exist to develop them. And lots of client–server software exists for specific parts of applications so we can more quickly buy parts of the application we need. For example, no one writes Shopping Carts anymore; it’s cheaper to buy a Shopping Carts software application and put it on an application server than it is to write your own.
- Client–server architectures also enable cloud computing. As we mentioned in cloud computing architectures, companies may choose to run a SaaS because of low price and high scalability compared to traditional client–server architecture hosted at home. One major issue that companies face when choosing SaaS is the security of the data. Each company has to evaluate the risk of its data being compromised and select its cloud provider carefully. However, SaaS is gaining popularity and companies are becoming more and more accustomed to this solution.

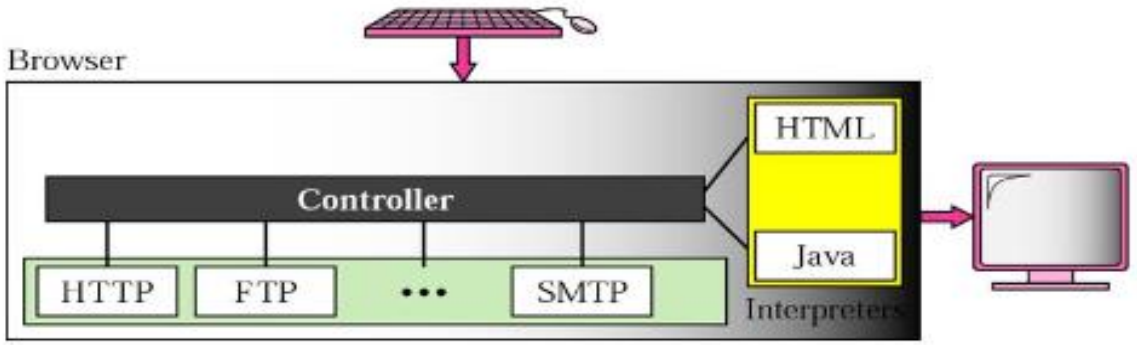
World Wide Web (WWW)

- The World Wide Web (WWW), or the Web, is a repository of information spread all over the world and linked together. The WWW has a unique combination of flexibility, portability, and user-friendly features that distinguish it from other services provided by the Internet. The WWW project was initiated by CERN (European Laboratory for Particle Physics) to create a system to handle distributed resources necessary for scientific research.
- Today it 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 websites, as shown in figure. Each website holds one or more documents, referred to as web pages. Each web page, however can contain some links to other web pages in the same or other websites. The pages can be retrieved and viewed by using browsers.



- Let us go through the scenario shown in figure. The client needs to see some information that it knows belongs to site A. It sends a request through its browser, a program that is designed to fetch Web documents. The request, among other information, includes the address of the site and the Web page, called the URL. The server at site A finds the document and sends it to the client. When the user views the document, s/he finds some references to other documents, including a Web page at site B. The reference has the URL for the new site. The user is also interested in seeing this document. The client sends another request to the new site, and the new page is retrieved.
- The above example shows the idea of hypertext and hypermedia. Hypertext means creating documents that refer to other documents. In a hypertext document, a part of text can be defined as a link to another document. When a hypertext is viewed with a browser, the link can be clicked to retrieve the other document. Hypermedia is a term applied to document that contains links to other textual document or documents containing graphics, video, or audio.

- **Browser:**
- A variety of vendors offer commercial browsers that interpret and display a web document, and all of them use nearly the same architecture. Each browser usually consists of three parts a controller, client programs, and interpreters as shown in figure. 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 client program can be one of the protocols described previously such as FTP, or TELNET, but it is usually HTTP. The interpreter is a language used today on the Internet such as HTML, Java, or JavaScript.

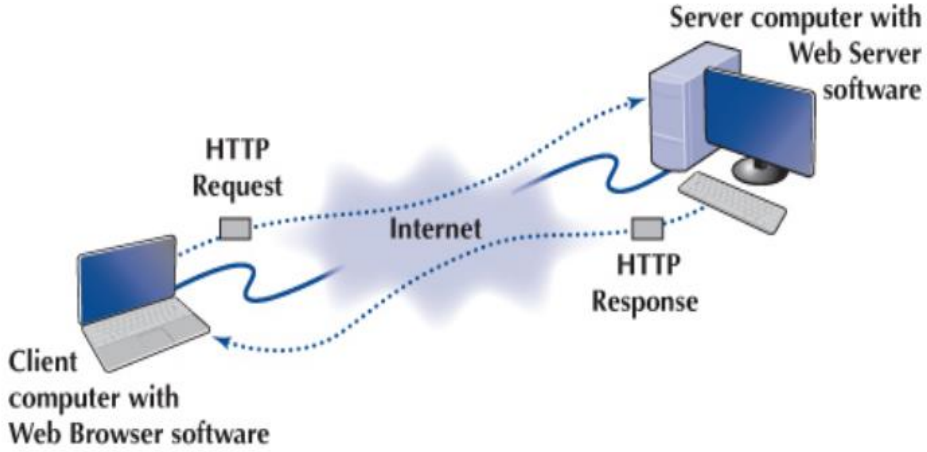


- **Web Server:**
- The server stores all pages belonging to the site. Each time a client request arrives, the corresponding document is sent to the client. To improve efficiency, servers normally store requested files in a cache in memory: memory is faster to access than disk. Some popular Web servers include Apache and Microsoft Internet Information Server.
- **Uniform Resource Locator (URL):**
- A client that wants to access a Web page needs the file name and the address. To facilitate the access of documents distributed throughout the world, HTTP uses locators. The uniform resource locator (URL) is a standard locator for specifying any kind of information on the Internet. The URL defines four things: protocol, host computer, port, and path.

- To combine these four pieces together, the uniform resource locator (URL) has been designed; it uses three different separators between the four pieces as shown below:
- protocol://host/path - Used most of the time
- protocol://host:port/path→ Used when port number is needed
- The protocol is the client-server application program used to retrieve the document. Many different protocols can retrieve a document; among them are Gopher, FTP, HTTP, News, and TELNET. The host is the domain name of the computer on which the information is located. 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.

