SUBJECT : IWT
BRANCH: COMPUTER SC & ENGG.
SEMESTER : 5TH SEM

INTERNET AND WEB TECHNOLOGY

Understanding the WWW and the Internet:

Internet: The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite (TCP/IP) to serve billions of users worldwide. It is a *network of networks* that consists of millions of private, public academic, business, and government networks.

WWW: The World Wide Web, abbreviated as WWW and commonly known as the Web, is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks.

Emergence of Web: Between the summers of 1991 and 1994, the load on the first Web server ("info.cern.ch") rose steadily by a factor of 10 every year.

In 1992 academia, and in 1993 industry, was taking notice. World Wide Web Consortium is formed in September 1994, with a base at MIT is the USA, INRIA in France, and now also at Keio University in Japan.

With the dramatic flood of rich material of all kinds onto the Web in the 1990s, the first part of the dream is largely realized, although still very few people in practice have access to intuitive hypertext creation tools.

The second part has yet to happen, but there are signs and plans which make us confident. The great need for information about information, to help us categorize, sort, pay for own information is driving the design of languages for the web designed for processing by machines, rather than people. The web of human readable document is being merged with a web of machine-understandable data. The potential of the mixture of humans and machines working together and communicating through the web could be immense.

WEB Servers: To view and browse pages on the Web, all you need is a web browser. To publish pages on the Web, you need a web server. A web server is the program that runs on a computer and is responsible for replying to web browser requests for files. You need a web server to publish documents on the Web. When you use a browser to request a page on a website, that browser makes a web connection to a server using the HTTP protocol. The browser then formats the information it got from the server. Server accepts the connection, sends the contents of the requested files and then closes.

WEB Browsers:

A web browser is the program you use to view pages and navigate the World Wide Web. A wide array of web browsers is available for just about every platform you can imagine. Microsoft Internet Explorer, for example, is included with Windows and Safari is included with Mac OS X. Mozilla Firefox, Netscape Navigator, and Opera are all available for free.

What the Browser Does The core purpose of a web browser is to connect to web servers, request documents, and then properly format and display those documents. Web browsers can also display files on your local computer, download files that are not meant to be displayed. Each web page is a file written in a language called the Hypertext Markup

Language (HTML) that includes the text of the page, a description of its structure, and links to other documents, images, or other media.

Protocols: In computing, a protocol is a set of rules which is used by computers to communicate with each other across a network. A protocol is a convention or standard that controls or enables the connection, communication, and data transfer between computing endpoints.

Internet Protocol Suite: The Internet Protocol Suite is the set of communications protocols used for the Internet and other similar networks. It is commonly also known as TCP/IP named from two of the most important protocols in it: The Transmission Control Protocol (TCP) and the Internet Protocol (IP), which were the first two networking protocols defined in this standard.

Building Web sites: It's a good idea to first think about and design your site. That way, you'll give yourself direction and you'll need to reorganize less later.

To design your site:

1. Figure out why you're creating this site. What do you want to convey?
2. Think about your audience. How can you tailor your content to appeal to this audience? For example, should you add lots of graphics or is it more important that your page download quickly?
3. How many pages will you need? What sort of structure would you like it to have? Do you want visitors to go through your site in a particular direction, or do you want to make it easy for them to explore in any direction?
4. Sketch out your site on paper.

What does *Internet Architecture Board (IAB)* mean?

The Internet Architecture Board (IAB) is a board of researchers and professionals that manages the engineering and technical development related to the Internet. IAB offers assistance and insight on a wide range of Internet-related concerns. Professional entities, standards agencies and other organizations frequently use IAB as a reference for network expertise.

IAB manages several task forces, including the Internet Research Task Force (IRTF) and the Internet Engineering Task Force (IETF). IAB was originally established as the Internet Configuration Control Board (ICCB) in 1979. Adopting several names afterwards, it finally became the IAB in 1992. Initially, the U.S. government and Federal Research Internet Configuration Committee (FRICC) supported IAB.

Network Standards

Networking standards define the rules for data communications that are needed for interoperability of networking technologies and processes. Standards help in creating and maintaining open markets and allow different vendors to compete on the basis of the quality of their products while being compatible with existing market products.

During data communication, a number of standards may be used simultaneously at the different layers. The commonly used standards at each layer are −

- **Application layer** − HTTP, HTML, POP, H.323, IMAP
- **Transport layer** − TCP, SPX
- **Network layer** −IP, IPX
- **Data link layer** − Ethernet IEEE 802.3, X.25, Frame Relay
- **Physical layer** −RS-232C (cable), V.92 (modem)

Types of Standards

Standards are of two types

- **De facto** − These are the standards that are followed without any formal plan or approval by any organization. They have come into existence due to traditions or facts. For example, the HTTP had started as a de facto standard.
- **De jure** − These standards are the ones which have been adopted through legislation by any officially recognized standards organization. Most of the communication standards that are used today are de jure standards.

Standards Organizations

Some of the noted standards organizations are

- International Standards Organization (ISO)
- International Telecommunication Union (ITU)
- Institute of Electronics and Electrical Engineers (IEEE)
- American National Standards Institute (ANSI)
- Internet Research Task Force (IETF)
- Electronic Industries Association (EIA)

What is an ISP?

ISP is an acronym that stands for *Internet Service Provider*. An Internet Service Provider is a company that provides Internet access to organizations and home users.

What exactly do ISPs do?

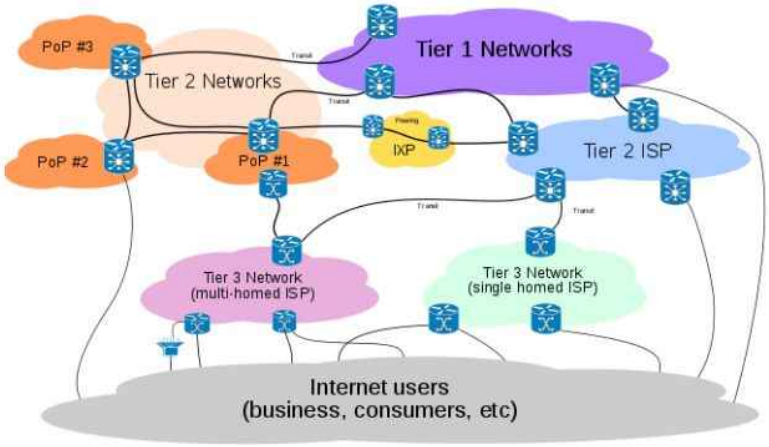
In short, an ISP provides you with Internet access, usually for a fee. Without an ISP, you wouldn't be able to shop online, access Facebook, or read this page. Connecting to the Internet requires specific telecommunications, networking, and routing equipment. ISPs allow users access to networks that contain the required equipment, enabling users to establish Internet connectivity.

ISPs are responsible for making sure you can access the Internet, routing Internet traffic, resolving domain names, and maintaining the network infrastructure that makes Internet access possible.

While the core function of an ISP is to provide Internet access, many ISPs do much more. ISPs also offer services like web hosting, domain name registration, and email services.

How do ISPs work?

At the top of the Internet access pyramid are Tier 1 Internet service providers. A Tier 1 Internet service provider is an ISP that has access to all the networks on the Internet using only network peering agreements they do not have to pay for. To help conceptualize what purpose Tier 1 ISPs serve, think of Tier 1 ISPs as the major highways of the Internet. These ISPs connect all corners of the World Wide Web. Some popular examples of Tier 1 ISPs include Vodacom, Bharti, Deutsche Telekom, British Telecommunications, and Verizon.



Tier 1 Internet service providers sell access to their networks to Tier 2 ISPs. Tier 2 ISPs then sell Internet access to organizations and home users. However, sometimes Tier 1 ISPs may sell Internet access directly to organizations and individuals. Additionally, a second intermediary ISP, referred to as a Tier 3 ISP, may purchase network bandwidth from a Tier 2 ISP before selling that bandwidth to end users.

When traffic is routed from your home network to the Internet, it goes through a number of hops before reaching its destination. For example, traffic may travel from your modem, to your Tier 3 ISP's network, to a Tier 2 ISP's network, to a Tier 1 ISP's network, then back down through a different set of ISPs before reaching the destination.

The underlying technology that ISPs use to establish connectivity can be based on analog telephone lines (dial-up), DSL, cable, satellite, Wi-Fi, fiber optics, or other connectivity mediums. The reason many cable and telephone providers are also ISPs is because their underlying infrastructure can accommodate Internet traffic.

Can I connect to the Internet without an ISP?

No, organizations and home users need an ISP to be able to access the Internet. If your ISP is down, you will not be able to access the Internet unless you have access through another ISP. Organizations that require redundant Internet connections may use a cellular service provider or secondary ISP connection to another provider for backup. A popular way for home users to work around Internet connectivity outages is to use their cell phone to continue working or as a mobile "hotspot".

Internet Architecture

The Internet is easily the largest computer system ever built, with tens of millions of nodes running hundreds of protocols. Examining its architecture is foremost about looking beyond the low-level system components and protocols and identifying the set of core functionalities that make it tick. The Internet is essentially a network for transporting digital data (i.e., bit streams) between computer processes. In the most abstract form, a network simply consists of nodes connected by links. In the Internet setting, the nodes are computers and the links are connections between computers. To help illustrate the set of necessary functionalities for providing communication services over the Internet, consider a typical computer communication scenario where process A running on one computer wants to transmit a file to process B running on another computer. For this transmission to be successful, the following functions are required:

Data Formatting: A and B must agree on a common data format so that B can extract and reassemble the content of the file from the bit streams received.

Addressing: Process A must have a means to both uniquely identify B from other processes and supply this identification, called B's address, to the network.

Routing: Methods must be in place for determining a feasible path for moving bits from A to B, based on the addresses of A and B.

Forwarding: Methods must be in place for actually moving bits from A to B, through a predetermined sequence of nodes.

Error recovery: Since no physical transmission medium is perfect and bits may be inverted or lost in transit, algorithms are required to detect and correct these errors.

Equally important is the division of work, regarding both the creation of distinct node types and the placement of the aforementioned key functions among the nodes. In one design, the network may consist of homogenous nodes, all of which implement one identical set of functions. While conceptually simple, this design may be inflexible and/or incur unnecessarily high cost. It is consideration of this kind of design trade-offs that has shaped the development of the Internet architecture.

The Internet began as an experimental network called ARPAnet, which was sponsored by the U.S. Department of Defense (DoD) initially for testing the viability of packet switched computer networks and later for demonstrating ways of combining packet networks that use different link technologies (leased phone lines, satellite, radio, etc.) into one integrated data communications infrastructure for the military. High on the requirement list for the Internet project were:
1. Robustness – Because of the military sponsorship, an emphasis is put on the ability of the network to continue to operate in the presence of link or node failures.
2. Link heterogeneity – Also important to the military is the network's ability to rapidly assimilate different link technologies so the network can be quickly deployed and extended.

Design Principles

To meet the overriding requirements of robustness and link heterogeneity, the original architects of the Internet made two important design decisions regarding how to organize the core computer networking functionalities. First they recognized that a monolithic network architecture where each switching node can cope with all link technologies will not scale. [Clark88] The concept of adding specialized packet switching nodes, called Internet Message Processors (IMPs), to the network architecture was developed to
address that problem and to take advantage of the then new store-and-forward communication paradigm.

Simplicity over efficiency: Whenever possible, trade efficiency for simplicity.
Datagram service: Provide simple connectionless packet forwarding service in the network core.
End-to-end argument: Functions placed at low levels of a system may be redundant or of little value when the cost of providing them at the low level is factored in.

All Networks are Equal:

Types of Networks in Use Today

1. Personal Area Network (PAN)

The smallest and most basic type of network, a PAN is made up of a wireless modem, a computer or two, phones, printers, tablets, etc., and revolves around one person in one building. These types of networks are typically found in small offices or residences, and are managed by one person or organization from a single device.

2. Local Area Network (LAN)

- We're confident that you've heard of these types of networks before – LANs are the most frequently discussed networks, one of the most common, one of the most original and one of the simplest types of networks. LANs connect groups of computers and low-voltage devices together across short distances (within a building or between a group of two or three buildings in close proximity to each other) to share information and resources. Enterprises typically manage and maintain LANs.
- Using routers, LANs can connect to wide area networks (WANs, explained below) to rapidly and safely transfer data.

3. Wireless Local Area Network (WLAN)

Functioning like a LAN, WLANs make use of wireless network technology, such as Wi-Fi. Typically seen in the same types of applications as LANs, these types of networks don't require that devices rely on physical cables to connect to the network.

INTERNET FUNDAMENTALS

- **The Motivation For Internetworking:** The technology, called internetworking, accommodates multiple, diverse underly- ing hardware technologies by providing a way to interconnect heterogeneous networks and a set of communication conventions that makes them interoperate. The internet technology hides the details of network hardware, and permits computers to communi- cate independent of their physical network connections. It is called open because anyone can build the software needed to communicate across an internet. The entire technology has been designed to foster communication among machines with diverse hardware architectures, to use almost any packet switched network hardware, to accommodate a wide variety of applications, and to accommodate multiple computer operating systems.

- **The lnternet Architecture Board(IAB):** The ARPA technology includes a set of network standards that specify the details of how computers communicate, as well as a set of conventions for interconnecting networks and routing traffic,referred to as TCP/IP , can be used to communicate across any set of interconnected networks.The IAB provides the focus and coordina- tion for much of the research and development underlying the TCP/IP protocols, and guides the evolution of the Internet. It decides which protocols are a required part of the TCP/IP suite and sets official policies.

- **lnternet Protocols And Standardization:**

- *Use existing protocol standards whenever such standards apply; invent new protocols only when existing standards are insufficient, and be prepared to use new standards when they become available and provide equivalent functionalities.*

- **Properties Of The Internet:** Internet must enable us to send data across intermediate networks even though they are not directly connected to the source or destination computers. All the computers in the internet must share a universal set of machine identifiers. Our notion of a unified internet also includes the idea of network independence in the user interface. That is, the set of operations used to establish communication or to transfer data must remain independent of the underlying network technologies and the destination computer. Certainly, a user should not have to understand the network interconnection topology when creating or using application programs to communicate.
- **Internet Architecture**: To have a viable internet, we need special computers that can transfer packets from one network to another. Computers that interconnect two networks and pass packets from one to the other are called internet gateways or internet routers.
- **Interconnection Through IP Router:** In a TCP/IP internet, special computers called IP routers or IP gate- ways provide interconnections among physical networks.
- Routers use the **destination network**, not the destination computer, when forwarding a packet.

Internet service provider (ISP):

An Internet service provider (ISP) is an organization that provides services for accessing or using the Internet. The ISP is equipped with all tools and technologies to provide access to the Internet services. The connection to ISP can be made over a telephone line, leased line or wireless/radio link connections.

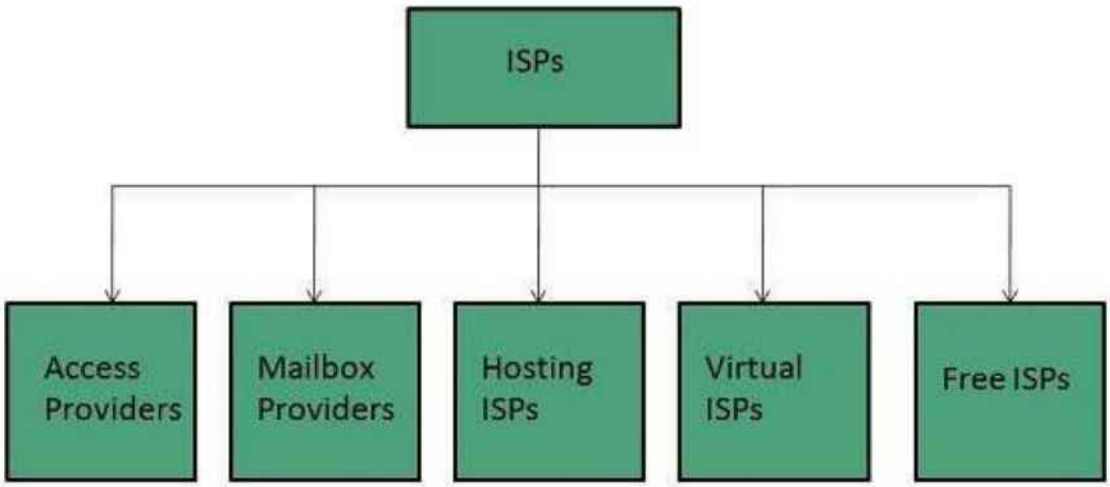
They offer various services:

- Internet Access

- Domain name registration
- Dial-up access
- Leased line access

ISP Types

ISPs can broadly be classified into five categories as shown in the following diagram:



Access providers

They provide access to internet through telephone lines, cable wi-fi or fiber optics.

Mailbox Provider

Such providers offer mailbox hosting services.

Hosting ISPs

Hosting ISPs offers e-mail, and other web hosting services such as virtual machines, clouds etc.

Virtual ISPs

Such ISPs offer internet access via other ISP services.

Free ISPs

Free ISPs do not charge for internet services.

Factors for choosing the right ISP :

1.Type of connection:

There are two primary types of internet service. Standard (also known as High Speed or broadband service) and High Availability service. The first group includes options like cable and DSL and typically offer higher speeds, but lower quality and reliability of service. This option is significantly cheaper in most cases. The High Availability class of services provides a Service Level Agreement for uptime that usually exceeds 99.99% or approximately 2 hours of downtime per year. These connections include Fiber Optic connections .According to our requirement of dependency on an internet connection, we can choose the Service.

2.**Speed:**

It is very important to ensure that we have enough speed for everyday use, including peak times (such as large meetings or training evolutions). when choosing an ISP, it's necessary to ensure they can provide the speeds we need. Based on our location, and the type of Internet access we are looking for (i.e. Fiber vs Fixed Wireless vs DSL etc.) bandwidth availability may fluctuate from carrier to carrier.

3.Availability:

While it would be ideal to have access to High Availability Fiber Optic, not every business has this option. Even the availability of

cable and DSL internet can be limited in new construction areas where lines of service have yet to be established. In these cases, it may take up to 6 months for construction and installation of service.

4.Redundancy:

In cases when the need to be continually connected is extreme and you cannot afford even one or two minutes of downtime, you want to secure some level of redundancy. Redundancy is a fail over internet connection that switches in the event your main line has gone down. This is more common with standard internet service, but can be important to have even with High Availability services with SLAs if a business cannot afford a moment of downtime.

5.Cost:

High Availability service comes with a high price tag and depending on the types of connections and service area – the price can be significant. As an example, a client that was quoted one High Availability fiber connection for $1000 per month for 20MB of service. Conversely, two diverse broadband connections (one cable at 100MB and one Broadband Fiber at 50MB) came in at only $400 per month – more speed, better availability for less cost. The service costs from your ISP will vary widely depending upon a number of factors.

6. Customer Support:

While in an ideal world , businesses would never have to engage with their ISP past service installation, Whether a client has billing questions, service issues, needs technical support, has upgrade inquiries or product add-ons, at some point or another, chances are a business will have to engage with an ISP's customer support team. Therefore, research what type of support the company offers.

A larger carrier, for example, might make you sit through an automated phone menu, place you on a lengthy hold, and eventually transfer you to a contracted employee outside of the U.S. Alternatively, a medium sized ISP, such as GeoLinks, offers 24/7 in-house customer support; customers are even able to ask for customer support reps by name.

Another element to consider is overall responsiveness. If your business does experience a technical issue, how long does it take a provider to respond and address the issue? Time is money, so whether it be hours wasted on hold, or weeks waiting on a repair, how an ISP handles customer relations directly affects its business customer's bottom line.

7. Agility and Flexibility

As a business grows and changes, its overall telecom needs will change as well. For example, if a law firm hires 10 more associates, they will likely need to upgrade their overall

bandwidth. Furthermore, if juggling multiple carriers and multiple bills becomes too large of a strain on a company's accounting apartment, a business may wish to streamline all their telecom needs with a single carrier.

Some ISPs offer additional services such as VoIP and SD-WAN, while others do not. Therefore, when selecting an ISP, make sure to explore their entire product suite and offerings. Choosing an aggregator, (an ISP that is capable of reselling multiple ISP products and services) such as GeoLinks, ensures that no matter the growth or changes in a business, a single provider will be able to upgrade and adapt to evolving business needs.

8. **Personal disk space:**

ISP s also provide desk space to your PC for the purpose of keeping your mails. The total size of your mail should not exceed the space reserved.

9. **Application Services Offered:**

Also enquire about the various application services like email, FTP,news groups and online chat services offered by the ISPs .

TYPES OF CONNECTIVITY

There exist several ways to connect to the internet. Following are these connection types available:

1. Dial-up Connection

2. ISDN
3. DSL
4. Cable TV Internet connections
5. Satellite Internet connections
6. Wireless Internet Connections

Dial-up Connection

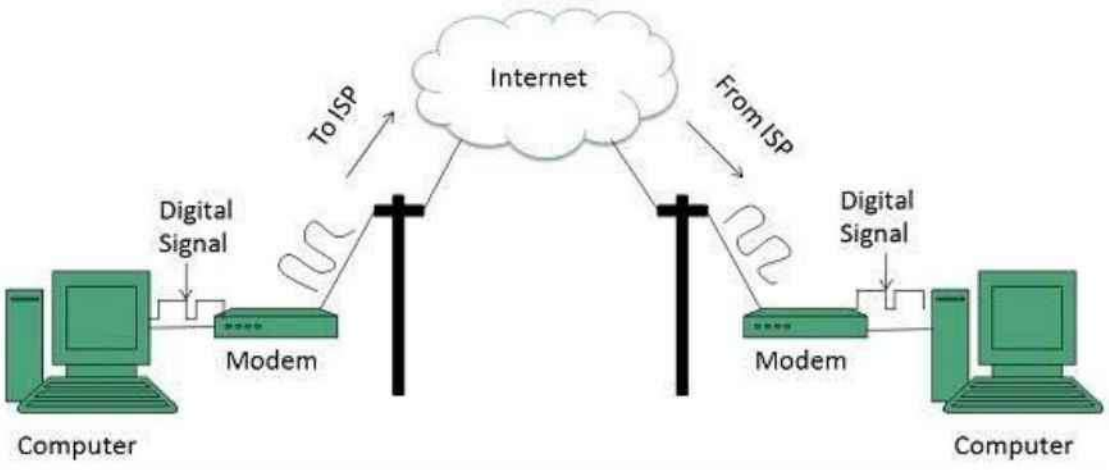
Dial-up connection uses telephone line to connect PC to the internet. It requires a modem to setup dial-up connection. This modem works as an interface between PC and the telephone line.

There is also a communication program that instructs the modem to make a call to specific number provided by an ISP.

Dial-up connection uses either of the following protocols:

1. Serial Line Internet Protocol (SLIP)
2. Point to Point Protocol (PPP)

The following diagram shows the accessing internet using modem:



ISDN

ISDN is acronym of **Integrated Services Digital Network.** It establishes the connection using the phone lines which carry digital signals instead of analog signals.

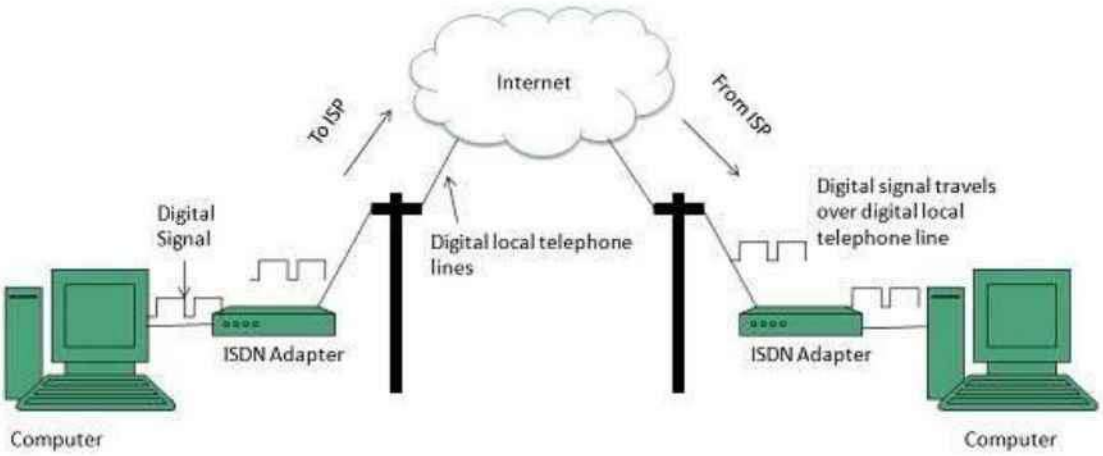
There are two techniques to deliver ISDN services:

1. Basic Rate Interface (BRI)
2. Primary Rate Interface (PRI)

Key points:

- The BRI ISDN consists of three distinct channels on a single ISDN line: t1o 64kbps B (Bearer) channel and one 16kbps D (Delta or Data) channels.
- The PRI ISDN consists of 23 B channels and one D channels with both have operating capacity of 64kbps individually making a total transmission rate of 1.54Mbps.

The following diagram shows accessing internet using ISDN connection:



DSL

DSL is acronym of **Digital Subscriber Line.** It is a form of broadband connection as it provides connection over ordinary telephone lines.

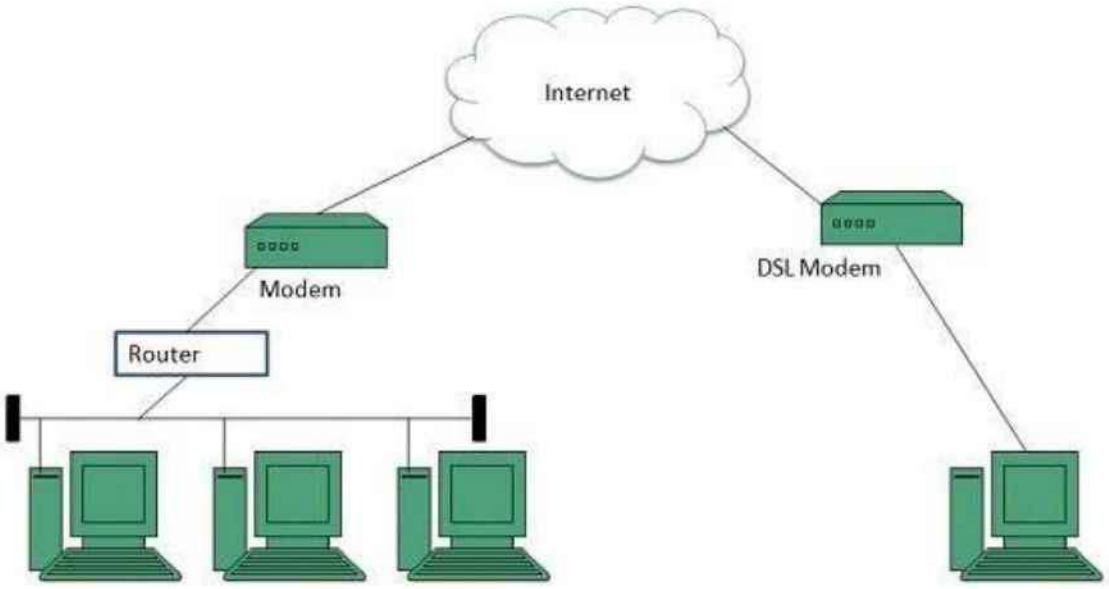
Following are the several versions of DSL technique available today:

1. Asymmetric DSL (ADSL)
2. Symmetric DSL (SDSL)
3. High bit-rate DSL (HDSL)

4. Rate adaptive DSL (RDSL)
5. Very high bit-rate DSL (VDSL)
6. ISDN DSL (IDSL)

All of the above mentioned technologies differ in their upload and download speed, bit transfer rate and level of service.

The following diagram shows that how we can connect to internet using DSL technology:



Internet Access Using DSL Modem

Cable TV Internet Connection

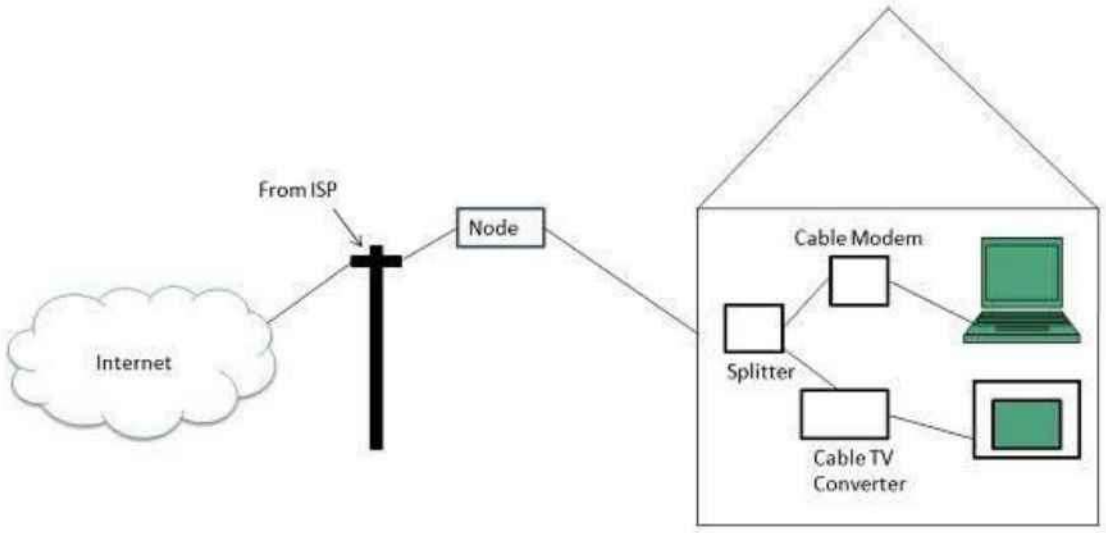
Cable TV Internet connection is provided through Cable TV lines. It uses coaxial cable which is capable of transferring data at much higher speed than common telephone line.