How the Web Works

- The Web is a good example of a two-tier client–server architecture. Each client computer needs an application layer software package called a Web browser. There are many different browsers, such as Microsoft's Internet Explorer. Each server on the network that will act as a Web server needs an application layer software package called a Web server. There are many different Web servers, such as those produced by Microsoft and Apache.
- To get a page from the Web, the user must type the Internet uniform resource locator (URL) for the page he or she wants (e.g., www.yahoo.com) or click on a link that provides the URL. The URL specifies the Internet address of the Web server and the directory and name of the specific page wanted. If no directory and page are specified, the Web server will provide whatever page has been defined as the site's home page.

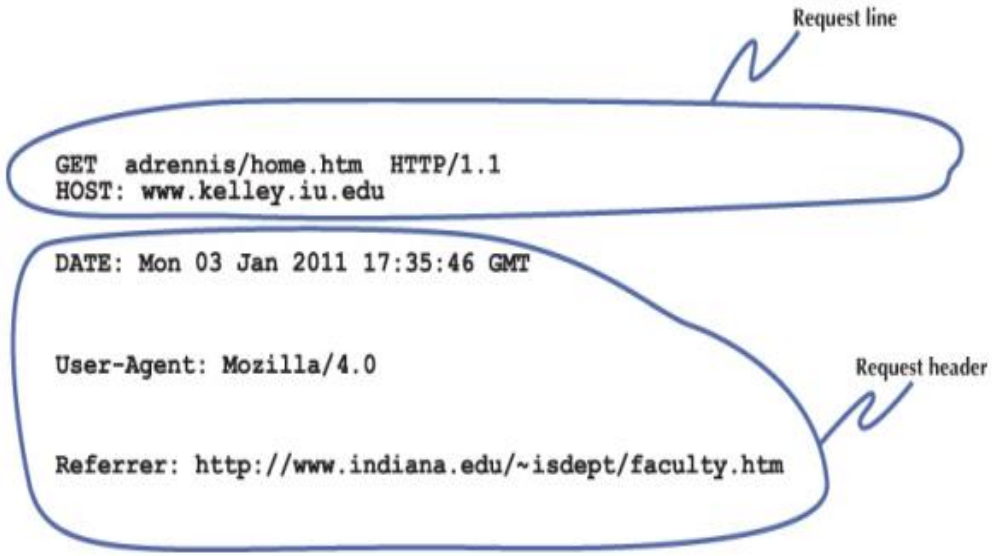


- For the requests from the Web browser to be understood by the Web server, they must use the same standard protocol or language. If there were no standard and each Web browser used a different protocol to request pages, then it would be impossible for a Microsoft Web browser to communicate with an Apache Web server, for example.
- The standard protocol for communication between a Web browser and a Web server is Hypertext Transfer Protocol (HTTP). To get a page from a Web server, the Web browser issues a special packet called an HTTP request that contains the URL and other information about the Web page requested. Once the server receives the HTTP request, it processes it and sends back an HTTP response, which will be the requested page or an error message.
- This request–response dialogue occurs for every file transferred between the client and the server. For example, suppose the client requests a Web page that has two graphic images. Graphics are stored in separate files from the Web page itself using a different file format than the HTML used for the Web page (e.g., in JPEG [Joint Photographic Experts Group] format). In this case, there would be three request–response pairs. First, the browser would issue a request for the Web page, and the server would send the response. Then, the browser would begin displaying the Web page and notice the two graphic files. The browser would then send a request for the first graphic and a request for the second graphic, and the server would reply with two separate HTTP responses, one for each request.

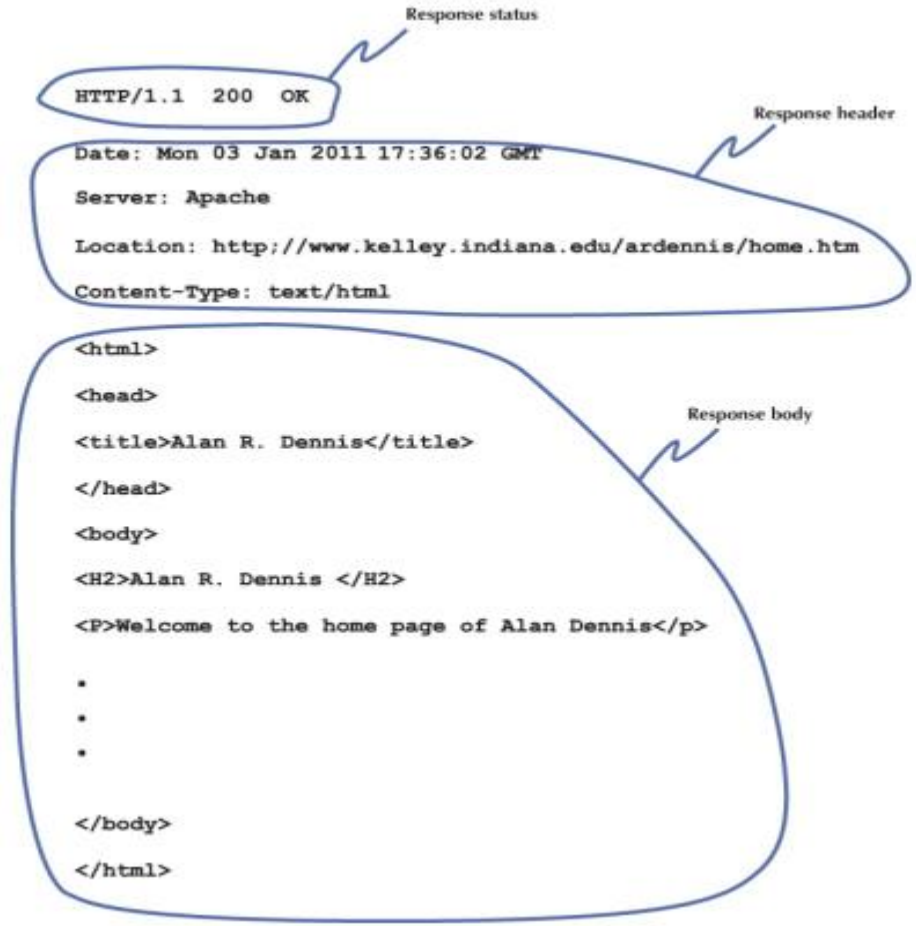
HTTP AND HTTPS

- The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the World Wide Web. The protocol transfers data in the form of plain text, hypertext, audio, video, and so on. It is called the Hypertext Transfer Protocol because its efficiency allows its use in a hypertext environment where there are rapid jumps from one document to another.
- An HTTP request from a Web browser to a Web server has three parts. The first two parts are required; the last is optional. The parts are as follows:
- **The request line,** which starts with a command (e.g., get), provides the Web page, and ends with the HTTP version number that the browser understands; the version number ensures that the Web server does not attempt to use a more advanced or newer version of the HTTP standard that the browser does not understand.
- **The request header**, which contains a variety of optional information such as the Web browser being used (e.g., Internet Explorer) and the date.
- **The request body,** which contains information sent to the server, such as information that the user has typed into a form.

- The format of an HTTP response from the server to the browser is very similar to that of the HTTP request. It, too, has three parts, with the first required and the last two optional:
- The **response status**, which contains the HTTP version number the server has used, a status code (e.g., 200 means “OK”; 404 means “not found”), and a reason phrase (a text description of the status code).
- The **response header**, which contains a variety of optional information, such as the Web server being used (e.g., Apache), the date, and the exact URL of the page in the response. The response body, which is the Web page itself.



HTTP REQUEST



HTTP RESPONSE

- The HTTP controls the transactions between a web client and a web server. The HTTP protocol transparently makes use of DNS and other Internet protocols to form connections between the web client and the web server, so the user is aware of only the web site's domain name and the name of the document itself. HTTP uses the services of TCP on well-known port 80.

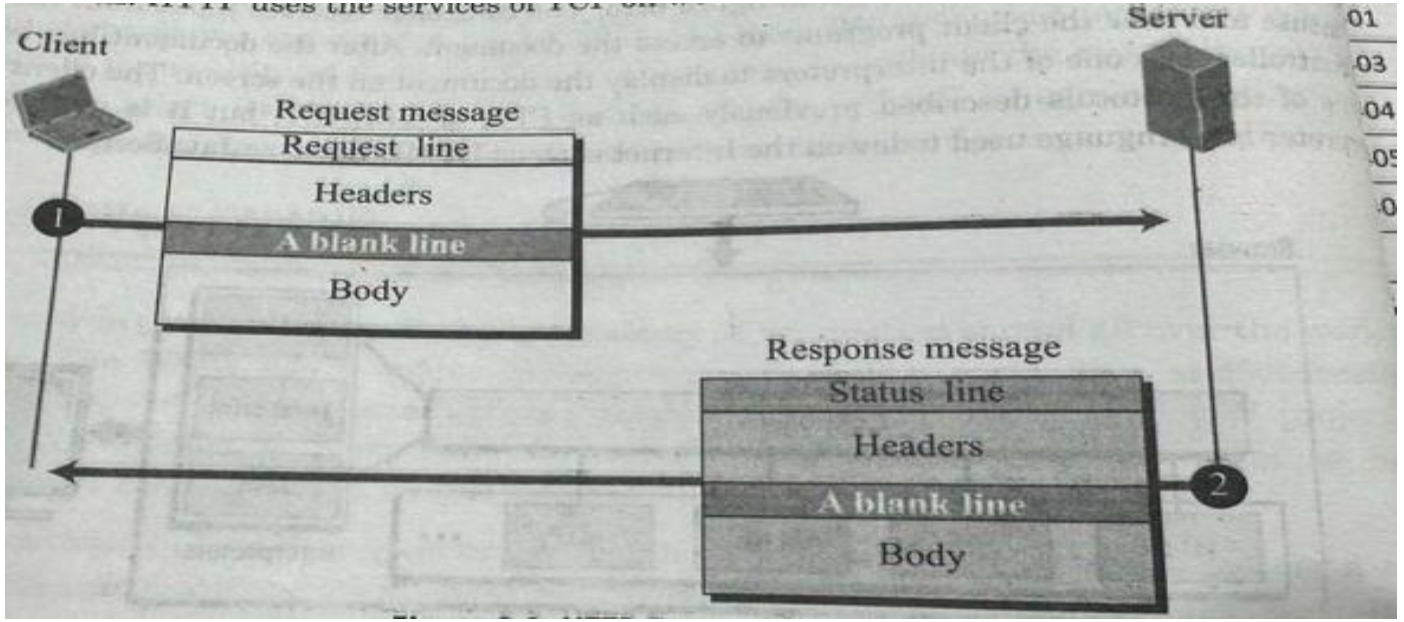


Table: Methods in request line of request messages

Method	Description
GET	Request for resource from server
POST	Submit data to the server
HEAD	Same as GET but does not return the body
PUT	The data within the request must be stored at the URL supplied, replacing any existing data.
DELETE	Delete a resource
OPTIONS	Return the HTTP methods supported by the server
CONNECT	Client requests the HTTP proxy to forward a TCP connection to some destination. Used to create a TCP/IP tunnel for secure connections using HTTP proxies.

Table: Status Code a of response message

Sr. No.	Code and Description
1.	**1xx: Informational** It means the request has been received and the process is continuing.
2.	**2xx: Success** It means the action was successfully received, understood and accepted.
3.	**3xx: Redirection** It means further action must be taken in order to complete the request.
4.	**4xx: Client Error** It means the request contains incorrect syntax or cannot be fulfilled.
5.	**5xx: Server Error** It means the server failed to fulfill an apparently valid request.