Key Points:

- A cable modem is used to access this service, provided by the cable operator.
- The Cable modem comprises of two connections: one for internet service and other for Cable TV signals.
- Since Cable TV internet connections share a set amount of bandwidth with a group of customers, therefore, data transfer rate

also depends on number of customers using the internet at the same time.

The following diagram shows that how internet is accessed using Cable TV connection:



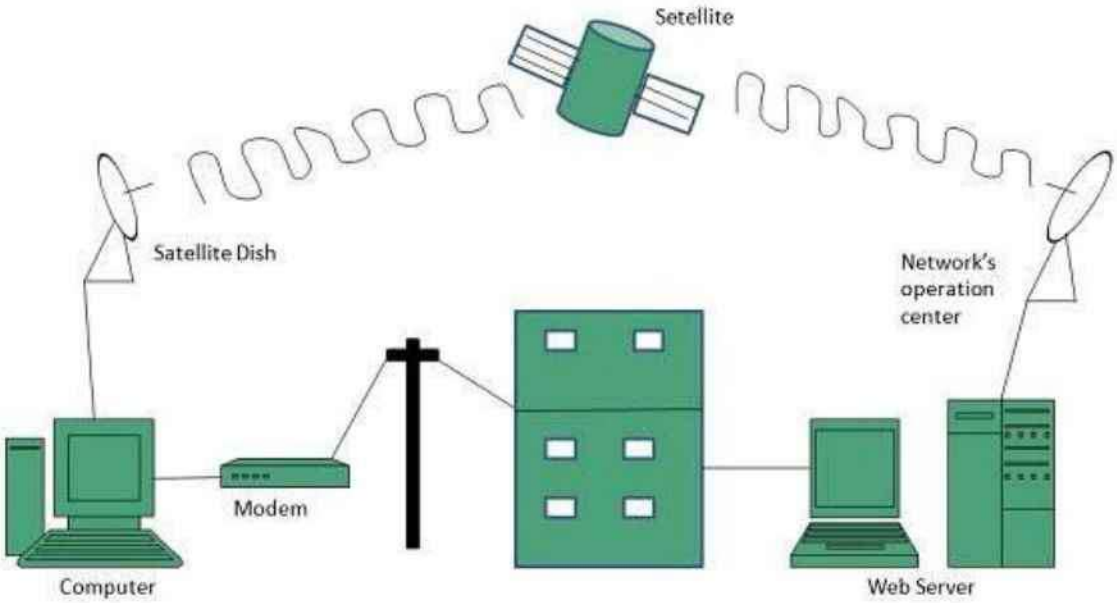
Satellite Internet Connection

Satellite Internet connection offers high speed connection to the internet. There are two types of satellite internet connection: one way connection or two way connection.

In one way connection, we can only download data but if we want to upload, we need a dialup access through ISP over telephone line.

In two way connection, we can download and upload the data by the satellite. It does not require any dialup connection.

The following diagram shows how internet is accessed using satellite internet connection:



VSAT:

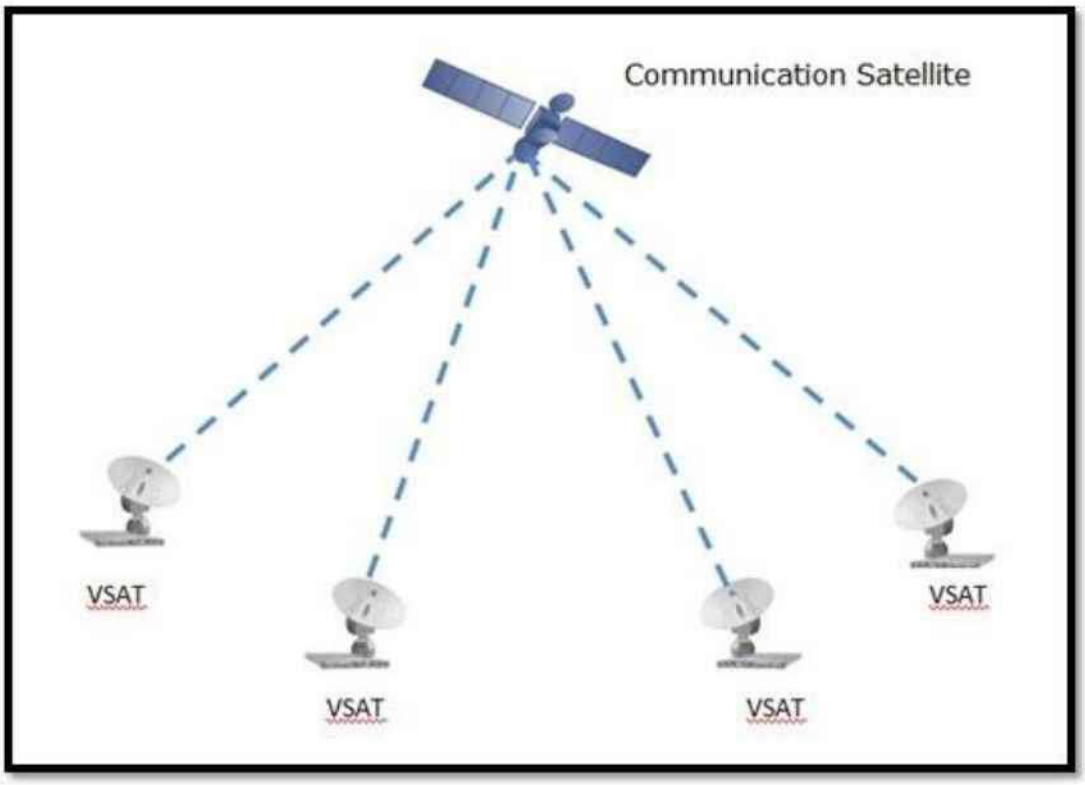
VSATs (Very Small Aperture Terminals) is a two way, lost cost, ground micro station for transmitting data to and from communication satellites. A VSAT has a dish antenna with diameters between 75 cm to 1 m, which is very small in comparison with 10 m diameter of a standard GEO antenna. It accesses satellites in geosynchronous orbits or geostationary orbits. Data rates in VSATs ranges from 4 Kbps to 16 Mbps.

VSAT (Very Small Aperture Terminal) is a satellite communications system that serves home and business users. A **VSAT** end user needs a box that interfaces between the user's computer and an outside antenna with a transceiver. The tranceiver receives or sends a signal to a satellite transponder in the sky.

VSATs access satellites in geosynchronous orbit or geostationary orbit to relay data from small remote Earth stations (terminals) to other terminals (in mesh topology) or master Earth station "hubs" (in star topology).

VSATs are used to transmit narrowband data (e.g., point-of-sale transactions using credit cards, polling or RFID data, or SCADA), or broadband data (for the provision of satellite

Internet access to remote locations, VoIP or video). VSATs are also used for transportable, on-the-move (utilising phased array antennas) or mobile maritime communications.



Configurations of VSATs

- **Star Topology** − This has a central uplink site which transmits data from and to each VSAT through the satellite.
- **Mesh Topology** − Each VSAT transmits data via the satellite to the other stations.
- Some VSAT networks are configured by having several centralized uplink sites (and VSAT stemming from it) connected in a multi-star topology with each star (and each terminal in each star) connected to each other in a mesh topology. Others configured in only a single-star topology sometimes will have each terminal connected to each other as well, resulting in each terminal acting as a central hub. These configurations are utilized to minimize the overall cost of the network, and to alleviate the amount of data that has to be relayed through a central uplink site (or sites) of a star or multi-star network.

Uses of VSATs

1. In narrowband data − e.g. point – of – sale transactions using debit cards or credit cards, RFID data.
2. In broadband data − for the provision of satellite Internet access to remote locations, VoIP or video). VSATs are also used for transportable, on-the-move (utilising phased array antennas) or mobile maritime communications.

Wireless Internet Connection

Wireless Internet Connection makes use of radio frequency bands to connect to the internet and offers a very high speed. The wireless internet connection can be obtained by either WiFi or Bluetooth.

Key Points:

- Wi Fi wireless technology is based on IEEE 802.11 standards which allow the electronic device to connect to the internet.
- Bluetooth wireless technology makes use of short-wavelength radio waves and helps to create personal area network (PAN).

TCP/IP

Network engineering is a complicated task, which involves software, firmware, chip level engineering, hardware, and electric pulses. To ease network engineering, the whole networking concept is divided into multiple layers. Each layer is involved in some particular task and is independent of all other layers. But as a whole, almost all networking tasks depend on all of these layers. Layers share data between them and they depend on each other only to take input and send output.

Layered Tasks

- The total task is divided in to small tasks. Each small task is then assigned to a particular layer which works dedicatedly to process the task only. Every layer does only specific work.
- In layered communication system, one layer of a host deals with the task done by or to be done by its peer layer at the same level on the remote host. The task is either initiated by layer at the lowest level or at the top most level. If the task is initiated by the topmost layer, it is passed on to the layer below it for further processing. The lower layer does the same thing, it processes the task and passes on to layer below it.
- Every layer clubs together all procedures, protocols, and methods which it requires to execute its piece of task. All layers identify their counterparts by means of encapsulation header and tail. Each layer gets service from the layer below it and provides services to the layer above it through interfaces.
- **Internet Model:** Internet uses TCP/IP protocol suite, also known as Internet suite. This defines Internet Model which contains four layered architecture. OSI Model is general communication model but Internet Model is what the internet

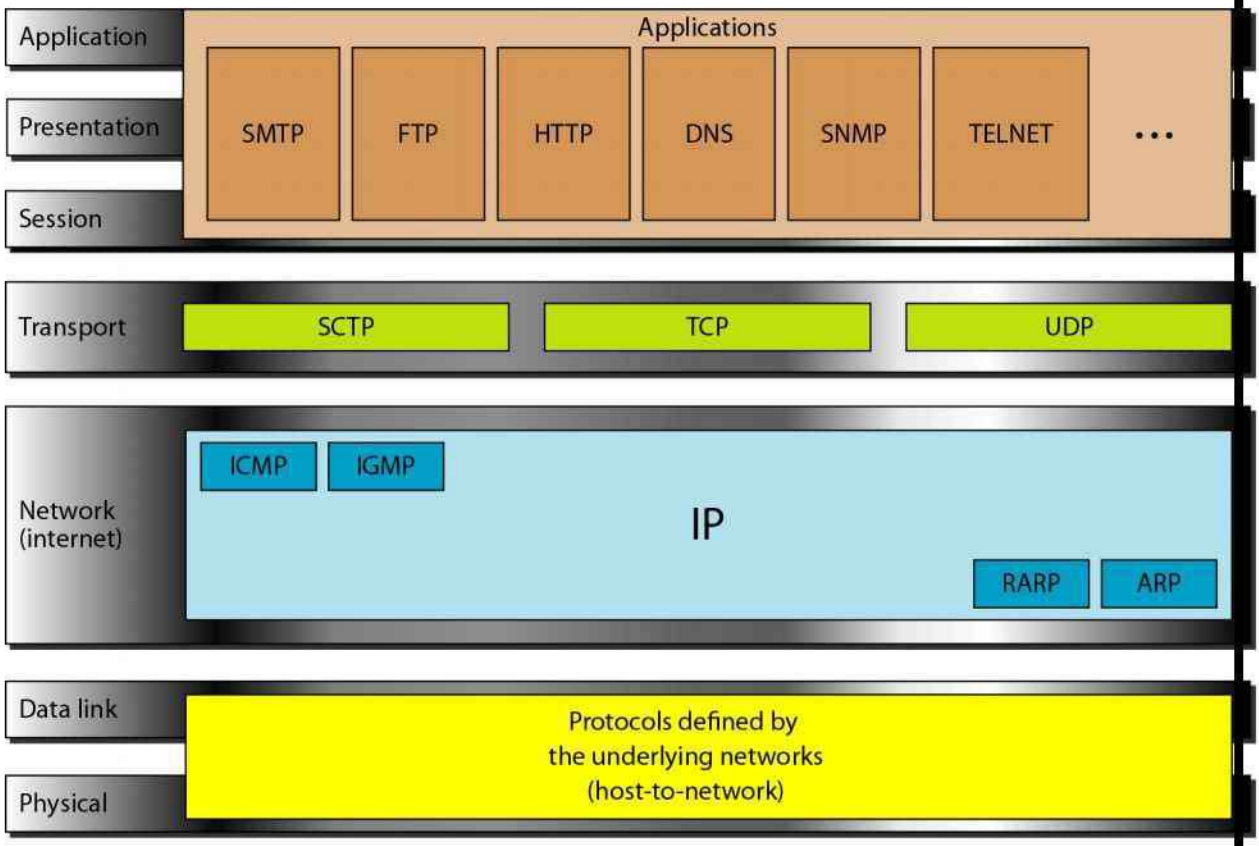
uses for all its communication. The internet is independent of its underlying network architecture so is its Model. This model has the following layers:

- **Application Layer:** This layer defines the protocol which enables user to interact with the network. 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, remote file access and transfer, shared database management, and other types of distributed information services. It is responsible for providing various services to the users like : Mail services, File transfer and access, Remote log in and accessing the World Wide Web.
- **Transport Layer:** This layer defines how data should flow between hosts. Major protocol at this layer is Transmission Control Protocol (TCP) and User Datagram Protocol(UDP). The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. The various responsibilities of transport layer are port addressing, segmentation and reassembly, connection control, flow control and error control .
- **Internet Layer:** Internet Protocol (IP) works on this layer. This layer facilitates host addressing and recognition. This layer defines routing and is responsible for end- to-end delivery. The network layer is responsible for the source-to-destination

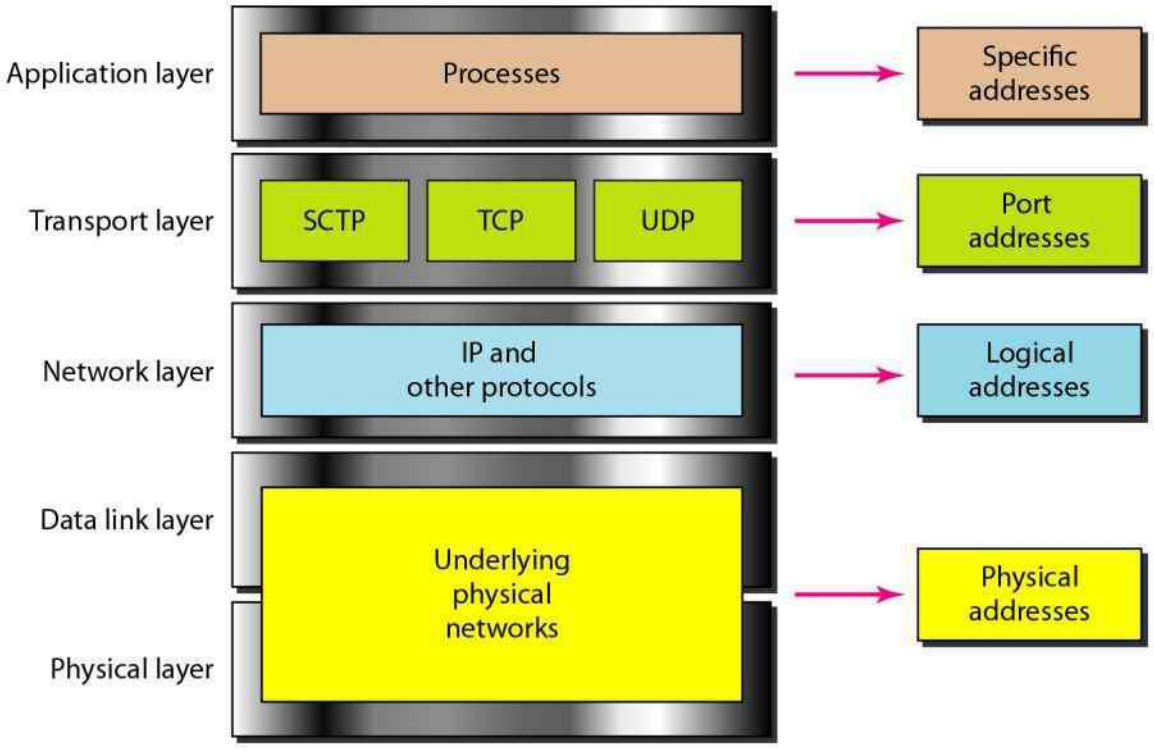
delivery of a packet, possibly across multiple networks (links). Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination. The various responsibilities of Internet layer are Logical Addressing and Routing.

- **Link Layer:** This layer provides mechanism of sending and receiving actual data. Unlike its OSI Model counterpart, this layer is independent of underlying network architecture and hardware. At the physical and data link layers, TCP/IP does not define any specific protocol. It supports all the standard and proprietary protocols.The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium. It also defines the procedures and functions that physical devices and interfaces have to perform for transmission to Occur. The physical layer is responsible for movements of individual bits from one hop (node) to the next. The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer (network layer). The datalink layer is responsible for moving frames from one hop (node) to the next. The various responsibilities of Datalink layer are : Framing, Physical addressing, Flow control, Error control and access control. The various responsibilities of physical layer are : Defines physical characteristics of interfaces and media, Representation of bits, Data rate and Synchronization of bits.

TCP/IP versus OSI:



RELATIONSHIP OF LAYERS AND ADDRESSES IN TCP/IP



Physical Addresses

The physical address, also known as the link address, is the address ofa node as defined by its LAN or WAN. It is included in the frame used by the data link layer. It is the lowest-level address.

Logical Addresses

Logical addresses are necessary for universal communications that are independent of underlying physical networks. A logical address in the Internet is currently a 32-bit address that can uniquely define a host connected to the Internet. No two publicly addressed and visible hosts on the Internet can have the same IP address.

Port Addresses

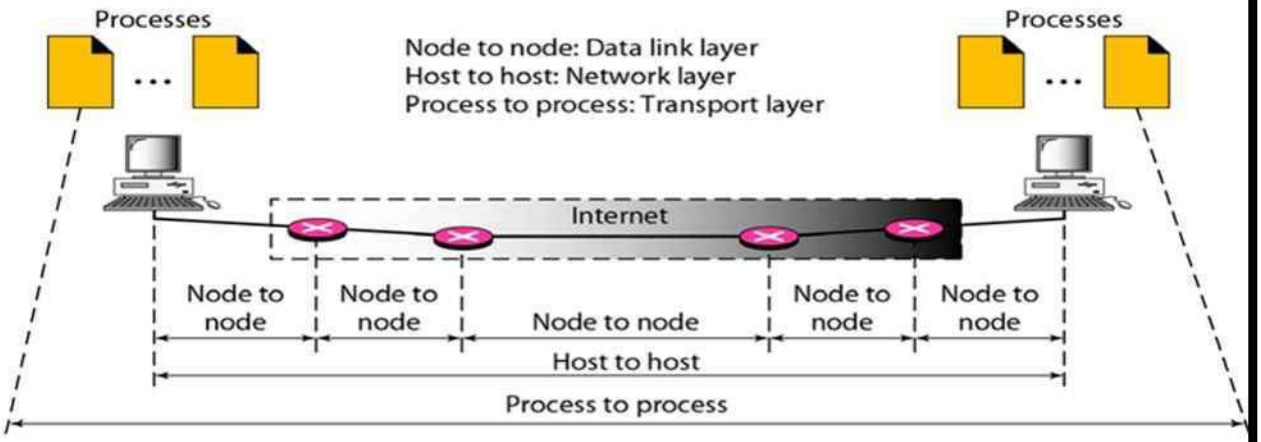
computers are devices that can run multiple processes at the same time. The end objective of Internet communication is a process communicating with another process which is made possible using port addresses. Process-to-process delivery needs two identifiers, IP address and the port number, at each end to make a connection. The combination of an IP address and a port number is called a socket address. The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely.

Specific Addresses

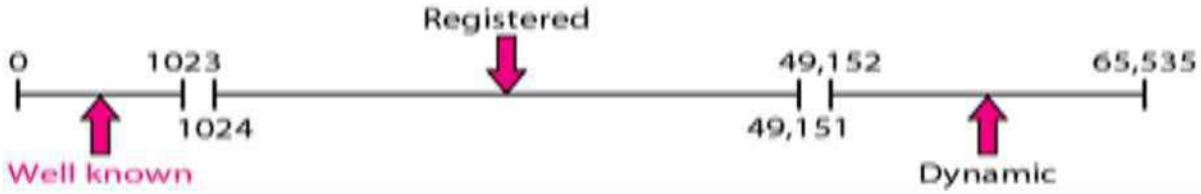
Some applications have user-friendly addresses that are designed for that specific address. Examples include the e-mail address and the Universal Resource Locator (URL).

TRANSPORT LAYER

- The transport layer is responsible for process-to- process delivery—the delivery of a packet, part of a message, from one process to another. Two processes communicate in a client/server relationship.



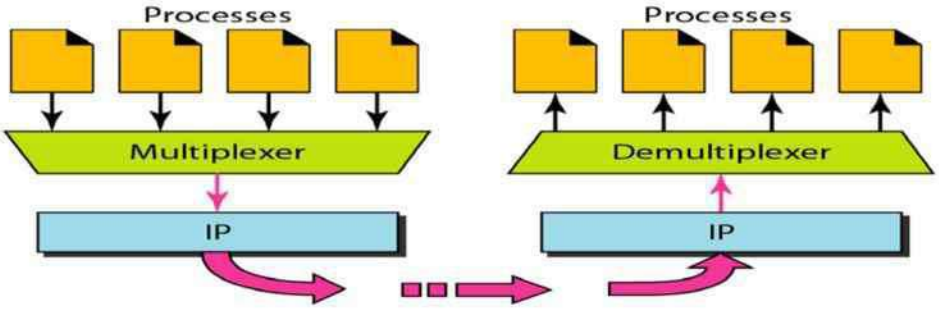
A process on the local host, called a client, needs services from a process usually on the remote host, called a server. Both processes (client and server) have the same name. For example, to get the day and time from a remote machine, we need a Daytime client process running on the local host and a Daytime server process running on a remote machine.



- **Addressing**
- TCP communication between two remote hosts is done by means of port numbers . Internet Assigned Number Authority(IANA) has divided the p orts numbers ranging from 0 – 65535 as follows:
- Well Known Ports (0 – 1023)
- Registered Ports (1024 – 49151)
- Ephermal/Dynamic Ports (49152 – 65535)

Multiplexing and Demultiplexing

The addressing mechanism allows multiplexing and demultiplexing by the transport layer.



Multiplexing

At the sender site, there may be several processes that need to send packets. However, there is only one transport layer protocol at any time. This is a many-to-one relationship and requires multiplexing. The protocol accepts messages from different processes, differentiated by their assigned port numbers. After adding the header, the transport layer passes the packet to the network layer.

Demultiplexing

At the receiver site, the relationship is one-to-many and requires de multiplexing. The transport layer receives datagrams from the network layer. After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number.

Connectionless Versus Connection-Oriented Service

A transport layer protocol can either be connectionless or connection-oriented.

Connectionless Service

- Because malicious use of ActiveX controls became such a widespread problem, Microsoft designed Internet Explorer 7 to display a warning every time a site attempts to use an ActiveX control. It's up to the user to decide whether or not the request comes from a trustworthy source. Today, ActiveX controls are far less common than they once were because so many browsers either disable ActiveX controls by default or do not support them at all.

Types of Active X Controls:

Different types of Active X controls offered by Visual Basic are:

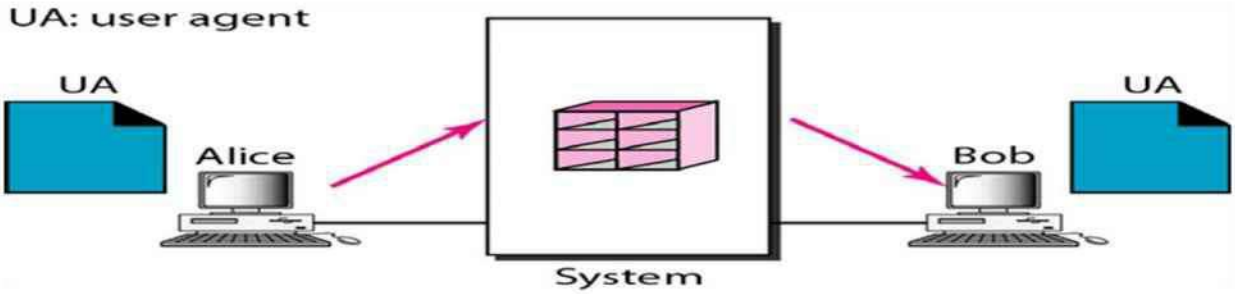
1. **User Drawn Controls**: These controls are developed from scratch for which user is responsible for designing the control's interface.

2. **Enhancement to existing VB controls**: Some controls can be developed based on existing controls and enhance their operation.

3. **New controls built with constituent controls**: A new control can be designed includes the existing controls totally.

ELECTRONIC MAIL

- One of the most popular Internet services is electronic mail(e-mail).
- **ARCHITECTURE:**
- Its architecture consists of several components. The general architecture of an e-mail system includes the three main components: user agent, message transfer agent, and message access agent.
- **FIRST SCENARIO:** Here the sender and the receiver of the e-mail are users (or application programs) on the same system; they are directly connected to a shared system, only two user agents are required. The administrator has created one mailbox for each user where the received messages are stored. A mail box is part of a local hard drive, a special file with permission restrictions. Only the owner of the mailbox has access to it.

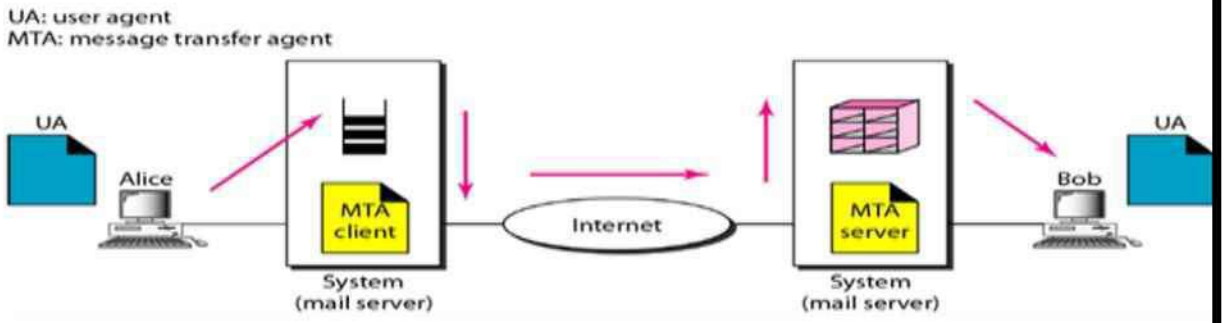


In the above example, Alice runs a user agent (UA) program to prepare the message and store it in Bob's mailbox. The message has the sender and recipient mailbox addresses (names offiles). Bob can retrieve and read the contents of his mailbox at his convenience, using a user agent.

When the sender and the receiver of an e-mail are on the same system, they need only two user agents.

SECOND SCENARIO IN ELECTRONIC MAIL

- In our example, Alice needs to use a user agent program to send her message to the system at her own site. The system (sometimes called the mail server) at her site uses a queue to store messages waiting to be sent. Bob also needs a user agent program to retrieve messages stored in the mailbox of the system at his site. The message, however, needs to be sent through the Internet from Alice's site to Bob's site. Here two message transfer agents are needed: one client and one server.



  - Here the sender and the receiver of the e-mail are users (or application programs) on two different systems.The message needs to be sent over the Internet.
  - **Here two user agents (UAs) and two message transfer agents (MTAs)(client and server) are needed.**

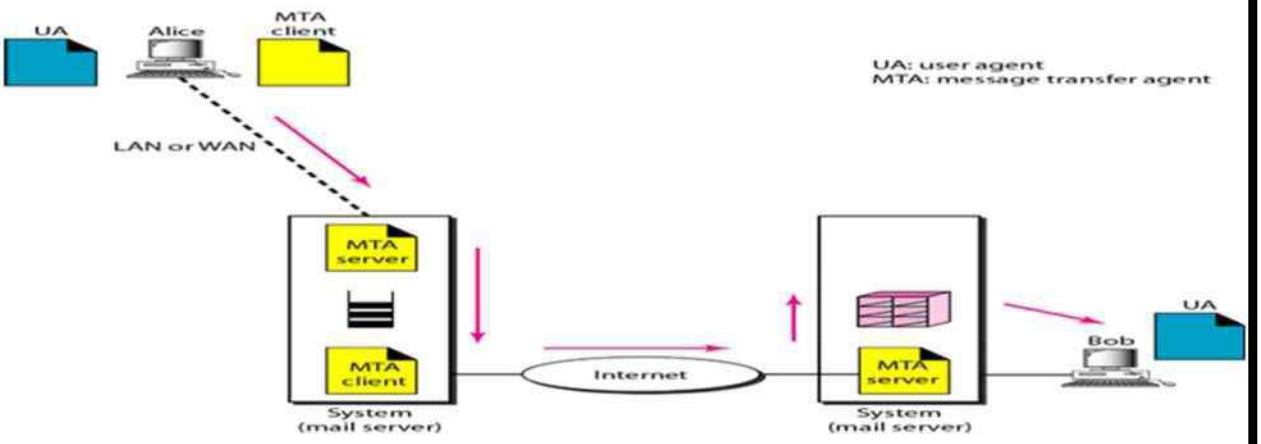
-

THIRD SCENARIO IN ELECTRONIC MAIL

- When the source is connected to the LAN or WAN, it needs a user agent to prepare its message, then the message is to be sent through the LAN or WAN. This can be done through a pair of message transfer agents (client and server). The source first calls the user agent which, in turn, calls the MTA client. The MTA client establishes a connection with the MTA server on the system, which is running all the time. The system at source's site queues all messages received. It then uses an MTA client to

send the messages to the system at Destination's site; the system receives the message and stores it in destination's mailbox.

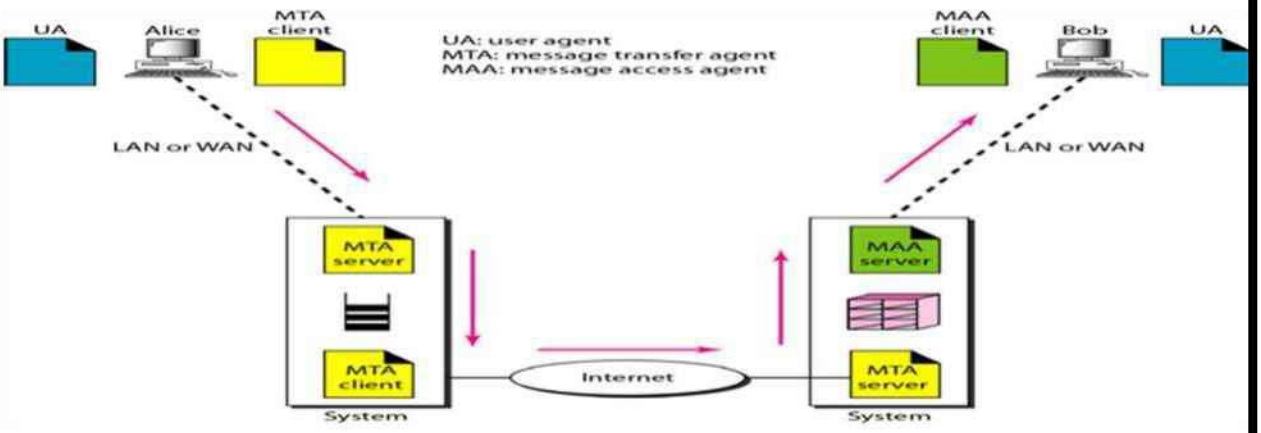
- At his convenience, the destined user uses his user agent to retrieve the message and reads it. Here we need two pairs of MTA client/server programs and two UAs.
- **When the sender is connected to the mail server via a LAN or a WAN, we need two UAs and two pairs of MTAs (client and server).**



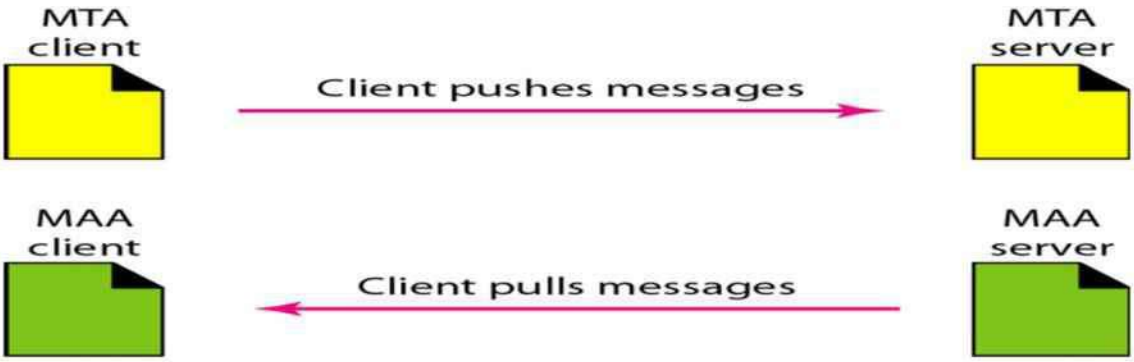
FOURTH SCENARIO IN ELECTRONIC MAIL

- In this case ,the destination is also connected to his mail server by a WAN or a LAN. After the message has arrived at destination's mail server, he needs to retrieve it. Here, we need another set of client/server agents, which we call message access agents (MAAs). Destination uses an MAA client to retrieve his messages. The client sends a request to the MAA server, which is running all the time, and requests the transfer of the messages.
- The destined user cannot bypass the mail server and use the MTA server directly. So he needs another pair of client/server programs: message access programs. This is so because an MTA client/server program is a push program: the client pushes the message to the server. Where as the destined user needs a pull program. The client needs to pull the message from the server.

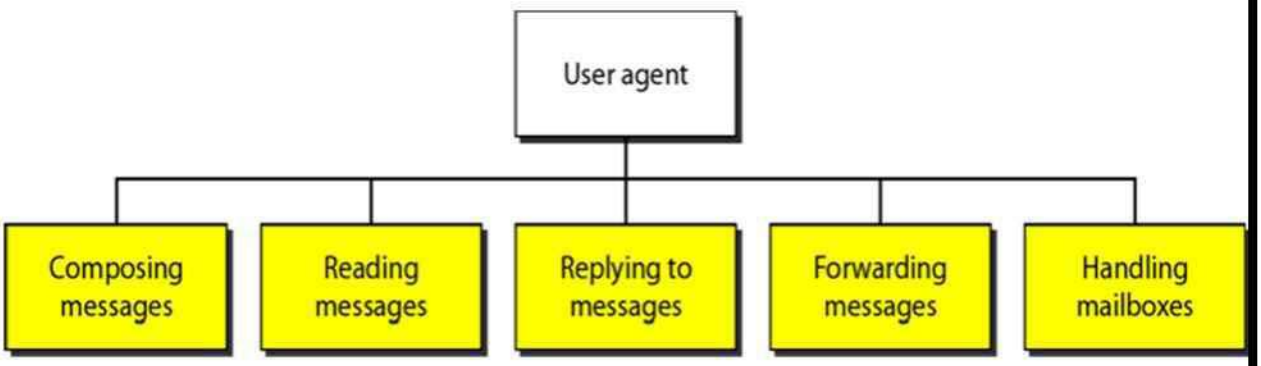
- **When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs and a pair of MAAs.**
- This is the most common situation today.



PUSH VERSUS PULL IN ELECTRONIC EMAIL



SERVICES OF USER AGENT:



Composing Messages : A user agent helps the user compose the e-mail message to be sent out. Most user agents provide a template on the screen to be filled in by the user. Some even have a built-in editor that can do spell checking, grammar checking, and other tasks expected from a sophisticated word processor.

Reading Messages : The second duty of the user agent is to read the incoming messages. When a user invokes a user agent, it first checks the mail in the incoming mailbox. Most user agents show a one-line summary of each received mail.

Replying to Messages: After reading a message, a user can use the user agent to reply to a message. A user agent usually allows the user to reply to the original sender or to reply to all recipients of the message.

Forwarding Messages : Replying is defined as sending a message to the sender or recipients ofthe copy. Forwarding is defined as sending the message to a third party. A user agent allows the receiver to forward the message.

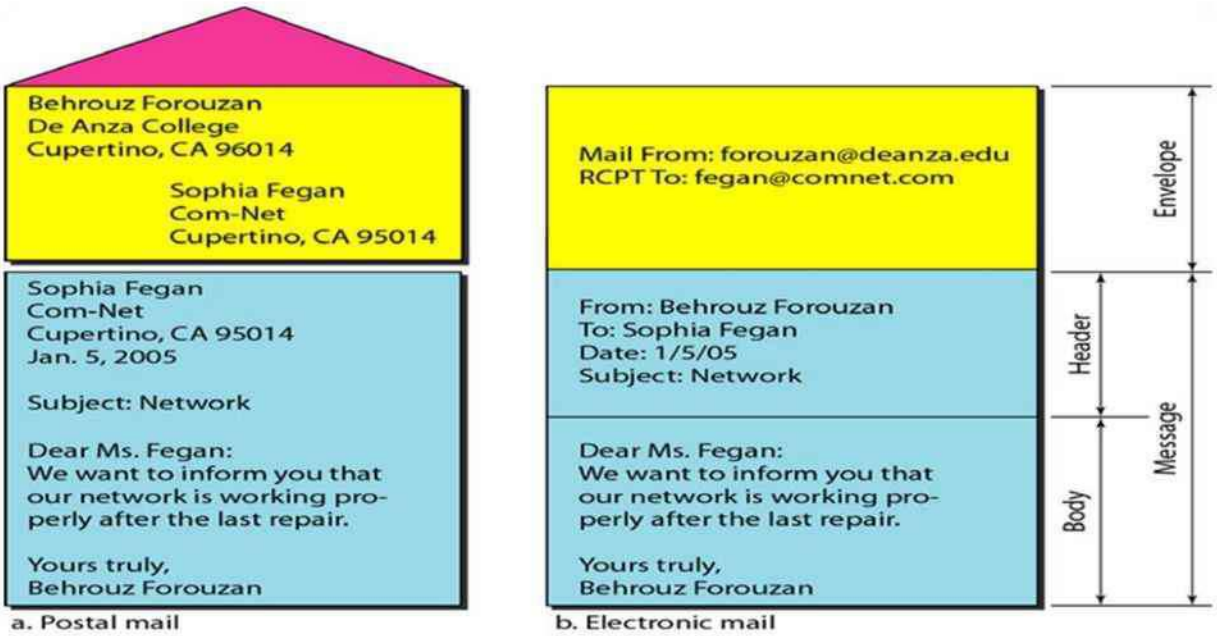
Handling Mailboxes : A user agent normally creates two mailboxes: an inbox and an outbox. Each box is a file with a special format that canbe handledby the user agent. The inbox keeps all the received e-mails until they are deleted by the user. The outbox keeps all the sent e-mails until the user deletes them. Most user agents today are capable of creating customized mailboxes.

User Agent Types Two types of user agents: command-driven and GUI-based.

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 user agent contain graphical-user interface (GUI) components that 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 ofGUI-based user agents are Eudora, Microsoft's Outlook, and Netscape.

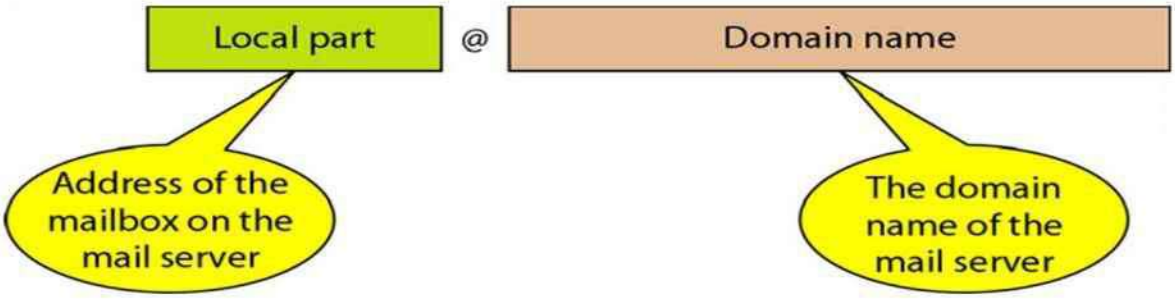
FORMAT OF AN E-MAIL



- **Sending Mail** :To send mail, the user, through the UA, creates mail that looks very similar to postal mail. It has an envelope and a message. The envelope usually contains the sender and the receiver addresses. The message contains the header and the body. The header of the message defines the sender, the receiver, the subject of the message while the body of the message contains the actual information to be read by the recipient.
- **Receiving Mail :** The user agent is triggered by the user (or a timer). If a user has mail, the UA informs the user with a notice. A list is displayed in which each line contains a summary of the

information about a particular message in the mailbox. The summary usually includes the sender mail address, the subject, and the time the mail was sent or received. The user can select any of the messages and display its contents on the screen.

E-MAIL ADDRESS



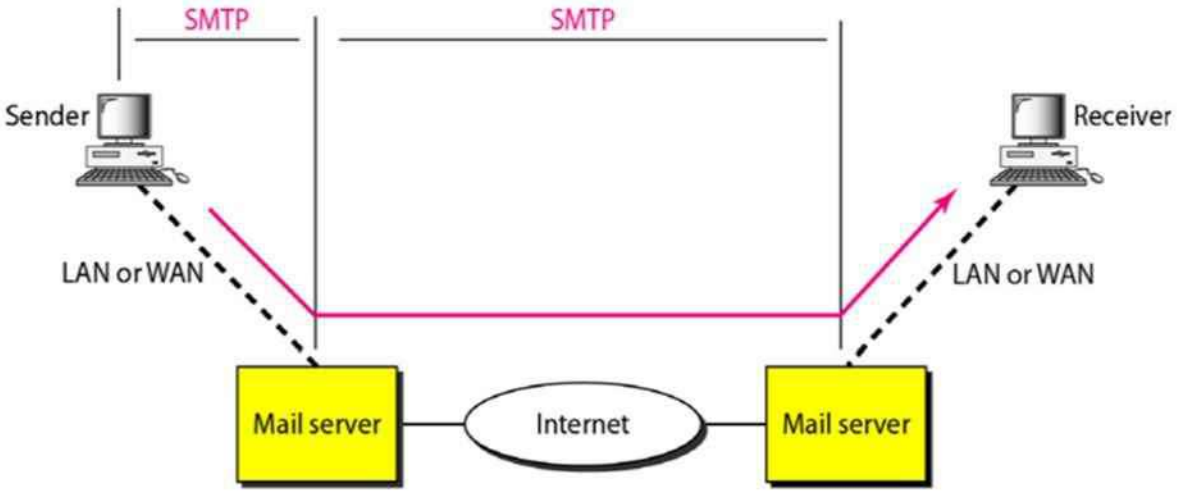
Addresses : To deliver mail, a mail handling system must use an addressing system with unique addresses. In the Internet, the address consists of two parts: a local part and a domain name, separated by an @ sign.

Local Part : The local part defines the name of a special file, called the user mailbox, where all the mail received for a user is stored for retrieval by the message access agent.

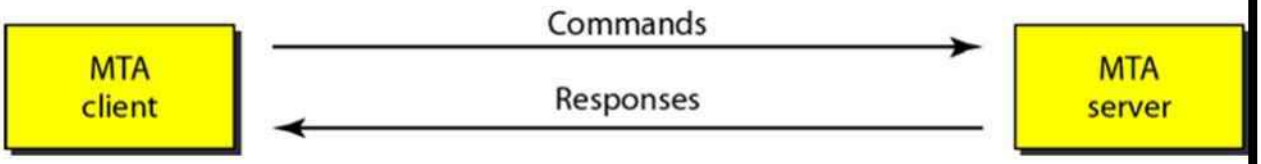
Domain Name : The second part of the address is the domain name. The domain name assigned to each mail exchanger either comes from the DNS database or is a logical name .

SMTP

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



SMTP simply defines how commands and responses must be sent back and forth. SMTP uses commands and responses to transfer messages between an MTA client and an MTA server. Each command or reply is terminated by a two-character (carriage return and line feed) end-of-line token. Commands are sent from the client to the server. SMTP defines 14 commands.



COMMANDS:

Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPT TO	Intended recipient of the message
DATA	Body of the mail
QUIT	
RSET	
VRFY	Name of recipient to be verified
NOOP	
TURN	
EXPN	Mailing list to be expanded
HELP	Command name

- **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.

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
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

POP3 and IMAP4

- **Post Office Protocol, version 3 (POP3) :**

The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server. Mail access starts with the client when the user needs to download e-mail from the mailbox on the mail server. The client opens a connection to the server on TCP port 110. It then sends its user name and password to access the mailbox. The user can then list and

retrieve the mail messages, one by one. POP3 has two modes: the delete mode and the keep mode. In the delete mode, the mail is deleted from the mailbox after each retrieval. In the keep mode, the mail remains in the mailbox after retrieval.

- **Internet Mail Access Protocol, version 4 (IMAP4):** IMAP4 is similar to POP3, but it provides the following extra functions:

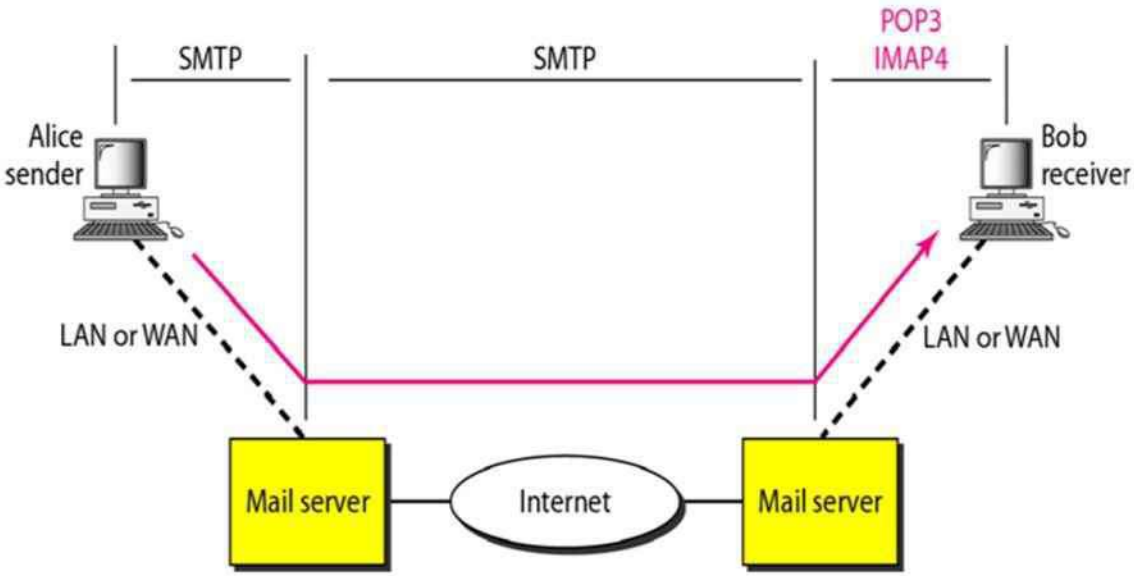
➢ A user can check the e-mail header prior to downloading.

➢ A user can search the contents ofthe e-mail for a specific string ofcharacters prior to downloading.

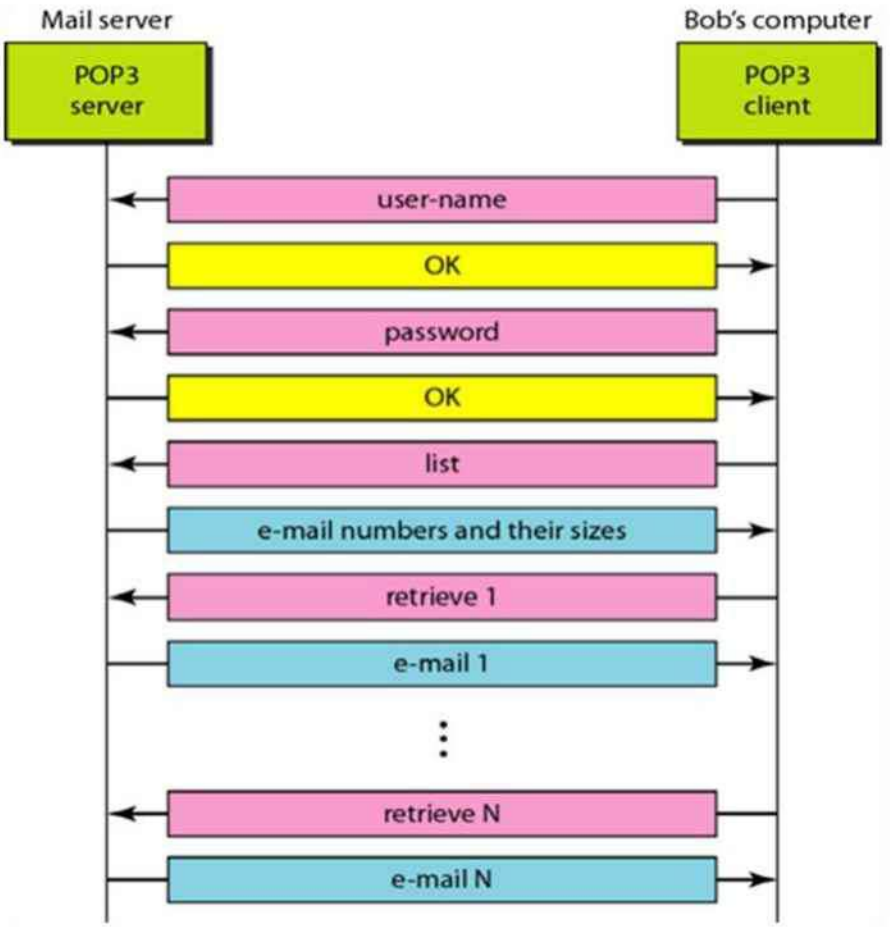
➢ A user can partially download e-mail. This is especially useful ifbandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.

➢ A user can create, delete, or rename mailboxes on the mail server.

➢ A user can create a hierarchy ofmailboxes in a folder for e-mail storage.

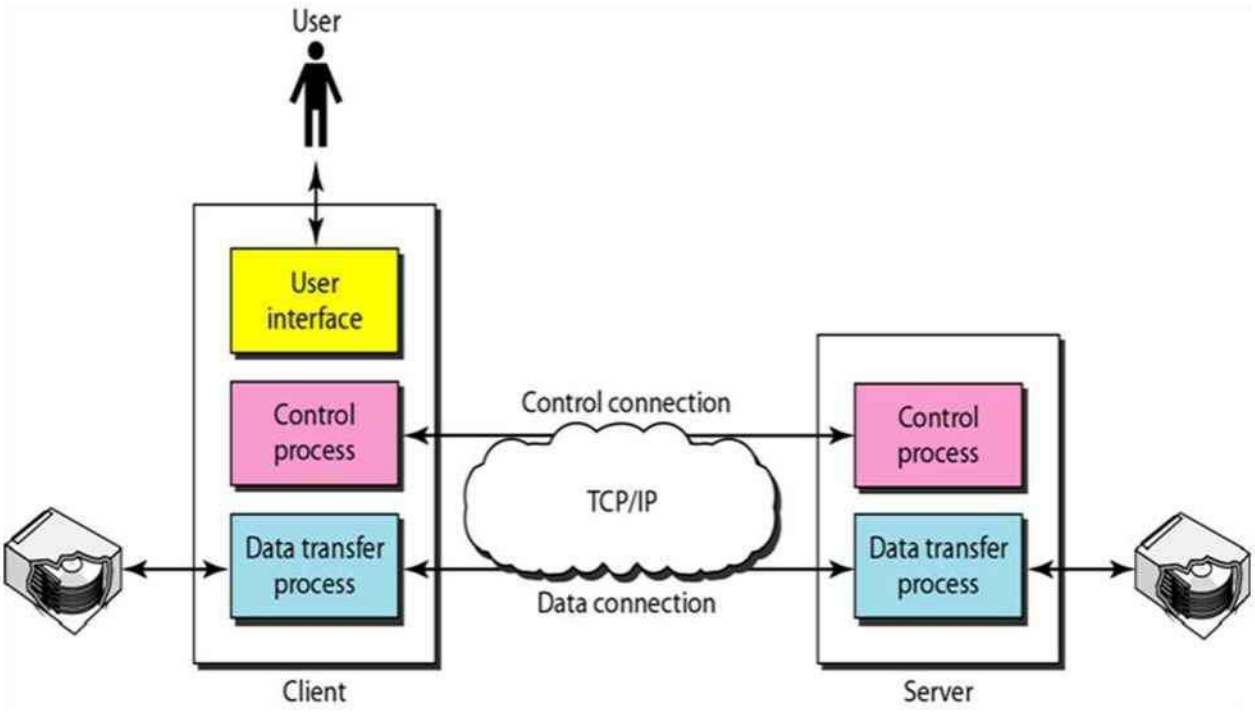


THE EXCHANGE OF COMMANDS AND RESPONSES IN POP3

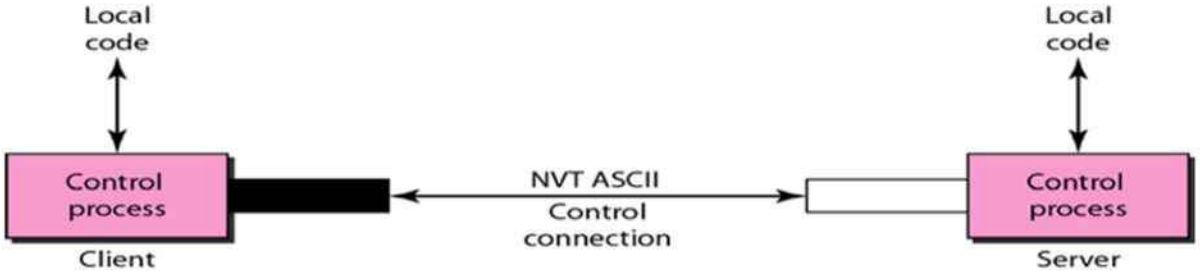


FILE TRANSFER PROTOCOL(FTP)

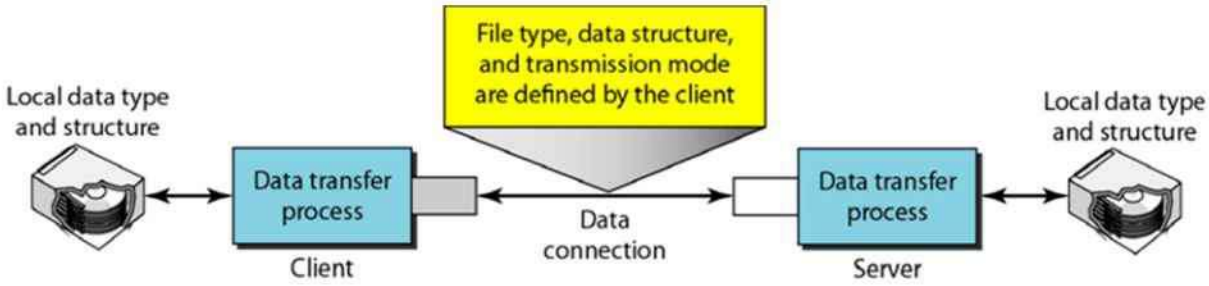
- The client has three components: user interface, client control process, and the client data transfer process. The server has two components: the server control process and the server data transfer process. The control connection is made between the control processes. The data connection is made between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



- **USING 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. This simple method is adequate for the control connection because we send one command (or response) at a time. Each line is terminated with a two-character (carriage return and line feed) end-of-line token.



- **USING DATA CONNECTION**: We want to transfer files through the data connection under the control of the commands sent over the control connection. file transfer in FTP means one of three things:
- 1. A file is to be copied from the server to the client. This is called retrieving and achieved using RETR command.
- 2. A file is to be copied from the client to the server. This is called storing and achieved using STOR command.
- 3.A list of directory or file names is to be sent from the server to the client. This is done using the LIST command. The heterogeneity problem is resolved by defining three attributes of communication: file type, data structure, and transmission mode.



- **File Type:** FTP can transfer one of the following file types across the data connection: an ASCII file, EBCDIC file, or image file. The ASCII file is the default format for transferring text files. Each character is encoded using 7-bit ASCII. The sender transforms the file from its own representation into ASCII characters, and the receiver transforms the ASCII characters to its own representation.
- **Data Structure:** FTP can transfer a file across the data connection by using one of the following interpretations about the structure of the data: file structure, record structure, and page structure. In the file structure format, the file 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.
- **Transmission Mode:** FTP can transfer a file across the data connection by using one of the following three transmission modes: stream mode, block mode, and compressed mode. The stream mode is the default mode. Data are delivered from FTP to TCP as a continuous stream of bytes. In block mode, data can be delivered from FTP to TCP in blocks. In the compressed mode, if the file is big, the data can be compressed.