- This example retrieves a document in. We use the GET method to retrieve an image with the path /usr/bin/image1. The request line shows the method (GET), the URL, and the HTTP version (1.1). The header has two lines that show that the client can accept images in the GIF of 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, MIME version, and length of the document. The body of the document follows the header.

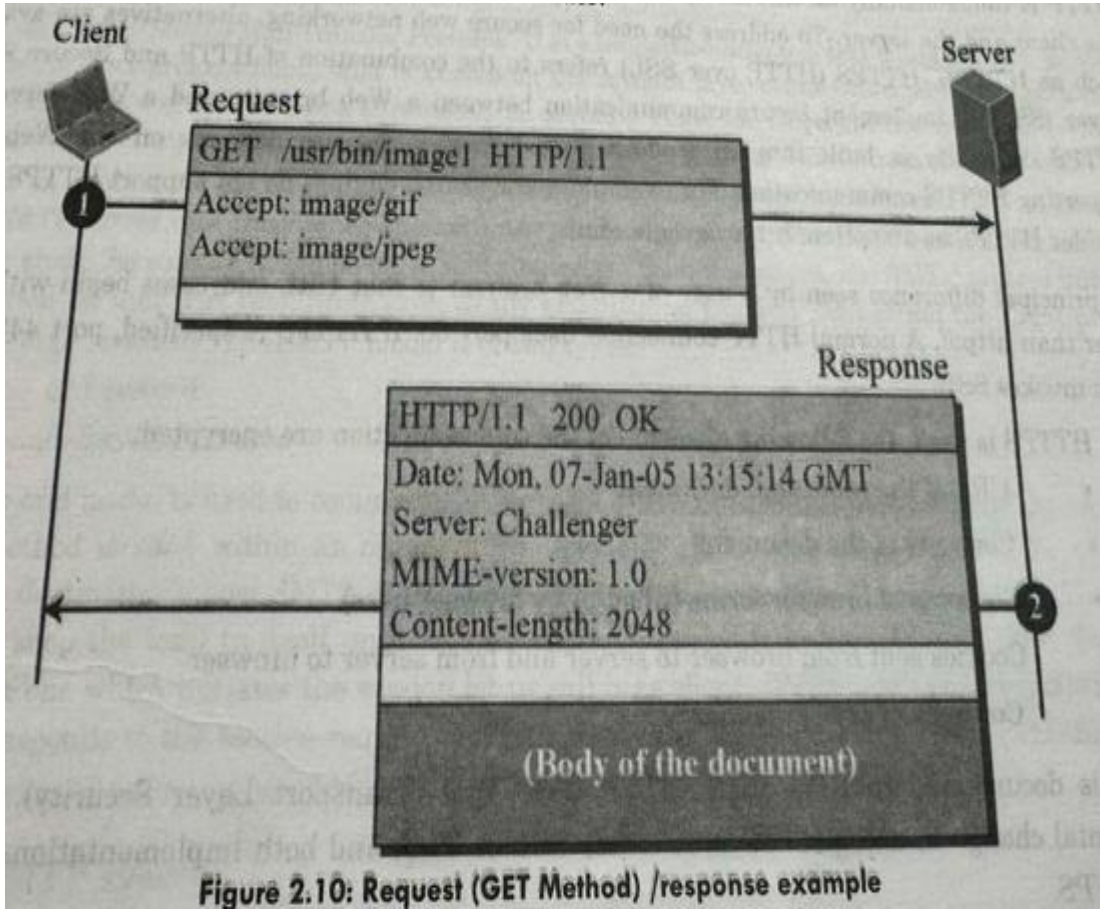


Figure 2.10: Request (GET Method) /response example

- In this example, the client wants to send data to the server. We use the POST method. The request line shows the method (POST), URL, and HTTP version (1.1). There are four lines of headers. The request body contains the input information. The response message contains the status line and four lines of headers. The created document, which is a CGI Common Gateway Interface document, is included as the body.