TErminaL NETwork(TELNET)

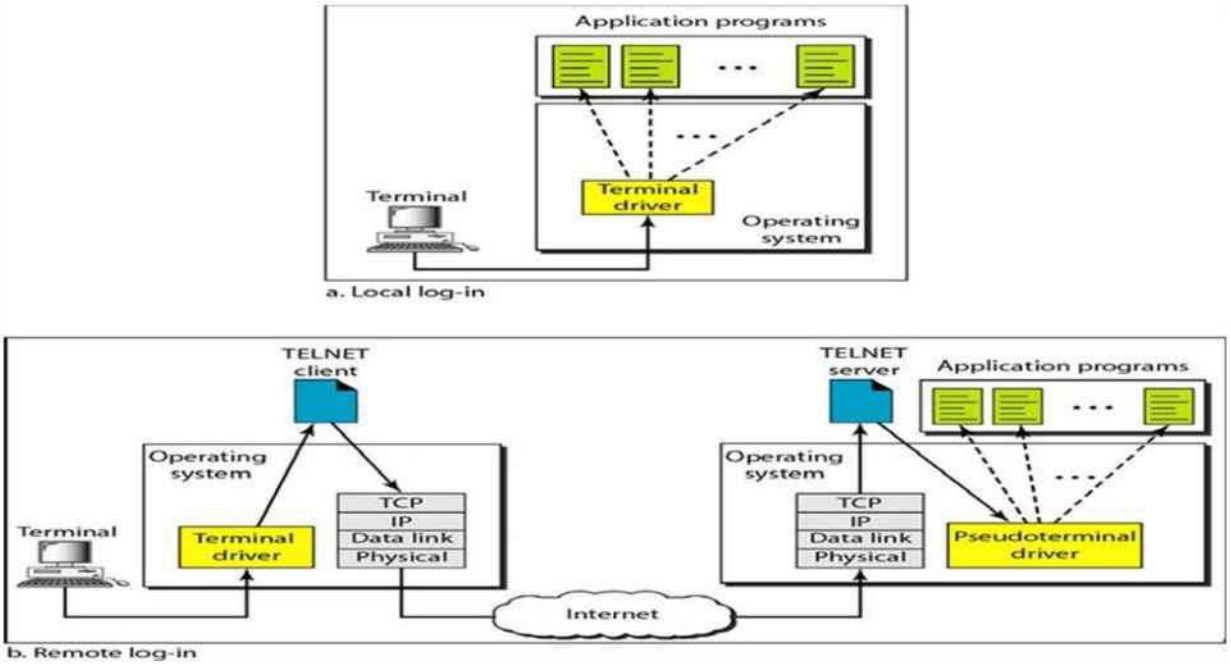
- TELNET is an abbreviation for TErminaL NETwork. It is the standard TCP/IP protocol for virtual terminal service as proposed by the International Organization for Standards (ISO). TELNET enables the establishment of a connection to a remote system in such a way that the local terminal appears to be a terminal at the remote system.
- TELNET is a general-purpose client/server application program.
- **Local and remote log-in:**

When a user logs into a local timesharing system, it is called local log-in. As a user types at a terminal or at a workstation running a terminal emulator, the key strokes 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.

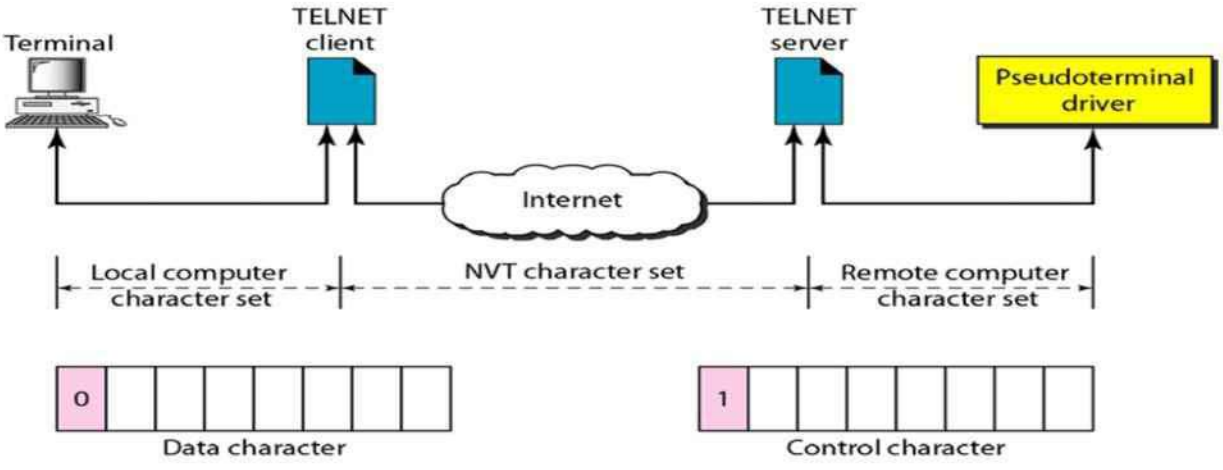
When a user wants to access an application program or utility located on a remote machine, she performs remote log-in. Here the TELNET client and server programs come into use. The user sends the keystrokes to the terminal driver, where the local operating system accepts the characters but does not interpret them. The characters are sent to the TELNET client, which transforms the characters to a universal character set called network virtual terminal (NVT) characters and delivers them to the local TCP/IP protocol stack. NVT uses two sets of characters, one for data and the other for control. Both are 8-bit bytes.

- The commands or text, in NVT form, travel through the Internet and arrive at the TCP/IP stack at the remote machine. Here 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. However, 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. The solution is to add a piece of software called a pseudo terminal driver which pretends that the characters are coming from a terminal. The operating system then passes the characters to the appropriate application program.



CONCEPT OF NVT



SOME NVT CONTROL CHARACTERS

Character	*Decimal*	*Binary*	*Meaning*
EOF	236	11101100	End of file
EOR	239	11101111	End of record
SE	240	11110000	Suboption end
NOP	241	11110001	No operation
DM	242	11110010	Data mark
BRK	243	11110011	Break
IP	244	11110100	Interrupt process
AO	245	11110101	Abort output
AYT	246	11110110	Are you there?
EC	247	11110111	Erase character
EL	248	11111000	Erase line
GA	249	11111001	Go ahead
SB	250	11111010	Suboption begin
WILL	251	11111011	Agreement to enable option
WONT	252	11111100	Refusal to enable option
DO	253	11111101	Approval to option request
DONT	254	11111110	Denial of option request
IAC	255	11111111	Interpret (the next character) as control

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.
- A party can offer to enable or disable an option if it has the right to do so. The offering can be approved or disapproved by the other party. To offer enabling, the offering party sends the WILL command, which means "Will I enable the option?" The other party sends either the DO command, which means "Please do," or the DONT command, which means "Please don't." To offer disabling, the offering party sends the WONT command, which means "I won't use this option any more." The answer must be the DONT command, which means "Don't use it anymore." A party can request from the other party the enabling

or the disabling ofan option. To request enabling, the requesting party sends the DO command, which means "Please do enable the option." The other party sends either the WILL command, which means "I will," or the WONT command, which means "I won't." To request disabling, the requesting party sends the DONT command, which means "Please don't use this option anymore." The answer must be the WONT command, which means "I won't use it anymore."

NVT CHARACTER SET FOR OPTION NEGOTIATION

Character	*Decimal*	*Binary*	*Meaning*
WILL	251	11111011	1. Offering to enable 2. Accepting a request to enable
WONT	252	11111100	1. Rejecting a request to enable 2. Offering to disable 3. Accepting a request to disable
DO	253	11111101	1. Approving an offer to enable 2. Requesting to enable
DONT	254	11111110	1. Disapproving an offer to enable 2. Approving an offer to disable 3. Requesting to disable

- TELNET uses only one TCP connection. The server uses the well-known port 23, and the client uses an ephemeral port. The same connection is used for sending both data and control characters. TELNET accomplishes this by embedding the control characters in the data stream. However, to distinguish data from control characters, each sequence of control characters is preceded by a special control character called interpret as control (lAC).

HTML

Planning for designing web pages:

Breaking Up Your Content into Main Topics

With your goals in mind, try to organize your content into main topics or sections, chunking related information together under a single topic.

Ideas for Organization and Navigation

At this point, you should have a good idea of what you want to talk about as well as a list of topics. The next step is to actually start structuring the information you have into a set of web pages. Before you do that, however, consider some standard structures that have been used in other help systems and online tools. This section describes some of these structures, their various features, some important considerations, including the following

Model and Structure of a Web site:

You need to know what the following terms mean and how they apply to the body of work you're developing for the Web:

Website: A collection of one or more web pages linked together in a meaningful way that, as a whole, describes a body of information or creates an overall effect.

Web server: A computer on the Internet or an intranet that delivers Web pages and other files in response to browser requests.

Web page: A single document on a website, usually consisting of an HTML document and any items that are displayed within that document such as inline images.

Home page: The entry page for a website, which can link to additional pages on the same website or pages on other sites.

Developing websites:

Designing a website, like designing a book outline, a building plan, or a painting, can sometimes be a complex and involved process. Having a plan before you begin can help you keep the details straight and help you develop the finished product with fewer false starts. Today, you learned how to put together a simple plan and structure for creating a set of web pages, including the following:

• Deciding what sort of content to present
• Coming up with a set of goals for that content
• Deciding on a set of topics
• Organizing and storyboarding the website

Basic HTML: HTML stands for Hypertext Markup Language. The idea here is that most documents have common elements for example, titles, paragraphs, and lists. Before you start writing, therefore, you can identify and define the set of elements in that document and give them appropriate names.

How Markup Works

HTML is a markup language. Writing in a markup language means that you start with the text of your page and add special tags around words and paragraphs. The tags indicate the different parts of the page and produce different effects in the browser. HTML has a defined set of tags you can use. You can't make up your own tags to create new styles or features.

What HTML Files Look Like

Pages written in HTML are plain text files (ASCII), which means that they contain no platform- or program-specific information. Any editor that supports text can read them. HTML files contain the following:

• The text of the page itself
• HTML tags that indicate page elements, structure, formatting, and hypertext links to

other pages or to included media. Most HTML tags look something like the following:
<thetagname>affected text</thetagname>
The tag name itself (here, thetagname) is enclosed in brackets (< >). HTML tags generally have a beginning and an ending tag surrounding the text they affect. The beginning tag "turns on" a feature (such as headings, bold, and so on), and the ending tag turns it off. Closing tags have the tag name preceded by a slash (/). The opening tag (for example, for paragraphs) and closing tag (for example, for paragraphs) compose what is officially called an HTML element.

Text Formatting and HTML

When an HTML page is parsed by a browser, any formatting you might have done by hand that is, any extra spaces, tabs, returns, and so on is ignored. The only thing that specifies formatting in an HTML page is an HTML tag. If you spend hours carefully editing a plain text file to have nicely formatted paragraphs and columns of numbers but don't include any tags, when a web browser loads the page, all the text will flow into one paragraph. All your work will have been in vain.
The advantage of having all white space (spaces, tabs, returns) ignored is that you can put your tags wherever you want. The following examples all produce the same output. Try them!

```
<h1>If music be the food of love, play on.</h1>
<h1>
If music be the food of love, play on.
</h1>
<h1>
If music be the food of love, play on. </h1>
<h1> If music be the food of love,
play on. </h1 >
```

Structuring Your HTML

The DOCTYPE Identifier

Although it's not a page structure tag, the XHTML 1.0 recommendation includes one additional requirement for your web pages. The first line of each page must include a DOCTYPE identifier that defines the XHTML 1.0 version to which your page conforms, and the document type definition (DTD) that defines the specification. This is followed by the <html>, <head>, and <body> tags. In the following example, the XHTML 1.0 Strict document type appears before the page structure tags:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/strict.dtd">
<html>
<head>
<title>Page Title</title>
</head>
<body>
...your page content...
</body>
</html>
```

Three types of HTML 4.01 document types are specified in the XHTML 1.0 specification:
Strict, Transitional, and Frameset.

The <html> Tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
...your page...
</html>
```

The <head> Tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>This is the Title. It will be explained later on</title>
</head>
...your page...
</html>
```

The <body> Tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>This is the Title. It will be explained later on</title>
</head>
<body>
...your page...
</body>
</html>
```

The Title

Each HTML page needs a title to indicate what the page describes. It appears in the title bar of the browser when people view the web page.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/transitional.dtd">
<html>
<head>
<title>The Lion, The Witch, and the Wardrobe</title>
</head>
<body>
...your page...
</body>
</html>
```

Headings

Headings are used to add titles to sections of a page. HTML defines six levels of headings.

Heading tags look like the following:

```
<h1>Movies</h1>
<h2>Action/Adventure</h2>
<h3>Caper</h3>
<h3>Sports</h3>
<h3>Thriller</h3>
```

<h3>War</h3>
<h2>Comedy</h2>
<h3>Romantic Comedy</h3>
<h3>Slapstick</h3>
<h2>Drama</h2>
<h3>Buddy Movies</h3>
<h3>Mystery</h3>
<h3>Romance</h3>
<h2>Horror</h2>

Paragraphs

As of the HTML 4.01 standard, paragraph tags are two-sided (...), and indicates the beginning of the paragraph. The closing tag is no longer optional, so rather than using to indicate where one paragraph ends and another begins, you enclose each paragraph within a tag.

Input

The dragon fell to the ground, releasing an anguished cry and seething in pain. The thrust of Enigern's sword proved fatal as the dragon breathed its last breath. Now Enigern was free to release Lady Aelfleada from her imprisonment in the dragon's lair.

Image:

Images displayed on the Web should be converted to one of the formats supported by most browsers: GIF, JPEG, or PNG. GIF and JPEG are the popular standards, and every graphical browser supports them. PNG is a newer image format that was created in response to some patent issue s with the GIF format.

The most important attribute of the <img> tag is src, which is the URL of the image you want to include. Paths to images are derived in the same way as the paths in the href attribute of links. So, to point to a GIF file named image.gif in the same directory as the HTML document, you can use the following HTML tag:

Input:

<h1>Welcome to The Halloween House of Terror!!</h1>

Output:

Links:

To create a link in an HTML page, you use the HTML link tag <a>.... The <a> tag often is called an anchor tag because it also can be used to create anchors for links.

Input

Go back to
Main Menu

Lists:

HTML 4.01 defines these three types of lists:

• Numbered or ordered lists, which are typically labeled with numbers
• Bulleted or unordered lists, which are typically labeled with bullets or some other symbol
• Glossary lists, in which each item in the list has a term and a definition for that term, arranged so that the term is somehow highlighted or drawn out from the text

List Tags

All the list tags have the following common elements:

• The entire list is surrounded by the appropriate opening and closing tag for the type of list (for example, <ul> and for unordered lists, or <ol> and for ordered lists).
• Each list item within the list has its own tag:
<dt> and <dd> for the glossary lists, and <li> for all the other lists.

Input

• Installing Your New Operating System
•
• Insert the CD-ROM into your CD-ROM drive.
• Choose RUN.
• Enter the drive letter of your CD-ROM (example: D:\), followed by SETUP.EXE.
• Follow the prompts in the setup program.
• Reboot your computer after all files are installed.
• Cross your fingers.
•

Customizing Ordered Lists

You can customize ordered lists in two main ways: how they're numbered and the number with which the list starts. HTML 3.2 provides the type attribute that can take one of five values to de fine which type of numbering to use on the list:
• "1" Specifies that standard Arabic numerals should be used to number the list (that is, 1, 2, 3, 4, and so on)
• "a" Specifies that lowercase letters should be used to number the list (that is, a, b, c, d, and so on)
• "A" Specifies that uppercase letters should be used to number the list (that is, A, B, C, D, and so on)
• "i" Specifies that lowercase Roman numerals should be used to number the list (that is, i, ii, iii, iv, and so on)
• "I" Specifies that uppercase Roman numerals should be used to number the list (that is, I, II, III, IV, and so on)
You can specify types of numbering in the <ol> tag, as follows: <ol type="a">. By default type="1" is assumed.

Input

The Days of the Week in French:
<ol type="I">
Lundi
Mardi
Mercredi
Jeudi
Vendredi
Samedi
Dimanche

Input

The Last Six Months of the Year (and the Beginning of the NextYear):
<ol type="I" start="7">
July
August

September
October
November
December
<li type="1">January

Tables:

Table Parts

Before getting into the actual HTML code to create a table, let's look at the following terms so that we both know what we're talking about:

• The caption indicates what the table is about: for example, "Voting Statistics" or "Toy Distribution Per Room" Captions are optional.

• The table headings label the rows, columns, or both. Usually they're in an emphasized font that's different from the rest of the table. They're optional.

• Table cells are the individual squares in the table. A cell can contain normal table data or a table heading.

• Table data is the values in the table itself. The combination of the table headings and table data makes up the sum of the table.

The <table> Element

The to create a table in HTML, you use <table>...</table> element to enclose the code for an optional caption, and then add the contents of the table itself:

<table>
...table caption (optional) and contents...
</table>

Rows and Cells

The cells within each row are created using one of two elements:

• <th>...</th> elements are used for heading cells. Generally, browsers center the contents of a <th> cell and render any text in the cell in boldface.

• <td>...</td> elements are used for data cells. TD stands for table data.

Input

<tr>
<th>Name</th>
<td>Alison</td>
<td>Tom</td>
<td>Susan</td>
</tr>
<tr>
<th>Height</th>
<td>5'4"</td>
<td>6'0"</td>
<td>5'1"</td>
</tr>
<tr>
<th>Weight</th>
<td>140</td>
<td>165</td>

<td>97</td>
</tr>
<tr>
<th>Eye Color</th>
<td>Blue</td>
<td>Blue</td>
<td>Brown</td>
</tr>

Setting Table Widths

To make a table as wide as the browser window, you add the width attribute to the table, as shown in the following line of code:

<table border="1" width="100%">

Changing Table Borders

You can change the width of the border drawn around the table. If border has a numeric value, the border around the outside of the table is drawn with that pixel width. The default is border="1". border="0" suppresses the border, just as if you had omitted the border attribute altogether.

Input

<table border="10" width="100%">

Cell Padding

The cell padding attribute defines the amount of space between the edges of the cells and the content inside a cell.

Input

<table cellpadding="10" border="1">

Cell Spacing

Cell spacing is similar to cell padding except that it affects the amount of space between cells that is, the width of the space between the inner and outer lines that make up the table border.

Input

<table cellpadding="10" border="4" cellspacing="8">

Spanning Multiple Rows or Columns

The tables you've created up to this point all had one value per cell or the occasional empty cell. You also can create cells that span multiple rows or columns within the table. Those spanned cells then can hold headings that have subheadings in the next row or column or you can create other special effects within the table layout.

Input

<html>
<head>
<title>Row and Column Spans</title>
</head>
<body>
<table border="1" summary="span example">
<tr>
<th colspan="2">Gender</th>
</tr>
<tr>
<th>Male</th>
<th>Female</th>

```
</tr>
<tr>
<td>15</td>
<td>23</td>
</tr>
</table>
</body>
</html>
```

Forms:

Using the <form> Tag

To accept input from a user, you must wrap all of your input fields inside a <form> tag. The purpose of the <form> tag is to indicate where and how the user's input should be sent. First, let's look at how the <form> tag affects page layout. Forms are block-level elements.

Input

```
<p>Please enter your username <form><input /> and password <input /></form> to log
in.</p>
```

The two most commonly used attributes of the <form> tag are action and method. Both of these attributes are optional. The following example shows how the <form> tag is typically used:

```
<form action="someaction" method="get or post">
content, form controls, and other HTML elements
</form>
```

action specifies the URL to which the form is submitted. Again, remember that for the form to be submitted successfully, the script must be in the exact location you specify and must work properly.

The method attribute supports two values: get or post. The method indicates how the form data should be packaged in the request that's sent back to the server. The get method appends the form data to the URL in the request.

Creating Text Controls

Text controls enable you to gather information from a user in small quantities. This control type creates a single-line text input field in which users can type information, such as their name or a search term.

Input

```
<p>Enter your pet's name:
<input type="text" name="petname" /></ p>
```

Creating Password Controls

The password and text field types are identical in every way except that the data entered in a password field is masked so that someone looking over the shoulder of the person entering information can't see the value that was typed into the field.

Input

```
<p>Enter your password: <input type="password" name="userpassword" size="8"
maxlength="8" /></p>
```

Creating Submit Buttons

Submit buttons are used to indicate that the user is finished filling out the form. Setting the type attribute of the form to submit places a submit button on the page with the

default label determined by the browser, usually Submit Query. To change the button text, use the value attribute and enter your own label, as follows:

<input type="submit" value="Send Form Data" />

Creating Reset Buttons

Reset buttons set all the form controls to their default values. These are the values included in the value attributes of each field in the form (or in the case of selectable fields, the values that are preselected). As with the Submit button, you can change the label of a Reset button to one of your own choosing by using the value attribute, like this:

<input type="reset" value="Clear Form" />

Creating Check Box Controls

Check boxes are fields that can be set to two states: on and off. To create a check box, set the input tag's type attribute to checkbox. The name attribute is also required, as shown in the following example:

Input

Check to receive SPAM email <input type="checkbox" name="spam" />

Creating Radio Buttons

Radio buttons, which generally appear in groups, are designed so that when one button in the group is selected, the other buttons in the group are automatically unselected. They enable you to provide users with a list of options from which only one option can be selected. To create a radio button, set the type attribute of an <input> tag to radio. To create a radio button group, set the name attributes of all the fields in the group to the same value. To cre ate a radio button group with three options, the following code is used:

Input

```
<p>Select a color:<br />
<input type="radio" name="color" value="red" /> Red<br />
<input type="radio" name="color" value="blue" /> Blue<br />
<input type="radio" name="color" value="green" /> Green<br />
</p>
```

Creating Menus with select and option

The select element creates a menu that can be configured to enable users to select one or more options from a pull-down menu or a scrollable menu that shows several options at once. The <select> tag defines how the menu will be displayed and the name of the parameter associated with the field. The <option> tag is used to add selections to the menu. The default appearance of select lists is to display a pull-down list that enables the user to select one of the options. Here's an example of how one is created:

Input

```
<p>Please pick a travel destination:
<select name="location">
<option>Indiana</option>
<option>Fuji</option>
<option>Timbuktu</option >
<option>Alaska</option>
</select>
</p>
```

Frames for designing a good website:

The first HTML document you need to create is called the frameset document. In this document, you define the layout of your frames, and the locations of the documents to be

initially loaded in each frame. Each of the three HTML documents other than the frameset document, the ones that load in the frames, contain normal HTML tags that define the contents of each separate frame area. These documents are referenced by the frameset document.

The <frameset> Tag

To create a frameset document, you begin with the <frameset> tag. When used in an HTML document, the <frameset> tag replaces the <body> tag, as shown in the following code:

```
<html>
<head>
<title>Page Title</title>
</head>
<frameset>
.. your frameset goes here ...
</frameset>
</html>
```

It's important that you understand up front how a frameset document differs from a normal HTML document. If you include a <frameset> tag in an HTML document, you cannot include a <body> tag also.

The cols Attribute

When you define a <frameset> tag, you must include one of two attributes as part of the tag definition. The first of these attributes is the cols attribute, which takes the following form:

```
<frameset cols="column width, column width, ...">
```

Input

```
<html>
<head>
<title>Three Columns</title>
</head>
<frameset cols="100,50%,*">
<frame src="leftcol.html">
<frame src="midcol.html">
<frame src="rightcol.html">
</frameset>
</html>
```

The rows Attribute

The rows attribute works the same as the cols attribute, except that it splits the screen into horizontal frames rather than vertical ones. To split the screen into two frames of equal height, you would write the following:

Input

```
<html>
<head>
<title>Two Rows</title>
</head>
<frameset rows="50%,50%">
<frame src="toprow.html">
<frame src="botrow.html">
</frameset>
```

</html>

The <frame> Tag

After you have your basic frameset laid out, you need to associate an HTML document with each frame by using the <frame> tag, which takes the following form:

<frame src="document URL">

For each frame defined in the <frameset> tag, you must include a corresponding <frame> tag, as shown in the following:

Input

```
<html>
<head>
<title>The FRAME Tag</title>
</head>
<frameset rows="*,*,*">
<frame src="document1.html" />
<frame src="document2.html" />
<frame src="document3.html" />
</frameset>
</html>
```

Changing Frame Borders

Start with the <frame> tag. By using two attributes, bordercolor and frameborder, you can turn borders on and off and specify their color. You can assign bordercolor any valid color value, either as a name or a hexadecimal triplet. frameborder takes two possible values:

1(to display borders) or 0 (to turn off the display of borders).

```
<html>
<head>
<title>Conflicting Borders</title>
</head>
<frameset frameborder="0" rows="*,*,*">
<frame frameborder="1" bordercolor="yellow" src="document1.html">
<frame bordercolor="#cc3333" src="document2.html">
<frame src="document3.html">
</frameset>
</html>
```

long questions:

1. What is internet? Want is WWW? What is the difference between them?
2. What are the different lists available explain briefly.
3. Explain the different tags and attributes available in table briefly.
4. What are the different tags available to create the elements of a form explain in detail.

Java Script, CSS and DOM

Java Script:

Programming Fundamentals:

JavaScript, originally called LiveScript, was developed by Brendan Eich at Netscape in 1995 and was shipped with Netscape Navigator 2.0 beta releases. JavaScript programs are used to detect and react to user-initiated events, such as a mouse going over a link or graphic. They can improve a Web site with navigational aids, scrolling messages and rollovers, dialog boxes, dynamic images, shopping carts, and so forth.

Client-side JavaScript programs are embedded in an HTML document between HTML head tags <head> and </head> or between the body tags <body> and </body>. Many developers prefer to put JavaScript code within the <head> tags, and at times, as you will see later, it is the best place to store function definitions and objects. If you want text displayed at a specific spot in the document, you may want to place the JavaScript code within the <body> tags. Or you may have multiple scripts within a page, and place the JavaScript code within both the <head> and <body> tags. In either case, a JavaScript program starts with a <script> tag, and and ends with a </script> tag. And if the JavaScript code is going to be long and involved, or may be reused, it can be placed in an external file (ending in .js) and loaded into the page.

```
1 <html>
2 <head><title>First JavaScript Sample</title></head>
3 <body bgcolor="yellow" text="blue">
4 <script language = "JavaScript" type="text/javascript">
4 document.writeln("<h2>Welcome to the JavaScript
World!</h1>");
5 </script>
6 <font size="+2">This is just plain old HTML stuff.</font>
7 </body>
8 </html>
```

Statements and Expressions:

Comments

Single line comments start with a double slash:

```
// This is a comment
```

For a block of comments, use the /* */ symbols:

```
/* This is a block of comments
that continues for a number of lines
*/
```

The <script> Tag

<script>
JavaScript statements...
</script>

<script>
document.write("Hello, world!
");
</script>

Attributes

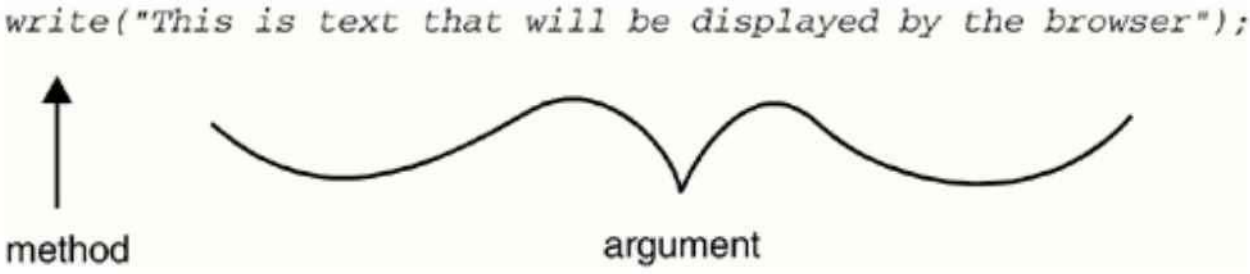
The <script> tag also has attributes to modify the behavior of the tag. The attributes are

- language
- type
- src

<script language="JavaScript"
type="text/javascript"
src="directory/sample.js">
</script>

String Concatenation

Concatenation is caused when two strings are joined together. The plus (+) sign is used to concatenate strings; for example:
"hot" + "dog or "San Francisco" + "</br>"
The write() and writeln() Methods



<html>
<head><title>Printing Output</title></head>
<body bgcolor="yellow" text="blue">
Comparing the document.write and document.writeln
 methods
<script language="JavaScript">
document.write("<h3>One, "); // No newline
document.writeln("Two, ");
document.writeln("Three, ");
document.write("Blast off...
"); // break tag
document.write("The browser you are using is " +
navigator.userAgent + "
");

```
</script>
<pre>
<script language="JavaScript">
document.writeln("With the <em>HTML &lt;pre&gt;
</em> tags, ");
document.writeln("the <em>writeln</em> method produces a newline.");
document.writeln("Slam");
document.writeln("Bang");
document.writeln("Dunk!");
</script>
</pre>
</body></html>
```

Data Types

Primitive Data Types

Primitive data types are the simplest building blocks of a program. They are types that can be assigned a single literal value such as the number 5.7, or a string of characters such as "hello". JavaScript supports three core or basic data types:

- numeric
- string
- Boolean

In addition to the three core data types, there are two other special types that consist of a single value:

- null
- undefined

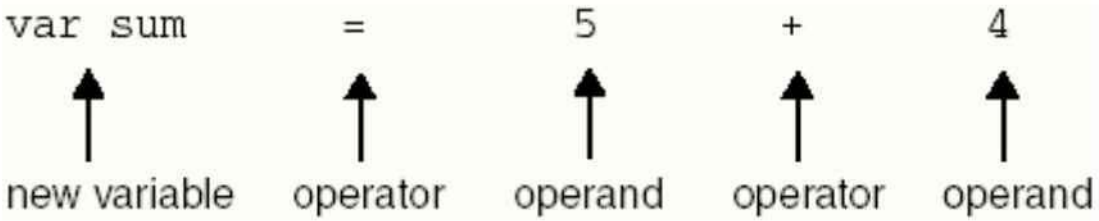
Variables

Variables are fundamental to all programming languages. They are data items that represent a memory storage location in the computer. Variables are containers that hold data such as numbers and strings. Variables have a name, a type, and a value. JavaScript variables can be assigned three types of data:

- numeric
- string
- Boolean

Operators:

Data objects can be manipulated in a number of ways by the large number of operators provided by JavaScript. Operators are symbols, such as +, –, =, >, and <, that produce a result based on some rules.



Precedence and associativity

Operator Description Associativity

() Parentheses Left to right

++ — Auto increment, decrement Right to left

! Logical NOT Right to left

* / % Multiply, divide, modulus Left to right

+ – Add, subtract Left to right

+ Concatenation Left to right

<<= Less than, less than equal to Left to right

>>= Greater than, greater than equal to Left to right

= = != Equal to, not equal to Left to right

= = = != = Identical to (same type), not identical to Left to right

&Bitwise AND Left to right

Operator Description Associativity

Bitwise OR

^ Bitwise XOR

~ Bitwise NOT

<< Bitwise left shift

>> Bitwise right shift

>>> Bitwise zero-filled, right shift

&& Logical AND Left to right

Logical OR Left to right

? : Ternary, conditional Right to left

= += – = *= /= %= <<= >>= Assignment Right to left

, (comma)

Types of Operators:

Arithmetic Operators

Arithmetic operators.

Operator/Operands	Function
x + y	Addition
x – y	Subtraction
x * y	Multiplication
x / y	Division
x % y	Modulus

Shortcut Assignment Operators

Assignment operators.

Operator	Example	Meaning
=	var x = 5;	Assign 5 to variable x.
+=	x += 3;	Add 3 to x and assign result to x.
–=	x –= 2;	Subtract 2 from x and assign result to x.
*=	x *= 4;	Multiply x by 4 and assign result to x.
/=	x /= 2;	Divide x by 2 and assign result to x.
**=	x **= 2;	Square x and assign result to x.
%=	x %= 2	Divide x by 2 and assign remainder to x.

Auto increment and Auto decrement Operators

To make programs easier to read, to simplify typing, and, at the machine level, to produce more efficient code, the auto increment (++) and auto decrement (– –) operators are provided. Auto increment and auto decrement operators.

Operator	Function	What It Does	Example
++x	Pre-increment	Adds 1 to x	x = 3; x++; x is now 4
x++	Post-increment	Adds 1 to x	x = 3; ++x; x is now 4
– –x	Pre-decrement	Subtracts 1 from x	x = 3; x – –; x is now 2
x– –	Post-decrement	Subtracts 1 from x	x = 3; – –x; x is now 2

Concatenation Operator

As shown in previous examples, the + sign is used for concatenation and addition. The concatenation operator, the + sign, is a string operator used to join together one or more strings. In fact, the concatenation operator is the only operator JavaScript provides to manipulate strings.

Operator Example Meaning

+ "hot" + "dog" Concatenates (joins) two strings; creates "hotdog".

"22" + 8 Converts number 8 to string "8", then concatenates resulting in "228". In statements involving other operators, JavaScript does not convert numeric values to strings.

+= x ="cow"; x+= "boy"; Concatenates two strings and assigns the result to x; x becomes "cowboy".

Comparison Operators

Operator/Operands Function

x == y	x is equal to y
x != y	x is not equal to y
x > y	x is greater than y
x >= y	x is greater than or equal to y
x < y	x is less than y
x <= y	x is less than or equal to y
x = = = y	x is identical to y in value and type
x != = y	x is not identical to y

Logical Operators

Operator/Operands Function

num1 && num2 True, if num1 and num2 are both true. Returns num1 if evaluated to false; otherwise returns num2. If operands are Boolean values, returns true if both operands are true; otherwise returns false.

num1 num2 True, if num1 is true or if num2 is true.

! num1 Not num1; true if num1 is false; false if num1 is true.

The Conditional Operator

FORMAT
conditional expression ? expression : expression

Examples:

x ?y : z If x evaluates to true, the value of the expression becomes y, else the value of the expression becomes z

big = (x > y) ? x : y If x is greater than y, x is assigned to variable big, else y is assigned to variable big

Bitwise Operators

Operator Function Example What It Does

&Bitwise AND x & y Returns a 1 in each bit position if both corresponding bits are 1.

Bitwise OR x y Returns a 1 in each bit position if one or both corresponding bits are 1.

^ Bitwise XOR x ^ y Returns a 1 in each bit position if one, but not both, of the corresponding bits are 1.

– Bitwise NOT –x Inverts the bits of its operands. 1 becomes 0; 0 becomes 1.

<< Left shift x << y Shifts x in binary representation y bits to left, shifting in zeros from the right.

>> Right shift x >> y Shifts x in binary representation y bits to right, discarding bits shifted off.

>>> Zero-fill right x shift >>> b Shifts x in binary representation y bits to the right, discarding bits shifted off, and shifting in zeros from the left.

Popup boxes:

JavaScript uses dialog boxes to interact with the user. The dialog boxes are created with three methods:

- alert()
- prompt()
- confirm()

The alert() Method

```
<html>
<head><title>Dialog Box</title></head>
<body bgcolor="yellow" text="blue">
```

```
<b>Testing the alert method</b><br>
<script language="JavaScript">
document.write("<font size='+2'>");
document.write("It's a bird, ");
document.write("It's a plane, <br>");
alert("It's Superman!");
</script>
</body></html>
```

The Prompt Box

Since JavaScript does not provide a simple method for accepting user input, the prompt dialog box and HTML forms are used. The prompt dialog box pops up with a simple text field box. After the user enters text into the prompt dialog box, its value is returned.

FORMAT

prompt(message);

prompt(message, defaultText);

Example:

prompt("What is your name? ", "");

prompt("Where is your name? ", name);

The Confirm Box

The confirm dialog box is used to confirm a user's answer to a question. A question mark will appear in the box with an OK button and a Cancel button. If the user presses the OK button, true is returned; if he presses the Cancel button, false is returned. This method takes only one argument, the question you will ask the user.

Example

```
<html>
<head>
<title>Using the JavaScript confirm box</title>
</head>
<body>
<script language = "JavaScript">
document.clear // Clears the page
if(confirm("Are you really OK?") == true){
alert("Then we can proceed!");
}
else{
alert("We'll try when you feel better? ");
}
</script>
</body>
</html>
```

Control Statements:
Conditionals

```
if (condition){
statements;
}
```

Example:

```
if ( age > 21 ){
alert("Let's Party!"); }
```

if/else

```
if (condition){
statements1;
}
else{
statements2;
}
```

Example:

```
if ( x > y ){
alert( "x is larger");
}
else{
alert( "y is larger");
}
```

Example

```
<html>
<head>
<title>Conditional Flow Control</title>
</head>
<body>
<script language=javascript>
<!-- Hiding JavaScript from old browsers document.write("<h3>");
var age=prompt("How old are you? ","");
if( age >= 55 ){
document.write("You pay the senior fare! ");
}
else{
document.write("You pay the regular adult fare. ");
}
document.write("</h3>");
//-->
</script>
</body>
</html>
```

if/else if

```
if (condition) {
statements1;
}
else if (condition) {
statements2;
}
else if (condition) {
statements3;
}
else{
statements4;
}
```

Switch/case

```
switch (expression){
case label :
statement(s);
break;
case label :
statement(s);
break;
...
default : statement;
}
```

Example:

```
switch (color){
case "red":
alert("Hot!");
break;
case "blue":
alert("Cold.");
break;
default:
alert("Not a good choice.");
break;
}
```

Example

```
<html>
<head>
<title>The Switch Statement</title>
</head>
<body>
<script language=javascript>
```

```
<!--
var color=prompt("What is your color?","");
switch(color){
case "red":
document.bgColor="color";
document.write("Red is hot.");
break;
case "yellow":
document.bgColor=color;
document.write("Yellow is warm.");
break;
case "green":
document.bgColor="lightgreen";
document.write("Green is soothing.");
break;
case "blue":
document.bgColor="#RRGGBB";
document.write("Blue is cool.");
break;
default:
document.bgColor="white";
document.write("Not available today. We'll use white");
break;
}
//-->
</script>
</body>
</html>
```

Loops

Loops are used to execute a segment of code repeatedly until some condition is met. JavaScript's basic looping constructs are

- while
- for
- do/while

The while Loop

The while statement executes its statement block as long as the expression after the while evaluates to true; that is, non-null, non-zero, non-false. If the condition never changes and is true, the loop will iterate forever (infinite loop). If the condition is false control goes to the statement right after the closing curly brace of the loop's statement block. The break and continue functions are used for loop control.

```
while (condition) {
statements;
increment/decrement counter;
}
```

Example

```
<html>
<head>
<title>Looping Constructs</title>
</head>
<body>
<h2>While Loop</h2>
<script language="JavaScript">
document.write("<font size='+2'>");
vari=0; // Initialize loop counter
while ( i< 10 ){ // Test
document.writeln(i);
i++; // Increment the counter
} // End of loop block
</script>
</body>
</html>
```

The do/while Loop

The do/while statement executes a block of statements repeatedly until a condition becomes false. Owing to its structure, this loop necessarily executes the statements in the body of the loop at least once before testing its expression, which is found at the bottom of the block.

```
do
{ statements;}
while (condition);
```

Example

```
<html>
<head>
<title>Looping Constructs</title>
</head>
<body>
<h2>Do While Loop</h2>
<script language="JavaScript">
document.write("<font size='+2'>");
vari=0;
do{
document.writeln(i);
i++;
} while ( i< 10 )
</script>
</body>
</html>
```

The for Loop

The for loop consists of the for keyword followed by three expressions separated by semicolons and enclosed within parentheses. Any or all of the expressions can be omitted, but the two semicolons cannot. The first expression is used to set the initial value of variables and is executed just once, the second expression is used to test whether the loop should continue or stop, and the third expression updates the loop variables; that is, it increments or decrements a counter, which will usually determine how many times the loop is repeated.

```
for(Expression1;Expression2;Expression3)
{statement(s);}
for (initialize; test; increment/decrement)
{statement(s);}
```

```
<html>
<head>
<title>Looping Constructs</title>
</head>
<body>
<h2>For Loop</h2>
<script language="JavaScript">
document.write("<font size='+2'>");
for(vari = 0; i< 10; i++ ){
document.writeln(i);
}
</script>
</body>
</html>
```

Try… Catch and Throw statements:

Catching errors in JavaScript:

It is very important that the errors thrown must be catched or trapped so that they can be handled more efficiently and conveniently and the users can move better through the web page.

Using try…catch statement:

The try..catch statement has two blocks in it:

- try block
- catch block

In the try block, the code contains a block of code that is to be tested for errors. The catch block contains the code that is to be executed if an error occurs. The general syntax of try..catch statement is as follows:

```
try
{
……
…… //Block of code which is to be tested for errors
}
catch (err)
{
……
…… //Block of code which is to be executed if an error occurs
}
```

When, in the above structure, an error occurs in the try block then the control is immediately transferred to the catch block with the error information also passed to the catch block. Thus, the try..catch block helps to handle errors without aborting the program and therefore proves user-friendly.

The concept of try…catch statement shown in an example:

```
<html>
<head>
<script type="text/javascript">
try
{
document.write(junkVariable)
}
catch(err)
{
document.write(err.message + "<br/>")
}
</script>
</head>
<body>
</body>
</html>
```

The output of the above program is 'junkVariable' is undefined

In the above program, the variable *junkVariable*is undefined and the usage of this in try block gives an error. The control is transferred to the catch block with this error and this error message is printed in the catch block.

throw in JavaScript:

There is another statement called throw available in JavaScript that can be used along with try…catch statements to throw exceptions and thereby helps in generating. General syntax of this throw statement is as follows:

```
throw(exception)
```
exception can be any variable of type integer or boolean or string.

for example:

```
<html>
<head>
<script type="text/javascript">
try
{
varexfor=10
if(exfor!=20)
{
throw "PlaceError"
}
}
catch(err)
{
if(err == "PlaceError")
document.write ("Example to illustrate Throw
Statement: Variable exfor not equal to 20.
<br/>")
}
</script>
</head>
<body>
</body>
</html>
```

The output of the above program is:

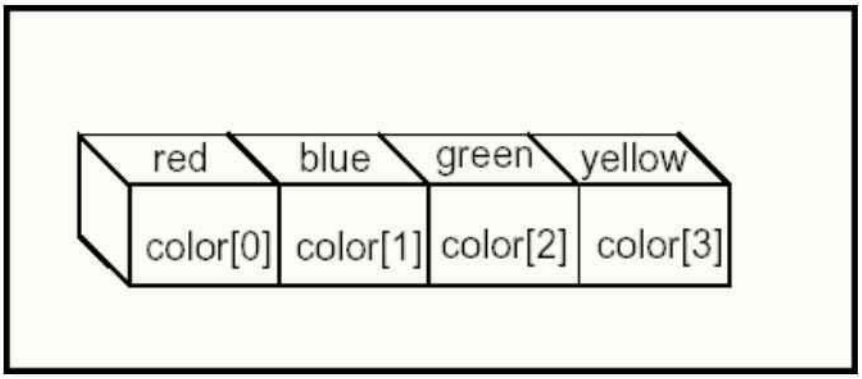
Example to illustrate Throw Statement: Variable exfor not equal to 20.

In the above example program, the try block has the variable exfor initialized to 10. Using the if statement, the variable value is checked to see whether it is equal to 20. Since exfor is not equal to 20, the exception is thrown using the throw statement. This is named Place Error and the control transfers to the catch block. The error catched is checked and since this is equal to the PlaceError, the statement placed inside the error message is displayed and the output is displayed as above.

Objects of java script:

Array Objects

An array is a collection of like values—called elements—such as an array of colors, an array of strings, or an array of images. Each element of the array is accessed with an index value enclosed in square brackets. An index is also called a subscript. There are two types of index values: a non-negative integer and a string. Arrays indexed by strings are called associative arrays. In JavaScript, arrays are built-in objects with some added functionality.



Declaring an Array

The following array is called array_name and its size is not specified.

vararray_name = new Array();

In the next example, the size or length of the array is passed as an argument to the Array() constructor. The new array has 100 undefined elements.

vararray_name = new Array(100);

And in the next example, the array is given a list of initial values of any data type:

vararray_name = new Array("red", "green", "yellow", 1 ,2, 3);

Example

```
<html>
<head><title>The Array Object</title>
<h2>An Array of Books</h2>
<script language="JavaScript">
var book = new Array(6); // Create an Array object
book[0] = "War and Peace"; // Assign values to its elements
book[1] = "Huckleberry Finn";
book[2] = "The Return of the Native";
book[3] = "A Christmas Carol";
book[4] = "The Yearling";
book[5] = "Exodus";
</script>
</head>
<body bgcolor="lightblue">
<script language="JavaScript">
document.write("<h3>");
for(vari in book){
document.write("book[" + i + "] "+ book[i] + "<br>");
}
</script>
</body>
</html>
```

Array Properties and Methods

Since an array is an object in JavaScript, it has properties to describe it and methods to manipulate it. The length of an array, for example, can be determined by the length property, and the array can be shortened by using the pop() method. For a complete list of array properties and methods

Array Object Properties

The Array object only has three properties. The most used is the length property which determines the number of elements in the array, that is, the size of the array.

Property What It Does

constructor References the object's constructor length Returns the number of elements in

the array prototype Extends the definition of the array by adding properties and methods

Array Methods

Whether you have an array of colors, names, or numbers, there are many ways you might want to manipulate the array elements. For example, you might want to add a new name or color to the beginning or end of the array, remove a number from the end of the array, or sort out all the elements, reverse the array, and so on. JavaScript provides a whole set of methods for doing all of these things and more

Method What It Does

concat() Concatenates elements from one array to another array join() Joins the elements of an array by a separator to form a string pop() Removes and returns the last element of an array push() Adds elements to the end of an array reverse() Reverses the order of the elements in an array sort() Sorts an array alphabetically, or numerically toString() Returns a string representation of the array

The Date Object

JavaScript provides the Date object for manipulating date and time. Like the String and Array objects, you can create as many instances as you like.

Example

```
var Date = new Date(); // The new constructor returns a Date object.
var Date = new Date("July 4, 2004, 6:25:22");
var Date = new Date("July 4, 2004");
var Date = new Date(2004, 7, 4, 6, 25, 22);
var Date = new Date(2004, 7, 4);
var Date = new Date(Milliseconds);
```

Using the Date Object Methods

Method What It Does

getDate Returns the day of the month (1–31)

getDay Returns the day of the week (0–6); 0 is Sunday, 1 is Monday, etc.

getFullYear Returns the year with 4 digits

getHours Returns the hour (0–23)

getMilliseconds Returns the millisecond

getMinutes Returns hours since midnight (0–23)

getMonth Returns number of month (0–11); 0 is January, 1 is February, etc.

getSeconds Returns the second (0–59)

setDate(value) Sets day of the month (1–31)

setFullYear() Sets the year as a four-digit number

setHours() Sets the hour within the day (0–23)

setHours(hr,min,sec,msec) Sets hour in local

setMilliseconds Sets the millisecond

setMinutes(min,sec, msec) Sets minute in local time

setMonth(month,date) Sets month in local time

setSeconds() Sets the second

setTime() Sets time from January 1, 1970, in milliseconds

setYear() Sets the number of years since 1900 (00–99)

toGMTString() Returns the date string in universal format

toString Returns string representing date and time

valueOf() Returns the equivalence of the Date object in milliseconds

Example

```
<html>
<head><title>Time and Date</title></head>
<body bgcolor="lightblue"><h2>Date and Time</h2>
<script language="JavaScript">
var now = new Date(); // Now is an instance of a Date object
document.write("<font size='+1'>");
document.write("<b>Local time:</b> " + now + "<br>");
var hours=now.getHours();
var minutes=now.getMinutes();
var seconds=now.getSeconds();
var year=now.getFullYear();
document.write("The full year is " + year +"<br>");
document.write("<b>The time is:</b> " +
```

```
hours + ":" + minutes + ":" + seconds);
document.write("</font>");
</script>
</body>
</html>
```

The Math Object

The Math object allows you to work with more advanced arithmetic calculations, such as square root, trigonometric functions, logarithms, and random numbers, than are provided by the basic numeric operators. If you are doing simple calculations, you really won't need it.

Math object methods.

Method Functionality

Math.abs(Number) Returns the absolute (unsigned) value of Number
Math.exp(x) Euler's constant to some power (see footnote)
Math.floor(Number) Rounds Number down to the next closest integer
Math.log(Number) Returns the natural logarithm of Number (base E)
Math.max(Number1, Number2) Returns larger value of Number1 and Number2
Math.min(Number1, Number2) Returns smaller value of Number1 and Number2
Math.pow(x, y) Returns the value of x to the power of y(x), where x is the base and y is the exponent
Math.random() Generates pseudorandom number between 0.0 and 1.0
Math.round(Number) Rounds Number to the closest integer
Math.sin(Number) Arc sine of Number in radians
Math.sqrt(Number) Square root of Number
Math.tan(Number) Tangent of Number in radians
Math.toString(Number) Converts Number to string

Example

```
<html>
<head><title>The Math Object</title></head>
<body>
<h2>Math object Methods--sqrt(),pow()<br>
Math object Property--PI</h2>
<P>
<script language="JavaScript">
varnum=16;
document.write("<h3>The square root of " +num+ " is ");
document.write(Math.sqrt(num),".<br>");
document.write("PI is ");
document.write(Math.PI);
document.write(".<br>"+num+" raised to the 3rd power is " );
document.write(Math.pow(num,3));
```

```
document.write(".</h3></font>");
</script>
</body></html>
```

The Boolean Object

The Boolean object was included in JavaScript 1.1. It is used to convert a non-Boolean value to a Boolean value, either true or false. There is one property, the prototype property, and one method, the toString() method, which converts a Boolean value to a string; thus, true is converted to "true" and false is converted to "false".

```
var object = new Boolean(value);
```

Example:

```
var b1 = new Boolean(5);
var b2 = new Boolean(null);
<html><head><title>Boolean Object</title>
</head>
<body bgcolor=aqua>
<font face="arial" size="+1"><b>
The Boolean Object<br>
<font size="-1">
<script language="JavaScript">
var bool1= new Boolean( 0);
var bool2 = new Boolean(1);
var bool3 = new Boolean("");
var bool4 = new Boolean(null);
var bool5 = new Boolean(NaN);
document.write("The value 0 is boolean "+ bool1 +"<br>");
document.write("The value 1 is boolean "+ bool2 +"<br>");
document.write("The value of the empty string is boolean "+ bool3+ "<br>");
document.write("The value of null is boolean "+ bool4+ "<br>");
document.write("The value of NaN is boolean "+ bool5 +"<br>");
</script>
</body></html>
```

Cascading Style Sheets:

Style can be delivered to a document by a variety of methods. The method with which style is connected with a document is referred to as *integration* . There are a variety of ways to integrate style, and how you decide to integrate style will depend largely upon what you are trying to accomplish with a specific document or number of documents.

Inline Style Sheets

The inline integration method allows you to take any tag and add a style to it. Using inline style gives you maximum control over a precise element of a web document, even just one character. Say you want to control the look and feel of a specific paragraph. You

could simply add a style="x" attribute to the paragraph tag, and the browser would display that paragraph using the style values you added to the code.

Example:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Inline Style Sample</title>
</head>
<body>
<h1 style="font-family: Arial" >Welcome!</h1>
</body>
</html>
```

Inline style is useful for getting precise control over something in a single document, but because it only applies to the element in question, you most likely won't be using inline style as frequently as other integration methods.

Internal Style Sheets

Embedding allows for control of a full document. Using the style element, which you place within the head section of a document, you can insert detailed style attributes to be applied to the entire page.
Embedding is an extremely useful way of styling individual pages that may also have other style methods influencing them. You can also style a single page or use multiple embedded sheets. The latter is especially useful if you'd like your document to have different styles for different media types.

Internal Style Sheet:

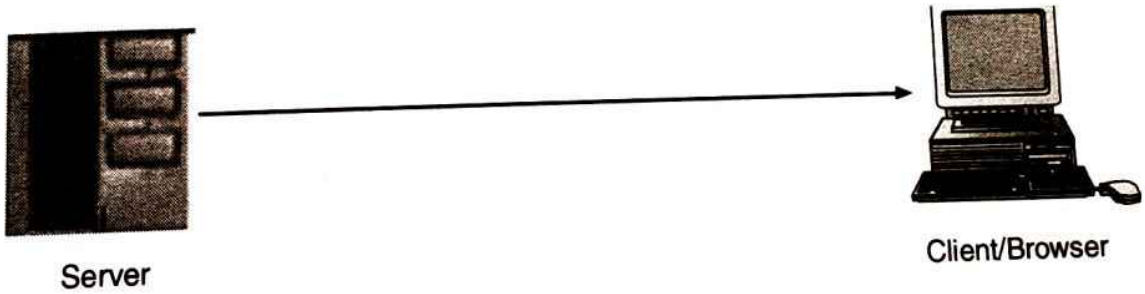
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Embedded Style Sample</title>
<style type="text/css" media="screen">
h1 {
font: Arial;
}
</style>
</head>
<body>
<h1>Welcome!</h1>
</body></html>
```

Websites classification

6.1. STATIC WEBSITES

static website contains web pages with fixed content. Each page is coded in HTML and displays I came information to every visitor. Static sites are the most basic type of website and are the easiest. create. It doesnot require any web programming or database design. A static site can be built by simply creating a few HTML pages and publishing them to a web server.

Since static web pages contain fixed code, the content of each page doesnot change unless it is manually updated by the webmaster. This works well for small websites, but it can make large sites with hundreds or thousands of pages difficult to maintain. Static sites that contain a lot of pages are often designed using templates. This makes it possible to update several pages at once, and also helps provide a consistent logout throughout the site.



Client/Browser Fig. 6.1. Static website

The static website is simple website design which is cost effective and beneficial for the small enterprises or individual to expand their business through web. Through static website individual to expand their business through web. Through static website individuals or small business houses can place simple information regarding their company and products in simple manner and at low cost. This type of Website is very useful for expanding market of company with its information and appearance on internet. Also termed as small business website and brochure websites. Static websites provides you with an online showcase where you can display your offerings to prospective visitors or clients. Static websites 1OW "What you see is what you have" concept, thus, ideal for those companies or individuals who just need the website to establish their web presence or use it as a contact platform for their clients.

TechnoGenx caters to all your static website development needs with perfection. Be it developing a simple and informative website or one with an attractive look and feel or both. We offer you our quality web services in the most attractive and cheapest prices. Examples of Static Websites

www.yashamanhospital.com

www.mukundamhospitalandkindeycentre.com

www.dlfgardencitygurgaon.net

www.rajputanasecurityguard.com

Advantages of Static Websites

* Quick to develop

• Cheap to develop

* Cheap to host

Disadvantages of Static Websites

* Requires web development expertise to update site.

* Site not as useful for the user.

* Content can get stagnant

. 6.2. DYNAMIC WEBSITES

Server A dynamic website contains information that changes, depending on the viewer of the site, the time of the day, the time zone, the native language of the country Client/Browser the viewer is in or many other factors.

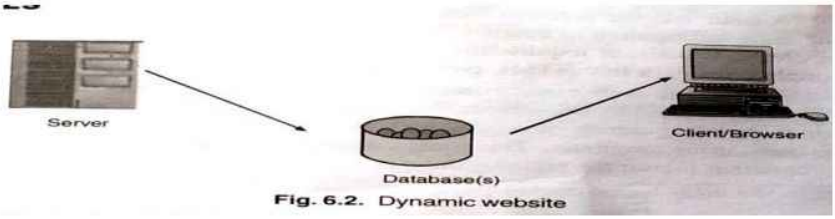


Fig. 6.2. Dynamic website

Database(s) Dynamic websites contain Fig. 6.2. Dynamic website web pages that are generated in real-time. These pages include web scripting code, such as PHP or ASP. When a dynamic page is accessed, the code within the page is passed on the web server and the resulting HTML is sent to the client web browser. Most large websites are dynamic. Since they are easier to maintain then static websites. This is because static pages each contain unique content, meaning must be manually opened, edited and published whenever a change is made. Dynamic pages, on the other hand, access information from a database. Therefore, to alter the content of a dynamic page, the web master may only need to update a database record. This is especially helpful for large sites that contain hundred or thousands of pages. It also makes it possible for multiple users to update the content of a website without editing the layout of the pages. Dynamic websites that access information from a database are also called database-driven websites. Dynamic sites can be more expensive to develop initially, but the advantages are numerous. At a basic level, a dynamic website can give the website owner the ability to simply update and add new content to the site.

197 Example of Dynamic Website es and events could be posted to the site through a simple browser interface. Dynamic features of site are only limited by imagination. Some examples of Dynamic Website would be: content management system, e-commerce system, bulletin/discussion pet facilities, ability for clients or users to upload documents ability for administrators or us ntent or add information to the site (dynamic publishing). features could be: content ma or extranet facilities, ability to create content or add

Advantages of Dynamic Website

Much more functional website.

• Much easier to update. New content brings people back to the site and helps in search engines. . Can work as a system to allow staff or users to collaborate.

Disadvantages of Dynamic Websites

• Slower/more expansive to develop.

* Hosting costs a little more.

6.3. WEB PORTALS

A web portal is most often one specially-designed webpage at a website which brings information together from diverse sources in a uniform way. Usually, each information source gets its dedicated area on the page for displaying information (a portlet) often the user can configure which ones to display. The extent to which content is displayed in a "uniform way" may depend on intended user and the intended purpose, as well as the diversity of the content.

A portal may use a search engine API to permit users to search intranet content as opposed to extranet content by restricting which domains may be searched. Apart from this common search engines feature, web portals may offer other services such as e-mail, news, stock quotes, information from databases and even entertainment content. Portals provide a way for enterprises and organizations to provide a consistent look and feel with access control and procedures for multiple applications and databases, which otherwise would have been different web entities at various URLs. The features available may be restricted by whether access is by an authorized and authenticated user (employee, member) or an anonymous site visitor. Examples of Early Public webportal were AOL, Excite, iGoogle, MSN, Rediff and Yahoo. Classification of Web Portals Web portals are classified as horizontal or vertical. A horizontal portal is used as a platform to several companies in same economic sector or to the same type of manufacturers or distributors. • A vertical portal (also known as vortal) is a specialized entry point to a specific market on industry niche, subject area or interest. Some vertical portals are known as "Vertical information portals" (VIPs). VIPs provide news, editorial content, digital publications and e-commerce capabilities.

6.4. SOCIAL NETWORKING SITES

Social networking platforms may allow organizations to improve communication and productivity disseminating information among different groups of employees in a more efficient manner.

Advantages of Social Networking Sites

* Facilitates open communication, leading to enhanced information discovery and delivery

* Allows employees to disucss ideas, post news, ask questions and share links.

* Provides an opportunity to widen business contacts.

* Targets a wide audience, making it a useful and effective recruitment tool.

* Improves business reputation and client base with minimal use of advertising.

* Expands market research, implements campaigns, delivers communications and directs inter people to specific websites. Disadvantages of Social Networking Sites

* Opens up the possibility for hackers to commit fraud and launch spam and virus attacks Increases the risk of people falling prey to online scams that seem genuine, resulting in data or identity theft.

* Potentially results in lost productivity, especially if employees are busy updating profiles etc Potentially results in negative comments from employees about the company or potentialle onsequences if employees use these sites to view objectionable, illicit or offensive material A Social Networking Use Policy Generally Define what social networking is particular to your organization, so employees know exactly what is meant by the term. * Establishes a clear and defined purpose for the polilcy. * Communicates benefits of social networking and of having a policy. * Provides a clear platform for educating employees. Refers to confidentiality of employer trade secrets and private or confidential information. Talks about productivity in temrs of social networking. * Privides examples of policy violations. * Outlines disciplinary measures to be taken for policy violations. What may be the most concerning aspect of social networking platforms is that they encourage people to share personal information. Even the most cautions and well-meaning individuals can give away information they should not, the same applies to what is posted on company approved social networking platforms. Examples of Social Networking sites Facebook, Flicker, Google+, HiS, ibibo, MyOpera, Orkut, Twitter etc.

6.5. RSS FEEDS RSS

is about getting live web feeds directly to your computer. RSS takes the latest headlines from different websites, and pushes those headlines down to your computer for quick scanning. The acronym RSS stands for many versions of the same thing. • Really simply syndication. * Rich site summary (RS 0.91) • RDF site summary (RS0.9 and 1.0) • Real-time simple syndication (RSS 2.0).

In each of the abov latest news directly sports, favourate combined ("aggrega Once published at the souro each of the above meaning, the purpose is the same: to have websites of vour choice deliver their news directly to your monitor. So instead of having to visit 14 different places to get your weather, favourate

photos, latest gossip or latest political debates, you just go to one screen and see it ined ("aggregated") into a single window. The RSS headlines and stories are effectively immediate. blished at the source server, RSS headlines take only moments to get to your screen. Why you would use RSS: 1. Hurricane Watch: If you live in "hurricaneally" of the USA, you will want to pay close attention to hurricane warning and evacuation tips. RSS is absolutely a good tool here. News: To get the freshest news on your favourate celebrity, the country you are about or your favourite sports team. 2 Hobby Interests: If you are a motor cyclist, a skier, or perhaps, a dog trainer, hundreds conversations and bits of hobby advice can be fed directly to your screen. 4. Photos: If you like to change your computer wallpaper daily, then RSS feeds are an excellent way to get the latest from photographers on the web. 5. Reading Your Friends Blogs: If you have loved ones around the globe who do blogging, then you can have all their latest entries fed directly to your screen. This is very helpful for families when one of their own is in the military serving in Iraq or Afghanistan or Africa. A good way to read about how they are doing during their military assignment. 6. Politics: If you are helping a political candidate get elected, then RSS in an invaluable tool for watching popular opinion and blog postings. 7. Jokes and Inspirational Quotes: Add a clean laugh to your morning, or a pick-me-up quote from a famous person__ RSS can do that for you. 8. Currency Exchange Rates: If you are planning a trip to another country, you can watch for when the best time is to buy the that currency.

How it Works:

1. Behind the Scenes: RSS headlines are really simple text files that the publishing web master submits to a special feed server. That RSS feed server, in turn, pushes the text file to the screens of its subscribers. Time lag is usually 30 seconds to 30 minutes before the subscribers see the updates. In most cases, the lag is not even noticeable.

2. To Get Started: You choose an RSS reader tool for yourself. Most RSS readers are free to use, and easy to learn.

3. Setup Your Screen: You load the RSS feeds into your reader tool. This is achieved through multiple different ways. You can visit the web feed site directly, you can copy-paste the special code from an e-mail, or you can load copies from your friend's RSS reader screen.

4. Then You Start Reading Your Web Feed News: You simply log in to your RSS reader page, or start your RSS software, and you can scan all your web feeds instantly. You can arrange the RSS feeds into folders, just like e-mail, and you can even set alert and sounds for when a particular web feed is updated. Examples of RSS Readers: Newsgator.com

* Bloglines.com

• Goodle Reader. * My Yahoo Reader

blogs 6.6. BLOG

A blog (a truncation of the expression web log) is a discussion or informational site published the world wide web and consisting of discrete entities (“Posts”) typically displayed in reverse chronological order (the most recent post appears first). Until, 2009 blogs were usually the work of a single individual occasionally of a small group, and often caused a single subject. More recently "multi-aut blogs” (MABs) have developed, with posts written by large numbers of authors and professionally edited The rise of Twitter and other "Microblogging” systems helps integrated MABs and single-author into societal newstreams. Blog can also be used as a verb, meaning to maintain or add content to a blog Blogging can be seen as a form of social networking service. Indeed, bloggers do not only produ only produce content to post on their blogs, but also build social relations with their readers and other bloggers. There are high-readership blogs which do not allow comments, such as Darling Fireball. Many blogs provide commentary on a particular subject, others function as more personal online diaries. Most blogs are primarily textual, although some focus on act (act blogs), photographs (photoblogs) video (video blogs or “Vlogs”), music (MP3 blogs and audio (Podcasts). Microblogging is another type of blogging, featuring very short posts. In education, blogs can be used as instructional resources. These blogs are referred to as edublogs. According to critics and other bloggers, Blogger is the most popular blogging service used today.

6.7. NETIQUETTE

The word netiquette derives from net and etiquette and as such, is a set of established conventions that have evolved over time on the internet and on the usenet news. Netiquette is most of all based on: * What has been found useful and proper in the electronic form of communication made possible by the Internet ? * What is appropriate in any form of communication between civilized human beings? and what is dictated by the common sense ? What Good might following the netiquette do you ? What you write and how you behave makes you internet personality, which then directly influences and alters your “real-world personality". Think carefully what kind of picture you wish to give of yourself. One day you will most probably have dealings with a few of the persons who have formed their picture of you on the internet in real life. Displaying a good familiarity of netiquette in one's postings can sometimes have a "secret handshake effect on the internet. If you pay attention to netiquette: * Your communication will be better you will gain more credibility. You will in all probability get helpful answers more readily when you have

questions to ask of need help on the internet. If, however, you repeatedly and deliberately ignore the netiquette, you will soon develop a reputation of a troublemaker and at least the more serious users will start avoiding you altogether.

4.1. INTRODUCTION OF INTERNET SECURITY

Internet security is a branch of computer security specifically related to the internet, often involving browser security but also network security on a more general level as it applies to other applications or operating systems on a whole. Its objective is to establish rules and measures the use agains attacks over the internet. The internet represents an insecure channel for exchanging information leading to a high risk of instruction or fraud such as phishing. Different methods have been used to protect the transfer of data including encryption.

4.2. TYPES OF SECURITY

4.2.1. Network Layer Security

TCP/IP can be made secure with the help of cryptographic methods and protocols that have been developed for securing communications on the internet. These protocols include SSL and TLS for web traffic, PGP for e-mail and IPsec for network layer security.

4.2.2. IPsec Protocol

This protocol is designed to protect communication in a secure manner using TCP/IP. It is a set of security extensions developed by IETF, and it provides security and authentication at the IP layer by using cryptography. To protect the content the data is transformed using encryption techniques. There are two main type of transformation that form the basis of IPsec: the Authentication Header (AH) and Encapsulating Security Payload (ESP). These two protocol provides data integrity, data origin authentication, and anti replay service.

The basic components of the IPsec security architecture are described in terms of following functionalities:

- Security protocol for AH and ESP.

- Security association for policy management and traffic processing.
- Manual and automatic key management for Internet Key Exchange (IKE).

The set of security services provided at the IP layer includes access control, data origin, integrity, protection against replays and confidentiality.

4.2.3. Electronic Mail Security (E-mail)

E-mail messages are composed, delivered and stored in a multiple step process, which starts with the message's composition. When the user finishes composing the message and sends it, the message is transformed into a standard format: an REC 2822 formatted message. Afterwards, the message can be transmitted. Using a network connection, the mail client, referred to as a mail user agent (MUA), connects to a mail transfer agent (MTA) operating on the mail server. The mail client then provides the sender's identity to the server. Next using the mail server commands, the client sends the recipient list to the mail server. The client then supplies the message. Once the mail server receives and processes the message, several events occur: recipient server identification, connection establishment, and message transmission. Using Domain Name System (DNS) services, the sender's mail server determines the mail server(s) for the recipient(s). Then, the server opens up a connection(s) to the recipient mail server(s) and sends the message employing a process similar to that used by the originating client, delivering the message to the recipient(s).

4.2.4. Pretty Good Privacy (PGP)

PGP provides confidentiality by encrypting messages to be transmitted or data files to be stored using an encryption also such as Triple DES. Email messages can be protected by using cryptography in various ways, such as the following:

- Signing an email message to ensure its integrity and confirm the identity of its sender.
- Encrypting the body of an e-mail message to ensure its confidentiality.
- Encrypting the communication between mail servers to protect the confidentiality of both the message body and message header.

The first two methods, message signing and message body encrypted are often used together, however, encrypting the transmission between mail server is typically used only when two organizations want to protect e-mails regularly sent between each other. For example, the organization could establish a Virtual Private Network (VPN) to encrypt the communication between their mail servers over the internet. Unlike methods that an only encrypt a message body, a VPN can encrypt entire messages, including email header, information such as sender, recipients and subjects. In some cases, organizations may need to protect header information. However, a VPN solution alone cannot provide a message signing mechanism, nor can it provide protection for email messages along the entire route from sender to recipient.

4.2.5. Multipurpose Internet Mail Extension (MIME)

MIME transforms non-ASCII data at the sender's site to Network Virtual Terminal (NVT) ASCII data and delivers it to client's Simple Mail Transfer Protocol (SMTP) to be sent through the internet. The server SMTP at the receivers side receive the NVT ASCII data and delivers it to MIME to be transformed back to the original non-ASCII data.

4.2.6. Message Authentication Code

A message authentication code is a cryptography method that uses a secret key to encrypt a message. This method outputs a MAC value that can be decrypted by the receiver, using the same secret key used by the sender. The Message Authentication code protects both a message's data integrity as well as its authenticity.

4.3. AUTHENTICATION AND AUTHORIZATION

DEFINITION

Authentication is a process for identifying and verifying who is sending a request.

DEFINITION

Authorization is a process to provide access rights to the individuals.

The process of identifying an individual usually based on a username and password. In security systems, authentication is distinct from *authorization*, which is the process of giving individuals access to system objects based on their identity. Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual.

The Figure 4.1 shows a simplified version of an authentication process.

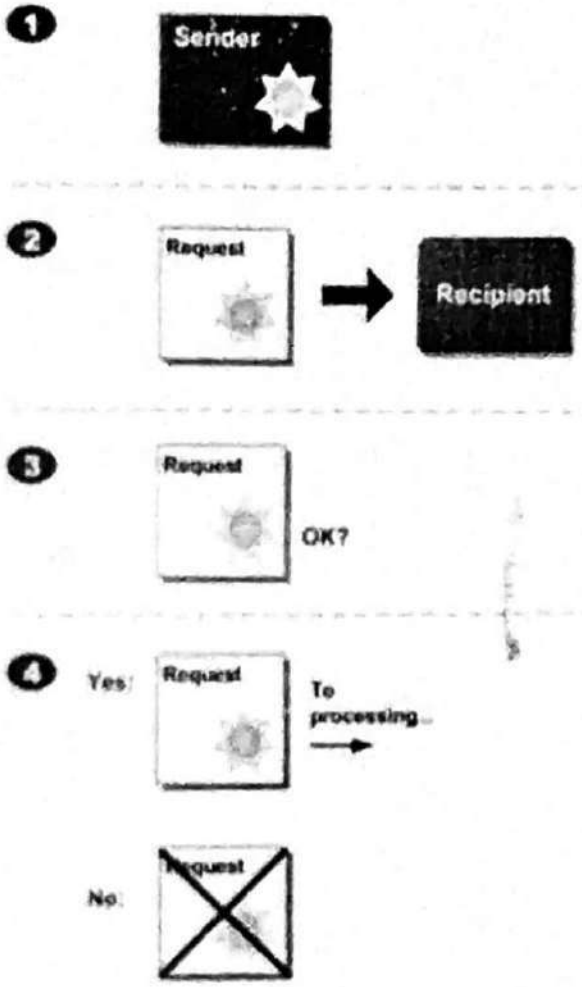


Fig. 4.1.

General Process of Authentication

1. The sender obtains the necessary credential.
2. The sender sends a request with the credential to the recipient.
3. The recipient uses the credential to verify the sender truly sent the request.
4. If yes, the recipient processes the request. If no, the recipient rejects the request and responds accordingly.

Normally authentication requires an entity to provide an identifier and then prove the identity by providing something you know (a password), something you have, or something you are.

Multifactor authentication combines multiple factors of information to improve security. Multifactor authentication normally combines something you know with something you are.

The process of authorization occurs once the user is authenticated. Authorization is the process which system uses to determine what a user is allowed to do once, he/she is authenticated. Whenever the authenticated user needs to access some files or response, the system verifies that operation against an Access control entries that define which users can or cannot perform certain operations. These operations include read a file, modify a file's contents, update a file properties, perform a backup, shut down a system etc.

4.3.1. Types of Authentication

As stated earlier, encryption is the process of taking all of the data that one computer is sending to another and encoding it into a form that only the other computer will be able to decode. Another

process, **authentication**, is used to verify that the information comes from a trusted source. Basically, if information is "authentic," we know who created it and we know that it has not been altered in any way since that person created it. These two processes, encryption and authentication, work hand-in-hand to create a secure environment.

There are several ways to authenticate a person or information on a computer:

1. Password : The use of a user name and password provides the most common form of authentication. You enter your name and password when prompted by the computer. It checks the pair against a secure file to confirm. If either the name or the password does not match, then you are not allowed further access.

2. Pass Cards : These cards can range from a simple card with a magnetic strip, similar to a credit card, to sophisticated smart cards that have an embedded computer chip.

3. Digital Signatures : A digital signature is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file) is authentic.

TIPS

Recently, more sophisticated forms of authentication have begun to show up on home and office computer systems. Most of these new systems use some form of biometrics for authentication. Biometrics uses biological information to verify identity. Biometric authentication methods include:

(a) Fingerprint scan
(b) Retina scan
(c) Face scan
(d) Voice identification

4.4. FIREWALLS

DEFINITION

A Firewall is a computer program that monitors the flow of information from the Internet to your computer.

There are two different types of firewall available for you to use: Hardware firewalls and Software firewalls.

Hardware Firewall

A Hardware Firewall is a physical piece of equipment that sits between the Internet and your computer. An example of a hardware firewall is a broadband router, a common form of Internet connection. The benefit of using a hardware firewall, is that it has the ability to protect multiple computer systems that are connected to it at the same time. This makes it an effective firewall for use in businesses that have multiple computers connected to the Internet, as well as in homes that have more than one computer system.

Software Firewall

Software Firewalls work in the same way as a hardware firewall, by monitoring and blocking information that comes to your computer via the Internet, however software firewalls must be installed as a program on your computer. These software firewalls can either be installed from a computer disk that you have purchased, or downloaded over the Internet. Software firewalls are the most common type of firewall. **Programs such as Norton 360, Norton Internet Security, ESET Smart Security, and Kaspersky Internet Security all have a firewall bundled within them.**

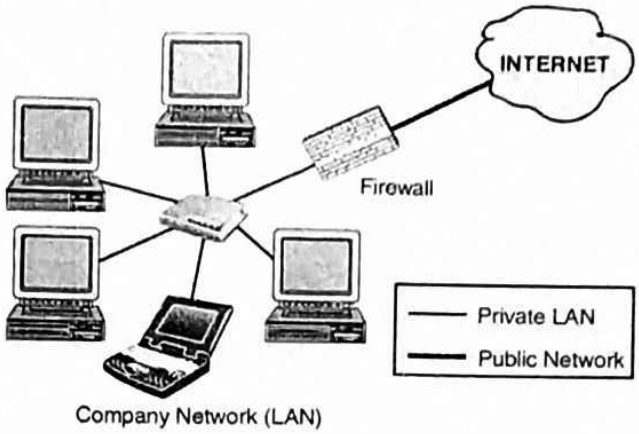


Fig. 4.2. Firewall.

Let us see the advantages and disadvantages of both of these :

Table 4.1. Advantages and Disadvantages of Firewall

Firewall Type	Advantages	Disadvantages
Hardware Firewall	• Operating system independent • Not vulnerable to malicious attacks • Better performance • Focuses on only firewall-related duties	• Can be single point of failure • Higher administrative overhead • Higher cost to implement and maintain
Software Firewall	• Less expensive to implement and maintain • Lower administrative overhead	• Dependent upon host operating system • Requires additional host hardware • Vulnerable to malicious attacks • Lower performance

Regardless of what type of firewall you choose to use, having one on your computer is a really good idea. A firewall helps to prevent computer hackers from accessing your computer through the Internet, and stealing sensitive information or infecting your computer system with some form of computer virus. A hacker can gain access to your system by "back doors" or open ports that connect your computer to the Internet. With a firewall in place these ports are blocked from inbound traffic, effectively closing the door in the hackers face.

TIPS

There are basically two ways that Firewalls work. Generally, data that comes in is analyzed by the firewall to determine the IP address it is coming from and the content that it contains. The firewall system then checks to see if this information is compliant with rules that you are able to configure. It can also analyze information at the application level. The firewall program will determine whether or not the application should be able to send or receive data through the port you are connected by.

4.4.1. Components of Firewall

Fundamentally, all firewalls consist of the following kinds of components:

1. Chokes

Computer or communications devices that restrict the free flow of packets between networks. Chokes are often implemented with routers, but they do not have to be. The use of the word "choke" is taken from the field of electronics: a choke is a device that exhibits great resistance to certain types of signals, but not to others.

2. Gates

Specially designated programs, devices, or computers within the firewall's perimeter that receive connections from external networks and handle them appropriately. Other texts on firewalls sometimes refer to single machines that handle all gate functions as *bastion hosts.*

Ideally, users should not have accounts on a gate computer. This restriction helps improve the computer's reliability and users' security.

3. Network Client Software

Client software includes programs such as telnet, ftp and mosaic. One of the simplest ways to give users limited access to the Internet is to allow them to log onto the gate machine and allow them to run network client software directly. This technique has the disadvantage that you must either create user accounts on the gate computer, or you must have users share a single account.

4. Proxy Server

A proxy is a program that poses as another. In the case of a firewall, a proxy is a program that forwards a request through your firewall, from the internal network to the external one.

5. Network Servers

Many network servers can also function as proxies. They can do so because they implement simple store-and-forward models, allowing them to forward queries or messages that they cannot handle themselves. Some servers that can operate easily as proxies include SMTP (because e-mail messages are automatically forwarded), NNTP (news is cached locally), NTP (time is maintained locally), and DNS (host addresses are locally cached). The following sections explore a variety of different kinds of firewall configurations in use today.

4.4.2. Firewall Features

When choosing your firewall it is important to pay attention to what features they offer you as these features can make a large difference in how your computer is protected. For some people certain features are more important than others, but in terms of security the most important are inbound and outbound filtering, application protection, notifications, stealth mode. These features and others will be discussed below:

1. Inbound and Outbound Filtering

Filtering is when a firewall examines information passing through it and determines if that information is allowed to be transmitted and received or should be discarded based on rules or filters that have been created. This function is the primary function of a firewall and how it handles these tasks if very important for your security.

2. Stealth Mode

It is important for your firewall to not only block requests to reach your computer, but to also make it appear as if your computer does not even exist on the Internet. When you are connected to the Internet and your computer can be not be detected via probes to your computer, you are in what is called Stealth mode. Hackers have the ability to detect if you are on the Internet by probing your machine with special

data and examining the results. When you are in Stealth mode the firewall does not send this information back making it seem like you are not even connected. Due to this hackers will not continue targeting your computer as they will think you are not online.

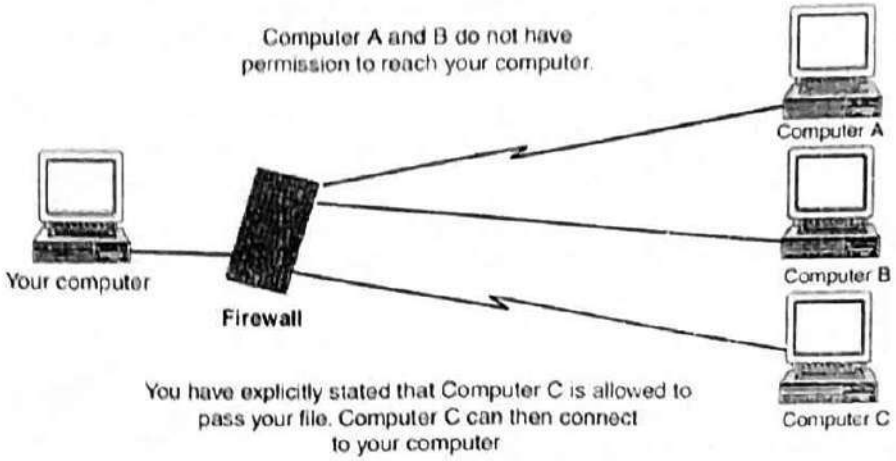


Fig. 4.3. Example of a Firewall Allowing a Remote Computer Access to a Computer Behind a Firewall.

3. Privacy Protection

Many firewalls now have the ability to block spyware, hijackers, and adware from reaching your computer. This allows you to protect your computer from being infected with software that is known to reveal private information about what you do on the Internet or other computing habits. These features are usually bundled into the commercial versions of the firewall software packages.

4. Application Integrity

Application integrity is when the firewall monitors the files on your computer for modification in the file or how they are launched. When it detects such a change it will notify the user of this and not allow that application to run or transmit data to the Internet. Many times these modifications may have been part of an upgrade, but if it was modified by a malicious program you will now be made aware of it.

5. Intrusion Detection

Intruders use various methods to penetrate the security of your computer. Intrusion detection scans incoming data for signatures of known methods and notifies you when such attacks are recognized. This allows you to see what means a hacker is trying to use to hack your computer.

6. Notifications

Notifications allow you to see the activity of what is happening on your firewall and for the firewall to notify you in various ways about possible penetration attempts on your computer.

4.4.3. Importance of Firewall

The Internet, like any other society, is plagued with the kind of jerks who enjoy the electronic equivalent of writing on other people's walls with spray paint, tearing their mailboxes off, or just sitting in the street blowing their car horns. Some people try to get real work done over the Internet, and others have sensitive or proprietary data they must protect. Usually, a **firewall's purpose is to keep the jerks out of our network while still letting us get our job done.**

Many traditional-style corporations and data centers have computing security policies and practices that must be followed. In a case where a company's policies dictate how data must be protected, a firewall is very important, since it is the embodiment of the corporate policy. Frequently, the hardest part of hooking to the Internet, if we are a large company, is not justifying the expense or effort, but convincing management that it's safe to do so. **A firewall provides not only real security—it often plays an important role as a security blanket for management.**

A firewall can act as your corporate "ambassador" to the Internet. Many corporations use their firewall systems as a place to store public information about corporate products and services, files to download, bug-fixes, and so forth. Several of these systems have become important parts of the Internet service structure and have reflected well on their organizational sponsors. Note that while this is historically true, most organizations now place public information on a Web server, often protected by a firewall, but not normally on the firewall itself.

4.4.4. Firewall Protection

Some firewalls permit only email traffic through them, thereby protecting the network against any attacks other than attacks against the email service. Other firewalls provide less strict protections, and block services that are known to be problems.

Generally, firewalls are configured to protect against unauthenticated interactive logins from the "outside" world. This, more than anything, helps prevent intruders from logging into machines on your network. More elaborate firewalls block traffic from the outside to the inside, but permit users on the inside to communicate freely with the outside. The firewall can protect us against any type of network-borne attack if we unplug it.

TIPS

Firewalls are also important since they can provide a single "choke point" where security and audit can be imposed. Unlike in a situation where a computer system is being attacked by someone dialing in with a modem, the firewall can act as an effective "phone tap" and tracing tool. Firewalls provide an important logging and auditing function; often they provide summaries to the administrator about what kinds and amount of traffic passed through it, how many attempts there were to break into it, etc.

Because of this, firewall logs are critically important data. They can be used as evidence in a court of law in most countries. We should safeguard, analyze and protect our firewall logs accordingly.

4.4.5. Limitations of Firewall

Firewalls can't protect against attacks that don't go through the firewall. Many corporations that connect to the Internet are very concerned about proprietary data leaking out of the company through that route. *For a firewall to work, it must be a part of a consistent overall organizational security architecture.* Firewall policies must be realistic and reflect the level of security in the entire network. For example, a site with top secret or classified data doesn't need a firewall at all: they shouldn't be hooking up to the Internet in the first place, or the systems with the really secret data should be isolated from the rest of the corporate network.

Another thing a firewall can't really protect us against is traitors or idiots inside our network. While an industrial spy might export information through our firewall, he's just as likely to export it through a telephone, FAX machine, or Compact Disc. CDs are a far more likely means for information to leak from our organization than a firewall. Firewalls also cannot protect

us against stupidity. Users who reveal sensitive information over the telephone are good targets for social engineering; an attacker may be able to break into our network by completely by-passing our firewall, if he can find a "helpful" employee inside who can be fooled into giving access to a modem pool.

Lastly, firewalls can't protect against bad things being allowed through them. For instance, many Trojan Horses use the Internet Relay Chat (IRC) protocol to allow an attacker to control a compromised internal host from a public IRC server. **If we allow any internal system to connect to any external system, then our firewall will provide no protection from this vector of attack.**

4.4.6. Technique of Firewall

Firewalls have evolved into several distinct types. The following passages will help clarify the functions of the most common types of firewalls in order to help us understand which basic features that we may specifically require. Once we are educated in firewall types then we should feel more confidence in deciding whether to purchase *firewall software vs. hardware*. Common types of firewalls are as follows :

(*i*) Application Gateways of Proxy Gateways.
(*ii*) Packet Filtering.
(*iii*) Circuit Gateways.
(*iv*) Stateful Packet Inspection.
(*v*) Internet Connection Firewall.
(*vi*) Hybrid Firewall.

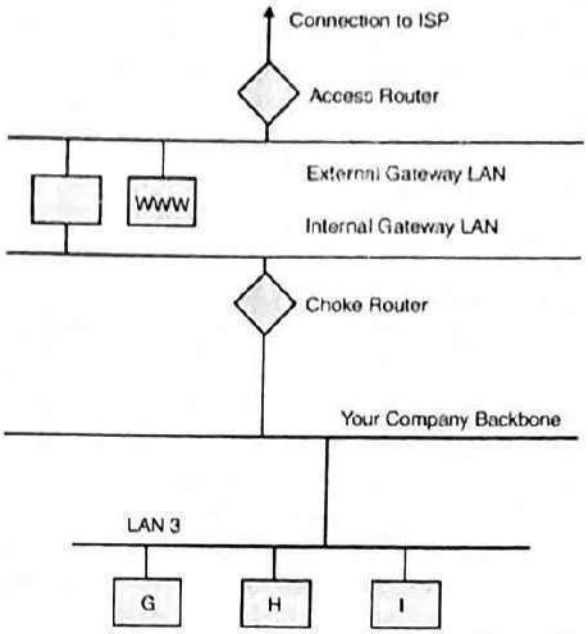


Fig. 4.4. A Sample Application Gateway.

4.4.6.1. Application Gateways or Proxy Gateways

The first firewalls were application gateways, and are sometimes known as proxy gateways. These are made up of hosts that run special software to act as a proxy server. This software runs at the *Application Layer* of our old friend the ISO/OSI Reference Model, hence the name. Clients behind the firewall must be *proxitized* (that is, must know how to use the proxy, and be configured to do so) in order to use Internet services. Traditionally, these have been the most secure, because they don't allow anything to pass by default, but need to have the programs written and turned on in order to begin passing traffic.

These are also typically the slowest, because more processes need to be started in order to have a request serviced. Figure 4.4 shows a application gateway.

4.4.6.2. Packet Filtering

TIPS

There is less overhead in packet filtering than with an application gateway, because the feature of access control is performed at a lower ISO/OSI layer (typically, the transport or session layer). Due to the lower overhead and the fact that packet filtering is done with routers, which are specialized computers optimized for tasks related to networking, a packet filtering gateway is often much faster than its application layer cousins.

Packet filtering is a technique whereby routers have *ACLs* (Access Control Lists) turned on. By default, a router will pass all traffic sent in and will do so without any sort of restrictions. Employing ACLs is a method for enforcing our security policy with regard to what sorts of access we allow the outside world to have to our internal network, and vice versa.

Figure 4.5 shows a packet filtering gateway.

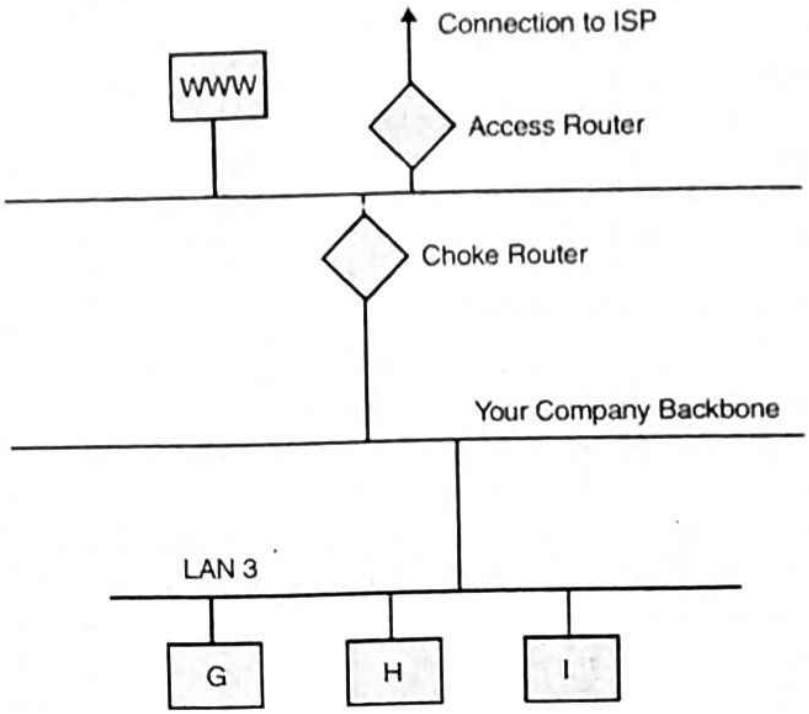


Fig. 4.5. A Sample Packet Filtering Gateway.

Because we're working at a lower level, supporting new applications either comes automatically, or is a simple matter of allowing a specific packet type to pass through the gateway.

There are problems with this method, though. Remember, TCP/IP has absolutely no means of guaranteeing that the source address is really what it claims to be. As a result, we have to use layers of packet filters in order to localize the traffic. We can't get all the way down to the actual host, but with two layers of packet filters, we can differentiate between a packet that came from the Internet and one that came from our internal network. **We can identify which network the packet came from with certainty, but we can't get more specific than that.**

4.4.6.3. Circuit Gateways

Circuit gateway firewalls work on the transport level of the protocol stack. They are fast and transparent, but really provide no protection from attacks. Circuit gateway firewalls also do not check the data in the packet. The one great benefit to this type of firewall is that they make the LAN behind the firewall invisible, as everything coming from within the firewall appears to have originated from the firewall itself. This is the least used type of firewall.

4.4.6.4. Stateful Packet Inspection

A fourth method that can be utilized by firewalls is called "Stateful Packet Inspection". It is called "Stateful" because it examines the contents of the packet to determine what the state of the communication is. It ensures that the stated destination computer has previously acknowledged the communication from the source computer. In this way all communications are initiated by the "receiving" computer and are taking place only with sources that are known or trusted from previous communication connections. In addition Stateful Packet Inspection firewalls are also more rigorous in their packet inspections. Stateful Packet Inspection firewalls also close off ports until an authorized connection is requested and acknowledged by the receiving computer. This allows for an added layer of protection from the threat of "port scanning" a method used by hackers to determine what PC services or applications are available to be utilized to gain access to the host computer.

4.4.6.5. Internet Connection Firewall

Windows XP provides Internet security in the form of the new Internet Connection Firewall (ICF). ICF makes use of active packet filtering, which means the ports on the firewall are opened for as long as needed to enable us to access the services we are interested in. The type of technology prevents hackers from scanning our computer's ports and resources. If we are hosting an Internet session, ICF allows us to open holes in the firewall that allow traffic on specific ports. This is called "port mapping."

4.4.6.6. Hybrid Systems

In an attempt to have the security of the application layer gateways with the flexibility and speed of packet filtering, some vendors have created systems that use the principles of both.

In some of these systems, new connections must be authenticated and approved at the application layer. Once this has been done, the remainder of the connection is passed down to the session layer, where packet filters watch the connection to ensure that only packets that are part of an ongoing (already authenticated and approved) conversation are being passed.

Other possibilities include using both packet filtering and application layer proxies. The benefits here include providing a measure of protection against your machines that provide services to the Internet (such as a public web server), as well as provide the security of an application layer gateway to the internal network.

4.5. ENCRYPTION AND DECRYPTION

When we think of encryption, we probably think of some sort of secret code that prevents others from reading our messages. While this privacy aspect of cryptography is important, it is only one of four aspects that are of particular importance in electronic communication.

(*i*) **Authentication** allows customers to be sure that the merchant they are sending their credit card details to is who they say they are. It can also allow merchant's to verify that the customer is the real owner of the credit card being used.

(*ii*) **Integrity** ensures that the messages have not been tampered with by a third party during transmission.

(*iii*) **Non-repudiation** prevents customers or merchants denying they ever received or sent a particular message.

(*iv*) **Privacy** prevents third parties from reading intercepted messages.

The translation of data into a secret code is known as encryption. It is the most effective way to achieve data security. To read an encrypted file, you must have access to a secret key or password that enables you to decrypt it. Unencrypted data is called plain text; encrypted data is referred to an cipher text. There are two main types of encryption: asymmetric encryption also called public key encryption and symmetric encryption.

4.5.1. Elements of an Encryption System

The main elements of an encryption system are the plaintext, the cryptographic algorithm, the key and the ciphertext.

(*i*) The **plaintext** is the raw message or data that is to be encrypted.

(*ii*) A **cryptographic algorithm or cipher** is a mathematical set of rules that defines how the plain text is to be combined with a key.

(iii) The **key** is a string of digits.

(iv) The **ciphertext** is the encrypted message.

These terms are probably best illustrated through a very simple example.

If we take the phrase "Web store" and add 2 characters to each letter the phrase becomes "ygd uvqtg". Here :

"Web store" is the plaintext

"add x characters to each letter" is the cryptographic algorithm

"2" is the key

"ygd uvqtg" is the ciphertext

There are two main types of encryption in common use today : **secret-key and public-key.**

4.5.2. Secret-Key Encryption Methods OR Symmetric Encryption

DEFINITION

Secret-key encryption, also known as single-key or symmetric encryption, involves the use of a single key that is shared by both the sender and the receiver of the message.

After creating the message, the sender encrypts it with their key and passes it to the recipient who then decrypts it by using a copy of the same key used to encrypt it.

A widely used method of secret-key encryption is **the data encryption standard or DES.**

Secret-key encryption does have some limitations, particularly with regard to key distribution. Let us see some of them as follows :

(*i*) For privacy to be maintained, every transmitter of messages would need to provide a different key to everyone they intended to communicate with, otherwise every potential recipient would be able to read all messages whether it was intended for them or not.

(*ii*) While this is manageable where a small number of parties are involved (for example, sending a private e-mail to a friend) it is not practical for Web commerce which can involve communicating with thousands of customers.

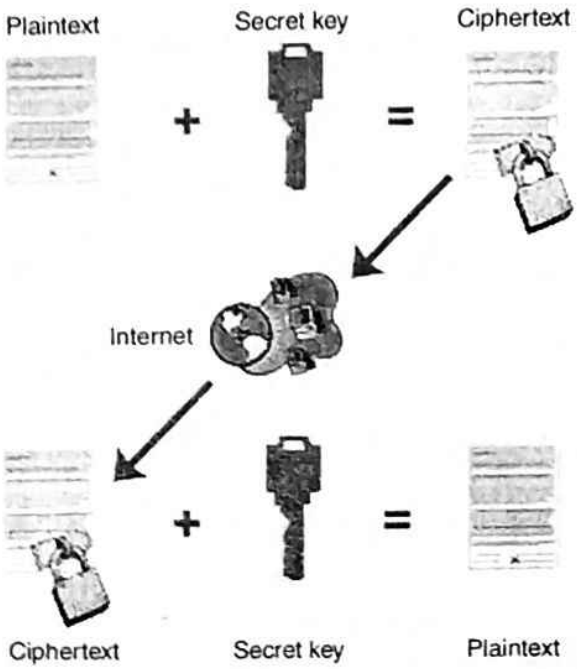


Fig. 4.6. Secret Key Encryption.

(*iii*) Another limitation with secret-key encryption is its inability to support non-repudiation. As both parties share the same key it is possible for one party to create a message with the shared secret key and falsely claim the other party had sent it.

(*iv*) **Secret-key encryption on its own is not suitable for Web commerce—instead a system known as public-key encryption is used.**

4.5.3. Public-Encryption Methods OR Asymmetric Encryption

Public-key encryption, or asymmetric encryption involves the use of two keys, one that can be used to encrypt messages (the public key) and one that can be used to either encrypt them or decrypt them (the private key).

These key pairs can be used in two different ways, to provide privacy or authentication.

Privacy is ensured by encoding a message with the public key as it can only be decoded by the holder of the private key.

TIPS

Authentication is achieved by encoding a message with the private key. Once the recipient has successfully decrypted it with the public key they can be assured it was sent by the holder of the private key.

As the public-key can be made widely available (from a server or third party), public-key cryptography does not suffer from the same key distribution and management problems as the secret-key system.

One disadvantage of the public-key system is that it is relatively slow, so when it is being used only for authentication it is not desirable to encrypt the whole message particularly if it is a long one. To get round this a digital signature is used.

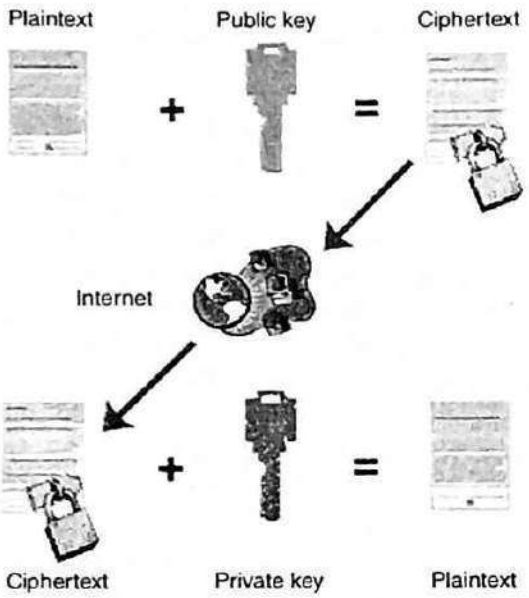


Fig. 4.7. Decryption with Private Key.

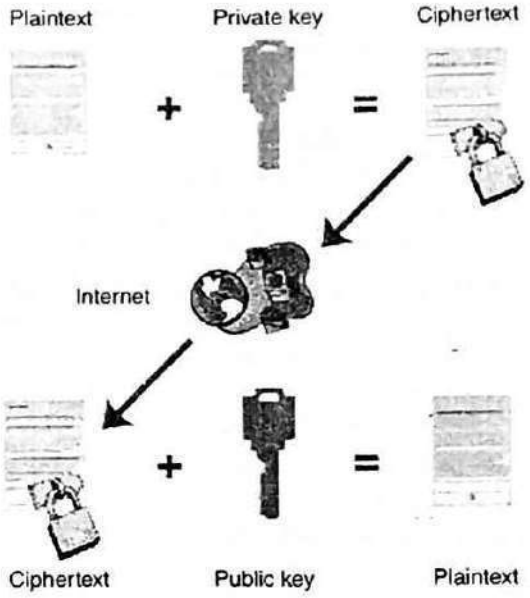


Fig. 4.8. Decryption with Public Key.

4.5.4. Symmetric Encryption vs. Asymmetric Encryption

Symmetric, or private-key, encryption (also known as conventional encryption) is based on a secret key that is shared by both communicating parties. The sending party uses the secret key as part of the mathematical operation to encrypt (or encipher) plain text to cipher text.

The receiving party uses the same secret key to decrypt (or decipher) the cipher text to plain text. Examples of symmetric encryption schemes are the RSA RC4 algorithm (which provides the basis for

Microsoft Point-to-Point Encryption (MPPE), Data Encryption Standard (DES), the International Data Encryption Algorithm (IDEA), and the Skipjack encryption technology proposed by the United States government (and implemented in the Clipper chip).

Asymmetric, or public-key, encryption uses two different keys for each user: one is a private key known only to this one user; the other is a corresponding public key, which is accessible to anyone.

The private and public keys are mathematically related by the encryption algorithm. One key is used for encryption and the other for decryption, depending on the nature of the communication service being implemented.

TIPS

In addition, public key encryption technologies allow digital signatures to be placed on messages.

A digital signature uses the sender's private key to encrypt some portion of the message. When the message is received, the receiver uses the sender's public key to decipher the digital signature to verify the sender's identity.

4.6. SECURE SOCKET LAYER (SSL)

DEFINITION

Secure Socket Layer (SSL) is a protocol designed and implemented by Netscape Communications. Netscape claims it is designed to work, as the name implies. At the socket layer, to protect any higher level protocol build on sockets, such as Telnet, FTP and HTTP.

Secure Socket Layer (SSL) technology allows web browsers and web servers to communicate over a secure connection. In this secure connection, the data that is being sent is encrypted before being sent and then is decrypted upon receipt and before processing. Both the browser and the server encrypt all traffic before sending any data. SSL addresses the following important security considerations.

(*i*) **Authentication:** During your initial attempt to communicate with a web server over a secure connection, that server will present our web browser with a set of credentials in the form of a server certificate. The purpose of the certificate is to verify that the site is who and what it claims to be. In some cases, the server may request a certificate that the client is who and what it claims to be (which is known as client authentication).

(*ii*) **Confidentiality:** When data is being passed between the client and the server on a network, third parties can view and intercept this data. SSL responses are encrypted so that the data cannot be deciphered by the third party and the data remains confidential.

(*iii*) **Integrity:** When data is being passed between the client and the server on a network, third parties can view and intercept this data. SSL helps guarantee that the data will not be modified in transit by that third party.

4.6.1. Use of SSL

To use SSL, an application server must have an associated certificate for each external interface, or IP address, that accepts secure connections. The theory behind this design is that a server should provide some kind of reasonable assurance that its owner is who you think it is, particularly before receiving any sensitive information. It may be useful to think of a certificate as a "digital driver's license" for an Internet address. It states with which company the site is associated, along with some basic contact information about the site owner or administrator.

10.1. INTRODUCTION OF SERVER SIDE SCRIPTING

Server side scripting is defined as "web server technology in which the user's request is fulfilled by running a script directly on a web server to generate dynamic web pages."

In simpler terms, a server-side scripts runs on the server rather than your computer. When you visit a website, the script will create the webpage on dynamically. In case of server side scripting, the client sends the request on the machine and then sends the result to the client machine.

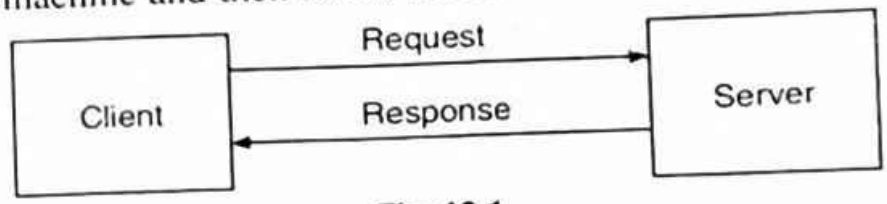


Fig. 10.1.

Popular server-side scripting languages include Perl PHP, ASP, JSP, Ruby, Cold fusion and Python.

Advantages:

1. Load times are generally faster than client side scripting.
2. It does not require the user to download plugins like JAVA or flash.
3. You can create a single website template for the entire website. Each new dynamic page you create will automatically use it.
4. Your scripts are hidden from view. Users only see HTML output, even when they view the source.

10.2. COMPONENTS OF SERVER SIDE SCRIPTING

Server side scripting is not only the collection of some programs that are executed on request, but it also contain some other below mentioned parts:

10.2.1. Web Server

A web server is a software that continuously monitors for any request that has been made from any client. It reads any request, it trees out the particular file or values from the database and sends the results to the client machine.

The web servers that are available these days are:

PWS (Personal Web Server): This web server is available for Window 98, 98 SE (Second Edition) and ME (Mellenium Edition).

IIS (Internet Information Server): This is available for Window NT (Server and workstation), 2000 (Server and Professional), XP, 2003 etc.

Apache: This web server is available for both windows and linux or Unix platforms.

Every web server creates a default web directory for storing its files *e.g.*, IIS creates.