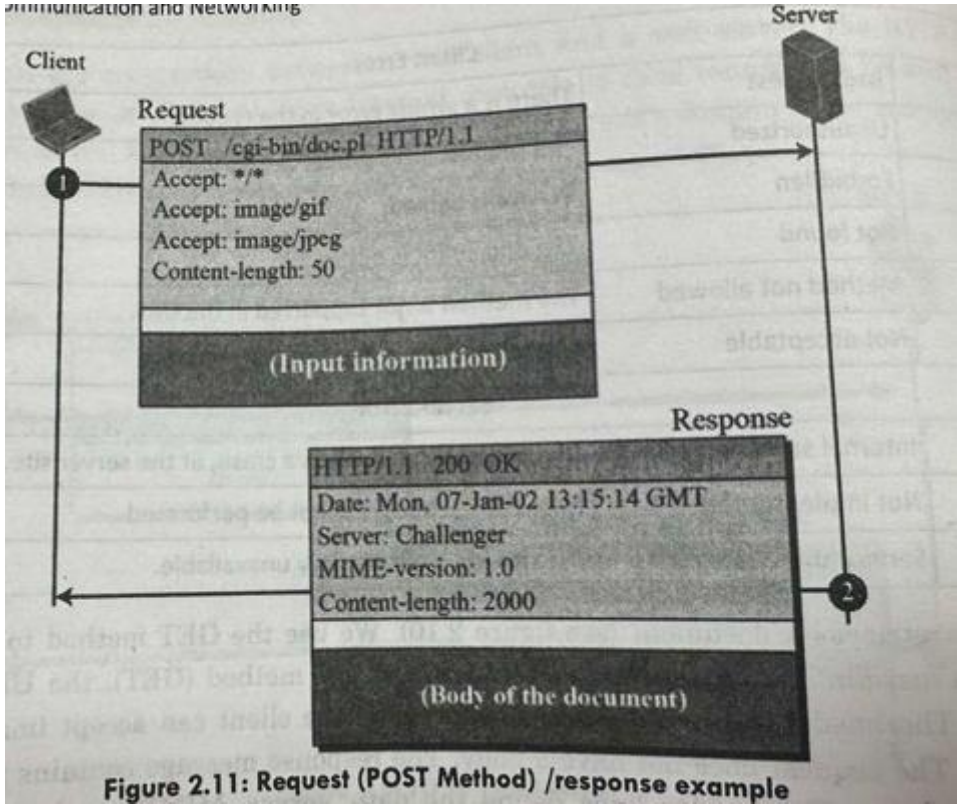


Figure 2.11: Request (POST Method) /response example

- HTTP is fundamentally an insecure protocol. Text-based information is sent in the clear" between the client and the server. To address the need for secure web networking, alternatives are available, such as HTTPS. HTTPS (HTTP over SSL) refers to the combination of HTTP and Secure Sockets Layer (SSL) to implement secure communication between a Web browser and a Web server. The HTTPS capability is built into all modern Web browsers. Its use depends on the Web server supporting HTTPS communication. For example, some search engines do not support HTTPS. Google provides HTTPS as an option: https://google.com.
- The principal difference seen by a user of a Web browser is that URL addresses begin with https:// rather than http://. A normal HTTP connection uses port 80. If HTTPS is specified, port 443 is used, which invokes SSL.

- When HTTPS is used, the following elements of the communication are encrypted:
- URL of the requested document
- Contents of the document
- Contents of browser forms (filled in by browser user)
- Cookies sent from browser to server and from server to browser
- Contents of HTTP header
- HTTPS is documented in RFC (Request for Comment) 2818, HTTP Over TLS (Transport Layer Security). There is no fundamental change in using HTTP over either SSL or TLS, and both implementations are referred to as HTTPS.

Electronic Mail

- Electronic mail, or e-mail, is the computerized version of writing a letter and mailing it at the local post office. Many people are so committed to using e-mail that if it were taken away tomorrow, some serious social and economic repercussions would be felt throughout the world.
- Many commercial e-mail programs are in existence, as well as a number of free ones that can be downloaded from the Internet. Although each e-mail program has its own unique foel and options, most offer the following services:
- Creating an e-mail message
- Sending an e-mail message to one recipient, multiple recipients, or a mailing list
- Receiving, storing, replying to, and forwarding e-mail messages
- Attaching a file such as a word processing document, a spreadsheet, an image, or a program to an outgoing e-mail message

- Most e-mail systems consist of two parts:

(1) the user agent, which is the portion of the e-mail program that allows a user to create, edit, store, and forward e-mail messages; and

(2) the message transfer agent, which is the portion of the e-mail program that prepares and transfers the e-mail message. Each transmitted e-mail also consists of two basic components: an envelope, which contains information describing the e-mail message, and the message, which is the contents of the envelope.

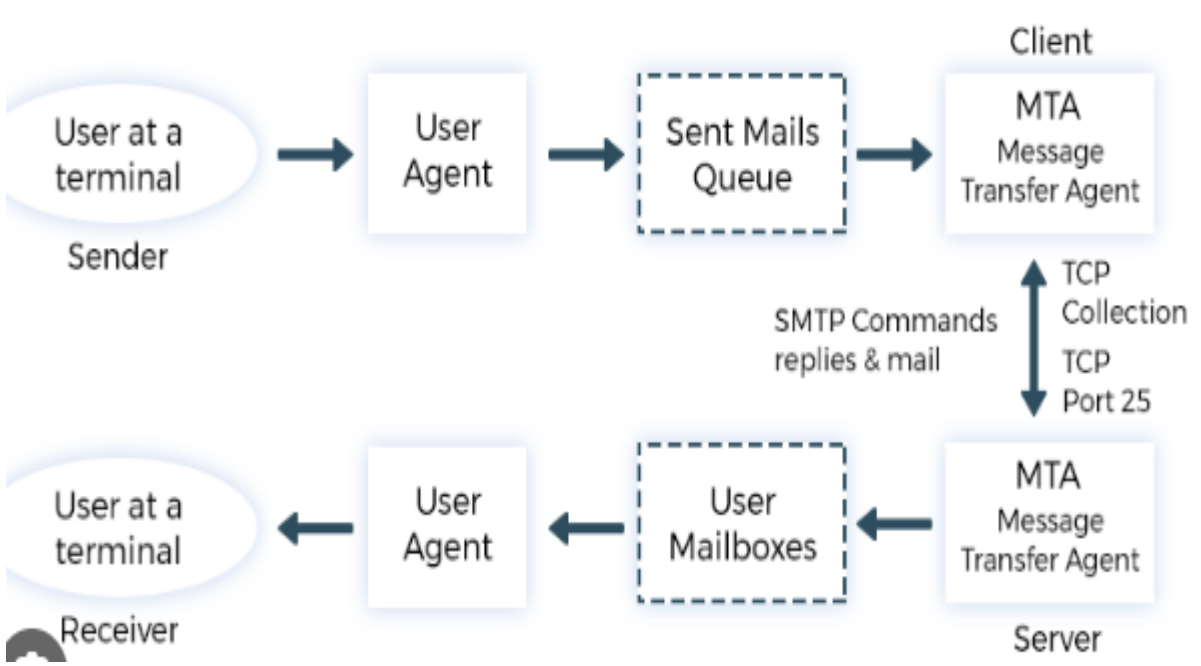
- The Simple Mail Transfer Protocol (SMTP) has always been the workhorse of the TCP/IP suite. However, SMTP has traditionally been limited to the delivery of simple text messages. In recent years, there has been a demand for the capability to deliver mail containing various types of data, including voice, images, and video clips. To satisfy this requirement, a new electronic mail standard, which builds on SMTP, has been defined: the Multi-Purpose Internet Mail Extension (MIME). In this section, we first examine SMTP, and then look at MIME.