```
{Root Directory}/inetpub/www root
```

where (Root Directory) is the name of the drive where the Window is installed.

Web server stores it's files in the particular web directory. The web server reads the request from the client (*i.e.*, browser) and sends the output to the client. Also, the particular web-directory is referred as local host from the browser if the server and client are running on the same machine.

10.2.2. Scripting Files

The second part of the server side scripting is the scripting files that are executed on the request of the client. These files are developed in any of the server side scripting languages that are available with us:

- ASP (Active Server Pages)
- JSP (Java Server Pages)
- PHP (Personal Home Pages)
- Servlets

These languages are having its own set of syntax to perform the specific tasks. Each language works under different requirements *e.g.*, ASP needs simply IIS installed and PHP can work with both IIS and Apache whereas ISP and servlets do require another software that work alongwith the web server so that JSP and servlets programs can be executed.

These softwares are like:

- Tomcat server
- Weblogics
- Apache Jserve etc.

10.2.3. Database Files

The third part of working of the server side scripting is the database files. These database files can be designed in any of the available database such that:

- MS Access
- SQL Server
- My SQL
- ORACLE

The job of the database is to store the information about various entities (any physically existing object is known as entity).

The job of the databases is to store the information about various entities (any physically existing object is known as entity).

A database is the collection of tables which inturn contains the records or rows of any existing object.

emp: Table

emp.no	*Name*	*Sol*
1.	ABC	1000
2.	XYZ	2000

Records

The database tables can be queried by using some standard SQL statements.

10.2.4. SQL

SQL is Structured Query Language that is used to interact with the databases. It is integral part of any Relational Database Management System (RDBMS) package system. SQL uses some predefined syntaxed statements to perform various database operations like.

10.2.5. Scripting Methods

The main scripting methods that are used in server side scripting are mentioned as under:

(*a*) ASP (*b*) JSP

(*a*) ASP: The active Server pages are the server side scripting language that executes on the windows platform only. When the IIS web server is installed on a machine, the ASP itself starts running on the computer (there is no need of installing the special compiler or parser for the ASP programs. The ASP code is developed by using the below mentioned syntax in the

```
<%
[                                ]  ASP code to perform a test
%>
```

HTML files ASP code is stored in the root directory and have a special extension *i.e.*, asp. In order to execute the files or scripts, the script is stored in the root directory (web server directory) and then referred from the client machine or browser by putting the following address in the address bar:

http: \\ local host\test.asp

The above mentioned address can execute the test.asp file stored in the root directory of the computer.

(*b*) JSP: Java Server Pages is the other scripting language that one is using instead of using the ASP. It is branch of java language. The knowledge of Java language is must before using the JSP. Without Java, one is not able to use JSP.

It works on the below mentioned Architecture.

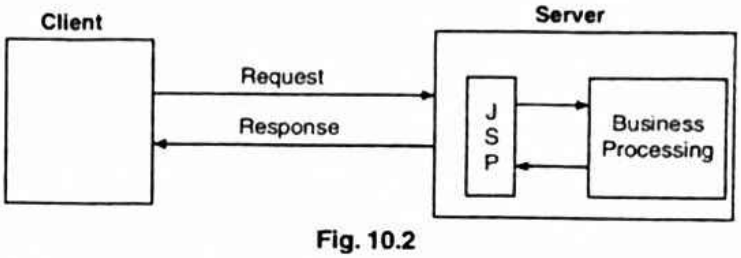


Fig. 10.2

JSP needs a special type of software package in order to run properly. This software package is always working in conjunction with the web server. One such software is known as Tomcat server.

10.3. DIFFERENCE IN CLIENT-SIDE AND SERVER SCRIPTING

Client-side scripting generally refers to the class of computer programs on the web that are executed client-sides by the user's web browser. Instead of server-side, client-side scripts are often embedded within an HTML document, but they may also be contained in a separate file, which is referred by the document that uses it. Client side scripting is used to change the interface behaviours within a specific web page in response to mouse or keyboard actions or at specified time events.

Server-side scripting is a next server technology in which a user's request is fulfilled by running a script directly on the web server to generate dynamic to the server, the server processes those requests HTML pages. In case of server side scripting, the client sends the requests on the machine and then sends the result to the client machine.

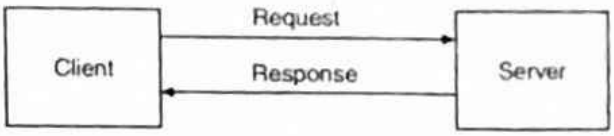


Fig. 10.3

With Client-Side Scripting, the scripts on the page are processed by the individual web browser that requests the page. **With Server-Side Scripting,** on the other hand, all scripts are processed on the server before the requested page is ever sent to the browser. This results in no actual code being sent to the client's machine.

Contrary to Server-Side scripts, Client-Side scripts can only run on browsers that support the specific scripting language that you are using. This means that you would need Netscape Navigator to view fully functioning JavaScript. On the contrary, you would need Internet Explorer to view fully functioning VB Script.

Server-Side Scripting has made it possible to "create platform-independent, easily deployable applications". The thought of creating a program or application that will run anywhere in the world has obvious advantages over Client-Side Scripting.

10.4. REMOTE SCRIPTING

There is a third kind of scripting which makes use of both client and server side technologies. Remote scripting takes the best of both worlds, the speed of client side scripts with the flexibility of server side code; however, it also creates its own problems because of the way it works.

Remote scripting works by allowing a client side script access to server side functions, with the functions being processed by the server. This allows client side code to use scripting technologies that wouldn't normally be available to them, while also calling server generated responses without the need to refresh the page.

Unfortunately the browser restrictions of the client side still apply. Microsoft's version of remote scripting uses Java for its functionality, which is quite ironic seeing how Windows XP doesn't come out of the box with Java support! There is also another JavaScript version of remote scripting that has been developed, which it could be argued now has a wider "active window of availability" than the Microsoft version.

10.5. SERVER SIDE SCRIPTING METHOD

Let us see some of the server side scripting methods :

1. **CGI (Common Gateway Interface) :** Simple scripting mechanisms supported by practically all web servers. Most commonly written in Perl, C/C++, or Python.

2. **Java Servlets :** Ability to execute Java objects on the server. Newer, less common technology but has dramatic performance advantages over CGI.

10.6. JAVASCRIPT ON THE SERVER

JavaScript extends the capabilities of the server. By providing a scripting language the software can do more without calling an external program. This makes it easier for Webmasters to add features to their sites that browsers can take advantage of. And it can reduce the load on the server by keeping the processing within the server software.

10.6.1. Request Object

Each time a browser wants more information it sends a request to a server. Data about these requests are available on the server from the request object in JavaScript. If you have written CGI scripts, you should be familiar with the properties that are built into the request object.

- **The Built-In Properties :** The request object is initialized with four properties: agent, ip, method, and protocol. A JavaScript function on the server can use this information to determine its response to the request. The following list provides details about each property.
  - (*i*) **Agent :** This property tells the server about the client's software, usually a browser. This information is a line of text, but there is no standard format to this text. The same browser always presents the same information. For example, if you use Netscape Navigator 2.0 and your friend uses Netscape Navigator 2.0, the same information is sent by either of you when you access this server.
  - (*ii*) **ip :** The ip property provides the IP address of the client. This tells the server from where on the Internet the request is sent and where to return a response. IP addresses are composed of a set of four numbers. Each is a value between 0 and 255. A sample IP number is 127.0.0.1.
  - (*iii*) **Method :** This property provides the HTTP method associated with the request. There are three methods currently used: get, post, and head.
  - (*iv*) **Protocol :** This is a standard text indicating the level of HTTP protocol used by the client. Currently this should be HTTP/1.0 in every case.
- **Other Request Properties :** A request may include additional information. Your browser's request may have resulted from submitting information from an HTML form. This information becomes a request property. It does not matter if the data is submitted with either the post or get method. The name of each element of the form becomes a property name.
- **Application of Request Properties:** These built-in properties allow the server to vary the response to the request. The agent property is commonly used to supply different HTML documents to different types of browsers. For example, an HTML page with tables might only be served to a limited number of browsers while others receive a different version of the page.

Servers may limit access to certain information to specific IP addresses. Using the ip property, the server might only provide sensitive company information to clients requesting it from known company computers. Usually you want additional security, just screening IP addresses is not enough.

10.6.2. Client Object

Each time a new client accesses the server application, a new client object is created. However, there are no built-in properties for the client object. If you need information retained about the client, then you create a property.

- **Client Properties:** A very common use for a client property is a client identification number. When the client first accesses the server, he can receive a form. The server can then process this information and assign the client an identification number. This might be randomly generated or looked up from a list of previous customers.

  As an example, a client can send a request. Part of the request can include a user supplied customer number in the text box named idnumber. You can then create a client property called custNo with the following line :

  ```
  Client.custNo = Request.idnumber
  ```

- **Client Objects Expire :** The server overflows with client objects unless they are properly deleted. Since client objects are automatically created for each and every client to access, then there must be a mechanism for deleting them.

LiveWire automatically deletes any client object with no property values. So if you don't use a client object, you don't have to worry about deleting it. In other words, you only have to clean up after yourself.

The default expiration is ten minutes of inactivity. If the client does not send another request to the server within ten minutes of the previous request, the object expires.

Obviously, in many cases ten minutes is insufficient time and, therefore, you might want to manually control this. You can simply use the expiration statement

```
client.expiration(seconds)
```

where seconds is the number of seconds before the client expires.

Another manual control is destroy. If you no longer need the client object, simply use the statement

```
client.destroy()
```

LiveWire looks at the client that sent the request and destroys its client object. This eliminates all of the client object's properties as well.

- **Cookies Store Information between Sessions:** Another technique for retaining client information is cookies. The browser must support the Netscape cookie protocol. If supported, the server sends the information to the client as name/value pairs. Obviously, this increases network traffic, but can offer substantial advantages to large access servers.

10.6.3. JavaScript and CGIs

Undoubtedly with your active interest in building websites, you have dealt with Common Gateway Interface (CGI) scripts. Prior to JavaScript, this was the primary means of creating interactive applications. Libraries of CGI scripts include counters, e-mailers, message boards, and many other functions.

LiveWire can replace CGI programming. Instead of calling external programs, the server software runs applications that are closely integrated to it. JavaScript is the language of these applications.

Applications are developed with three tools. You build these applications using LiveWire's Site Manager. The source files for the applications are developed using the same HTML editors used to build browser JavaScript pages. The applications run in response to requests from Netscape Navigator.

10.6.4. The JavaScript Balancing Act

When interactivity was only done with a CGI, all of the processing was done on the server. Because JavaScript can run on both the server and the browser, writing a successful application requires you to properly allocate the processing between the two.

1. **Browser Handle the Work :** In general, let the browser code "polish" the information before sending it to the server. The following are several different tasks best handled by the browser :

- Validate user inputs
- Check that the values are within range
- Prompt for confirmations
- Verify inputs that are valid, but not usual
- Perform aggregate calculations (totals, averages, means, sales tax)
- Conditionalize HTML
- Perform other functions that require no server data

2. **More Work for the Server :** Though the browser code can take much of the burden off the server, the server still must perform. Given the increasingly interactive nature of Web applications, the demands on the server are increasing. The following types of tasks should be performed by the server :
   - Direct the flow from one page to another
   - Maintain data about the client from one request to the next
   - Maintain the shared data among clients and applications
   - Access various databases
   - Access various files on the server (for example, multimedia)
   - Call C libraries, as needed
   - Dynamic customization of Java applets

Obviously, the server is a busy fellow. And this load is expected to grow dramatically as servers provide multimedia material and more interactivity.

10.6.5. Exchanging Information between the Browser and Server

In making your application come alive, the server and browser must exchange information. The client, or browser, typically sends user responses. These can first be "polished" by JavaScript routines on the browser side. The server in turn sends data back to the browser as HTML pages.

From Client to Server : User responses are submitted just as you currently handle forms. The user completes the form and clicks the submit button. The data from the radio buttons, checkboxes, textboxes, and textarea are sent to the server. The server then places this data into the request object. Each element of the form has a corresponding property.

From Server to Client : Usually a server only returns a static page in response to a browser request. In LiveWire, the response is still a page, but the contents of the page vary. User input can result in changes to default form values, new values of hidden form elements, or direct substitutions.

A server-side JavaScript can dynamically build in the HTML code for a form element that is part of the page. As an example, you can have the following statement in a source document:

```
<INPUT TYPE="text"  NAME="example" VALUE='request.agent'>
```

In this case the default value of the text is the browser agent information.

You use an identical procedure for hidden form elements. The only difference is that the type is hidden instead of text. Your client-side JavaScript can then use this value as part of any function.

Another means of making your pages come alive is to change part of the JavaScript code. When you send a page to the browser it can contain JavaScript code for the browser to execute as part of the page. There is no reason that this code has to be static.

The server can modify the page being sent to change the JavaScript code embedded in the page. The page on the server side might include

```
<SERVER>
write ("<SCRIPT>var luck = " + client.winnings + "</SCRIPT>")
</SERVER>
```

Assuming the value of client.winnings is 1000, the browser sees a line of

```
<SCRIPT>var luck = 1000 </SCRIPT>
```

10.7. SQL (STRUCTURED QUERY LANGUAGE)

It is used to interact with the databases. It is integral part of any Relational Database Management System (RDBMS) package system. SQL uses some predefined syntaxed statements to perform various database operations like.

10.7.1. Retrieving Information from a Table

SQL uses the select statement for this purpose.

Syntax :