- **Simple Mail Transfer Protocol (SMTP)**
- SMTP stands for "Simple Mail Transfer Protocol." It is a connection-oriented, text-based network protocol from the internet protocol family and is located on the seventh layer of the OSI model: the application layer. Just like any other network protocol, it contains rules for correct communication between computers in a network. SMTP is responsible for feeding and forwarding e-mails from sender to recipient. SMTP is an application layer protocol. The client who wants to send the mail opens a TCP connection to the SMTP server and then sends the mail across the connection. The SMTP server is always on listening mode. As soon as it listens for a TCP connection from any client, the SMTP process initiates a connection on that port (25). After successfully establishing the TCP connection the client process sends the mail instantly. The SMTP model is of two types:

1. End-to-end method
2. Store-and- forward method.

- The end-to-end model is used to communicate between different organizations whereas the store and forward method is used within an organization. A SMTP client who wants to send the mail will contact the destination's host SMTP directly in order to send the mail to the destination. The SMTP server will keep the mail to itself until it is successfully copied to the receiver's SMTP. The client SMTP is the one which initiates the session let us call it as client- SMTP and the server SMTP is the one which responds to the session request and let us call it as receiver-SMTP. The client- SMTP will start the session and the receiver-SMTP will respond to the request

- Model of SMTP system:
- In the SMTP model user deals with the user agent (UA) for example Microsoft outlook, Netscape, Mozilla etc. In order to exchange the mail using TCP, MTA is used. The users sending the mail do not have to deal with the Message Transfer Agent (MTA) it is the responsibility of the system admin to set up the local MTA. The MTA maintains a small queue of mails so that it can schedule repeat delivery of mail in case the receiver is not available. The MTA delivers the mail to the mailboxes and the information can later be pulled from the server by message access agents (MAA). Currently two message access protocols are available: Post Office Protocol, version 3 (POP3) and Internet Mail Protocol, Version 4 (IMAP4)



- **IMAP:(Internet Mail Access Protocol)**
- An Internet standard protocol for storing and retrieving messages from Simple Mail Transfer Protocol (SMTP) hosts.

How It Works

- SMTP provides the underlying message transport mechanism for sending e-mail over the Internet, but it does not provide any facility for storing and retrieving messages.
- SMTP hosts must be continuously connected to one another, but most users do not have a dedicated connection to the Internet.
- IMAP4 provides mechanisms for storing messages received by SMTP in a receptacle called a mailbox.
- IMAP4 server stores messages received by each user until the user connects to download and read them using an IMAP4 client such as Microsoft Outlook Express.

Specifically, IMAP4 allows users to

- Access multiple folders, including public folders
- Create hierarchies of folders for storing messages
- Leave messages on the server after reading them so that they can access the messages again from another location
- Search a mailbox for a specific message to download
- Flag messages as read
- Selectively download portions of messages or attachments only
- Review the headers of messages before downloading them

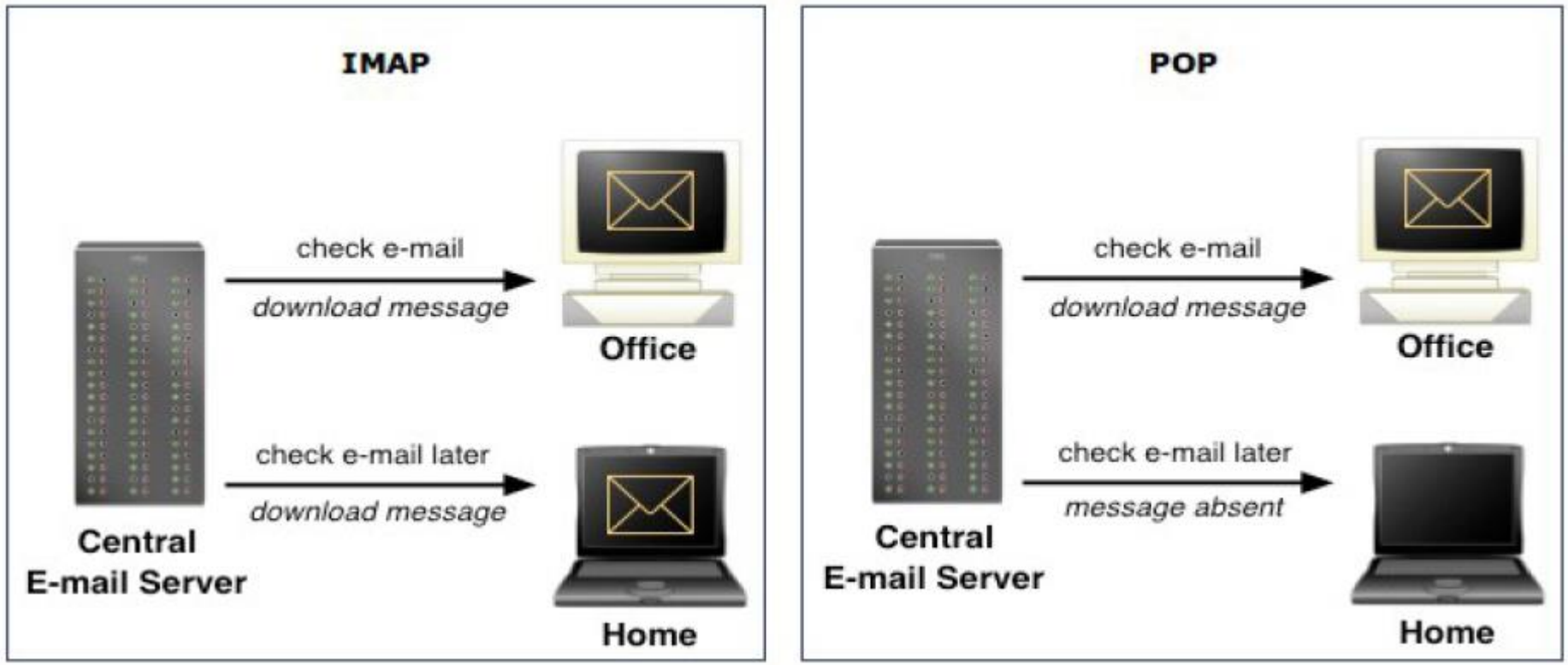
- **Post Office Protocol version 3 (POP3)**
- An Internet standard protocol for storing and retrieving messages from Simple Mail Transfer Protocol (SMTP) hosts.

How It Works

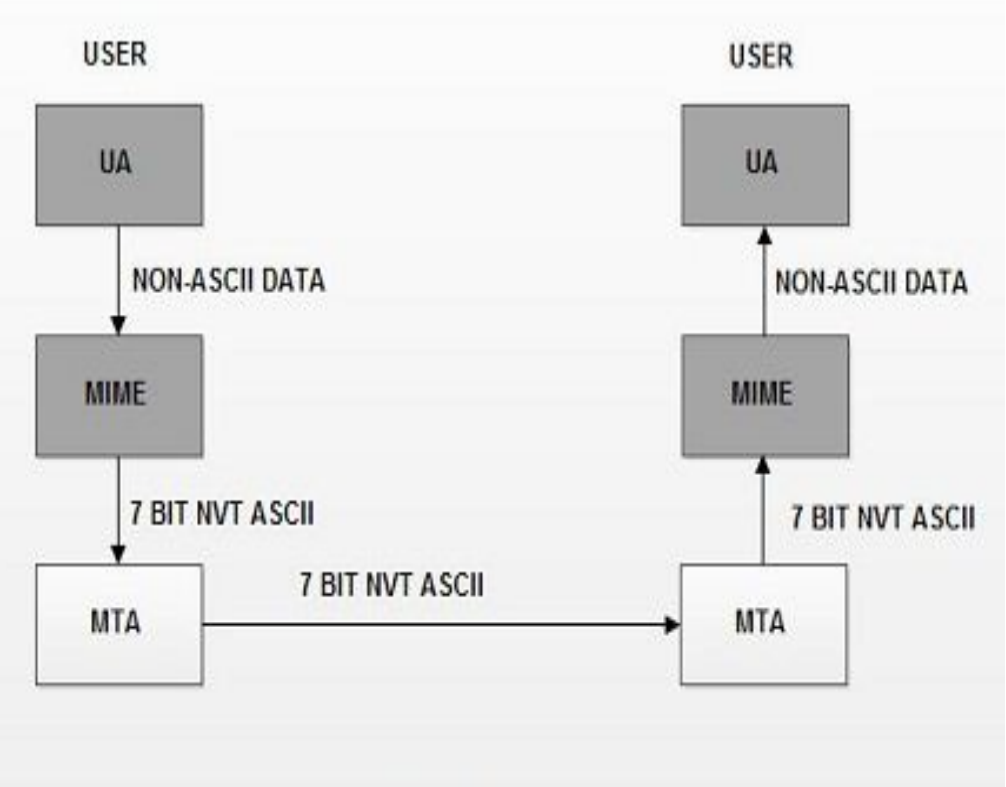
- SMTP provides the underlying transport mechanism for sending e-mail messages over the Internet, but it does not provide any facility for storing messages and retrieving them.
- SMTP hosts must be continuously connected to one another, but most users do not have a dedicated connection to the Internet.
- Post Office Protocol version 3 (POP3) provides mechanisms for storing messages sent to each user and received by SMTP in a receptacle called a mailbox.
- A POP3 server stores messages for each user until the user connects to download and read them using a POP3 client such as Microsoft Outlook 98, Microsoft Outlook Express, or Microsoft Mail and News.
- To retrieve a message from a POP3 server, a POP3 client establishes a Transmission Control Protocol (TCP) session using TCP port 110, identifies itself to the server, and then issues a series of POP3 commands

- **IMAP VS POP:**

- The main difference, as far as we are concerned here, is the way in which IMAP or POP controls your e-mail inbox.
- When you use IMAP you are accessing your inbox on the mail server. IMAP does not actually move messages onto your computer. You can think of an e-mail program using IMAP as a window to your messages on the server. Although the messages appear on your computer while you work with them, they remain on the central mail server.
- POP does the opposite. Instead of just showing you what is in your inbox on the U's mail server, it checks the server for new messages, downloads all the new messages in your inbox onto your computer, and then deletes them from the server. This means that every time you use POP to view your new messages, they are no longer on the central mail server.


IMAP
check e-mail
download message
Office
check e-mail later
download message
Central
E-mail Server
Home
POP
check e-mail
download message
Office
check e-mail later
message absent
Central
E-mail Server
Home

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



Other Applications

- **Telnet:**

- Telnet is a protocol that allows user to log in to remote computers (called hosts) over a TCP/IP network (such as the internet). Using telnet client software on your computer, you can make s connection to a telnet server (that is, the remote host). Once your telnet client establishes a connection to the remote host, your client becomes a virtual terminal, allowing you to communicate with the remote host from your computer. In most cases, you'll need to log into the remote host, which requires that you have an account on that system. Occasionally, you can log in as guest or public without having an account.
- Although 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 in favor of another protocol, Secure Shell (SSH).

- Although Telnet is almost replaced by SSH, it is still widely used today for two reasons:
- The simple plaintext architecture of Telnet allows us to explain the issues and challenges related to the concept of remote logging, which is also used in SSH when it serves as a remote logging protocol.
- Network administrators often use Telnet for diagnostic and debugging purposes.