```
Select column names    /* from tablenames;
```

where;

(*i*) Column names is the list of columns (separated by (,) comma) or * for listing all columns.

(*ii*) and Tablename is the name of the table.

Hence, the statement can be written as:

```
Select * from emp;
```

and the above statement retrieves all the columns and rows from the emp table.

10.7.2. Inserting Records into a Table

For inserting rows or records into the database tables, the SQL uses following syntax:

```
insert into tablename values (set of values);
```

Example: To insert any record in a table having name emp and having field empno and sol, we can issue a SQL statement:

```
insert into emp values (1, 1000)
```

10.7.3. Deletion of Records

This SQL statement is used for the deletion of the records from a table. It uses the following syntax:

```
Delete from tablename;
```

Example: If one wants to delete all the records present in the emp table, the SQL statement would be like

```
Delete from emp;
```

10.7.4. Updation of Records

For updating the already existing records:

```
Update tablename set column name = new value;
```

All of there parts *i.e.*, web server, scripting files and databases files work together to perform the server side scripting property. The figure given below shows the working:

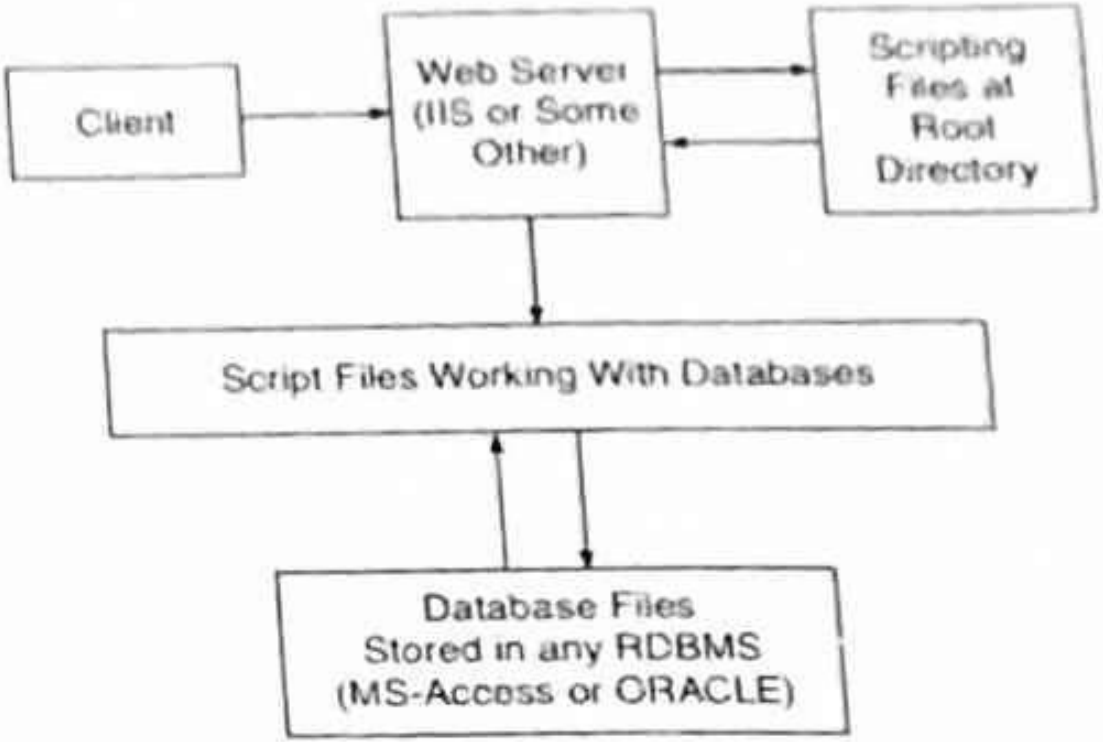

Client
Web Server (IIS or Some Other)
Scripting Files at Root Directory
Script Files Working With Databases
Database Files Stored in any RDBMS (MS-Access or ORACLE)

Fig. 10.4

11.1. INTRODUCTION

PHP is an open-source server-side scripting langauge. PHP version 5.2.14 is available on web server (WWW.indiana.edu, WWW.iue.edu). You can create dynamic web pages with the PHP scripting language. A dynamic web page interacts with the user, so that each user visiting the page sees customized information. PHP can also be used to create dynamic web pages that are generated from information accessed from a MySQL database. You can embed PHP commands within a standard HTML page. PHP's syntax is similar to that of C and perl, making it easy to learn for anyone with basic programming skills another feature that PHP offers is connectivity to most of the common databases. PHP also offers integration with various external libraries, which allow the develops to do anything from generating PDF documents to parsing XML.

Syntax: Place a file containing the code below inside the WWW directory of your web server account and save it as info.php

```
<? php
phpinfo( );
? >
```

To view the file, you would then provide the website address to the file.

Syntax:

```
http : //www.indiana.edu/~account/info.php
```

It will output information about the current state of PHP, which includes information about PHP compilation options and extensions, the PHP version, server information and environment (if compiled as a module), the PHP environment, OS version information, paths, master and local values of

configuration options, HTTP headers, and the PHP license. Because every system is setup differently, phpinfo() can be used to check configuration settings and for available predefined variables on a given system.

11.2. VARIABLES IN PHP

Sometimes it is convenient to be able to have variable names. Variables in PHP are represented by a dollar sign followed by the name of variable. The variable name is case-sensitive.

Variable names follow the same rules as other labels in PHP. A valid variable name starts with a letter or underscore, followed by any number of letters, numbers, or underscores.

For example,

```
<? php
$ var = 'Bob';
$ var = 'joe';
echo "$var, $var";      //outputs "Bob, joe"
$4site = 'not yet';     //invalid; starts with numbers
$_4site = 'not yet';    //valid; starts with an underscore
$tayte = 'manikka';     //valid; 'a' is (extended) ASCII 228
? >
```

By default, variables are always assigned by value. That is to say, when you assign an expression to a variable, the entire value of the original expression is copied into the destination variable. This means, for instance, that after assigning one variable's value to another changing one of those variables will have no effect on the other.

11.3. STRINGS

A string is series of characters where a character is the same as a byte. This means that PHP only supports a 256 character set, and hence does not offer native unicode support.

NOTE

String can be as large as upto 2 GB (2147483647 bytes maximum)

A string literal can be specified in four different ways:

1. Single Quoted

The simplest way to specify a string is to enclose it in single quotes (the character ').

To specify a literal single quote, escape it with a backslash (\). To specify a literal backslash, double it (\\). All other instances of backslash will be treated as a literal backslash; this means that the other escape sequences you might be used to, such as /r or /n, will be output literally as specified rather than having any special meaning.

For example

```
< ? php
echo 'this is a simple string';
? >
```

2. Double Quoted

If the string is enclosed in doubel quotes (" "), the PHP will interpret more escape sequences for special characters:

Escaped Characters

Sequence	Meaning
\n	linefeed
\r	Carriage return
\t	horizontal tab
\v	vertical tab
\e	escape sequence
\f	form feed
\\	backslash
\$	dollar sign
\"	double quote
\[0-1] {1, 3}	The sequence of characters matching the regular expression is a character in octal notation.
\x[0-9A-Fa-f] {1, 2}	The sequence of characters matching the regular expression is a character in hexadecimal notation.

As in single quoted strings, escaping any other character will result in the backslash being printed too. Before PHP 5.1.1, the backslash in \{$var} had not been printed.

The most important feature, of double-quoted strings is the fact that variable names will be expanded

3. Heredoc

A third way to delimit strings is the heredoc syntax : <<<. After this operator, an identifier is provided then a newline. The string itself follows, then the same identifier again to close the quotation.

The closing identifier must begin in the first column of the line. Also, the identifier must follow the same naming rules as any other label in PHP, it must contain only alphanumeric characters and underscores, and must start with a non-digit character or underscrore.

Here doc text behaves just like a double quoted string, without the double quotes. This means that quotes in a heredoc don't need to be escaped, but the escape codes listed above can still be used. Variables are expanded, but the same care must be taken when expression complex variables inside a heredoc as with strings.

For example,

Example#1

Heredoc in arguments: It is also possible to use the heredoc syntax to pass data to function arguments:

```
<?php
var_dump (array(<<<EDD
footbar!
EDD
));
?>
```

Example 2. Using double quotes in Heredoc

Starting with PHP 5.3.0, the opening heredoc identifier may optionally be enclosed in double quotes:

```
<? php
echo <<<"FOOBAR"
Hello world!
FOOBAR;
?>
```

4. NOWdoc

NOWdocs are to single-quoted strings what heredocs are to double-quoted strings. A nowdoc is specified similarly to a heredoc; but no parsing is done inside a nowdoc. The construct is ideal for embedding PHP code or others large blocks of text without the need for escaping.

A NOWdoc is identified with same <<< sequence used for heredoces, but the identifier which follows is enclosed in signel quotes, *e.g.*, <<<'EOT'. All the rules for heredoc identifier also apply to nowdoc identifier, especially those regarding the appearance of the closing identifier.

Example:

```
static data example
<? php
class foo {
    public $bar = <<<'EOT'
    bar EOT;
        }
        ?>
```

11.4. PHP OPERATOR TYPE

What is Operator ?

Simple answer can be given using expression 4 + 5 is equal to 9. Here 4 and 5 are called operands and '+' is called operator. PHP language support following type of operator.

1. Arithmetic Operators

There are following arithmetic operators supported by PHP language:

Assume variable A and B holds value 5 and 10 respectively then

Show Example

Operator	Description	Example
+	Adds two operands	A + B will give 15
*	Multiply both operand	A*B will get 50
–	Subtracts second operand from the first	A – B will get –5
/	Divide numerator by denumerator	B/A will get 2
%	Modulus operator and remainder of after an integer division	B%A will give 0
++	Increment operator, increases integer value by 1	A++ give 6
– –	Decrement operator, decreases: integer value by one	A – – give 4

2. Comparison Operators

There are following comparison operators supported by PHP language. Assume variable A holds 10 and variable B holds 10 then:

Operator	Description	Example
= =	Checks if the value of two operands are equal or not, if yes then condition becomes true.	(A = = B) is not true.
! =	Checks if the value of two operands are equal or not, if values are not equal then condition becomes true	(A! = B) is true.
>	Checks if the value of left operand is greater then the value of right operand, if yes then condition becomes true	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater then or equal to the value of right operand, if yes then condition becomes true	(A > =B) is not true.
<=	Checks if the value of left operant is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

3. Logical Operator

There are following logical operators supported by PHP languae. Assume variable A holds 10 and B holds 20 then:

Show Examples:

Operator	Description	Example
AND	Called logical AND operator. If both the operands are true then condition becomes true.	(A and B) is true.
OR	Called logical OR operator. If any of the two operands are non-zero then condition becomes true.	(A or B) is true.
%%	Called logical AND operator. If both the operands are non-zero then condition becomes true.	(A && B) is true.
ll	Called logical OR operator. If any of the two operands are non-zero then condition becomes true.	
!	Called logical NOT operator, use to reverse the logical state of its operand. If a condition is true then logical NOT operator will make false.	!(A && B) is false

4. Assignment Operators

There are following assignment operators supported by PHP language:

Examples:

Operator	Description	Example
=	Simple assignment operator, assigns values from right side operands to left side operand	C = A + B, will assign value of A + B into C
+=	Add AND assignment operator, it adds right operand to the left operand and assign the result to the left operand	C += A is equivalent to C = C + A
-=	Subtract AND assignment operator, it subtract right operand from the left operand and assign the result to the left operand	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator, it multiply right operand to the left operand and assign the result to the left operand	C *= A is equivalent to C = C * A
/=	Divide AND assignment operator, it divides left operand with the right operand and assign the result to left operand	C /= A is equivalent to C = C/A
%=	Modulus AND assignment operator, it takes modulus using two operands and assign the result to left operand.	C% = A is equivalent to C = C % A

5. Conditional Operator

There is one more operator called conditional operator. This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation. The conditional operator has this syntax.

Show Example:

Operator	Description	Example
?:	Conditional expression	If condition is true ? then value x otherwise value y

Operator Categories

All the operators we have discussed above can be categorised into following categories:

- Unary prefix operators, which precede a single operand.
- Binary operators, which take two operands and perform a variety of arithmetic and logical operations.
- The conditional operator (a ternary operator), which takes three operands and evaluates either the second or third expression, depending on the evaluation of the first expression.
- Assignment operators, which assign a value to a variabel.

Precedence of PHP Operators

Operator precedence determines the grouping of terms in an expression. This affects how an expression in evaluated certain operator have higher precedence than others for example, the multiplication operator has higher precedence than addition operator so multiple is performed before the addition operation. Within an expression, higher precedence operators will be evaluated first.

Category	Operator	Associativity
Unary	!++ – –	Right to left
Multiplicative	* / %	Left to right
Additive	+ –	Left to right
Relational	<<= >>=	Left to right
Equality	= = !=	Left to right
Logical AND	&&	Left to right
Logical OR	ll	Left to right
Conditional	?:	Right to left
Assignment	= += –= *= /= %=	Right to left

11.5. PHP DECISION MAKING

The if, elseif_else and switch statements are used to take decision based on different condition.

You can use conditional statements in your code to make your decisions. PHP supports following three decision making statements:

The if .... else statement

If you want to execute some code if a condition is true and another code if a condition is false, use the if ... else statement.

Syntax:

```
If (condition)
    Code to be executed if condition is true;
else
    Code to be executed if condition is false;
```

Example: The following example will output "Have a nice weekend?" If the current day is Friday, otherwise, it will output "Have a nice day!"

```
<html>
<body>
<? php
$d = date ("D");
If ($d = "Fri")
echo "Have a nice weekend!";
else
echo "Have a nice day!";
    ?>
    </body>
        </html>
```

The Elseif Statement

If you want to execute some code if one of several conditions are true use the elseif statement.

Syntax:

```
If (condition)
Code to be executed if condition is true;
elseif (condition)
Code to be executed if condition is true;
else
code to be executed if condition is false;
```

For example, the following example will output "Have a nice weekend!" If the current day is Friday, and "Have a nice Sunday" If the current day is Sunday, otherwise it will output "Have a nice day".

```
<html>
<body>
<?php
$d = date ("D");
If ($d = = "Fri")
echo "Have a nice weeked!";
elseif ($d = "Sun")
echo "Have a nice sunday";
else
echo "Have a nice day!";
?>
</body>
</html>
```

The Switch Statement

If you want to select one of many blocks of code to be executed, use the switch statement the switch statement is used to avoid long blocks of if ... elseif ... else code.

Syntax:

```
Switch (expression)
{
case label1:
code to be executed if expression = label1;
break;
case label2:
code to be executed if expression=label2;
break;
default:
code to be executed if expression is different from both label1 and label2;
}
```

Example: The switch statement works in an unusual way. First it evaluate given expression then seeks a label to match the resulting value. If a matching value is found then the code is associated with the matching label will be executed or if none of the labels match then statement will execute any specified default code:

```
<html>
    <body>
    <?php
$d = date ("D");
    switch ($d)
    {
    case "Mon":
    echo "Today is Monday";
    break;
case "The":
echo "Today is Tuesday";
    break;
case "Wed":
echo "Today is Wednesday";
    break;
    case "Thur":
echo "Today is Thursday";
    break;
case "Fri":
echo "Today is Friday";
case "Sat":
echo "Today is saturday";
    break;
case "Sun":
echo "Today is Sunday";
    break;
default:
    echo "wonder which day is this?";
}
    ?>
    </body>
        </html>
```

11.6. PHP LOOP TYPES

Loops in PHP are used to execute the same block of code a specified number of times. PHP supports following four loop types:

- **For:** Loops through a block of code a specified number of times. The for statement is used when you know how many times you want to execute a statement or a block of statements.

Syntax:

```
for (initialization; condition; increment/decrement)
{
    code to be executed;
    }
```

The initializer is used to set the start value for the counter of the number of loop iterations. A variable may be declared here for this purpose and it is traditional to name it $i.

Example: The following example makes five iterations and changes the assigned value of two variables on each pass of the lop:

```
<html>
    <body>
    <?php
    $a = 0;
    $b = 0;
for ($i = 0; $i < 5; $i++)
    {
    $a += 10;
    $b += 5;
}
echo ("At the end of loop a = $a and b = $b");
  ?>
  </body>
  </html
```

This will produce the following result.

At the end of loop a = 50 and b = 25.

While Loop

Loops through a block of code if and as long as a specified condition is true. If the test expression is true then the code block will be executed. After the code has executed the test expression will again be evaluated and the loop will continue until the test expression is found to be false.

Syntax:

```
While (condition)
{
code to be executed;
}
```

Example: This example decrements a variable value on each iteration of the loop and the counter increments until it reaches 10 when the evaluation is false and the loop ends.

```
<html>
<body>
<?php
$i = 0i
$num = 50;
while($ i<10)
    {
    $num - -i
    $i++;
    }
```

```
echo ("Loop stopped at i = $i and num = $num");
    ?>
    </body>
    </html>
```

This will produce following result:

```
Loop stopped at i = 10 and num = 40
```

The do .... while loop statement: It will execute a block of code at least once—it then will repeat the loop as loop as a condition is true.

Syntax:

```
do
{ code to be executed;
} while (condition);
```

Example: The following example will increment the value of i at least once, and it will continue incrementing the variable i as long as it has a value of less than 10.

```
<html>
<body>
<?php
$i = 0;
$num = 0;
do
{
    $i++;
}
while ($i < 10);
echo ("Loop stopped at i = $i");
    ?>
    </body>
    </html>
```

This will produce following result

```
Loop stopped at i = 10
```

The foreach Loop Statement

The foreach statement is used to loop through arrays. For each pass the value of the current array element is assigned to $value and the array pointer is moved by one and in the next pass next element will be processed.

Syntax:

```
foreach(array as value)
{
    code to be executed;
}
```

Example:

```
<html>
    <body>
    <? php
    $array = array (1, 2, 3, 4, 5);
    foreach($ array as $ value)
    {
    echo "value is $value <br/>";
    }
    ?>
    </body>
</html
```

This will produce following result:

```
value is 1
value is 2
value is 3
value is 4
value is 5
```

11.7. PHP ARRAYS

Array is a data structure that store one or more simiklar type of values in a single value.

For example: If you want to store 100 numbers then instead of denining 100 variables, its easy to define an array of 100 length.

There are three different kind of array and each array value is accessed using an IDs which is called array index.

1. Numeric Array

An array with a numeirc index. Values are stored and accessed in linear fashion. These array can store numbers, strings and any object but their index will be presented by numbers. By default array index starts from zero.

Example: Here we have used array ( ) function to create array.

```
<html>
<body>
<? php
/* first method to create array */
$ numbers = array (1 2 3 4 5);
    foreach($number as $value)
    {
    echo "value is $value <br/>";
    }
/* second method to create array */
    $numbers[0] = "One";
```

```
$numbers[1] = "Two";
$numbers[2] = "Three";
$numbers[3] = "Four";
$numbers[4] = "Five";
foreach ($numbers as $value)
{
echo "value is $value <br/>";
}
?>
</body>
</html>
```

This will produce following result:

```
Value is 1
Value is 2
Value is 3
Value is 4
Value is 5
Value is one
Value is two
Value is three
Value is four
Value is five
```

2. Associative Arrays

An array with strings as index. This stores element value in association with key values rather than a strict linear index order. These arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values.

Example: To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead we could use employees names as the key in our associative array, and the value would be their respective salary.

```
<html>
<body>
<?php
/*First method to associate create array*/
$salaries = array(
        "Mohammad"=> 2000,
        "Zara"=> 1000;
        "Pakhi"=> 10000;
        "Vinay"=> 50000
        );
echo "Salary of mohammad is". $salaries ['mohammad'] . "<br/>";
echo "Salary of Zara is".$salaries['Zara']. "<br/>";
```

```
echo "Salary of Paakhi is".$salaries ['Paakhi']. "<br/>";
echo "Salary of Vinay is".$salaries ['Vinay']."<br/>";

    /* Second Method to create array */
        $Salaries ['Mohammad'] = "medium";
        $Salary ['Zara'] = "Very low";
        $Salary ['Paakhi'] = "High";
        $ salaries ['Vinay'] = "Very light";
echo "Salary of mohammad is".$salaries ['mohammad']."<br/>";
echo "Salary of Zara is".$salaries ['Zara']."<br/>";
echo "Salary of Paakhi is".$salaries ['Paakhi']."<br/>";
echo "Salary of Vinay is".$salaries ['Vinay']. "<br/>;
    ?>
    </body>
        /html
```

This will produce following result:

```
Salary of mohammad is 2000
Salary of Zara is 1000
Salary of Paakhi is 10000
Salary of Vinay is 50000
Salary of Mohammad is medium
Salary of Zara is very low
Salaryof Paakhi is high
Salary of Vinay is very high
```

3. Multidimensional Array

An array containing one or more arrays and values are accessed using multiple indices. A multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and so on. Values in the multi-dimensional array accessed using multiple index.

Example: In this example, we create a two dimensional array to store marks of three students in three subjects:

```
<html>
<body>
<?php
    $marks = array(
        "Mohammad" => array
        (
        "Physics" => 35
        "Maths" => 30
        "Chemistry" => 39
        );
        "Paakhi" => array
```

```
(
"Physics" => 40
"Maths" => 42
"Chemistry" => 45
);
"Vinay" => array
(
"Physics" => 41
"Maths" => 42
"Chemistry" => 49
)
);
/*Accessing multi-dimensional array values*/
echo "Marks for mohammad in physics:";
echo "$Marks ['mohammad'] ['physics']. "<br/>";
echo "Marks for Paakhi in Maths:";
echo "Marks ['Paakhi'] ['Maths']. "<br/>";
echo "Marks for Vinay in Chemistry:";
echo "$marks['Vinay'] ['Chemistry']. "<br/>";
    ?>
    </body>
        </html>
```

This will produce result:

```
Marks for Mohammad in Physics: 35
Marks for Paakhi in Maths : 42
Marks for Viney in Chesmitry: 49
```

11.8. PHP GET AND POST METHODS

There are two ways the browser client can send information to the web server.

- The GET method
- The POST method

Before the browser sends the information, it encodes it using a scheme called URL encoding. In this scheme, name/value pairs are joined with equal signs and different pairs are separated by the ampereand

```
name1 = value1 & name2 = value2 & name3 = value3
```

Spaces are removed and replaced with the + character and any other nonalphanumeric characters are replaced with a hexadecimal values. After the information is encded it is sent to the server.

The GET Method

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the ? character.

```
http://www.test.com/index.htm?name1=value1&name2=value2
```

- The GET method produces a long string that appears in your server logs, in the browser's location box.
- The GET method is restricted to send upto 1024 characters only.
- Never use GET method if you have password or other sensitive information to be sent to the server.
- GET can't be used to send binary data, like images or word documents, to the server.
- The data sent by GET method can be accessed using QUERY_STRING environment variable.
- The PHP provides $_GET associative array to access all the sent information using GET method.

Example:

```
<?php
If ($_GET ["Name"] || $_GET ["age"])
    {
    echo "Welcome". $_GET['name']."<br/>";
echo "you are". $_GET['age']. "years old";
    exit( )
    }
?>
<html>
    <body>
<form action = "<?php $_PHP_SELF ?>" method = "GET">
Name : <input type = "text" name ="name"/>
Age : <input type = "text" name = "age"/>
<input type = "submit"/>
</form>
</body>
</html>
```

The POST Method

The POST method transfers information via HTTP headers. The information is encoded as described in case of GET method and put into a header called QUERY_STRING.

- The POST method does not have any restriction on data size to be sent.
- The POST method can be used to send ASCII as well as binary data.
- The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using secure HTTP you can make sure that your information is secure.
- The PHP provides $-Post associative array to access all the sent information using POST method.

Example:

```
<?php
If ($_POST ["name"] || $_POST["age"];
{
{
```

```
echo "Welcome". $_POST['name'] "<br/>";
echo "you are. $_POST['age']. "years old";
        exit( ),
        }
        ?>
        <html>
        <body>
    <form action = "<?php $_PHP_SELF?>" method = "POST">
    Name: <input type = "text" name = "name"/>
    Age: <input type = "text" name = "age"/>
    <input type = "Submit"/>
    </form>
    </body>
    </html
```

11.9. PHP SESSIONS

An alternative way to make data accessible across the various page of an entire website is to use a PHP session.

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit. The location of the temporary file is determined by setting in the php.ini file called session.save.path.

When a session is started following things happen:

- PHP first creates a unique identifier for that particular session which is a random string of 32 hexadecimal numbers.
- A cookie called PHPSESSID is automatically sent to the user's computer to store unique session identification string.
- A file is automatically created on the server in the designated temporary directory and bears the name of the unique identifier prefixed by sees–ie.

When a PHP script wants to retrieve the value from a session variable, PHP automatically gets the unique session identifier string from the PHPSESSID cookie and then looks in its temporary directory for the file bearing that name and a validation can be done by comparing both values.

A session ends when the user loses the browser or after leaving the site, the server will temrminate the session after a predetermined period of time, commonly 30 minutes duration.

Starting a PHP Session

A PHP session is easily started by making a call to the session_start( ), function. This function first checks if a session is already started and if none is started then it starts one. It is recommended to put the call to session_start( ) at the beginning of the page.

Session variables are stored in associative array called $_SESSION[ ]. These variables can be accessed during lifetime of a session. The following example starts a session then register a variable called counter that is incremented each time the page is visited during the session. Make use of isset( ) function to check if session variable is already set or not. Put this code in a test.php file and load this file many times to see the result.

```
<?php
session_start( );
if (isset($_SESSION['Counter']))
{
    $_SESSION['Counter']+=1;
}
else
{
    $_SESSION ['Counter'] = 1;
}
$msg = "you have visited this page".$_SESSION['Counter'];
$msg = "in this session";
?>
    <html>
    <head>
    <title> Setting up a PHP session </title>
    </head>
    <body>
    <?php echo ($msg);?>
    </body>
    </html>
```

Destroying a PHP Session

A PHP session can be destroyed by session_destroy ( ) function. This function doesnot need any argument and a single call can destroy all the session variables. If you want to destroy a single session variable then you can use unset ( ) function to unset a session variable.

Here is the example to unset a single variable

```
<? php
unset ($_SESSION ['Counter']);
?>
```

Here is the call which will destroy all the session variables:

```
<? php
session_destroy( );
?>
```

PHP Cookies

Cookies are text files stored on the client computer and they are kept of use tracking purpose. PHP transparently supports HTTP cookies. There are three steps involved in identifying returning users:

- Server script sends a set of cookies to the browser. For example name, age or identification number etc.
- Browser stores this information on local machine for future use.

- When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

Setting Cookies with PHP

PHP provided setcookie ( ) function to set a cookies. This function, requires upto six arguments and should be called before <html> tag. For each cookie this function has to be called separately.

```
Setcookie(name, value, expire, path, domain, security);
```

Here is the detail of all the arguments:

- **Name:** This sets the name of the cookie and is stored in an environment variable called HTTP_COOKIE_VARS. This variable is used while accessing cookies.
- **Value:** This sets the value of the named variable and is the content that you actually want to store.
- **Expiry:** This specify a future time in seconds since 00:00:00 GMT on Ist Jan. 1970. After this time cookie will become inaccessible. If this parameter is not set then cookie will automatically expire when the web browser is closed.
- **Path:** This specifies the directories for which the cookie is valid. A single forward slash character permits the cookie to be valid for all directories.
- **Domain:** This can be used to specify the domain name is very large domains and must contain at least two periods to be valid. All cookies are only valid for the host and domain which created them.

Security: This can be to 1 to specify that the cookie should only be sent by secure transmission using HTTPs othersise set to 0 which mean cookie can be sent by regular HTTP.

Example: This example will create two cookies name and age these cookies wil be expired after one hour.

```
<?php
Setcookie ("name", "John Watkin", time( ) + 3600, "/", " ", 0);
Setcookie ("age", "36", time\) + 3600, "/", " ", 0);
    ?>
    <html>
        <head>
    <title> setting cookies with PHP </title>
        </head>
    <body>
    <? php echo "set cookies"?>
    </body>
    </html>
```

Accessing Cookies with PHP

PHP provides many ways to access cookies. Simplest way is to use either $_cookie or SHTTP_COOKIE_VARS variables. Following example will access all the cookies set in above example.

```
<html>
<head>
<title> Accessing cookies with PHP </title>
</head>
<body>
<?php
echo$_COOKIE ["name"]."<br/>";
/* is equivalent to */
echo $HTTP_COOKIE_VARS ["name"]. "<br/>";
echo$_COOKIE ["age"]."<br/>";
/* is equivalent to */
echo $HTTP_COOKIE_VARS ["name"]."<br/>";
?>
</body>
</html>
```

Deleting Cookie With PHP

Officially, to delete a cookie you should call setcookie( ) with the name argument only but this does not always work well, however, and should not be relied on.

It is safest to set the cookie with a date that has already expired:

```
<?php
setcookie("name", " ", time( ) – 60, "/", " ", 0);
setcookie ("age", " ", time( ) – 60, "/", " ", 0);
?>
<html>
<head>
<title> Deleting Cookies with PHP </title>
</head>
<body>
<?php echo "Deleted Cookies"?>
</body>
</html>
```