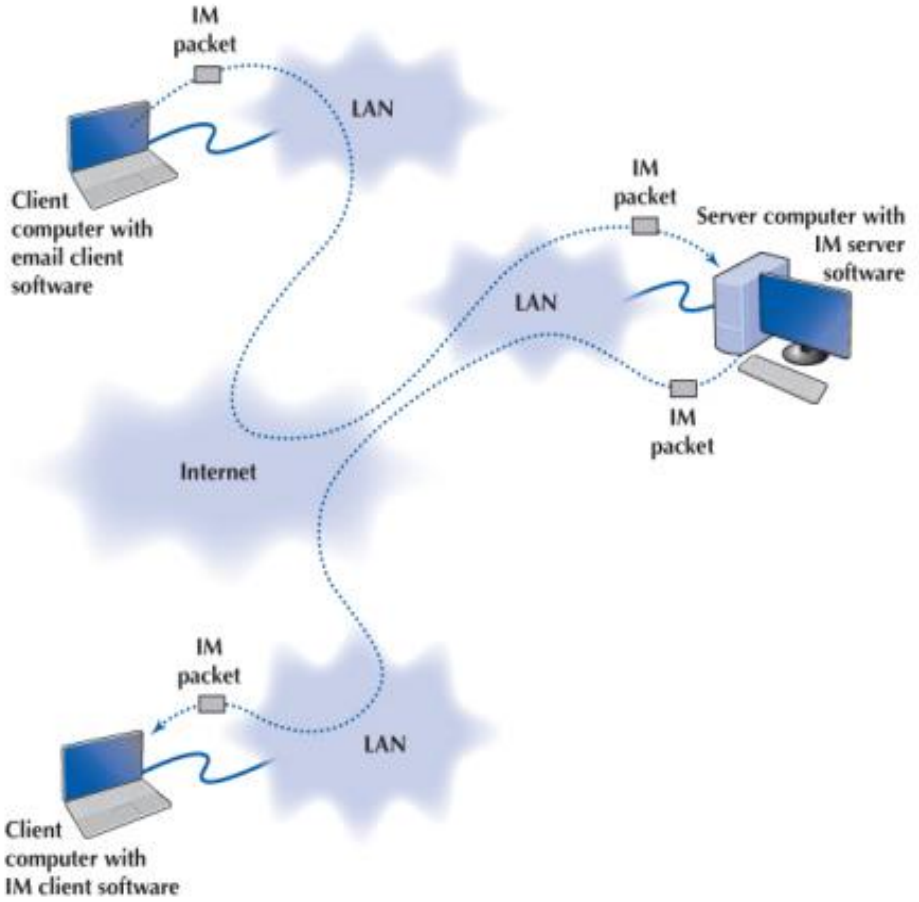
Telnet clients are available for all major operating systems. Command-line telnet clients are built into most versions of macOS, Windows, Unix, and Linux.

- **How telnet works?**
- Telnet also operates on the client/server principle. The local computer uses a telnet client program and the remote, or host, computer uses a telnet server program.
- Telnet uses software installed on your computer, to create a connection with the remote host.
- The telnet client (local computer), at your command, will send a request to the telnet server (remote host).
- The server will reply asking for a user name and password. If accepted, the Telnet client will establish a connection to the host, thus making your computer a virtual terminal and allowing you complete access to the host's computer.
- Telnet requires the use of a user name and password, which means you need to have previously set up an account on the remote computer.
- In some cases, however, computers with telnet will allow guests to log on with restricted access.

- **Instant Messaging**

- One of the fastest growing Internet applications has been instant messaging (IM). With IM, you can exchange real-time typed messages or chat with your friends. Some IM software also enables you to verbally talk with your friends in the same way as you might use the telephone or to use cameras to exchange real-time video in the same way you might use a videoconferencing system. Several types of IM currently exist, including Google Talk and AOL Instant Messenger.
- IM works in much the same way as the Web. The client computer needs an IM client software package, which communicates with an IM server software package that runs on a server. When the user connects to the Internet, the IM client software package sends an IM request packet to the IM server informing it that the user is now online. The IM client software package continues to communicate with the IM server to monitor what other users have connected to the IM server. When one of your friends connects to the IM server, the IM server sends an IM packet to your client computer so that you now know that your friend is connected to the Internet. The server also sends a packet to your friend’s client computer so that he or she knows that you are on the Internet.

- With the click of a button, you can both begin chatting. When you type text, your IM client creates an IM packet that is sent to the IM server. The server then retransmits the packet to your friend. Several people may be part of the same chat session, in which case the server sends a copy of the packet to all of the client computers. IM also provides a way for different servers to communicate with one another, and for the client computers to communicate directly with each other. Additionally, IM will do voice and video.



- **Videoconferencing:**

- Video conferencing is a visual communication session between two or more users regardless of their location, featuring audio and video content transmission in real time. In broad terms, video conferencing software is a solution that allows two or more persons to talk and see each other remotely. Initially video conferencing software only enabled users to make video calls or hold group video conferences. However, as technology advanced, video conferencing softwares acquired plenty of helpful tools and features for remote communication and learning.
- Today, video conferencing is rather a tool for integrated into video collaboration and unified communications platforms that also offer screen sharing, slideshow, recording, instant messaging. project management tools, telephony integration among other features. As an increasing number of companies switch to remote work, video conferencing in gradually becoming the new normal, shaping the way we communicate, learn and work.
- The transmission of video requires a lot of network capacity. Most videoconferencing uses data compression to reduce the amount of data transmitted. Surprisingly, the most common complaint is not the quality of the video image but the quality of the voice transmissions. Special care needs to be taken in the design and placement of microphones and speakers to ensure quality sound and minimal feedback

- Most videoconferencing systems were originally developed by vendors using different formata, so many products were incompatible. The best solution was to ensure that all hardware and software used within an organization was supplied by the same vendor and to hope that any other organizations with whom you wanted to communicate used the same equipment. Today, three standards are in common use: H.320, H.323, and MPEG-2 (also called ISO 13818-2). Each of these standards was developed by different organizations and is supported by different products. They are not compatible, although some application software packages understand more than one standard.
- H.320 is designed for room-to-room videoconferencing over high-speed telephone lines.
- H.323 is a family of standards designed for desktop videoconferencing and just simpleaudio conferencing over the Internet.
- MPEG-2 is designed for faster connections, such as a LAN or specially designed, privately operated WAN
- Webcasting is a special type of one-directional videoconferencing in which content is sent from the server to the user. The developer creates content that is downloaded as needed by the users and played.